KECERDASAN BUATAN (TEORI, ALGORITMA, DAN IMPLEMENTASINYA)

Penulis:
Rahmadi Asri
Wirawan Istiono
Yusuf Wahyu Setiya Putra
Hermila A.
Suwito Pomalingo
Rudi Sutomo
Rahmat Taufik R.L Bau
Jayanti Yusmah Sari
Fahmy Rinanda Saputri
Melissa Indah Fianty
Erick Fernando
Irmawati
Yulius Denny Prabowo
Budi Wijaya Rauf
Angga Aditya Permana

GET PRESS INDONESIA

KECERDASAN BUATAN (TEORI, ALGORITMA, DAN IMPLEMENTASINYA)

Penulis :
Rahmadi Asri
Wirawan Istiono
Yusuf Wahyu Setiya Putra
Hermila A.
Suwito Pomalingo
Rudi Sutomo
Rahmat Taufik R.L Bau
Jayanti Yusmah Sari
Fahmy Rinanda Saputri
Melissa Indah Fianty
Erick Fernando
Irmawati
Yulius Denny Prabowo
Budi Wijaya Rauf
Angga Aditya Permana

ISBN :

Editor : Drs. Amna, MIT
Penyunting: Ariyanto M.Pd
Desain Sampul dan Tata Letak :
Atyka Trianisa, S.Pd

Penerbit : Get Press Indonesia
Anggota IKAPI No. 033/SBA/2022
Redaksi :
Jl. Palarik Air Pacah RT 001 RW 006
Kelurahan Air Pacah Kecamatan Koto Tangah
Padang Sumatera Barat
Website : www.getpress.co.id
Email : globaleksekutifteknologi@gmail.com

Cetakan pertama, Oktober 2023

KATA PENGANTAR

Dengan mengucapkan puji syukur kehadirat Allah SWT, atas limpahan rahmat dan hidayahNya, maka Penulisan Buku dengan judul "KECERDASAN BUATAN (TEORI, ALGORITMA, DAN IMPLEMENTASINYA)" dapat diselesaikan. Buku ini berisikan bahasan kecerdasan buatan dan betapa pentingnya kercerdasan buatan dalam kehidupan modern, kami akan memperkenalkan Anda pada dunia yang sangat menarik dan beragam dari kecerdasan buatan (AI). Kami akan membahas konsep-konsep dasar AI, perkembangan sejarahnya, dan dampaknya yang luar biasa terhadap berbagai aspek kehidupan manusia. Buku ini akan membantu Anda memahami esensi AI, mulai dari konsep dasar, algoritma hingga aplikasi yang lebih canggih, dan memberikan wawasan mendalam tentang bagaimana AI akan terus mengubah cara kita bekerja, bermain, dan hidup di masa depan.

Buku ini masih banyak kekurangan dalam penyusunannya. Oleh karena itu, kami sangat mengaharapkan kritik dan saran demi perbaikan dan kesempurnaan buku ini selanjutnya. Kami mengucapkan terima kasih kepada berbagai pihak yang telah membantu dalam proses penyelesaian Buku ini. Semoga Buku ini dapat menjadi sumber referensi dan literatur yang mudah dipahami serta bermanfaat.

Penulis, 14 November2023

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENGENALAN KECERDASAN BUATAN

Oleh Rahmadi Asri

1.1 Pengantar Kecerdasan Buatan

Perkembangan Ilmu Teknologi saat ini semakin canggih, Teknologi tersebut merupakan suatu produk dari ilmu komputer. Ilmu komputer memiliki cabang begitu luas salah satunya adalah kecerdasan buatan atau disebut *Artificial Intelligence (AI)*. Contoh produk AI yang sering digunakan sehari hari seperti Peta dan Navigasi, Media Sosial, *Voice Assistant*, Editor Tulisan, Rekomendasi Pencarian, *Face ID* dan *E-Payment*.

Kecerdasan merupakan keahlian khusus yang dimiliki oleh setiap manusia, melalui keahlian tersebut manusia bisa menyelesaikan suatu pekerjaan dengan mudah. Ada dua jenis metode yang digunakan untuk menyelesaikan pekerjaan tersebut yaitu dengan cara manual yang dilakukan oleh human itu sendiri dan secara modernisasi menggunakan mesin cerdas yang disebut sebagai kecerdasan buatan.

Kecerdasasan Buatan *(Artificial Intelligence) AI* disebut juga sebagai ilmu interdisipliner dengan berbagai pendekatan, kemajuan dalam pembelajaran mesin dan pembelajaran mendalam, khususnya, menciptakan suatu perubahan paradigma dalam kehidupan sehari-hari, di hampir setiap sektor industri teknologi saat ini.

Kecerdasan buatan memanfaatkan komputer dan mesin untuk meniru kemampuan pikiran manusia dalam memecahkan

masalah dan mengambil keputusan. Kecerdasan buatan diciptakan oleh manusia diimplementasikan ke dalam sistem komputer sehingga kecerdasan buatan dapat melakukan pekerjaaan seperti halnya yang dapat dilakukan oleh manusia.

1.2 Perbedaan Kecerdasan Buatan dan Kecerdasan Alami

Kecerdasan buatan dengan kecerdasan alami memiliki keunggulan masing masing.

1.2.1 Kecerdasan Buatan

1. Bersifat permanen
   Kecerdasan buatan tidak akan pernah berubah jika tidak dilakukan pembaharuan sistem yang baru. Sedangkan kecerdasan alami dapat berubah ubah karena pada hakikatnya manusi memiliki sifat pelupa dan kreatif.
2. Mudah
   Kecerdasan buatan sangat mudah untuk dilakukan duplikat sehingga bisa langsung untuk mentransfer data dari *Personal Computer* 1, ke *Personal Computer* yang lain, dengan cepat dan akurat. Sedangkan kecerdasan alami belum dapat meminimalisir waktu tempuh dengan optimal dalam melakukan transfer data.
3. Murah
   Optimasi biaya yang digunakan relatif lebih kecil dibandingkan dengan kecerdasan alami pada proses penyediaan layanan komputer sedangkan pekerjan yang dilakukan dengan cara manual dapat memakan waktu dalam proses pengerjaan nya.
4. Konsisten
   Bersifat permanen, melakukan perkerjaan dengan teliti, cepat, tepat dan akurat sedangkan kecerdasan alami bersifat tidak tetap atau berubah-ubah.

5. Dapat didokumentasi

Dapat melacak semua rekam jejak digital pada sistem cerdas dengan cara mengecek seluruh aktifitas yang dilakukan oleh sistem tersebut, sedangkan sistem alami sulit untuk di dokumentasikan.

6. Kecepatan
   Kemampuan mengerjakan beberapa pekerjaan lebih cepat dibandingkan dengan manusia.

1.2.2 Kecerdasan Alami

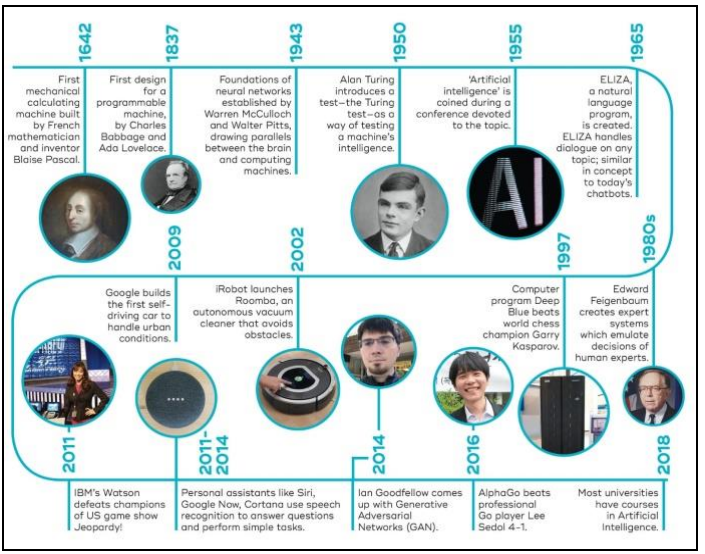
1. Kreatif
   Ilmu pengetahuan manusia terus bertambah secara signifikan, karena manusia mampu beradaptasi dengan lingkungan sekitar, sehingga ide yang dihasilkan manusia dapat diimplementasikan langsung. Sedangkan kecerdasan buatan harus membuat sistem baru sebagai perbaikan daripada sistem lama untuk perubahan.
2. Implementasi
   Ide manusia dapat diterapkan secara langsung, berdasarkan dari hasil pengalaman, sedangkan kecerdasan buatan menerima inputan yang bersipat simbolik.

1.3 Sejarah Kecerdasan Buatan

AI mulai dikenal pada tahun 1950-an. Awalnya para ilmuan merintis AI sebagai solusi untuk memecahkan masalah matematika kompleks dengan menciptakan mesin berfikir. Dukungan terhadap AI mulai menjadi pusat perhatian oleh peneliti, karena melihat adanya pendekatan yang baik terhadap perkembangan AI. Sehingga pemerintah memberikan dana penelitian yang signifikan. Namun pada kenyataannya penelitian tersebut tidak berhasil. Akibatnya pada tahun 1970 penelitian dan komunitas AI berkurang. Tahun 1980 para ilmuan mulai melakukan perbaikan kembali terhadap AI dimana permasalahan sebelumnya dianggap sulit dicapai dan AI kembali tanpak menjanjikan. Namun harapan tersebut kembali

mengalami kegagalan, sehingga pada tahun 1990 para peneliti AI mulai berkurang secara dramatis.

Pada tahun 2000-an AI sudah mulai memasuki ke era baru, minat peneliti AI kembali melonjak tinggi, AI juga mengalami kemajuan yang sangat pesat dimana AI didukung oleh perkembangan algoritma dan teknologi komputer modern. Para ilmuan berpendapat untuk terus meningkatkan pemahaman AI, tujuannya adalah agar AI dapat menjadi solusi untuk memecahkan suatu permasalahan dalam kehidupan sehari-hari, AI juga dapat diimplementasikan disemua sector pada masa yang akan datang. Sejarah kecerdasan buatan dapat dilihat pada gambar dibawah ini:



Gambar 1.1 Sejarah Kecerdasan Buatan
(Sumber: https://qbi.uq.edu.au/brain/intelligent-machines/history-artificial-intelligence)

1.4 Fondasi Ilmu Kecerdasan Buatan

1. Matematika

- Mempelajari dan mendalami Perhitungan antara lain himpunan, aljabar, statistik dan estimasi. geometri dan pengkuran.
- Penalaran logis
- Menguasai logika dan algoritma

2. Ekonomi
   Optimalisasi suatu pencapaian pada proses
3. Filsafat
   Penggunaan ilmu pengetahuan dalam melakukan sesuatu tindakan dan mempresentasikan pikiran sebagai mesin.
4. Ilmu Syaraf
   Bagaimana cara kerja fungsi otak untuk memproses suatu informasi, ilmu syaraf ini lebih dikenal sebagai Jaringan Syaraf Tiruan. Penggunaan jaringan syaraf tiruan untuk menemukan pola pada data yang dimodelkan yaitu data statistik non linier. Jaringan syaraf tiruan hadir untuk meniru cara kerja otak manusia untuk menemukan solusi.
5. Psikologi
   Menerapkan ilmu kognetif. Ilmu kognetif merupakan ilmu yang mempelajari tentang data, dimana data tersebut untuk dapat di olah menjadi suatu informasi yang bermanfaat. Ilmu kognetif juga mempelajari tentang bagaimana pola fikir manusia berkerja, bagaimana pola fikir manusia itu dapat berfungsi dan berprilaku.
6. Teknik Komputer
   Menyediakan perangkat keras dan perangkat lunak untuk pembuatan aplikasi kecerdasan buatan.
7. Teori Kontrol
   Penerapan algoitma dalam meyelesaikan masalah yang ada pada lingkungan sistem kecerdasan buatan, teori kontrol ini dapat memberikan solusi pada sistem cerdas untuk melakukan perkembangan.
8. Bahasa
   - *Knowledge representation language.*
   - *Natural Language Processing* (*NLP*)
     NLP dapat Memadukan teknologi komputasional, lingustik dan pembelajaran mesin.

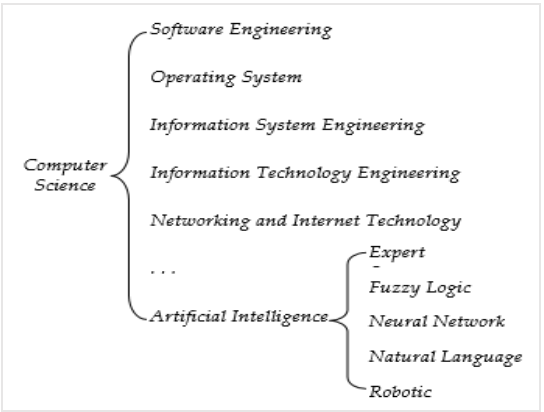
- Pemrosesan bahasa manusia yang dapat dimengerti oleh komputer yaitu berupa *text* dan maupun suara.

1.5 Kedudukan Ilmu Kecerdasan Buatan

Kecerdasan buatan merupakan salah satu dari bagian ilmu komputer yang paling populer saat ini, kecerdasan buatan memberikan dampak positif terhadap penggunanya, salah satu output sistem cerdas adalah pemeliharaan data berbasis *online* seperti *cloud computing.*

Ilmu komputer *(computer science)* khusus membuat produk aplikasi yang bermanfaat, dimana aplikasi tersebut dapat dipakai oleh pengguna sebagaimana halnya komputer dapat meniru fungsi otak manusia.

Kedudukan ilmu kecerdasan buatan dapat dilihat pada bagan dibawah ini:



Gambar 1.2 Bagan Kedudukan Ilmu Kecerdasan Buatan
(Sumber:https://repository.uinjkt.ac.id/dspace/bitstream/123456789/44538/2/naskah%20kecerdasan%20buatan.pdf)

Kedudukan dasar kecerdasan buatan tidak terlepas dari algoritma, struktur data dan teknik bahasa pemrograman. Sedangkan perinsip dasar kecerdasan buatan terdiri dari privasi

untuk keamanan, bersifat transparan, berkeadilan dalam membuat keputusan, memiliki sistem yang handal, inklusivitas dapat menangani kebutuhan manusia dan akuntabilitas yaitu bertanggung jawab atas perkembangan sistem kepada bagian pengembang sistem kecerdasan buatan.

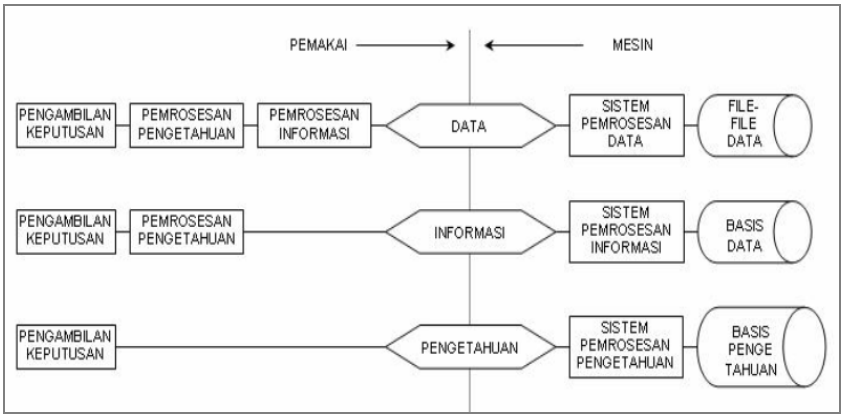
1.6 Ruang Lingkup Kecerdasan Buatan

1. Sistem pakar
   Sistem pakar mulai diterapkan pada tahun 1980-an, sistem pakar merupakan hasil perkembagan riset dari kecerdasan buatan. Sistem pakar adalah perangkat lunak komputer yang berisi tentang pengetahuan atau keahlian seseorang yang bersifat spesifik kemudian diimplementasikan kedalam komputer.
2. Pengenalan ucapan
   Komunikasi manusia dengan komputer menggunakan suara, suara manusia tersebut tersimpan di *memory* komputer sehingga komputer dapat mengenali suara manusia.
3. Robotika dan Sistem sensor
   Sekumpulan Alat mekanik yang dapat melakukan pekerjaan fisik dengan pengawasan sedangkan sistem sensor dapat mengubah sinyal fisik menjadi sinyal elektronik.
4. *Computer Vision*
   Teknologi baru yang merupakan pengembangan dari kecerdasan buatan, teknologi ini memanfaatkan kecerdasan buatan untuk memperoleh data atau inforasi dari input visual yaitu photo dan video.
5. *Intelligence Computer-aided instruction*
   Teknologi informasi pengembangan dari multimedia, teknologi ini sering digunakan sebagai sarana pendidikan, *computer-aided instruction* disebut sebagai bentuk pendidikan didunia maya. Ciri-ciri *computer-aided instruction* tidak dibatasi oleh waktu, ruang dan tempat tidak terbatas, materi-materi pembelajaran dapat diakses langsung oleh pelajar, dapat mendokumentasikan kegiatan

belajar dan mengajar serta materi yang diajarkan berupa (text, gambar, audio, video dan animasi).

1.7 Revolusi Pengolahan Data Oleh Komputer

Sebelum kecerdasan berkembang pesat seperti saat ini, awal nya komputer digunakan untuk pengolahan data. Seiring berkembangnya teknologi komputer dapat menghasilkan informasi yang bermanfaat besar terhadap penggunanya, sehingga sistem komputer dapat memproses dan membantu semua pekerjaan manusia yang lebih efektif dan efesien. Revolusi pengolah data oleh komputer dapat dilihat pada gambar dibawah ini:



Gambar 1.3 Pengolahan Data, Informasi dan Pengetahuan
(Sumber: https://repository.uinjkt.ac.id/dspace/bitstream/123456789/44538/2/naskah%20kecerdasan%20buatan.pdf)

Entitas DATA pada Gambar 1.3, di atas menunjukan bahwa pada sisi pemakai terdiri dari tiga tingkatan proses yaitu pemrosesan informasi, pemrosesan pengetahuan dan pengambilan keputusan, sedangkan pada mesin terdiri dari dua tingkatan proses yaitu sistem pemrosesan data dan file penyimpanan data. Entitas INFORMASI pada Gambar 1.3, di atas menunjukan bahwa dari sisi pemakai tersebut memiliki

dua tingkatan proses yaitu pemrosesan pengetahuan dan pengambilan keputusan, sedangkan pada mesin terdiri dari dua tingkatan proses yaitu sistem pemrosesan informasi dan database. Bagian ketiga entitas PENGETAHUAN pada Gambar 1.3, di atas menunjukan bahwa dari sisi pemakai tersebut memiliki satu entitas yaitu pengambilan keputusan, sedangkan pada mesin memiliki dua tingkatan proses yaitu sistem pemrosesan pengetahuan dan *database* pengetahuan.

Penjelasan pada gambar diatas bahwa pemakai merupakan sebagai pengguna data yaitu manusia, sedangkan mesin sebagai alat yang digunakan untuk memproses data. Data yang diolah oleh komputer berupa data text, data gambar, data audio dan data video. Data data tersebut diproses menggunakan komputer sehingga menghasilkan *output* yang bernilai, *output* tersebut berupa keputusan yang logis sehingga membantu manusia dalam mencari solusi untuk menyelesikan suatu kendala dalam dunia perkerjaan.

1.8 Faktor Pendorong Perkembangan Kecerdasan Buatan

Faktor-faktor yang mempengaruhi perkembangan kecerdasan buatan adalah:

- Perkembangan teknologi pada *Hardware* komputer
- Perkembangan Software Komputer
- Perkembangan komputer khusus untuk pengguna pribadi
- Partisifasi investor untuk mengembangkan sistem AI

1.9 Contoh Penerapan Kecerdasan Buatan

Beberapa contoh aplikasi atau program *output* dari kecerdasan buatan dapat dilihat sebagai berikut:

1. *Deep Blue*
   Deep blue yaitu program Catur, program tersebut dapat mengalahkan pecatur kelas dunia yang bernama Garyy Kasparov, pada tahun 1997.
2. *Logic Theorist*
   Dapat menyelesaikan masalah logika pada matematika menggunakan algoritma.
3. *Systran*
   Perangkat lunak yang ini digunakan untuk penerjemah bahasa, bahasa yang diterjemahkan terdiri dari bahasa Peranci, Jerman, Korea, Jepang, Italia, Rusia, Portugis dan Spanyol.
4. *Delco Electronics*
   Pengemudian mobil secara mandiri tanpa dikemudikan oleh manusia, teknologi ini menggunakan alat pendekteksi canggih untuk tetap berada pada jalur-jalur jalan yang digunakan

DAFTAR PUSTAKA

http://eprints.binadarma.ac.id/15964/1/UAS%20METODOLOGI%20PENELITIAN%20Ummi%20Kalsum.pdf

http://repository.uinsu.ac.id/10772/1/DIKTAT%20KECERDASAN%20BUATAN%20EDISI%20REVISI%20I.pdf

https://stekom.ac.id/artikel/apa-itu-ai-kecerdasan-buatan-pengertian-kelebihan-kekurangan

https://algorit.ma/blog/produk-artificial-intelligence-2022/

https://sisfo.itp.ac.id/bahanajar/BahanAjar/Minarni/kecerdasan_buatan/BAB%20I%20Pendahuluan.pdf.

https://qbi.uq.edu.au/brain/intelligent-machines/history-artificial-intelligence

http://repository.uin-suska.ac.id/20880/7/BAB%202.pdf

https://rifqimulyawan.com/blog/pengertian-cognitive-science/

https://repository.uinjkt.ac.id/dspace/bitstream/123456789/44538/2/naskah%20kecerdasan%20buatan.pdf

BAB 2
Sistem Cerdas

Oleh Wirawan Istiono

2.1 Apa Itu Sistem Cerdas?

Sistem cerdas mengacu pada perangkat teknologi yang sangat canggih yang mampu memahami dan bereaksi terhadap lingkungan sekitarnya. Sistem cerdas mencakup berbagai manifestasi, mulai dari penyedot debu otonom seperti Roomba hingga algoritme pengenalan wajah dan rekomendasi belanja pada aplikasi Amazon.

Disiplin sistem cerdas juga menekankan pada cara di mana sistem ini terlibat dengan pengguna manusia dalam konteks fisik dan sosial yang berkembang dan dinamis. Pada tahap awal, robot menunjukkan otonomi terbatas dalam pengambilan keputusan, karena mereka beroperasi di bawah asumsi lingkungan yang dapat diprediksi dan melakukan tindakan berulang secara konsisten dalam keadaan yang sama. Dalam wacana kontemporer, robot umumnya didefinisikan sebagai entitas otonom yang mampu melihat sekelilingnya dan melakukan tindakan fisik dalam dunia fisik dengan tujuan mencapai tujuan tertentu (Unr.edu, 2020).

Sistem cerdas mengacu pada sistem komputer canggih yang memiliki kemampuan untuk mengumpulkan, menganalisis, dan menghasilkan respons berdasarkan data yang diperolehnya dari lingkungan terdekatnya. Sistem memiliki kemampuan untuk berinteraksi dan bertukar informasi secara efektif dengan banyak entitas, termasuk pengguna dan sistem komputer

lainnya. Selain itu, ia memiliki kemampuan untuk memperoleh pengetahuan melalui pengalaman dan menyesuaikan perilakunya berdasarkan informasi terkini yang tersedia. Selain itu, sistem cerdas berpotensi memfasilitasi pemantauan dan pengelolaan jarak jauh.

Konsep sistem cerdas tidak memiliki definisi yang disepakati secara universal. Terminologi ini digunakan dengan cara yang berbeda oleh sumber yang berbeda. Definisi sistem cerdas tidak memiliki norma yang diterima secara universal. Penentuan atribut individu seringkali didasarkan pada berbagai tingkat kapasitas intelektual daripada seperangkat sifat yang kaku. Sistem cerdas dicirikan oleh empat atribut utama. Pertama, ia berfungsi dalam lingkungan yang mencakup agen lain. Kedua, menunjukkan kemampuan kognitif seperti persepsi, kontrol tindakan, penalaran deliberatif, dan penggunaan bahasa. Ketiga, menganut prinsip-prinsip perilaku yang didasarkan pada rasionalitas dan norma-norma sosial. Terakhir, ia memiliki kemampuan untuk beradaptasi dan meningkatkan kinerjanya melalui pembelajaran (Robert Sheldon dan Ivy Wigmore, 2022).

Dalam artikel ilmiahnya yang berjudul "What is an intelligent system?", Martin Molina, seorang anggota fakultas yang terhormat dalam departemen kecerdasan buatan di Technical University of Madrid, menyajikan penjelasan komprehensif tentang sistem cerdas. Tanggal publikasi sumber tersebut adalah Februari 2022. Lebih lanjut Molina menegaskan bahwa konsepnya tidak boleh ditafsirkan sebagai karakterisasi yang ketat untuk tujuan menentukan ada atau tidaknya kecerdasan dalam sistem tertentu. Sebaliknya, definisi menggambarkan atribut khas yang mungkin terwujud dalam sistem cerdas. Contoh dari suatu sistem dapat dianggap cerdas berdasarkan pada kepemilikan persepsi dan kemampuannya untuk memanipulasi tindakan, bahkan tanpa adanya penalaran deliberatif atau kapasitas untuk belajar (Molina, 1998).

Menurut individu, munculnya gagasan sistem cerdas di bidang teknologi informasi dapat dikaitkan dengan keberhasilan penerapan kecerdasan buatan. Memang, kecerdasan buatan (AI) sering dirujuk sebagai atribut mendasar dari setiap makhluk cerdas. Menurut Molina, faktor penting dalam mengembangkan sistem cerdas adalah pemilihan metodologi kecerdasan buatan (AI) yang cermat yang melengkapi sistem dengan kemampuan kognitif yang diperlukan untuk beroperasi sebagai entitas yang komprehensif.

Sistem pintar mencakup integrasi penginderaan, aktuasi, dan kemampuan kontrol untuk menggambarkan dan mengevaluasi situasi tertentu secara efektif. Sistem ini memanfaatkan data yang tersedia untuk membuat penilaian berdasarkan informasi, baik melalui cara prediktif maupun adaptif, yang menghasilkan pelaksanaan tindakan cerdas. Kecerdasan sistem sering dikaitkan dengan kemampuannya untuk beroperasi secara mandiri melalui kontrol loop tertutup, efisiensi energinya, dan kemampuan jaringannya (Wikipedia, 2023).

Namun demikian, penting untuk dicatat bahwa sistem cerdas tidak boleh digabungkan dengan kecerdasan buatan (AI), meskipun terkadang konsep-konsep ini dapat dipertukarkan. Kecerdasan buatan (AI) adalah upaya komputer dan perangkat lain untuk mereplikasi kecerdasan manusia, meskipun keberadaan AI semata tidak secara inheren menunjukkan adanya sistem cerdas. Misalnya, pemanfaatan komputer untuk menjalankan aplikasi berdasarkan kecerdasan buatan tidak berarti bahwa komputer memiliki kecerdasan bawaan. Namun, itu dapat berfungsi sebagai agen dalam sistem cerdas, sehingga berkontribusi pada keseluruhan fungsinya (Robert Sheldon dan Ivy Wigmore, 2022).

2.2 Aplikasi Sistem Cerdas

Pemanfaatan sistem cerdas telah menemukan beberapa aplikasi dalam domain yang berbeda. Sistem ini telah digunakan di bidang-bidang seperti kesehatan, keuangan, transportasi, dan manufaktur, antara lain. Integrasi sistem cerdas telah memfasilitasi kemajuan. Sistem cerdas diposisikan untuk memikul tanggung jawab yang semakin meningkat dalam masyarakat kontemporer, meliputi (Unr.edu, 2020):

- Penerapan sistem otomasi di fasilitas manufaktur, yang biasa disebut otomasi pabrik, semakin lazim dalam beberapa tahun terakhir.
- Topik bahasannya berkaitan dengan bidang robotika, khususnya bidang robotika servis.
- Bidang robotika bantu meliputi pengembangan dan penerapan sistem robotik yang dirancang untuk membantu penyandang disabilitas atau keterbatasan dalam
- Pemanfaatan teknologi militer dalam berbagai konteks dan skenario telah menjadi perhatian dan analisis yang signifikan di kalangan akademisi. Eksplorasi dari
- Penyediaan perawatan medis
- Topik pendidikan dan hiburan sangat penting dalam wacana akademik.
- Tindakan memeriksa objek atau sistem secara visual.
- Proses pengenalan karakter mengacu pada identifikasi otomatis dan interpretasi karakter atau simbol dalam gambar digital atau dokumen cetak.
- Identifikasi individu melalui penggunaan modalitas biometrik yang berbeda, seperti pengenalan wajah, analisis sidik jari, pemindaian iris, dan geometri tangan.
- Topik yang menarik adalah pengawasan visual.

- Transportasi cerdas mengacu pada penerapan teknologi dan sistem canggih di sektor transportasi untuk meningkatkan efisiensi, keselamatan, dan keberlanjutan.

Sistem cerdas sangat lazim dan menghadirkan peluang tanpa batas. Kecerdasan buatan (AI) memiliki potensi untuk menghasilkan beberapa keunggulan di berbagai sektor, memberikan dampak pada bidang individu dan pekerjaan kita. Tabel berikutnya menyajikan contoh bagaimana sistem cerdas dapat digunakan secara efektif di banyak industri. Tinjauan singkat ini memberikan wawasan tentang kemungkinan pemanfaatan sistem cerdas (Robert Sheldon dan Ivy Wigmore, 2022).

Tabel 2.1 Aplikasi potensial dari sistem cerdas.

Industri	**Contoh**
Transportation	Topik yang diminati meliputi sistem kontrol lalu lintas, studi arus lalu lintas dan kemacetan, koordinasi armada pengiriman, kendaraan otonom, sistem angkutan umum, dan pencegahan kecelakaan.
Aerospace	Subjek yang diminati meliputi kontrol yang ditemukan di kokpit maskapai, kendaraan udara tak berawak (drone), pemantauan pesawat ruang angkasa, perencanaan misi, pengembangan sistem panduan dan navigasi yang canggih, dan bidang kontrol lalu lintas udara.
Human identification	Pemanfaatan pengawasan visual, pemantauan biometrik (yaitu pengenalan wajah atau sidik jari), serta teknik pengenalan karakter atau ucapan, lazim dalam wacana akademis kontemporer.

Manufacturing	Bidang robotika dan otomatisasi telah melihat kemajuan yang signifikan dalam beberapa tahun terakhir. Kemajuan ini telah mengarah pada pengembangan sistem otonom yang mampu melakukan tugas tanpa campur tangan manusia. Selain itu, ada peningkatan signifikan dalam diagnostik dan perbaikan sistem robot, memastikan fungsionalitas optimalnya. Bidang kemajuan lainnya adalah manajemen inventaris, di mana sistem otomatis digunakan untuk melacak dan mengontrol tingkat inventaris secara efisien. Terakhir, teknik inspeksi visual telah ditingkatkan, memungkinkan robot menilai dan menganalisis data visual secara akurat untuk berbagai aplikasi.
Education	Bimbingan, proses memberikan bantuan akademik yang dipersonalisasi kepada siswa, melibatkan pendeteksian pola belajar individu dan menggunakan sistem tanya jawab untuk memfasilitasi perolehan pengetahuan. Selain itu, perencanaan dan penjadwalan kursus memainkan peran penting dalam mengatur kegiatan pendidikan, sementara pendidikan online interaktif menawarkan platform yang menarik bagi siswa untuk berpartisipasi aktif dalam proses pembelajaran.
Climatology	Peramalan, sistem peringatan dini (seperti tornado atau tsunami), penelitian dan pemantauan lapangan,pencitraan dan analisis satelit

Retail	Makalah akademik ini bertujuan untuk memberikan rekomendasi untuk belanja online, terminal point-of-sale (POS), digital signage, serta perencanaan dan pengiriman produk.
Home automation	Pemanfaatan peralatan cerdas, sistem pengaturan iklim, dan mekanisme kontrol pencahayaan menjadi semakin umum di masyarakat kontemporer. Selain itu, integrasi teknologi ini telah berkontribusi pada peningkatan pengalaman hiburan.
Public health	Bidang fokus yang tercakup dalam diskusi ini meliputi perawatan medis, diagnostik medis, telemonitoring, manajemen krisis kesehatan (khusus mengacu pada pandemi COVID-19), pengenalan pola kekerasan, dan rekayasa biomedis.

Sumber: (Robert Sheldon dan Ivy Wigmore, 2022)

Penggabungan kemampuan kognitif bawaan adalah elemen fundamental dari internet of things (IoT). Perangkat Internet of Things (IoT) memiliki kemampuan untuk mengirimkan data secara mandiri melalui jaringan tanpa memerlukan campur tangan manusia. Ketika dikombinasikan dengan kecerdasan buatan dan teknologi generasi kelima (5G), Internet of Things (IoT) memiliki kapasitas untuk mengembangkan sistem canggih yang memiliki kemampuan untuk mempengaruhi berbagai aspek keberadaan manusia secara signifikan (Robert Sheldon dan Ivy Wigmore, 2022).

2.3 Tantangan dalam Sistem Cerdas

Bidang penelitian sistem cerdas menemui banyak kendala, terutama yang berkaitan dengan representasi komputer dari lingkungan fisik yang berubah.

1. Ketidakpastian muncul dari keterbatasan yang melekat pada sensor fisik dan efektor, yang menawarkan kemampuan informasi dan tindakan yang terbatas, tidak tepat, dan bising. Oleh karena itu, potensi ketidakakuratan dalam tindakan sistem muncul dari kebisingan sensor dan kendala dalam melakukan tindakan tersebut.
2. Dunia fisik mengalami perubahan konstan, memerlukan kebutuhan pengambilan keputusan yang cepat untuk beradaptasi dengan lingkungan yang berkembang.
3. Perhitungan yang membutuhkan banyak waktu. Proses menemukan rute yang paling efisien menuju tujuan yang diinginkan memerlukan eksplorasi menyeluruh dari serangkaian kemungkinan keadaan yang luas, yang menghasilkan biaya pemrosesan yang signifikan. Salah satu batasan potensial yang terkait dengan mendedikasikan waktu berlebihan untuk tugas komputasi adalah kemungkinan lanskap global dapat mengalami transformasi selama periode ini, sehingga menyebabkan rencana yang dihitung menjadi usang.
4. Proses pemetaan melibatkan kehilangan informasi yang signifikan saat beralih dari representasi tiga dimensi (3D) ke representasi dua dimensi (2D). Visi komputer menghadapi berbagai masalah, seperti perubahan perspektif, kondisi pencahayaan, dan skala. Selain itu, ia harus mengatasi masalah yang berkaitan dengan kekacauan latar belakang atau gerakan, serta tugas mengelompokkan entitas yang menunjukkan variasi intra-kelas dan antar-kelas.

Ada banyak masalah di bidang Kecerdasan Buatan, yang sekarang akan kita lanjutkan untuk membahas dan mengusulkan solusi potensial (Vadapalli, 2022).

1. Daya Komputer

Konsep daya komputasi mengacu pada kapasitas sistem komputer untuk melakukan tugas komputasi dan memproses informasi. Ini adalah aspek mendasar dari teknologi komputer yang menentukan kecepatan dan

Konsumsi daya dari algoritme ini, didorong oleh selera sumber daya yang tak terpuaskan, berfungsi sebagai pencegah bagi sebagian besar pengembang. Pembelajaran Mesin dan Pembelajaran Jauh berfungsi sebagai komponen dasar Kecerdasan Buatan, yang membutuhkan jumlah inti dan GPU yang semakin besar untuk mencapai kinerja optimal. Ada bidang lain di mana penerapan kerangka kerja pembelajaran mendalam, seperti pemantauan asteroid, penerapan layanan kesehatan, dan pelacakan badan kosmik, dapat diterapkan untuk memanfaatkan konsep dan pengetahuan yang ada (Vadapalli, 2022).

Pemanfaatan superkomputer memerlukan sejumlah besar kapasitas komputasi, yang harus dicatat, membutuhkan biaya yang cukup besar. Pemanfaatan Cloud Computing dan sistem pemrosesan paralel telah meningkatkan efisiensi pengembang yang mengerjakan sistem AI; meskipun demikian, penting untuk mengakui bahwa kemajuan ini bukannya tanpa biaya. Terlepas dari peningkatan masuknya volume data yang tak terduga dan proliferasi cepat dari algoritma yang rumit, tidak mungkin secara finansial bagi semua individu untuk menanggung biaya tersebut.

2. Kurangnya Kepercayaan

Sifat ketidakpastian dari proses prediksi dalam model deep learning merupakan perhatian yang signifikan untuk kecerdasan buatan. Memahami cara di mana serangkaian input tertentu

dapat menghasilkan solusi untuk berbagai jenis masalah menimbulkan tantangan bagi individu yang tidak memiliki pengetahuan khusus dalam materi pelajaran.

Sebagian besar populasi global tidak memiliki kesadaran tentang tujuan dan keberadaan Kecerdasan Buatan (AI), serta integrasinya ke dalam objek biasa, seperti telepon pintar, Smart TV, sistem perbankan, dan bahkan kendaraan, hingga berbagai tingkat otomatisasi (Vadapalli , 2022).

3. Pengetahuan Terbatas

Individu memiliki pemahaman terbatas atau keakraban dengan materi pelajaran tertentu. Ada banyak aplikasi komersial di mana Kecerdasan Buatan dapat berfungsi sebagai pilihan unggul untuk teknologi konvensional. Perhatian utama terletak pada pemahaman dan pemahaman tentang Kecerdasan Buatan. Selain individu yang memiliki minat besar pada teknologi, mahasiswa, dan peneliti, terdapat demografi terbatas yang memiliki kesadaran mengenai kemungkinan kecerdasan buatan (Vadapalli, 2022).

Misalnya, banyak Usaha Kecil dan Menengah (UKM) memiliki kesempatan untuk menerapkan jadwal kerja terstruktur dan memperoleh pengetahuan tentang strategi baru untuk meningkatkan kemampuan produksi mereka. Selain itu, mereka dapat mengembangkan kemahiran dalam manajemen sumber daya, penjualan online dan manajemen produk, serta mendapatkan wawasan tentang perilaku pelanggan dan berhasil merespons dinamika pasar. Selain itu, pengetahuan mereka tentang penyedia layanan dalam bisnis teknologi, seperti Google Cloud, Amazon Web Services, dan entitas serupa lainnya, terbatas.

4. Tingkat Manusia

Konsep kecerdasan tingkat manusia mengacu pada kemampuan sistem buatan untuk melakukan tugas dan menunjukkan

kemampuan kognitif yang sebanding. Ini adalah tantangan yang signifikan dalam bidang kecerdasan buatan, yang telah menimbulkan rasa antisipasi di antara para peneliti yang terlibat dalam pengembangan layanan AI baik untuk perusahaan mapan maupun perusahaan baru yang baru muncul. Sementara perusahaan-perusahaan ini mungkin mengklaim telah mencapai tingkat akurasi melebihi 90%, penting untuk dicatat bahwa kinerja manusia melampaui kinerja mereka dalam semua situasi ini. Sebagai ilustrasi, biarkan model kita membuat prediksi mengenai klasifikasi gambar sebagai anjing atau kucing. Manusia memiliki kemampuan untuk secara akurat memperkirakan hasil yang diinginkan dengan tingkat presisi yang luar biasa, mencapai tingkat akurasi yang luar biasa lebih dari 99% (Vadapalli, 2022).

Untuk mencapai kinerja yang sebanding dalam model pembelajaran yang mendalam, diperlukan tingkat penyempurnaan yang tak tertandingi, optimalisasi hyperparameter, pemanfaatan kumpulan data yang substansial, penerapan algoritme yang terdefinisi dengan baik dan akurat, serta akses ke kapasitas komputasi yang kuat. . Selain itu, penting untuk memastikan pelatihan tanpa gangguan pada data pelatihan dan pengujian berikutnya pada data uji. Tugas yang dihadapi tampaknya cukup padat karya, melebihi ekspektasi awal dengan selisih yang signifikan dalam hal kompleksitas.

Salah satu strategi potensial untuk mengurangi beban tenaga kerja adalah dengan melibatkan layanan penyedia, karena mereka memiliki kemampuan untuk melatih model pembelajaran mendalam khusus dengan memanfaatkan model yang sudah ada sebelumnya. Model menjalani pelatihan ekstensif pada kumpulan data besar yang terdiri dari jutaan foto dan kemudian disesuaikan untuk mengoptimalkan akurasinya. Namun, terlepas dari upaya ini, mereka masih menunjukkan kesalahan dan menghadapi tantangan yang signifikan dalam

mencapai tingkat kinerja yang sebanding dengan manusia (Vadapalli, 2022).

5. Privasi dan Keamanan Data

Topik privasi dan keamanan data sangat penting dalam masyarakat kontemporer.

Penentu mendasar yang mendasari semua model pembelajaran mendalam dan mesin adalah aksesibilitas data dan sumber daya yang diperlukan untuk pelatihan mereka. Memang, meskipun kami memiliki sejumlah besar data, penting untuk diketahui bahwa data ini berasal dari sejumlah besar pengguna yang tersebar di berbagai lokasi geografis. Akibatnya, sangat penting untuk mengenali potensi risiko yang terkait dengan pemanfaatan data ini untuk niat jahat (Vadapalli, 2022).

Misalnya, pertimbangkan skenario hipotetis di mana penyedia layanan medis memperluas layanannya ke populasi satu juta orang di wilayah perkotaan tertentu. Dalam skenario ini, serangan dunia maya terjadi, yang mengakibatkan perolehan data pribadi tanpa izin milik satu juta pelanggan. Informasi yang dikompromikan ini kemudian dapat diakses oleh individu yang beroperasi di dalam web gelap. Kumpulan data mencakup informasi yang berkaitan dengan berbagai penyakit, kondisi kesehatan, riwayat medis, dan faktor terkait lainnya. Selain itu, saat ini kita dihadapkan pada tantangan untuk mengelola data dalam skala planet. Mengingat banyaknya arus masuk informasi dari berbagai sumber, kemungkinan besar akan terjadi kebocoran data.

Organisasi tertentu telah menerapkan strategi baru untuk mengatasi hambatan tersebut. Data dilatih pada perangkat pintar, sehingga menghilangkan kebutuhan transmisi data ke server. Hanya model terlatih yang dikirim kembali ke perusahaan (Vadapalli, 2022).

6. Isu Bias

Implikasi etis dari sistem AI bergantung pada sejauh mana data pelatihan yang diterimanya. Oleh karena itu, perolehan data yang akurat dan andal muncul sebagai solusi utama untuk pengembangan sistem kecerdasan buatan yang efektif di masa mendatang. Namun, penting untuk dicatat bahwa data yang dikumpulkan oleh organisasi dalam situasi sehari-hari seringkali berkualitas rendah dan kurang penting.

Individu yang bersangkutan menunjukkan prasangka karena mereka secara selektif menggambarkan karakteristik dan atribut dari kelompok terbatas individu yang memiliki afiliasi serupa yang berkaitan dengan agama, etnis, jenis kelamin, komunitas, dan kecenderungan ras lainnya. Penyelesaian efektif dari masalah ini memerlukan pembentukan algoritme yang mampu memantaunya dengan mahir (Vadapalli, 2022).

7. Kelangkaan Data

Fenomena kelangkaan data mengacu pada terbatasnya ketersediaan data untuk kepentingan analisis dan penelitian. Perusahaan terkemuka, seperti Google, Facebook, dan Apple, telah dihadapkan dengan tuduhan yang berkaitan dengan penggunaan data yang dibuat pengguna secara tidak etis. Akibatnya, negara-negara tertentu, seperti India, telah menerapkan peraturan teknologi informasi yang ketat untuk membatasi penyebaran data tersebut. Akibatnya, perusahaan-perusahaan ini saat ini dihadapkan pada tantangan untuk memanfaatkan data lokal untuk menghasilkan aplikasi global, sehingga berpotensi menimbulkan bias.

Data memainkan peran penting dalam bidang kecerdasan buatan (AI), karena berfungsi sebagai komponen dasar untuk mesin pelatihan untuk memperoleh pengetahuan dan menghasilkan prediksi. Secara khusus, data berlabel digunakan untuk memfasilitasi proses pembelajaran sistem AI. Beberapa perusahaan saat ini sedang mengejar inovasi, dengan penekanan khusus pada pengembangan model kecerdasan buatan (AI) yang

mampu menghasilkan hasil yang tepat bahkan dalam situasi ketika data terbatas. Kehadiran informasi yang bias berpotensi merusak integritas seluruh sistem (Vadapalli, 2022).

2.4 Mempelajari Sistem Cerdas

Untuk melanjutkan studi sistem cerdas, siswa harus memiliki kapasitas untuk memahami dan mensintesis pengetahuan yang berasal dari beberapa domain disiplin ilmu, meliputi (Unr.edu, 2020):

- Materi pelajaran yang ada berkaitan dengan bidang pemrograman.
- Topik diskusi berkaitan dengan struktur data.
- Algoritma adalah konsep dasar dalam ilmu komputer dan matematika. Itu adalah prosedur langkah demi langkah
- Pengenalan pola adalah proses kognitif untuk mengidentifikasi dan mengkategorikan keteraturan atau struktur berulang dalam data atau informasi. Ini melibatkan kemampuan untuk
- Pembelajaran mesin adalah subbidang kecerdasan buatan yang berfokus pada pengembangan algoritme dan model yang memungkinkan sistem komputer untuk belajar dan membuat
- Kecerdasan buatan (AI) mengacu pada bidang ilmu komputer yang berfokus pada pengembangan mesin cerdas yang mampu melakukan tugas-tugas itu
- Bidang studi yang berhubungan dengan prinsip dasar materi, energi, dan interaksinya
- Metode numerik mengacu pada teknik matematika yang digunakan untuk memecahkan masalah dan

mendapatkan solusi perkiraan melalui penggunaan perhitungan numerik. Pendekatan ini

- Bidang psikologi mencakup studi ilmiah tentang perilaku manusia dan proses mental. Ini melibatkan

Selain itu, perolehan keterampilan matematika memegang peranan penting. Trigonometri, aljabar linier, dan kalkulus secara teratur digunakan dalam aplikasi praktis. Statistik dan probabilitas adalah kompetensi penting dalam semua disiplin ilmu yang berkaitan dengan sistem cerdas. Mengembangkan kemahiran dalam Linux juga sangat menguntungkan.

2.5 Perbedaan antara Kecerdasan Manusia dan Mesin

Kesenjangan antara kecerdasan manusia dan kecerdasan mesin adalah topik yang menarik dan menarik (Tutorialspoint.com, 2020).

- Persepsi manusia dicirikan oleh kemampuan untuk mengenali dan menginterpretasikan pola, sedangkan persepsi mesin bergantung pada penerapan aturan yang telah ditentukan dan analisis data.
- Proses penyimpanan dan pengambilan informasi pada manusia bergantung pada pengenalan pola, sedangkan mesin menggunakan algoritma pencarian untuk tujuan ini. Sebagai gambaran, nilai numerik 40404040 memiliki atribut yang memudahkan dalam menghafal, menyimpan, dan mengambil kembali karena polanya yang lugas.
- Manusia memiliki kemampuan untuk secara akurat melihat dan memahami suatu item secara keseluruhan, bahkan dalam kasus di mana bagian-bagian tertentu mungkin tidak ada atau terdistorsi. Sebaliknya, robot kekurangan kapasitas untuk melakukan tugas ini dengan tingkat akurasi yang sama.

2.5.1 Sisi Natural

Kecerdasan Manusia berusaha untuk secara efektif menyesuaikan diri dengan lingkungan baru dengan menggunakan beragam proses kognitif. Sebaliknya, Kecerdasan Buatan berupaya membangun robot yang memiliki kemampuan untuk meniru perilaku manusia dan melakukan tindakan yang mirip dengan yang dilakukan oleh manusia. Otak manusia memiliki karakteristik analog, sedangkan mesin beroperasi secara digital. (Pavan Vadapalli, 2022)

Bidang Kecerdasan Buatan (AI) Kecerdasan Buatan (AI) berupaya untuk meningkatkan produktivitas tenaga kerja dengan meniru kecerdasan manusia, sehingga memfasilitasi proses pemecahan masalah dengan kerumitan minimal. Kemampuan untuk memecahkan berbagai macam masalah dengan cepat merupakan karakteristik dari sistem ini, berbeda dengan kecerdasan manusia yang membutuhkan banyak waktu untuk beradaptasi dengan berbagai prosedur. Untuk memastikan perbedaan utama antara kecerdasan alami dan buatan, penting untuk mempertimbangkan variasi dalam proses fungsionalnya dan persyaratan temporal yang terkait dengan masing-masingnya.

2.5.2. Sisi Functioning

Kemampuan kognitif manusia mencakup pemanfaatan kapasitas komputasi otak, retensi memori, dan penalaran kognitif, sedangkan robot yang digerakkan oleh AI sangat bergantung pada input data dan instruksi tepat yang diberikan ke sistem (Pavan Vadapalli, 2022).

Selain itu, pemrosesan kognitif dan pemahaman tentang isu-isu kompleks oleh manusia membutuhkan banyak waktu, serta kebutuhan untuk adaptasi dan keakraban dengan isu-isu tersebut. Saat mempertimbangkan Kecerdasan Buatan, pemanfaatan input yang tepat dan pemeriksaan menyeluruhnya memfasilitasi pembuatan hasil yang tepat.

2.5.3. Learning Power Side

Kecerdasan manusia mencakup kapasitas untuk memperoleh pengetahuan dan keterampilan melalui asimilasi berbagai situasi dan pengalaman masa lalu. Konsepnya berkaitan dengan perolehan pengetahuan melalui proses membuat kesalahan dan selanjutnya menyempurnakan pendekatan seseorang selama masa hidup individu. Kecerdasan Manusia pada dasarnya didasarkan pada adanya pemikiran cerdas dan aktivitas cerdas. Namun, Kecerdasan Buatan tertinggal dalam hal ini karena komputer tidak memiliki kemampuan untuk terlibat dalam proses kognitif. Oleh karena itu, ketika membandingkan AI dengan otak manusia, terbukti bahwa kecerdasan manusia memiliki kapasitas kognitif yang jauh lebih besar daripada kecerdasan buatan. Selain itu, manusia menunjukkan kemampuan pemecahan masalah yang luar biasa yang bergantung pada keadaan khusus yang dihadapi (Pavan Vadapalli, 2022).

Sistem kecerdasan buatan memiliki kapasitas untuk memperoleh pengetahuan dari data dan meningkatkan kinerjanya melalui pelatihan berkelanjutan. Namun, penting untuk dicatat bahwa sistem ini tidak dapat mereplikasi proses kognitif khas yang melekat pada manusia. Meskipun sistem yang didukung AI menunjukkan kinerja yang kuat dalam tugas tertentu, perolehan serangkaian kemampuan baru untuk area aplikasi yang berbeda dapat menjadi proses yang memakan waktu selama bertahun-tahun.

Kecerdasan Buatan memiliki kapasitas untuk memperoleh pengetahuan dari data dan menyempurnakan kinerjanya melalui proses pelatihan berulang. Namun, pada dasarnya tidak mampu mencapai proses kognitif yang rumit yang eksklusif untuk manusia. Meskipun sistem bertenaga AI menunjukkan kinerja yang kuat dalam tugas tertentu, perolehan serangkaian kemampuan baru untuk area aplikasi yang berbeda dapat

menjadi proses yang memakan waktu selama bertahun-tahun (Pavan Vadapalli, 2022).

2.6 Apa yang Tidak Dapat Dilakukan Oleh Kecerdasan Buatan – Faktor "Manusia"

Perdebatan antara Kecerdasan Buatan dan Kecerdasan Manusia pada dasarnya tidak seimbang. Memang benar bahwa AI telah membuat kemajuan signifikan dalam penciptaan robot cerdas yang melampaui kemampuan manusia di bidang tertentu (seperti AlphaGo dan DeepBlue), masih ada celah yang cukup besar untuk dijembatani guna mencapai potensi penuh dari otak manusia. . Sementara sistem AI secara khusus direkayasa dan dididik untuk mereplikasi dan meniru perilaku manusia, mereka tidak mampu membuat keputusan rasional seperti yang dibuat oleh manusia (Pavan Vadapalli, 2022).

Selain itu, ketika membahas perbedaan antara kecerdasan manusia dan kecerdasan mesin, penting untuk diketahui bahwa kecerdasan manusia memainkan peran penting dalam membentuk simulasi yang dihasilkan oleh kecerdasan mesin. Perbedaan utama antara Kecerdasan Alam dan Kecerdasan Buatan terletak pada data yang diberikan kepada mereka, bersama dengan keterbatasan yang melekat pada kemampuan pemecahan masalah yang menyertai masing-masing bentuk kecerdasan ini (Romeroet al., 2020).

Kemampuan pengambilan keputusan sistem AI sebagian besar berasal dari analisis kejadian, data yang telah dilatih, dan relevansi kontekstualnya dengan kejadian tertentu. Komputer AI tidak dapat memahami gagasan "sebab dan akibat" karena kurangnya akal sehat yang melekat. Menurut Nick Burns, seorang Ilmuwan Data yang berspesialisasi dalam Layanan SQL, dengan tepat dinyatakan bahwa keefektifan model bergantung pada kualitas data yang mendasarinya.

Manusia memiliki kapasitas khusus untuk memperoleh pengetahuan dan kemudian menggunakannya dalam hubungannya dengan pemikiran logis, rasionalitas, dan pemahaman. Situasi dunia nyata memerlukan strategi yang komprehensif dan masuk akal yang mempertimbangkan faktor logis dan emosional, yang disesuaikan dengan karakteristik unik manusia. Oleh karena itu, karakteristik tertentu dari perbedaan antara kecerdasan manusia dan mesin menunjukkan bahwa kecerdasan manusia menunjukkan kelayakan yang lebih besar dibandingkan dengan fitur lainnya (Pavan Vadapalli, 2022).

2.7 Implikasi Masa Depan Kecerdasan Buatan Dibandingkan dengan Kecerdasan Manusia

Saat ini, kecerdasan buatan (AI) sedang dalam kondisi pengembangan dan kemajuan yang berkelanjutan. Durasi yang diperlukan untuk melatih sistem kecerdasan buatan (AI) sangat panjang, membuatnya tidak dapat dicapai tanpa keterlibatan bantuan manusia. Kendaraan otonom dan robotika, serta teknologi canggih seperti pemrosesan bahasa alami dan analisis gambar, bergantung pada kecerdasan manusia (Pavan Vadapalli, 2022).

Kecerdasan manusia bertanggung jawab atas perkembangan pertama kecerdasan buatan. Oleh karena itu, sejauh mana kecerdasan manusia memodifikasi kecerdasan buatan untuk memasukkan fungsionalitas baru merupakan faktor penentu. Oleh karena itu, untuk memeriksa wacana lama seputar manfaat komparatif, terbukti bahwa kecerdasan manusia memiliki validitas yang lebih besar dalam merancang pendekatan untuk Pembelajaran Mesin.

Saat ini, otomatisasi adalah aplikasi kecerdasan buatan (AI) dominan yang dengan cepat menembus beberapa industri. Menurut sebuah laporan yang diterbitkan oleh World Economic Forum (WEF) pada tahun 2018, Think Tank Swiss membuat

proyeksi bahwa pada tahun 2022, kecerdasan buatan (AI) diperkirakan akan mengakibatkan perpindahan sekitar 75 juta pekerjaan dalam skala global. . Namun, juga diantisipasi bahwa kemajuan teknologi ini secara bersamaan akan memberikan sekitar 133 juta lapangan kerja baru. Peran karir yang muncul akan membutuhkan individu untuk memiliki kemampuan khusus dalam Ilmu Data, termasuk pemahaman yang kuat tentang Matematika dan Statistik, keakraban dengan algoritma Pembelajaran Mesin, keahlian dalam pemrograman, kompetensi dalam penambangan data dan teknik perselisihan data, pengetahuan tentang prinsip-prinsip rekayasa perangkat lunak, dan kemampuan untuk mengomunikasikan wawasan data secara efektif melalui visualisasi data (Romeroet al., 2020).

Forum Ekonomi Dunia (WEF) telah merumuskan serangkaian agenda untuk tahun 2022, menyoroti pentingnya Kecerdasan Buatan (AI) sebagai kekuatan transformatif dalam membentuk masa depan. Agenda ini menekankan potensi AI untuk memberikan kontribusi besar pada pengembangan lingkungan yang ditandai dengan ketersediaan data yang tepat dan andal. Disiplin telah menghadirkan prospek pekerjaan yang layak bagi individu, sehingga memfasilitasi transformasi positif dalam lintasan profesional mereka. Domain Kecerdasan Buatan telah berkembang secara signifikan, sangat diuntungkan dengan munculnya berbagai subbidang yang berspesialisasi dalam aspek Ilmu Data tertentu. Saat membandingkan Kecerdasan Buatan (AI) dengan Kecerdasan Manusia (HI), terbukti bahwa HI memiliki tingkat penyesuaian yang lebih tinggi dan mungkin tidak cukup layak untuk mengatasi masalah kompleks secara keseluruhan.

Di zaman sekarang, organisasi yang menggunakan teknologi Big Data dan Ilmu Data dikelola oleh para profesional yang mahir seperti Insinyur ML, Ilmuwan Data, Insinyur Data, dan spesialis terkait lainnya yang memiliki pengetahuan dan keahlian komprehensif dalam seluk-beluk Kecerdasan Buatan dan

Pembelajaran Mesin. Nilai yang diperoleh dari Big Data sebagian besar dikaitkan dengan keahlian domain dan keterampilan fleksibel yang dimiliki oleh para profesional ini (Georges Akhras, 2000).

2.8 Penutup Pembahasan Kecerdasan Manusia dan Kecerdasan Buatan

Kecerdasan buatan (AI) adalah instrumen bernilai tinggi yang secara signifikan memengaruhi industri. Diantisipasi bahwa otomatisasi, bersama dengan alur kerja yang cerdas, akan menjadi praktik yang berlaku di semua industri di masa mendatang. Meskipun AI telah mencapai tingkat kemahiran yang tinggi dalam menunjukkan perilaku cerdas, AI tetap tidak mampu mereplikasi proses kognitif yang rumit dari pemikiran manusia. Dalam skenario khusus ini, kecerdasan buatan (AI) telah menunjukkan kekurangan relatif dibandingkan dengan otak manusia saat terlibat dalam persaingan langsung. AI terbatas pada pemecahan masalah dalam batasan antarmuka yang telah dibuat untuk tujuan ini. Otak manusia terus berfungsi sebagai sumber utama untuk menghasilkan berbagai simulasi dan masukan yang sangat penting bagi pengembangan kecerdasan buatan (AI) dan kemajuan pembelajaran mesin.

Mengingat kurangnya pemahaman saat ini di antara para ilmuwan dan peneliti mengenai sifat misterius dari proses berpikir manusia, prospek pengembangan komputer yang mampu meniru kemampuan kognitif seperti manusia sangat tidak mungkin terjadi di masa mendatang. Kesimpulannya, lintasan kecerdasan buatan (AI) sebagian besar akan dibentuk oleh kemampuan manusia. Ini akan ditingkatkan oleh kecerdasan dan kesadaran manusia.

2.9 Potensi Kecerdasan Buatan dalam Meningkatkan Operasi Bisnis di Masa Depan

Perusahaan terkemuka seperti Apple dan Google telah melakukan investasi besar dalam kemajuan Kecerdasan Buatan. Selain industri yang disebutkan di atas, kecerdasan buatan (AI) seringkali tidak dimanfaatkan secara maksimal di berbagai sektor lain seperti manufaktur, pendidikan, ritel, dan kesehatan (Romero*et al.*, 2020).

Banyak perusahaan menghasilkan data dalam jumlah besar setiap hari; namun, pemanfaatan kecerdasan buatan (AI) untuk menganalisis kumpulan data yang luas dan mengekstrak wawasan dari pola dan karakteristik yang mendasari data tersebut jarang terjadi. Penyelidikan utama adalah alasan-alasan untuk menonjolnya masalah ini secara signifikan. Faktor utama yang berkontribusi terhadap masalah ini adalah akses yang terbatas, pemahaman yang tidak memadai, dan kemampuan yang tidak memadai. Kami telah membiasakan diri dengan tantangan utama yang terkait dengan kecerdasan buatan (AI). Sangat penting untuk memperoleh pengetahuan tentang cara mengatasi perbedaan antara tantangan kecerdasan buatan dan profitabilitas bisnis dapat ditangani secara efektif (Vadapalli, 2022).

Salah satu tantangan utama di bidang kecerdasan buatan adalah terbatasnya aksesibilitas sumber daya komputasi canggih dan mahal, yang menjadi penghalang signifikan bagi sebagian besar organisasi. Selain itu, tidak ada akses ke pengetahuan AI yang mahal dan terbatas yang diperlukan untuk pemanfaatan sumber daya tersebut secara efisien.

Pada tahun 2022, sebagian besar perusahaan, khususnya 37%, telah menerapkan layanan kecerdasan buatan (AI) dan terus menggunakannya. Berdasarkan sebuah penelitian, diproyeksikan bahwa bisnis kecerdasan buatan (AI) akan

menghasilkan pendapatan tahunan sebesar $126 miliar pada tahun 2025.

Menurut sebuah laporan oleh Forbes, diproyeksikan bahwa pasar kecerdasan buatan (AI) akan mencapai nilai substansial sebesar $15,7 triliun pada tahun 2030. Selain itu, investasi di sektor ini diperkirakan akan mencapai sekitar $500 miliar pada tahun 2024.

Ada tiga metode potensial di mana kecerdasan buatan (AI) dapat memfasilitasi perluasan perusahaan dan mengatasi tantangan yang terkait dengan AI (Vadapalli, 2022).

2.9.1 Manfaatkan teknologi kecerdasan buatan yang sudah ada sebelumnya.

Di zaman sekarang, bisnis tidak lagi dipaksa untuk memulai proses pelatihan AI dari awal karena banyaknya pekerjaan AI yang dilakukan di platform cloud. Aksesibilitas ke solusi AI berbasis cloud ini sangat kontras dengan model sebelumnya yang menghadapi tantangan di bidang kecerdasan buatan. Perusahaan memiliki potensi untuk memperoleh keuntungan finansial dari tenaga kerja yang sebelumnya telah diselesaikan oleh perusahaan lain. Mereka memiliki kemampuan untuk memodifikasi teknologi kecerdasan buatan yang sudah ada sebelumnya agar sesuai dengan kebutuhan spesifik mereka. Namun, kemampuan mereka untuk menyelesaikan tugas ini terhalang karena tidak adanya antarmuka yang ramah pengguna dan intuitif.

2.9.2 Terus perbarui teknologi AI sesering mungkin

Potensi untuk pembelajaran berkelanjutan dan peningkatan dapat dicapai melalui pemanfaatan kecerdasan buatan (AI). Karakteristik inilah yang membedakannya sebagai jenis teknologi yang luar biasa. Pemilik Tesla menyadari ketersediaan rutin pembaruan perangkat lunak, yang merupakan ciri khas dari pengalaman kepemilikan mereka. Fenomena ini muncul

karena kehadiran beberapa kendaraan Tesla yang beroperasi secara substansial, yang secara aktif mengumpulkan data yang secara teratur digunakan untuk meningkatkan performa setiap mobil. Untuk memfasilitasi pembelajaran dan penyebaran pengetahuan, pemanfaatan kecerdasan buatan (AI) sangat penting dalam beberapa domain aplikasi dan bisnis. Kemajuan teknologi yang terus meningkat dapat secara signifikan memfasilitasi ekspansi bisnis dan secara efektif mengatasi tantangan yang terkait dengan kecerdasan buatan (Vadapalli, 2022).

2.9.3 Manfaatkan kemajuan terbaru dalam teknologi.

Terlepas dari sifat terobosan awal mereka, pendekatan kecerdasan buatan (AI) yang digunakan belakangan ini menjadi tidak efektif. Ada banyak tantangan di bidang kecerdasan buatan yang berkaitan dengan versi lama. Kemajuan berkelanjutan dalam model kecerdasan buatan (AI) dan jaringan saraf mencerminkan proses perolehan keterampilan dan pengetahuan manusia, di mana individu secara konsisten berusaha untuk belajar dan memperluas kemampuan mereka dari waktu ke waktu. Namun demikian, untuk memanfaatkan keuntungan dari kemajuan ini, sangat penting untuk mengembangkan desain prosesor baru dan model pemrograman yang mampu menjalankan algoritme AI dan non-AI. Era mendatang yang ditandai dengan munculnya solusi AI yang lebih praktis dan layak secara ekonomi diantisipasi untuk mencakup beragam aplikasi dan industri. Dalam waktu dekat, diantisipasi bahwa kemajuan akan memungkinkan kita untuk mengatasi keterbatasan yang ada terkait dengan kekuatan, kompleksitas, dan biaya (Vadapalli, 2022).

DAFTAR PUSTAKA

Georges Akhras (2000) 'Smart Materials and Smartsystems for the Future', *Canadian Military Journal*, Autumn 200(1), pp. 25–32.

Molina, M. (1998) 'What is an Intelligent System?', *Fuzzy Control of Industrial Systems*, (December), pp. 1–3. doi: 10.1007/978-1-4757-2813-2_1.

Pavan Vadapalli (2022) *AI vs Human Intelligence: Difference Between AI & Human Intelligence*, *upgrad.com*. Available at: https://www.upgrad.com/blog/ai-vs-human-intelligence/ (Accessed: 18 Agustus 2023).

Robert Sheldon and Ivy Wigmore (2022) *Definition Intelligent System*, *Techtarget.com*. Available at: https://www.techtarget.com/whatis/definition/intelligent-system (Accessed: 18 Agustus 2023).

Romero, M. *et al.* (2020) 'Towards a Characterisation of Smart Systems: A Systematic Literature Review', *Computers in Industry*, 120, pp. 1–23. doi: 10.1016/j.compind.2020.103224.

Tutorialspoint.com (2020) *Artificial Intelligence - Intelligent Systems*, *tutorialspoint.com*. Available at: https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligent_systems.htm (Accessed: 18 Agustus 2023).

Unr.edu (2020) *What are intelligent systems?*, *unr.edu*. Available at: https://www.unr.edu/cse/undergraduates/prospective-students/what-are-intelligent-systems#:~:text=Intelligent systems are technologically advanced,to Amazon's personalized shopping suggestions. (Accessed: 18 Agustus 2023).

Vadapalli, P. (2022) *Top 7 Challenges in Artificial Intelligence in 2023*, *upgrad.com*. Available at: https://www.upgrad.com/blog/top-challenges-in-

artificial-intelligence/ (Accessed: 18 August 2023).
Wikipedia (2023) *Smart system*, *wikipedia.org*. Available at: https://en.wikipedia.org/wiki/Smart_system (Accessed: 18 Agustus 2023).

BAB 3
PENYELESAIAN MASALAH DAN PENCARIAN

Oleh Yusuf Wahyu Setiya Putra

3.1 Pengertian Penyelesaian Masalah

Penyelesaian masalah umumnya dikenal sebagai metode untuk mencapai tujuan yang diinginkan atau menemukan solusi terhadap suatu situasi tertentu. Dalam ilmu komputer, penyelesaian masalah mengacu pada teknik kecerdasan buatan, termasuk berbagai teknik seperti pembentukan algoritma yang efisien, heuristik, dan melakukan analisis akar penyebab untuk menemukan solusi yang diinginkan. penyelesaian masalah dalam Kecerdasan Buatan biasanya mengacu pada penelitian solusi suatu masalah dengan melakukan algoritma logis, memanfaatkan persamaan polinomial dan diferensial, dan mengeksekusinya menggunakan paradigma pemodelan. Ada berbagai solusi untuk satu masalah, yang dicapai dengan heuristik berbeda. Selain itu, beberapa masalah mempunyai solusi unik. Itu semua tergantung pada sifat permasalahan yang ada.

Gambar 3.1 Ilustrasi Penyelesaian Masalah
(Sumber : https://s29840.pcdn.co)

Proses penyelesaian masalah sering kali digunakan untuk mencapai tujuan atau menyelesaikan situasi tertentu. Dalam ilmu komputer, istilah "penyelesaian masalah" mengacu pada metode kecerdasan buatan, yang mungkin mencakup perumusan memastikan kesesuaian, menggunakan algoritma, dan melakukan analisis akar penyebab untuk mengidentifikasi solusi yang masuk akal. penyelesaian masalah dengan kecerdasan buatan (AI) sering kali melibatkan penyelidikan potensi solusi terhadap masalah melalui teknik penalaran, memanfaatkan persamaan polinomial dan diferensial, dan melaksanakannya serta menggunakan kerangka pemodelan. Masalah yang sama memiliki sejumlah solusi, yang semuanya diselesaikan dengan menggunakan algoritma unik. Selain itu, permasalahan tertentu mempunyai solusi orisinal. Itu semua tergantung pada bagaimana situasi tertentu dibingkai.

Kecerdasan buatan digunakan oleh pemrogram di seluruh dunia untuk mengotomatisasi sistem demi manajemen sumber daya dan waktu yang efektif. Permainan dan teka-teki dapat menimbulkan beberapa masalah yang paling sering terjadi dalam kehidupan sehari-hari. Penggunaan algoritma AI dapat mengatasi hal ini secara efektif. Berbagai metode penyelesaian

masalah diterapkan untuk menciptakan solusi bagi berbagai teka-teki kompleks, termasuk tantangan matematika seperti kripto-aritmatika dan kotak ajaib, teka-teki logis termasuk rumus Boolean serta N-Queens, dan permainan yang cukup baik seperti Sudoku dan Catur. Oleh karena itu, berikut ini adalah beberapa masalah paling umum yang telah diatasi oleh kecerdasan buatan:

1. Permainan Catur
2. N-Queen
3. Menara Hanoi
4. Travelling Salesman Problem

3.2 Jenis dan Penyelesaian Masalah dalam Kecerdasan Buatan

Dalam kecerdasan buatan atau AI, dapat dikelompokkan jenis-jenis masalah yang yang terjadi. Pada dasarnya ada tiga jenis masalah dalam kecerdasan buatan yaitu:

1. Ignorable (Dapat Diabaikan)
2. Recoverable (Dapat Dipulihkan)
3. Irrecoverable (Tidak dapat dipulihkan)

Ketiga jenis masalalah pada kecerdasan buatan tersebut akan dijelaskan dalam sub bab berikut ini.

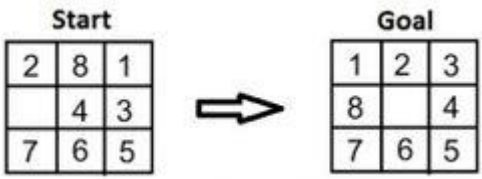
3.2.1 *Ignorable* (Dapat Diabaikan)

Dimana langkah-langkah penyelesaiannya dapat diabaikan (misalnya, Pembuktian Teorema). Misalkan kita mencoba membuktikan teorema matematika. Pertama kita lanjutkan dengan membuktikan lemma yang menurut kita akan berguna. Belakangan kita menyadari bahwa itu tidak berguna. Jadi, apakah kita sedang dalam masalah serius?. Tidak, Kita masih bisa membuktikan teorema tersebut, hanya saja kita perlu

mengabaikan pendekatan pertama dan mulai dengan pendekatan lain untuk membuktikan teorema tersebut.

3.2.2 *Recoverable* (Dapat Dipulihkan)

Di mana langkah-langkah solusi dapat dibatalkan. (misalnya, Soal 8 Teka-Teki). Misalkan kita mempunyai soal 8 teka-teki seperti yang ditunjukkan pada gambar di atas. Saat mencoba menyelesaikan soal 8 teka-teki, secara keliru kita melakukan tindakan yang salah dan menyadari kesalahan tersebut. Sekarang untuk memperbaiki kesalahan kita, kita perlu membatalkan langkah yang salah.



Gambar 3.2 Teka-teki 8 Kotak
(Sumber : https://praveen1302.wordpress.com)

Untuk membatalkan langkah yang salah, mekanisme kontrol untuk pemecah 8 teka-teki harus melacak urutan langkah yang dilakukan, sehingga kita dapat mundur ke keadaan awal dan memulai dengan beberapa langkah yang benar.

3.2.3 *Irrecoverable* (Tidak dapat dipulihkan)

Dimana langkah penyelesaiannya tidak dapat dibatalkan. (misalnya, Catur).

Gambar 3.3 Permainan Catur
(Sumber : https://praveen1302.wordpress.com)

Perhatikan masalah bermain catur. Misalkan program bermain catur membuat gerakan yang salah dan menyadarinya setelah beberapa gerakan. Ia tidak bisa mengabaikan kesalahannya begitu saja. Ia juga tidak dapat membatalkan langkahnya dan memulai permainan lagi. Di sini, sekali kita mengambil tindakan, kita tidak akan pernah bisa pulih dari langkah tersebut. Hanya kami yang bisa berusaha memberikan yang terbaik dari situasi saat ini.

3.2.4 Langkah-langkah Penyelesaian Masalah

Langkah-langkah penyelesaian masalah dalam AI: Masalah AI berhubungan langsung dengan sifat manusia dan aktivitasnya. Jadi kita memerlukan sejumlah langkah terbatas untuk memecahkan masalah yang memudahkan pekerjaan manusia. Ini adalah langkah-langkah berikut yang diperlukan untuk memecahkan masalah:

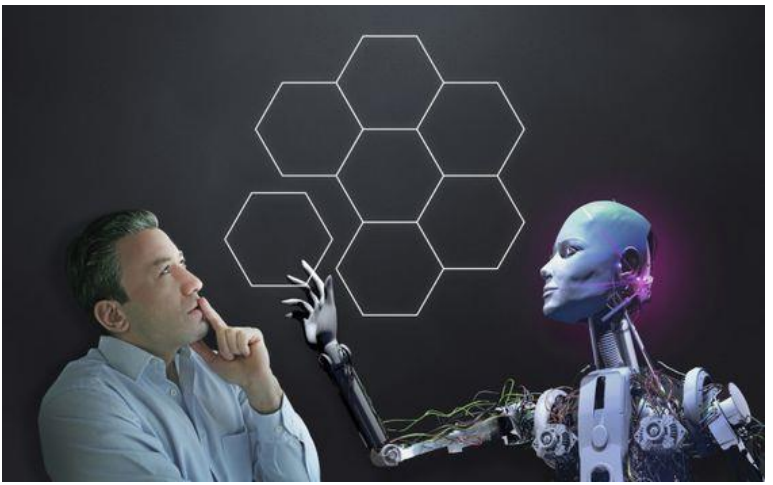
1. Definisi masalah: Spesifikasi input yang terperinci dan solusi sistem yang dapat diterima.
2. Analisis masalah: Analisis masalah secara menyeluruh.
3. Representasi Pengetahuan: mengumpulkan informasi rinci tentang masalah dan menentukan semua teknik yang mungkin.
4. penyelesaian masalah: Pemilihan teknik terbaik.

Kemudian untuk komponen dalam merumuskan masalah terkait akan dijelaskan sebagai berikut:

1. Keadaan Awal: Keadaan ini memerlukan keadaan awal untuk masalah yang memulai agen AI menuju tujuan tertentu. Dalam keadaan ini metode baru juga menginisialisasi penyelesaian domain masalah oleh kelas tertentu.
2. Tindakan: Tahap perumusan masalah ini bekerja dengan fungsi dengan kelas tertentu yang diambil dari keadaan awal dan semua kemungkinan tindakan dilakukan pada tahap ini.
3. Transisi: Tahap perumusan masalah ini mengintegrasikan tindakan aktual yang dilakukan oleh tahap tindakan sebelumnya dan mengumpulkan tahap akhir untuk meneruskannya ke tahap berikutnya.
4. Uji tujuan: Tahap ini menentukan apakah tujuan tertentu tercapai oleh model transisi terintegrasi atau tidak, setiap kali tujuan tercapai, hentikan tindakan dan lanjutkan ke tahap berikutnya untuk menentukan biaya untuk mencapai tujuan.
5. Penetapan biaya jalur (path costing): Komponen numerik penyelesaian masalah ini menetapkan berapa biaya yang akan dikeluarkan untuk mencapai tujuan. Hal ini membutuhkan semua perangkat lunak perangkat keras dan biaya kerja manusia.

3.3 Agen Refleks Kecerdasan Buatan

Agen refleks AI secara langsung memetakan negara ke dalam tindakan. Setiap kali agen ini gagal beroperasi dalam lingkungan di mana keadaan pemetaan terlalu besar dan tidak mudah dilakukan oleh agen, maka masalah yang disebutkan akan hilang dan dikirim ke domain penyelesaian masalah yang memecah masalah besar yang tersimpan ke dalam area penyimpanan yang lebih kecil dan diselesaikan satu per satu. Tindakan terpadu terakhir akan menjadi hasil yang diinginkan.



Gambar 3.4 Ilustrasi Agen Kecerdasan Buatan
(Sumber : https://pin.it/WRzhQKM)

Ada lima agen utama yang digunakan dalam Kecerdasan Buatan berdasarkan kemampuannya dalam memahami kecerdasan. Agen-agen tersebut adalah sebagai berikut:

1. Agen Refleks Sederhana
2. Agen Refleks Berbasis Model
3. Agen Berbasis Tujuan
4. Agen Berbasis Utilitas
5. Agen Pembelajaran

Berdasarkan masalah dan domain kerjanya, berbagai jenis agen penyelesaian masalah didefinisikan dan digunakan pada tingkat atom tanpa keadaan internal apa pun yang terlihat dengan algoritma penyelesaian masalah. Agen penyelesaian masalah bekerja secara tepat dengan mendefinisikan masalah dan beberapa solusi. Jadi kita dapat mengatakan bahwa penyelesaian masalah adalah bagian dari kecerdasan buatan yang mencakup sejumlah teknik seperti tree, B-tree, algoritma heuristik untuk menyelesaikan suatu masalah.

3.3.1 Agen Refleks Sederhana

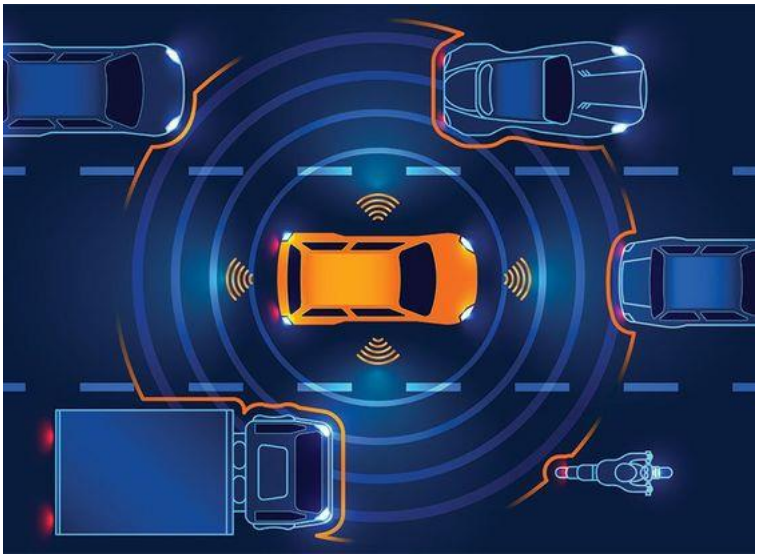
Agen refleks sederhana adalah agen cerdas paling dasar yang ada. Ia melakukan tindakan berdasarkan situasi saat ini. Ketika sesuatu terjadi di lingkungan agen refleks sederhana, agen tersebut dengan cepat memindai basis pengetahuannya untuk mengetahui bagaimana merespons situasi yang ada berdasarkan aturan yang telah ditentukan sebelumnya.

Agen refleks sederhana bekerja melalui sensor dan bertindak terhadap lingkungan melalui aktuator. Agen refleks sederhana memilih tindakan berdasarkan persepsi agen saat ini atau dunia dan bukan berdasarkan persepsi masa lalu. Mereka bekerja ketika lingkungan dapat diamati sepenuhnya.

Sebagai contoh, jika Anda meletakkan tangan Anda di atas kompor yang panas, suatu refleks menyebabkan Anda segera melepaskan tangan Anda sebelum berkata "Hei, ini panas!" pesan bahkan sampai ke otak Anda. Refleks pelindung lainnya adalah berkedip ketika ada sesuatu yang terbang ke arah mata Anda atau mengangkat tangan jika ada bola yang dilempar ke arah Anda.

3.3.2 Agen Refleks Berbasis Model

Agen refleks berbasis model adalah agen yang menggunakan memori internal dan riwayat persepsi untuk membuat model lingkungan di mana ia beroperasi dan membuat keputusan berdasarkan model tersebut. Agen refleks berbasis model dibuat untuk menangani aksesibilitas parsial, mereka melakukan ini dengan mencatat bagian dunia yang dapat dilihatnya sekarang. Hal ini dilakukan dengan menjaga keadaan internal yang bergantung pada apa yang telah dilihat sebelumnya sehingga menyimpan informasi mengenai aspek-aspek keadaan saat ini yang tidak teramati.



Gambar 3.5 Ilustrasi Mobil *self-driving*
(Sumber : https://pin.it/5A3Dhzi)

Mobil self-driving adalah contoh bagus dari agen refleks berbasis model. Mobil tersebut dilengkapi dengan sensor yang mendeteksi adanya rintangan, seperti lampu rem mobil di depannya atau pejalan kaki yang berjalan di trotoar. Saat mobil melaju, sensor-sensor ini memasukkan persepsi ke dalam memori mobil dan model internal lingkungannya.

3.3.3 Agen Berbasis Tujuan

Agen AI berbasis tujuan dapat didefinisikan sebagai program cerdas yang dapat mengambil keputusan berdasarkan pengalaman sebelumnya, pengetahuan, masukan pengguna, dan tujuan yang diinginkan. Agen berbasis tujuan memperluas kemampuan agen berbasis model dengan memiliki informasi "tujuan". Mereka memilih suatu tindakan, sehingga mereka dapat mencapai tujuan. Agen-agen ini mungkin harus mempertimbangkan serangkaian tindakan yang mungkin

dilakukan sebelum memutuskan apakah tujuannya tercapai atau tidak.

Mobil tanpa pengemudi Waymo dari Google adalah contoh yang baik dari agen berbasis tujuan ketika mereka diprogram dengan mempertimbangkan tujuan akhir, atau sasaran. Mobil kemudian akan ''berpikir'' dan mengambil keputusan yang tepat guna mengantarkan penumpangnya ke tempat tujuan.

3.3.4 Agen Berbasis Utilitas

Agen berbasis utilitas adalah agen yang bertindak tidak hanya berdasarkan pada apa tujuannya, namun juga cara terbaik untuk mencapai tujuan tersebut. Singkatnya, kegunaan (atau kegunaan) agen itulah yang membedakan dirinya dari agen lainnya. Agen yang dikembangkan dengan kegunaan akhirnya sebagai bahan penyusun disebut agen berbasis utilitas. Ketika ada beberapa alternatif yang mungkin, maka untuk memutuskan mana yang terbaik, agen berbasis utilitas digunakan. Mereka memilih tindakan berdasarkan preferensi (utilitas) untuk setiap negara bagian.

Sebagai contoh agen berbasis utilitas adalah dapat berupa termostat rumah yang mengetahui cara mulai memanaskan atau mendinginkan rumah Anda berdasarkan pencapaian suhu tertentu.

3.3.5 Agen Pembelajaran

Agen pembelajaran dalam AI adalah jenis agen yang dapat belajar dari pengalaman masa lalunya atau memiliki kemampuan belajar. Ia mulai bertindak dengan pengetahuan dasar kemudian mampu bertindak dan beradaptasi secara otomatis melalui pembelajaran. Agen ini menggunakan elemen

pembelajaran tambahan untuk secara bertahap meningkatkan dan menjadi lebih berpengetahuan dari waktu ke waktu tentang suatu lingkungan. Elemen pembelajaran menggunakan umpan balik untuk memutuskan bagaimana elemen kinerja harus diubah secara bertahap untuk menunjukkan peningkatan.

Berikut adalah unsur-unsur dari Agen Pembelajaran:

1. Basis Pengetahuan : Basis Pengetahuan adalah bagian mendasar dari setiap agen yang bekerja pada Kecerdasan Buatan dan oleh karena itu harus ada di dalam setiap sistem tersebut. Semua informasi, tindakan masa lalu, dan dampaknya, serta segala jenis data yang digunakan agen ada di dalam Basis Pengetahuan ini.
2. Sensor : Untuk Belajar dari lingkungan dan lingkungan sekitar, agen harus mampu mengamati dan mempersepsikan informasi dari lingkungan. Ini dilakukan melalui sensor. Sensor merupakan bagian yang sangat penting dari setiap agen berbasis AI dalam meningkatkan kinerjanya dan membantu agen tersebut bertindak seperti manusia.
3. Prosesor : Prosesor yang menjadi otak agen ini agak mirip dengan prosesor komputer. Perbedaannya adalah prosesor ini jauh lebih canggih daripada prosesor biasa karena dapat memproses situasi kompleks tertentu seperti ketidakpastian dalam data, informasi yang tidak lengkap tentang tugas apa pun, dll. Hal ini membuatnya jauh lebih efisien untuk digunakan dalam situasi kehidupan nyata. Selain itu, pemroses dalam agen itulah yang menjadikan suatu agen bisa secara manusiawi. Semakin baik kinerja yang kami inginkan pada agen kami, semakin baik pula prosesor yang dirancang untuk itu.
4. Satuan Logika : Unit Logika bertanggung jawab untuk menggambar dan menerapkan logika dalam setiap keputusan. Unit ini mulai digunakan setiap kali agen mencoba menyimpulkan hasil dari berbagai angka dan pernyataan yang tersedia.
5. Memori : Semua informasi yang ada di dalam Basis Pengetahuan harus disimpan di suatu tempat. Jadi, Memori

juga merupakan bagian penting bagi setiap agen untuk menyimpan semua data dan fakta, serta hasil dan kesimpulan yang dihasilkan oleh prosesor dan unit logika.

6. Aktuator : Agen harus melakukan tindakan tertentu atau menghasilkan beberapa hasil setelah semua analisis dan pemrosesan. Hal ini dilakukan oleh aktuator. Dalam robotika, aktuator ini adalah bagian bergerak yang melakukan tugas fisik tertentu, sedangkan dalam sistem lain, aktuator ini mirip dengan unit keluaran yang menghasilkan data yang diproses sebagai keluaran.

Selain itu, ada juga banyak elemen lain dari Agen Pembelajaran seperti Perangkat Keras, Desain sistem, sistem tenaga, sistem pendingin, dll. Sebagai contoh, manusia adalah contoh agen pembelajaran. Misalnya, manusia bisa belajar mengendarai sepeda, padahal saat lahir belum ada manusia yang memiliki keterampilan tersebut.

3.4 Pencarian dalam Kecerdasan Buatan

Pencarian adalah salah satu metode utama untuk memecahkan masalah apa pun di AI. Agen rasional atau agen penyelesaian masalah menggunakan algoritma pencarian ini untuk menemukan solusi optimal. Agen penyelesaian masalah ini seringkali berbasis tujuan dan memanfaatkan representasi atom. Selain itu, algoritma pencarian ini memiliki sifat kelengkapan, optimalitas, kompleksitas waktu, dan kompleksitas ruang berdasarkan kualitas solusi yang diberikan.

Ada dua jenis utama algoritma pencarian dalam kecerdasan buatan, berikut adalah jenisnya dan akan dijelaskan pada sub-bab ini:

1. Pencarian Informasi
2. Pencarian Tanpa Informasi

3.4.1 Pencarian Informasi

Algoritma ini menggunakan pengetahuan domain dasar dan memahami informasi yang tersedia mengenai masalah tertentu sebagai pedoman untuk solusi optimal. Solusi yang diberikan oleh algoritma informed search lebih efisien dibandingkan dengan algoritma uninformed search.

Jenis Algoritma Pencarian yang diinformasikan ada dua jenis algoritma pencarian informasi yaitu :

1. Greedy Search : adalah algoritma pencarian AI yang mencoba menemukan jalur paling menjanjikan dari titik awal tertentu menuju tujuan. Algoritma ini bekerja dengan mengevaluasi biaya setiap jalur yang mungkin dan kemudian memperluas jalur dengan biaya terendah. Greedy adalah paradigma algoritmik yang membangun solusi sedikit demi sedikit, selalu memilih solusi berikutnya yang menawarkan manfaat paling jelas dan langsung. Jadi permasalahan dimana memilih optimal lokal juga mengarah pada solusi global adalah yang paling cocok untuk Greedy. Misalnya pertimbangkan Masalah Knapsack Pecahan.
2. A* Search : Algoritma Pencarian A* adalah algoritma pencarian sederhana dan efisien yang dapat digunakan untuk menemukan jalur optimal antara dua node dalam sebuah grafik. Ini akan digunakan untuk menemukan jalur terpendek. Ini merupakan perpanjangan dari algoritma jalur terpendek Dijkstra (Dijkstra's Algorithm). Algoritma ini menggunakan fungsi heuristik untuk memandu pencarian menuju solusi optimal. Pencarian A*, salah satu algoritma pencarian heuristik paling populer, menggunakan biaya sebenarnya untuk mencapai sebuah node dan perkiraan biaya untuk mencapai tujuan dari node tersebut.

3.4.2 Pencarian Tanpa Informasi

Algoritme ini tidak memiliki hak istimewa untuk menggunakan pengetahuan domain dasar, seperti kedekatan tujuan yang diinginkan. Ini berisi informasi mengenai melintasi pohon dan mengidentifikasi daun dan simpul tujuan. Pencarian tanpa informasi juga dikenal dengan nama pencarian buta karena saat melintasi, tidak ada informasi spesifik tentang keadaan awal dan pengujian untuk tujuan tersebut. Pencarian ini melewati setiap node hingga mencapai tujuan yang diinginkan.

Jenis Algoritma Pencarian Tanpa Informasi. Ada lima jenis utama algoritma pencarian tanpa informasi berikut ini:

1. Pencarian Luas-Pertama : Pencarian Breadth-first (BFS) adalah algoritma untuk mencari struktur data pohon untuk sebuah node yang memenuhi properti tertentu. Ini dimulai dari akar pohon dan mengeksplorasi semua node pada kedalaman saat ini sebelum berpindah ke node pada tingkat kedalaman berikutnya.
2. Pencarian Kedalaman Pertama : Pencarian mendalam pertama (DFS) adalah algoritma untuk melintasi atau mencari struktur data pohon atau grafik. Algoritme dimulai dari simpul akar (memilih beberapa simpul sembarang sebagai simpul akar dalam kasus grafik) dan mengeksplorasi sejauh mungkin sepanjang setiap cabang sebelum melakukan backtracking.
3. Pencarian Biaya Seragam : Pencarian biaya seragam (uniform-cost search) adalah algoritma pencarian tanpa informasi yang menggunakan biaya kumulatif terendah untuk menemukan jalur dari sumber ke tujuan. Node diperluas, mulai dari root, sesuai dengan biaya kumulatif minimum. Pencarian biaya seragam kemudian diimplementasikan menggunakan Antrean Prioritas.
4. Pencarian Pertama Kedalaman Pendalaman Iteratif : Dalam ilmu komputer, pencarian pendalaman berulang atau lebih khusus lagi pencarian kedalaman-pertama pendalaman berulang (IDS atau IDDFS) adalah strategi pencarian

ruang/grafik keadaan di mana versi pencarian kedalaman-pertama yang dibatasi kedalaman dijalankan berulang kali dengan batas kedalaman yang meningkat hingga tujuan ditemukan.

5. Pencarian Dua Arah : Pencarian dua arah adalah algoritma pencarian graf yang mencari jalur terkecil dari titik sumber ke titik tujuan. Ini menjalankan dua pencarian simultan – Meneruskan pencarian dari sumber/simpul awal menuju simpul tujuan. Pencarian mundur dari titik tujuan/target menuju titik sumber.

3.5 Komputasi Evolusioner

Komputasi Evolusioner, metode penyelesaian masalah ini memanfaatkan konsep evolusi yang terkenal. Teori evolusi bekerja berdasarkan prinsip survival of the fittest. Dinyatakan bahwa organisme yang dapat mengatasi lingkungannya dengan baik dalam lingkungan yang menantang atau berubah dan bereproduksi, generasi masa depannya secara bertahap mewarisi mekanisme penanggulangannya, menghasilkan keragaman pada organisme anak baru. Dengan cara ini, organisme baru bukan hanya salinan dari organisme lama tetapi memiliki campuran karakteristik yang sejalan dengan lingkungan yang keras tersebut. Manusia adalah contoh paling menonjol dari proses evolusi yang telah berevolusi dan terdiversifikasi karena akumulasi mutasi yang menguntungkan selama beberapa generasi.

Gambar 3.6 Ilustrasi Komputasi Evolusioner
(Sumber : https://pin.it/6cXb4iW)

Dalam AI, konsep evolusi mengacu pada teknik trial-error yaitu, ubah solusi lama secara acak untuk menghasilkan solusi baru. Apakah ini memberikan hasil yang lebih baik? Jika ya, simpan dan buang sisa larutannya. Jika tidak, buang saja. Lanjut ke poin 1.

Model komputasi yang menggunakan algoritma evolusi menerapkan proses evolusi untuk memecahkan masalah yang kompleks. Proses evolusi ini diilhami oleh teori evolusi biologis. Algoritme yang berkembang menggunakan prinsip-prinsip seperti pewarisan dari generasi sukses sebelumnya, dan seleksi alam di mana solusi terbaik mewariskan sifat-sifatnya ke generasi berikutnya.

Cara kerja komputasi evolusioner adalah kumpulan awal solusi yang mungkin dibuat dengan dimulainya komputasi evolusioner. Solusi-solusi yang pernah dicoba disempurnakan ketika solusi-solusi yang lebih lemah dihilangkan secara stokastik dan perubahan-perubahan kecil secara acak diperkenalkan ke generasi-generasi berikutnya. Seiring berlalunya generasi, solusinya menjadi semakin disempurnakan. Pada akhirnya solusi yang dihasilkan oleh komputasi evolusioner dapat dioptimalkan secara ketat, meskipun pada awalnya pendekatannya tidak dipahami.

3.6 Algoritma Genetika

Teori evolusi adalah dasar dari algoritma genetika. Algoritma ini menggunakan metode pencarian acak langsung. Pengembang menghitung fungsi kecocokan untuk menyilangkan dua opsi yang paling cocok untuk menciptakan anak yang disukai. Pengembang mengumpulkan data populasi dan kemudian mengevaluasi setiap anggota untuk menghitung kebugaran setiap orang. Diperkirakan berdasarkan seberapa cocok setiap anggota dengan kebutuhan yang diinginkan. Kemudian pihak pengembang menggunakan berbagai cara seleksi untuk mempertahankan anggota terbaiknya. Beberapa caranya adalah sebagai berikut:

1. Seleksi Peringkat
2. Seleksi Turnamen
3. Seleksi Mantap
4. Pemilihan Roda Roulette (Seleksi Proporsional Kebugaran)
5. Elitisme

Meskipun algoritma genetika tidak selalu bekerja dengan baik, algoritma ini tidak mudah rusak, dan masukannya sedikit berubah. Para pengembang biasanya menggunakan algoritma genetika untuk menghasilkan solusi tingkat tinggi untuk masalah optimasi dan pencarian dengan mengandalkan operasi yang terinspirasi oleh bio seperti mutasi, persilangan, dan seleksi.

Gambar 3.7 Ilustrasi Algoritma Genetika
(Sumber : https://pin.it/3pV3OyK)

Teknik penyelesaian masalah membantu meningkatkan kinerja program. Algoritme pencarian berbasis AI memberikan hasil dengan presisi tinggi dan akurasi maksimum. Algoritma ini lebih cepat dalam eksekusi dibandingkan algoritma lainnya dan menawarkan beberapa metode pencarian tergantung pada masalah yang dihadapi. Penerapan heuristik memungkinkan perancangan algoritma yang secara konseptual lebih mudah dengan biaya komputasi yang lebih murah dibandingkan dengan menggunakan algoritma yang optimal. Komputasi evolusioner juga membantu dalam masalah optimasi dan pencarian. Secara keseluruhan, teknik-teknik ini menjadi dasar untuk memecahkan masalah tingkat tinggi dalam AI seperti algoritma catur, dan masalah mendaki bukit.

3.7 Penutup

Yang terakhir pada bab ini adalah pertanyaan, mengapa penyelesaian masalah penting dalam AI? Jawabannya adalah menguraikan segala jenis masalah memerlukan tindakan terorganisir khusus yang harus diperhatikan. Hal yang sama adalah masalah penyelesaian masalah dengan AI. Berikut rinciannya:

1. Sasaran : Dalam fase ini, segera setelah krisis muncul, agen AI menetapkan tujuan atau tanda. Hal ini memerlukan agen untuk memeriksa dan mengklarifikasi masalah secara menyeluruh. Hal ini merupakan tindakan yang vital, karena jika tujuan dari permasalahan tersebut tidak dikembangkan dengan baik maka segala tindakan yang dilakukan untuk mencapai tujuan tersebut akan sia-sia.
2. Deskripsi Masalah : Ini adalah salah satu tahap utama penyelesaian masalah. Setiap kali terjadi masalah, maka agen memilih tindakan apa yang harus dilakukan agar dapat berjalan menuju tujuan yang dikembangkan. Hal ini dicapai dalam tindakan selanjutnya:
3. Menggambarkan Negara : Wilayah negara dapat digambarkan sebagai kumpulan semua kondisi akurat yang dapat digunakan oleh seorang agen ketika menemukan kunci krisis.
4. Menentukan Negara Bagian Utama : Agar agen dapat mulai memecahkan masalah, agen harus memulai dari suatu negara bagian. Keadaan utama tempat agen mulai bekerja diarahkan ke keadaan utama.
5. Kumpulkan Detail : Sekarang agen mengumpulkan data dan menggunakan data yang dibutuhkannya untuk memperbaiki masalah. Rincian ini akan dikumpulkan dengan kejadian satu kali serta potongan pengetahuan saat ini.
6. Merancang Perubahan : Beberapa permasalahan berukuran terlalu kecil sehingga permasalahan ini dapat diuraikan secara efisien. Namun seringkali permasalahannya adalah perencanaan dan implementasi yang baik diperlukan. Oleh karena itu, hal ini memerlukan struktur data yang tepat dan proses pengelolaan jauh sebelumnya.

Dapat disimpulkan bahwa salah satu tahap dasar penyelesaian masalah yang menentukan ukuran apa yang harus diambil untuk memenuhi tujuan yang dikembangkan. Perumusan masalah merupakan tahapan dalam uraian masalah yang digunakan untuk memahami dan memilih suatu jalannya kegiatan yang harus dievaluasi untuk mencapai suatu tujuan. Jika ada lebih

dari satu metode yang dapat digunakan agen untuk mencapai tujuannya, maka hal tersebut akan menimbulkan kerumitan dalam mencapai tujuan karena akan ada terlalu banyak tindakan dan tindakan yang dapat diambil oleh entitas AI untuk mencapai tujuan sehingga menyebabkan kekacauan dan kekacauan. penurunan efisiensi yang sangat besar. Perumusan masalah dapat diselesaikan melalui banyak tahapan seperti deskripsi kondisi awal agen, memilih langkah-langkah yang mungkin dapat ditanggung oleh agen, dan merancang standar transisi untuk menentukan upaya agen.

DAFTAR PUSTAKA

Gebser, M, Kaminski, R, Kaufmann, B, & Schaub, T 2022, *Answer set solving in practice*, Springer Publishing, New York.

Natarajan, M, & Kolobov, A 2022, *Planning with Markov decision processes: An AI perspective*, Springer Publishing, New York.

Finlay, J 2020, *An introduction to artificial intelligence*, CRC Press, London.

BAB 4
PENCARIAN INFORMAD (TERPADU DAN HEURISTIK) DAN IMPLEMENTASINYA

Oleh Hermila A.

4.1 Pendahuluan

Metode pencarian informatif (Informative search), juga dikenal sebagai pencarian informad, menggunakan informasi yang diketahui atau diperoleh sebelumnya untuk membantu proses pencarian menuju tujuan atau solusi yang diinginkan. Tujuan pencarian informad adalah untuk mempersempit ruang pencarian dan mencapai solusi dengan lebih efisien dengan memanfaatkan pengetahuan yang ada atau heuristik yang relevan. Berbeda dengan pencarian non-informad, yang hanya bergantung pada informasi langsung dari masalah tanpa memanfaatkan pengetahuan tambahan, pencarian informad biasanya menggunakan heuristik atau informasi yang dapat memberikan petunjuk tentang sejauh mana suatu tindakan atau langkah mendekatkan kita ke solusi terbaik.

Fungsi utama dari pencarian informad (informative search) dalam kecerdasan buatan adalah untuk membimbing proses pencarian menuju tujuan atau solusi yang diinginkan dengan lebih efisien. Pencarian informad memanfaatkan informasi tambahan atau pengetahuan yang tersedia untuk membantu dalam mengurangi ruang pencarian yang harus dieksplorasi dan mempercepat pencapaian hasil yang diinginkan.

Pencarian informad berguna dalam situasi di mana ada pengetahuan tambahan yang dapat membantu dalam

membimbing pencarian menuju solusi. Namun, harus diingat bahwa heuristik yang digunakan harus relevan dengan masalah yang dihadapi dan tidak selalu menjamin solusi optimal. Keberhasilan pencarian informad bergantung pada kualitas heuristik yang digunakan dan bagaimana informasi tersebut digunakan untuk membimbing pencarian. Model pencarian informad yang akan dibahas disini adalah pencarian terpadu dan pencarian heuristik

4.2 Pencarian Terpadu

Dalam kecerdasan buatan, pencarian terpadu adalah proses penggabungan dan koordinasi berbagai teknik dan sumber daya untuk melakukan pencarian informasi yang lebih efisien dan efektif. Tujuan pencarian terpadu adalah untuk mengatasi kompleksitas tugas pencarian yang melibatkan berbagai jenis data, sumber informasi, dan konteks.

Beberapa bagian dan teknik yang biasa digunakan dalam penelitian kecerdasan buatan terpadu adalah:

4.2.1. Pemrosesan Bahasa Alami (NLP)

Natural Language Processing atau NLP adalah cabang ilmu komputer dan kecerdasan buatan yang berkaitan dengan interaksi antara komputer dan bahasa manusia yang alami (Amazon, 2023). Tujuan utama dari NLP adalah untuk mengizinkan komputer memahami, memanipulasi, dan menghasilkan bahasa manusia dengan cara yang bermakna. Ini mencakup berbagai tugas yang melibatkan pemahaman dan penggunaan bahasa, seperti analisis teks, terjemahan mesin, pengenalan suara, generasi teks, serta pemahaman dan penghasilan bahasa yang lebih maju. Teknik NLP memungkinkan mesin untuk memahami dan menghasilkan teks manusia. Dalam pencarian terpadu, ini dapat digunakan untuk memahami

pertanyaan atau permintaan pengguna dalam bahasa alami dan mencari informasi yang relevan dari berbagai sumber.

Proses Natural Language Processing (NLP) terdiri dari beberapa tahap, dan ini adalah beberapa tahap umum implementasi yang terlibat dalam NLP:

a. Tokenisasi

Tahap pertama adalah memecah teks menjadi unit-unit yang lebih kecil, yang disebut "token." Token ini bisa berupa kata-kata, frasa, atau bahkan karakter tergantung pada tingkat tokenisasi yang diinginkan.

b. Pembersihan Teks (Text Cleaning)

Tahap ini melibatkan menghapus karakter khusus, tanda baca, dan elemen yang tidak relevan atau mengganggu dari teks. Ini membantu mengurangi noise dalam teks dan membuatnya lebih mudah untuk diproses.

c. Normalisasi

Normalisasi melibatkan mengubah teks menjadi bentuk standar, misalnya mengubah semua huruf menjadi huruf kecil, mengganti singkatan dengan bentuk lengkap, atau menghapus karakter ganda.

d. Stopword Removal

Kata-kata umum yang tidak memberikan banyak informasi, seperti "dan," "atau," atau "di," sering dihapus karena tidak banyak memberikan makna penting dalam analisis teks.

e. Pemenggalan Kata (Stemming atau Lemmatization)

Tahapan ini adalah proses mengubah kata-kata menjadi bentuk dasar (lemmatization) atau menghilangkan akhiran (stemming), sehingga kata-kata dengan akar yang sama dapat dianggap sama meskipun dalam bentuk yang berbeda.

f. Pemahaman Teks (Text Understanding)

Bagian ini adalah tahap yang lebih kompleks di mana komputer mencoba untuk memahami struktur dan makna teks. Ini melibatkan mengidentifikasi entitas seperti nama orang atau tempat, serta memahami hubungan antara kata-kata dalam kalimat.

g. Analisis Sentimen

Jika diperlukan, tahap ini melibatkan menganalisis teks untuk menentukan sentimen atau emosi yang terkandung dalamnya, apakah itu positif, negatif, atau netral.

h. Ekstraksi Informasi

Dalam tahap ini, informasi khusus yang diinginkan diekstraksi dari teks. Misalnya, dalam analisis berita, judul, tanggal, dan isi berita dapat diekstraksi sebagai entitas yang berbeda.

i. Evaluasi dan Pemahaman

Setelah data diolah, tahap evaluasi melibatkan menganalisis hasil dan memastikan bahwa tujuan dari analisis NLP tercapai.

j. Visualisasi dan Pelaporan

Hasil analisis NLP sering disajikan dalam bentuk visualisasi atau laporan untuk mempermudah pemahaman dan pengambilan keputusan.

Contoh Penerapan NLP

a. Penerjemah Bahasa Otomatis

Penerjemahan bahasa otomatis ini memanfaatkan pemahaman tentang tata bahasa, struktur kalimat, serta kosa kata dalam berbagai bahasa. Google Translate adalah salah satu contoh terkenal dari penerjemahan bahasa otomatis yang menggunakan teknologi NLP. Sistem ini mampu menerjemahkan teks dari satu bahasa ke bahasa lain dengan cepat dan akurat

b. Asisten Virtual

Asisten virtual menggunakan teknologi NLP untuk memahami bahasa manusia, menjawab pertanyaan, melaksanakan perintah,

dan memberikan informasi yang diperlukan. Hal ini memungkinkan interaksi lebih alami antara manusia dan mesin. Contoh Penerapannya:

1. Siri (Apple)

Siri adalah asisten virtual yang dikembangkan oleh Apple. Ini digunakan di perangkat-perangkat Apple seperti iPhone, iPad, dan Mac. Contoh penggunaan:

Pertanyaan: "Cuaca hari ini di Jakarta?"

Respon: Siri akan memberikan informasi cuaca saat ini di Jakarta.

2. Google Assistan (Google)

Google Assistant adalah asisten virtual dari Google, yang hadir di berbagai perangkat seperti ponsel cerdas, speaker pintar, dan perangkat lainnya. Contoh penggunaan:

Pertanyaan: "Berikan resep kue cokelat."

Respon: Google Assistant akan memberikan resep kue cokelat yang relevan.

3. Amazon Alexa (Amazon)

Alexa adalah asisten virtual yang diintegrasikan dengan perangkat-perangkat Amazon seperti Echo. Contoh penggunaan:

Perintah: "Alexa, putar musik jazz."

Respon: Alexa akan memutar musik jazz sesuai permintaan.

4. Cortana (Microsoft)

Cortana adalah asisten virtual yang dikembangkan oleh Microsoft, awalnya hadir di Windows 10 dan perangkat Microsoft lainnya. Contoh penggunaan:

Pertanyaan: "Apakah ada pertemuan di jadwal hari ini?"

Respon: Cortana akan menampilkan informasi mengenai jadwal pertemuan pengguna.

c. Analisis Sentimen

NLP digunakan untuk menganalisis sentimen atau perasaan yang terkandung dalam teks. Ini bisa digunakan untuk melacak opini publik tentang suatu topik, produk, atau merek. Berikut contoh penerapannya adalah :

1. Analisis sentimen Ulasan Produk

Dalam e-commerce atau industri yang menjual produk, NLP digunakan untuk menganalisis sentimen dari ulasan pelanggan tentang produk tertentu. Contoh:

Ulasan: "Saya sangat senang dengan kualitas produk ini. Sangat bagus dan tahan lama!"

Analisis Sentimen: NLP akan menganalisis ulasan ini sebagai sentimen positif karena mencerminkan kepuasan pengguna terhadap produk.

2. Monitoring Media Sosial

Perusahaan atau merek menggunakan analisis sentimen untuk melacak opini publik tentang mereka di media sosial. Contoh:

Tweet: "Pengalaman buruk dengan layanan pelanggan perusahaan XYZ. Tidak pernah akan menggunakan lagi."

Analisis Sentimen: NLP akan menganalisis tweet ini sebagai sentimen negatif, mengindikasikan kekecewaan pengguna terhadap layanan pelanggan.

3. Analisis Feedback Layanan

Bisnis dapat menganalisis sentimen dari umpan balik pelanggan untuk memahami bagaimana pelanggan merasa tentang layanan mereka. Contoh:

Umpan Balik: "Sangat mudah berkomunikasi dengan tim dukungan. Mereka sangat membantu!"

Analisis Sentimen: NLP akan menganalisis umpan balik ini sebagai sentimen positif karena mencerminkan pengalaman positif dengan layanan pelanggan.

4. Analisis Berita dan Artikel

Media dan organisasi berita dapat menggunakan NLP untuk menganalisis sentimen dari artikel berita dan berbagai opini. Contoh:

Judul Berita: "Pasar Saham Menguat Berkat Pertumbuhan Ekonomi Yang Stabil."

Analisis Sentimen: NLP akan menganalisis judul ini sebagai sentimen positif karena mencerminkan kondisi ekonomi yang baik.

5. Pencarian Web (Search Engines)

Dalam pencarian web NLP memainkan peran penting dalam memahami bahasa alami manusia dan menghubungkan pertanyaan atau permintaan pengguna dengan informasi yang relevan dalam basis data atau internet secara keseluruhan. Contoh mesin pencari Google menggunakan teknologi NLP untuk memahami pertanyaan atau kata kunci yang dimasukkan oleh pengguna dan menghasilkan hasil pencarian yang relevan. Contoh:

Pencarian: "Restoran sushi terbaik di dekat saya"

Hasil: Mesin pencari akan menampilkan daftar restoran sushi terbaik di dekat lokasi pengguna

e. Chatbot

Dalam pemanfaatan Chatbot, NLP memungkinkan chatbot untuk memahami pertanyaan dan perintah pengguna dalam bahasa alami, serta memberikan respons yang relevan dan berguna.

f. Text To Speech (Sintesis Ucapan)

Sintesis Ucapan dalam Teknologi NLP memainkan peran penting dalam menghasilkan suara yang alami dan dapat dipahami oleh manusia berdasarkan teks yang diberikan. Contoh Audiobook dan Pembaca Teks.

Aplikasi dan perangkat lunak menggunakan sintesis ucapan berbasis NLP untuk membaca teks, seperti dalam audiobook atau bacaan teks di perangkat elektronik. Contoh:

Teks: "Dalam bab ini, kami akan menjelajahi konsep dasar kecerdasan buatan..."

Sintesis Ucapan: Sistem NLP akan mengubah teks menjadi suara yang dapat didengarkan, sehingga pengguna bisa mendengarkan teks tersebut.

4.2.2. Penggalian Data (Teknik Data Mining)

Data mining, juga disebut Knowledge Discovery in Database (KDD), adalah proses pengumpulan dan pengolahan data dengan tujuan mengekstrak informasi penting dari data. Proses ini dapat dilakukan dengan menggunakan perangkat lunak yang memanfaatkan perhitungan matematika, statistika, atau AI (Setiawan, 2021).

Teknik penggalian data (data mining) membantu dalam menemukan pola, hubungan, dan informasi yang tersembunyi dalam data besar. Ini dapat membantu dalam mengidentifikasi tren atau kesamaan dalam sumber-sumber informasi yang berbeda.

Metode Data Mining

Terdapat beberapa metode data mining yang digunakan untuk menganalisis data dan menghasilkan informasi yang berharga. Berikut adalah beberapa metode data mining yang umum digunakan:

1. Klasifikasi (Classification)

Metode ini digunakan untuk mengelompokkan data ke dalam kategori atau kelas berdasarkan atribut yang diberikan. Contohnya adalah pengklasifikasian email menjadi "spam" atau "non-spam".

2. Regresi (Regression)

Regresi digunakan untuk memahami hubungan antara variabel input dan variabel output kontinu. Ini membantu dalam meramalkan atau memprediksi nilai-nilai berdasarkan pola data historis.

3. Klastering (Clustering)

Metode ini mengelompokkan data menjadi kelompok atau klaster yang memiliki kesamaan tertentu. Klastering membantu dalam mengidentifikasi pola alami dalam data tanpa memerlukan label kelas sebelumnya.

4. Asosiasi (Association)

Asosiasi mencari hubungan antara item dalam kumpulan data. Ini sering digunakan dalam analisis keranjang belanja, di mana tujuannya adalah untuk menemukan keterkaitan antara item yang sering dibeli bersamaan.

5. Anomali (Anomaly Detection):

Metode ini bertujuan untuk mengidentifikasi data yang tidak biasa atau kelainan dalam suatu kumpulan data. Ini sering digunakan dalam deteksi fraud, keamanan jaringan, atau pemeliharaan prediktif.

6. Penggalian Pola Berurutan (Sequential Pattern Mining):

Metode ini menemukan pola-pola urutan dalam data, seperti urutan transaksi pembelian yang sering terjadi.

7. Analisis Deret Waktu (Time Series Analysis):

Digunakan untuk menganalisis data yang dikumpulkan dalam interval waktu tertentu. Ini membantu dalam memahami tren, musiman, dan pola lainnya dalam data waktu.

8. Analisis Tekstual (Text Mining):

Fokus pada analisis teks dan data berstruktur atau tidak berstruktur lainnya. Ini melibatkan ekstraksi informasi, kategorisasi teks, analisis sentimen, dan lainnya.

Tahapan Data Mining

Berikut ini adalah beberapa tahapan umum dalam proses data mining yang dilakukan untuk menggali informasi berharga dari kumpulan data yang sangat besar:

1. Pemahaman Bisnis (Business Understanding):

Tahapan Ini adalah langkah pertama di mana Anda berkolaborasi dengan pemangku kepentingan bisnis untuk memahami tujuan analisis data, pertanyaan yang perlu dijawab, dan bagaimana hasil analisis akan digunakan untuk mengambil keputusan bisnis.

2. Pemahaman Data (Data Understanding):

Tahapan ini mengumpulkan dan memahami data yang akan digunakan dalam analisis. Ini melibatkan pemahaman tentang sumber data, format, struktur, dan kualitas data.

3. Pembersihan Data (Data Preparation):

Data sering kali perlu dibersihkan dan disiapkan sebelum analisis. Ini termasuk menghapus data duplikat, mengisi nilai yang hilang, mengatasi outliers, dan mengubah format data jika diperlukan.

4. Transformasi Data (Data Transformation):

Data mungkin perlu diubah bentuknya agar sesuai dengan tujuan analisis. Ini mungkin melibatkan normalisasi, agregasi, pengkodean ulang, dan langkah transformasi lainnya.

5. Pemilihan Teknik (Modeling)

Tahapan ini, memilih teknik atau algoritma data mining yang sesuai untuk pertanyaan bisnis yang ingin jawab. Ini bisa mencakup regresi, klasifikasi, klastering, asosiasi, dan lainnya.

6. Pengolahan dan Analisis Data (Data Processing and Analysis):

Di tahap ini, menerapkan teknik yang dipilih pada data yang telah disiapkan. Melakukan analisis dan memperoleh hasil yang dapat membantu menjawab pertanyaan bisnis yang diajukan.

7. Evaluasi Model (Model Evaluation)

Bagian ini mengevaluasi kinerja model atau hasil analisis dengan menggunakan metrik yang relevan. Tujuan dari langkah ini adalah memastikan bahwa model atau analisis memberikan hasil yang akurat dan berguna.

8. Implementasi (Deployment)

Jika solusi data mining Anda memberikan nilai bisnis yang signifikan, Anda dapat mengimplementasikan solusi tersebut dalam lingkungan bisnis nyata.

9. Monitoring dan Pemeliharaan (Monitoring and Maintenance)

Setelah implementasi, penting untuk memantau kinerja solusi data mining dan memperbarui model atau analisis jika diperlukan. Bisnis dan lingkungan data terus berubah, jadi pemeliharaan berkala diperlukan.

4.2.3. Pengenalan Pola

Salah satu konsep penting dalam kecerdasan buatan adalah pengenalan pola, yang berkaitan dengan kemampuan komputer untuk mengidentifikasi, memahami, dan mengklasifikasikan pola dalam data. Pola-pola ini dapat berupa struktur atau hubungan tertentu antara elemen-elemen dalam data, seperti gambar, teks, suara, atau data lainnya. Tujuan dari pengenalan pola adalah untuk memungkinkan komputer untuk memahami

data yang tidak terstruktur dan kompleks, sehingga komputer dapat menggunakan data yang lebih baik untuk tujuan ini.

Tahapan Pengenalan Pola

1. Pengumpulan dan Persiapan Data: Langkah pertama adalah mengumpulkan data yang relevan untuk analisis. Data ini bisa berupa gambar, teks, suara, atau jenis data lainnya. Setelah data dikumpulkan, langkah selanjutnya adalah mempersiapkannya untuk analisis. Ini termasuk pembersihan data dari noise, mengisi nilai-nilai yang hilang, dan mengkonversi data ke format yang sesuai.
2. Ekstraksi Fitur: Fitur-fitur penting dari data perlu diekstraksi untuk mewakili karakteristik yang relevan. Misalnya, dalam pengenalan gambar, fitur-fitur ini bisa berupa tepi, tekstur, atau pola warna. Ekstraksi fitur dilakukan agar algoritma dapat bekerja dengan representasi data yang lebih sederhana dan terstruktur.
3. Seleksi Fitur (Opsional): Terkadang, data dapat mengandung banyak fitur yang tidak semua relevan atau penting. Langkah seleksi fitur melibatkan pemilihan subset fitur yang paling berkontribusi terhadap pengenalan pola. Ini membantu mengurangi dimensi data dan meningkatkan efisiensi komputasi.
4. Pemilihan Algoritma: Pilihan algoritma sangat tergantung pada jenis data dan tugas yang akan diselesaikan. Misalnya, untuk pengenalan gambar, algoritma Convolutional Neural Networks (CNN) sering digunakan. Pada tahap ini, Anda harus memilih algoritma pembelajaran mesin atau teknik pengenalan pola yang sesuai dengan masalah Anda.
5. Pelatihan Model: Setelah algoritma dipilih, langkah selanjutnya adalah melatih model dengan menggunakan data yang telah dipersiapkan. Proses pelatihan melibatkan memberikan data pelatihan yang berisi contoh-contoh pola dan jawaban yang diharapkan (label). Model akan menyesuaikan parameter-parameter internalnya selama proses pelatihan untuk dapat mengenali pola dengan lebih baik.

6. Validasi dan Evaluasi: Setelah model dilatih, penting untuk mengukur kinerjanya dengan menggunakan data yang tidak pernah dilihat sebelumnya (data validasi atau uji).
7. Penyetelan Hyperparameter: Beberapa algoritma memiliki parameter yang disebut hyperparameter yang perlu disetel untuk mencapai hasil terbaik. Ini melibatkan eksperimen dengan berbagai nilai hyperparameter untuk melihat bagaimana mereka mempengaruhi kinerja model.
8. Prediksi dan Inferensi: Setelah model dilatih dan disetel dengan baik, dapat menggunakannya untuk melakukan prediksi atau inferensi pada data baru yang tidak memiliki label. Model akan menerapkan apa yang telah dipelajari selama pelatihan untuk mengenali pola dalam data baru.

4.3 Pencarian Heuristik

Pencarian heuristik dalam konteks kecerdasan buatan (AI) berarti menggunakan aturan atau strategi berbasis pengetahuan untuk menemukan solusi. Ini memungknkan tidak selalu memberi solusi terbaik, tetapi dapat membantu menemukan solusi dengan lebih cepat dan efisien. Heuristik adalah panduan praktis atau metode pendekatan yang cenderung menghasilkan solusi yang cukup baik, meskipun kadang-kadang tidak optimal. Pencarian heuristik sangat berguna ketika mencari solusi terbaik terlalu sulit atau memakan waktu. Berikut ini adalah beberapa contoh algoritma pencarian heuristik yang sering digunakan dalam AI:

4.3.1. Algoritma Genetika

John Holand dari New York, Amerika Serikat, mengembangkan algoritma genetika pertama kali. Penemuan ini diterbitkan dalam bukunya yang berjudul "Adaption in Natural and Artificial Systems" pada tahun 1975. Algoritma genetika adalah metode untuk menemukan solusi terbaik untuk setiap masalah. Metode ini akan mencoba beberapa solusi sebelum menemukan yang

terbaik untuk kriteria yang ditetapkan, atau fungsi fitness. Algoritma ini termasuk dalam kelompok algoritma evolusioner karena menggunakan pendekatan evolusi Darwin di bidang biologi seperti pewarisan sifat, seleksi alam, mutasi gen, dan kombinasi (crossover). Karena itu, algoritma ini juga termasuk dalam kelompok algoritma metaheuristik karena merupakan teknik pencarian optimal dalam ilmu komputer.

Langkah-Langkah Umum dalam Algoritma Genetika

1. *Population Initiation* (Insiasi Populasi)
   Di mulai dengan menghasilkan populasi awal yang terdiri dari beberapa objek tertentu, masing-masing mewakili satu potensi solusi untuk masalah yang sedang dipecahkan. Ini seringkali dibuat secara acak atau dengan teknik inisialisasi yang sesuai dengan masalah yang sedang dihadapi.
2. Evaluation
   Untuk mengetahui sejauh mana solusi yang mereka tawarkan adalah solusi yang efektif, evaluasi setiap anggota populasi dilakukan dengan menghitung nilai kebugaran setiap anggota. Nilai kebugaran menunjukkan seberapa baik solusi yang ditawarkan setiap anggota dalam menyelesaikan masalah.
3. Selection
   Individu dengan nilai fitness yang lebih tinggi cenderung lebih banyak berkontribusi pada generasi berikutnya. Akibatnya, objek dengan nilai fitness yang lebih tinggi dipilih untuk menjadi orang tua dalam proses reproduksi.
4. Reproduction
   Reproduksi adalah proses pembentukan generasi baru dalam populasi berikutnya. Mutasi dan crossover (persilangan) merupakan dua orang dipilih sebagai objek tua, dan potongan solusi mereka digabungkan untuk menghasilkan satu atau lebih keturunan. Mutasi: Variasi

populasi disebabkan oleh penerapan perubahan acak pada individu.

5. Replacement
   Populasi baru menggantikan populasi lama. Proses ini dapat bersifat elitis, yang berarti sebagian besar individu terbaik tetap ada, atau tidak elitis, yang berarti semua orang digantikan.
6. Termination
   Algoritma genetika berlangsung selama beberapa generasi atau hingga kriteria berhenti terpenuhi. Kriteria berhenti dapat berupa jumlah generasi maksimum atau penemuan solusi yang memenuhi batasan tertentu.
7. Optimal Solution
   Algoritma genetika menghasilkan individu terbaik dalam populasi terakhir, yang seharusnya merupakan solusi terbaik untuk masalah.

Contoh Penerapan Algoritma Genetika

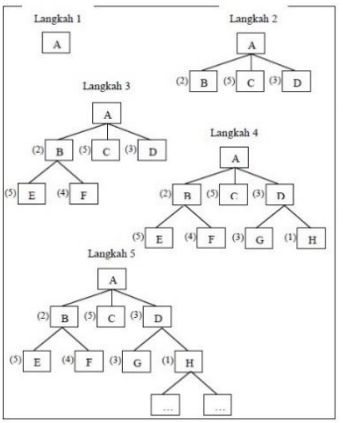
1. Perencanaan Jadwal (Scheduling)
   Algoritma genetika dapat digunakan untuk mengoptimalkan jadwal penugasan sumber daya dalam berbagai situasi seperti jadwal kuliah, penjadwalan produksi, atau penjadwalan transportasi.
2. Optimasi Rute Perjalanan
   Dalam kasus ini, algoritma genetika dapat digunakan untuk menemukan rute terpendek yang mengunjungi beberapa kota dan kembali ke kota awal. Setiap individu dalam populasi mewakili sebuah rute, dan evolusi populasi bertujuan untuk menemukan rute terpendek.
3. Perancangan Struktur
   Dalam bidang perancangan struktur, seperti desain bangunan atau jembatan, algoritma genetika dapat digunakan untuk mencari konfigurasi yang optimal dari elemen-elemen struktural, yang memenuhi persyaratan teknis dan biaya yang diberikan.
4. Desain Jaringan (Network Design)

Dalam desain jaringan komunikasi, seperti desain jaringan telekomunikasi atau jaringan komputer, algoritma genetika dapat digunakan untuk mengoptimalkan topologi jaringan dan alokasi sumber daya.

4.3.2. Pencarian Terbaik Pertama (Best-First Search)

Menggunakan konsep pencarian heuristik (Heuristic Search) dan queue prioritas, Best First Search dapat memperluas simpul dari simpul sebelumnya sesuai dengan aturan tertentu. Algoritma ini bertujuan untuk mencapai titik tujuan atau titik akhir dengan cara yang paling singkat dari titik awal.

Di antara semua leaf node (simpul pada level terdalam) yang pernah dibangkitkan, pilihan pertama akan memilih simpul atau node baru dengan biaya paling rendah. Fungsi evaluasi f(n) digunakan untuk mendapatkan simpul terbaik. Evaluasi Best First Search dapat mencakup biaya perkiraan dari suatu simpul menuju tujuan atau kombinasi biaya sebenarnya dan perkiraan.



Gambar 4.1 Langkah Proses *Best First Search*
(Sumber. https://www.trivusi.web.id/2022/03/algoritma-best-first-search.html)

a. Bangkitkan node A, kemudian semua successor A, dan cari biaya terendah.

b. Node B dipilih karena memiliki biaya paling rendah, yaitu.
c. Semua successor B dibangkitkan. Kemudian biaya node C dan D dibandingkan, dan node D memiliki biaya yang paling kecil dibandingkan dengan biaya node C, E, F, dan G. Oleh karena itu, node D dipilih dan selanjutnya akan dibangkitkan semua successor D.
d. Biaya node H paling kecil dibandingkan dengan biaya node C, E, F, dan G, sehingga semua successor H Kemudian lanjutkan sampai Anda menemukan node tujuan.

Contoh Penerapan Algoritma Best-First Search

1. Permainan: BFS digunakan dalam permainan seperti catur, permainan video, dan permainan papan lainnya. Dalam catur, misalnya, BFS digunakan untuk menghitung langkah-langkah yang mungkin dan mengevaluasi kemungkinan konsekuensi langkah-langkah tersebut untuk mencari jalur terbaik menuju kondisi menang (mengambil raja lawan).
2. Navigasi dan Pemetaan: BFS digunakan dalam aplikasi navigasi seperti GPS dan peta digital untuk mencari jalur terpendek atau tercepat antara dua lokasi. Dalam hal ini, BFS membantu menemukan rute terbaik berdasarkan nilai heuristik yang mewakili jarak atau waktu tempuh antara lokasi-lokasi tersebut.
3. Pencarian dalam Ruang Keadaan: BFS digunakan dalam perencanaan perjalanan dan pencarian dalam ruang keadaan, di mana ada banyak kemungkinan tindakan yang dapat diambil dan Anda ingin mencapai tujuan tertentu. Contohnya termasuk perencanaan rute robot, perencanaan produksi, dan perencanaan perjalanan di bidang logistik.

4.3.3. Pencarian Berlekuk (Hill Climbing)

Menghadapi masalah optimasi, algoritma pencarian hill climbing digunakan untuk mencari solusi terbaik atau paling optimal dalam ruang pencarian. Dalam ruang pencarian ini, setiap solusi memiliki nilai numerik yang dapat diukur, biasanya

dalam bentuk fungsi objektif, dan Anda ingin mencari solusi yang memiliki nilai maksimum atau minimum, tergantung pada jenis masalah yang dihadapi. Ketika mereka mendaki bukit, mereka mengambil solusi pertama dan kemudian memperbaruinya ke solusi yang lebih baik di sekitarnya. Ini dicapai dengan mengganti solusi saat ini dengan yang lebih dekat dan lebih baik.

Langkah-langkah Hill Climbing

1. Inisialisasi
   Langkah pertama adalah memilih solusi awal yang bisa acak atau didasarkan pada pengetahuan awal tentang masalah.
2. Evaluasi
   Solusi awal dievaluasi menggunakan fungsi objektif yang diberikan. Fungsi ini akan memberikan skor atau nilai berdasarkan seberapa baik solusi tersebut.
3. Iterasi
   Proses iteratif dimulai, di mana algoritma akan mencoba meningkatkan solusi saat ini dengan mengubahnya sedikit demi sedikit. Ini dilakukan dengan menghasilkan solusi yang berdekatan dalam ruang pencarian.
4. Pemilihan Langkah
   Ada beberapa strategi yang dapat digunakan untuk memilih langkah perubahan solusi saat ini. Beberapa di antaranya adalah:
   - Hill Climbing Stochastic (Random Restart): Memilih langkah acak dari solusi saat ini.
   - Hill Climbing Steepest Ascent: Memilih langkah yang memiliki perubahan terbesar dalam nilai fungsi objektif.
   - Hill Climbing dengan Simulated Annealing: Menggunakan probabilitas untuk menerima langkah yang lebih buruk dengan tujuan menghindari jatuh ke minimum lokal.

DAFTAR PUSTAKA

Amazon, A. (2023). *Apa itu Pemrosesan Bahasa Alami (NLP)? - Penjelasan tentang NLP - AWS*. Amazon Web Services, Inc. https://aws.amazon.com/id/what-is/nlp/

Setiawan, R. (2021). *Apa itu Data Mining dan Bagaimana Metodenya? - Dicoding Blog*. https://www.dicoding.com/blog/apa-itu-data-mining/

BAB 5
Pencarian Adversarial

Oleh Suwito Pomalingo

5.1 Pendahuluan

5.1.1 Definisi Pencarian Adversarial

Pencarian Adversarial adalah sebuah subbidang dalam kecerdasan buatan yang berfokus pada pemecahan masalah di mana dua atau lebih agen berkompetisi untuk mencapai tujuan yang berlawanan. Dalam konteks ini, "adversarial" mengacu pada sifat kompetitif antara agen-agen yang terlibat (Russell and Norvig, 2009). Dalam model ini, setiap agen berusaha untuk memaksimalkan fungsi utilitas atau keuntungannya sendiri, seringkali pada biaya agen lain. Ini berbeda dari model pencarian tradisional atau algoritma optimasi, di mana tujuannya adalah untuk menemukan solusi terbaik dari sebuah masalah tanpa adanya kompetisi antar agen.

Dalam konteks permainan, agen biasanya dibagi menjadi dua kelompok: "Maximizer" dan "Minimizer". Maximizer berusaha untuk memaksimalkan nilai dari fungsi evaluasi, sementara Minimizer berusaha untuk meminimalkannya. Kedua agen ini beraksi secara bergantian, memilih langkah-langkah dari suatu himpunan langkah yang memungkinkan dengan tujuan untuk mencapai keadaan akhir yang paling menguntungkan bagi mereka.

Penting untuk diingat bahwa dalam Pencarian Adversarial, keputusan yang diambil oleh satu agen akan mempengaruhi

keadaan lingkungan, yang pada gilirannya akan mempengaruhi keputusan agen lainnya. Ini menciptakan sebuah dinamika yang kompleks dan seringkali tidak deterministik, memerlukan algoritma dan teknik yang canggih untuk menavigasi (Browne *et al.*, 2012).

Konsep ini juga telah diterapkan di luar permainan, termasuk dalam simulasi keamanan siber, di mana sistem keamanan (defender) berupaya untuk mengidentifikasi dan menetralkan potensi tindakan dari penyerang (attacker), yang juga menggunakan strategi yang sama untuk mengeksploitasi sistem (Lye and Wing, 2005).

Dengan demikian, Pencarian Adversarial adalah sebuah pendekatan yang sangat fleksibel dan kuat untuk memodelkan berbagai jenis interaksi kompetitif, baik dalam permainan maupun aplikasi praktis lainnya.

5.1.2 Signifikansi dalam Kecerdasan Buatan

Pencarian Adversarial tidak hanya merupakan sebuah teknik atau algoritma, tetapi juga sebuah paradigma yang membentuk dasar dari banyak aplikasi kecerdasan buatan (AI). Salah satu signifikansi utamanya adalah dalam memahami dan memodelkan keputusan dalam lingkungan yang kompetitif atau antagonis. Ini adalah sebuah langkah maju dari model keputusan tradisional yang seringkali hanya mempertimbangkan satu agen dan mengabaikan interaksi dengan agen lain.

1. Kompleksitas dan Realisme: Dalam banyak situasi dunia nyata, keputusan dibuat dalam konteks yang melibatkan kompetisi atau konflik. Pencarian Adversarial menawarkan kerangka kerja untuk memodelkan dan memahami dinamika ini, memberikan solusi yang lebih realistis dan robust.
2. Optimalitas dan Robustness: Algoritma yang digunakan dalam pencarian adversarial, seperti Minimax dan Alpha-Beta Pruning, seringkali menghasilkan solusi yang optimal atau hampir optimal. Ini penting dalam aplikasi seperti

permainan papan, di mana keputusan yang suboptimal bisa dengan cepat mengarah pada kekalahan.

3. Interdisipliner: Signifikansi pencarian adversarial juga terlihat dalam berbagai disiplin ilmu, termasuk ekonomi (teori permainan), psikologi (pengambilan keputusan), dan bahkan biologi (evolusi strategi). Ini menunjukkan fleksibilitas dan kegunaan metode ini dalam memahami berbagai jenis sistem yang kompleks.
4. Pembelajaran Mesin dan Data Science: Dalam era data besar, pencarian adversarial juga menjadi penting dalam pengembangan algoritma pembelajaran mesin yang robust terhadap manipulasi data, seringkali disebut sebagai "adversarial machine learning".
5. Keamanan dan Etika: Dalam konteks keamanan siber dan etika AI, memahami bagaimana agen bisa bertindak secara adversarial adalah kunci untuk membangun sistem yang lebih aman dan etis.

Pencarian Adversarial memegang peranan penting dalam evolusi dan aplikasi kecerdasan buatan. Ini adalah salah satu dari sedikit paradigma yang memungkinkan mesin untuk membuat keputusan yang optimal atau hampir optimal dalam situasi yang kompetitif. Teknik ini sangat relevan dalam berbagai aplikasi, mulai dari permainan papan seperti catur dan Go, hingga aplikasi yang lebih kompleks seperti keamanan siber, negosiasi otomatis, dan bahkan di bidang keuangan untuk model prediksi harga saham (Sutton and Barto, 2018).

5.1.3 Aplikasi Umum

Pencarian Adversarial tidak hanya terbatas pada permainan tetapi juga memiliki aplikasi yang luas di berbagai bidang. Misalnya, dalam keamanan siber, teknik ini digunakan untuk memodelkan tindakan potensial dari penyerang dan mempertahankan sistem dari serangan (Papernot *et al.*, 2016). Selain itu, dalam bidang robotika, algoritma ini digunakan untuk simulasi dan pelatihan robot dalam lingkungan yang kompetitif (Kochenderfer, 2015).

Pencarian Adversarial memiliki aplikasi yang sangat luas dan beragam, mencakup berbagai sektor dan disiplin ilmu. Berikut adalah beberapa aplikasi umum yang menunjukkan fleksibilitas dan kegunaan dari paradigma ini:

1. Permainan Papan dan Video Games: Ini adalah salah satu aplikasi paling awal dan paling terkenal dari pencarian adversarial. Dari catur hingga Go, dan dari poker hingga video games strategi, algoritma seperti Minimax dan Monte Carlo Tree Search digunakan untuk menciptakan lawan buatan yang kuat.
2. Keamanan Siber: Pencarian adversarial digunakan untuk memodelkan dan memitigasi potensi serangan pada jaringan dan sistem. Ini membantu dalam merancang kebijakan keamanan yang lebih efektif dan dalam mendeteksi potensi celah keamanan.
3. Robotika: Dalam robotika, algoritma ini digunakan untuk pelatihan dan simulasi. Misalnya, dua robot yang berkompetisi untuk mencapai tujuan tertentu bisa dilatih menggunakan paradigma ini.
4. Sistem Rekomendasi: Meskipun kurang intuitif, pencarian adversarial juga digunakan dalam sistem rekomendasi untuk memodelkan kompetisi antara item yang berbeda untuk mendapatkan perhatian pengguna.
5. Bidang Medis: Dalam pengembangan obat atau terapi, model adversarial digunakan untuk memprediksi bagaimana patogen atau sel kanker akan bereaksi terhadap berbagai jenis pengobatan, memungkinkan perancangan strategi pengobatan yang lebih efektif.
6. Otomotif dan Transportasi: Dalam desain kendaraan otonom, pencarian adversarial digunakan untuk memodelkan interaksi antara kendaraan dan lingkungannya, termasuk kendaraan lain dan pejalan kaki, untuk meningkatkan keamanan dan efisiensi.
7. Pasar Keuangan: Algoritma ini digunakan untuk memodelkan perilaku trader dan pasar, membantu dalam pembuatan keputusan investasi dan manajemen risiko.

5.2 Dasar Teori

5.2.1 Teori Permainan

Teori Permainan adalah cabang dari matematika yang digunakan untuk memodelkan, menganalisis, dan memahami interaksi antara berbagai agen yang membuat keputusan. Ini adalah fondasi teoretis dari banyak algoritma dalam pencarian adversarial dan memiliki aplikasi yang luas dalam kecerdasan buatan (Osborne and Rubinstein, 1994).

Teori Permainan pertama kali diperkenalkan oleh John von Neumann dan Oskar Morgenstern pada tahun 1944. Sejak itu, teori ini telah berkembang menjadi alat yang sangat kuat untuk memahami dinamika sosial, ekonomi, dan keputusan.

Konsep Dasar

1. Pemain: Agen yang terlibat dalam permainan.
2. Strategi: Rencana aksi yang diambil oleh pemain.
3. Payoff: Keuntungan atau kerugian yang diterima oleh pemain berdasarkan strategi yang dipilih.
4. Equilibrium: Keadaan di mana tidak ada pemain yang bisa meningkatkan payoff-nya dengan mengubah strategi jika strategi pemain lain tetap.

Jenis-Jenis Permainan

1. Permainan Kooperatif vs Non-Kooperatif: Dalam permainan kooperatif, pemain bisa membentuk aliansi, sementara dalam permainan non-kooperatif, setiap pemain bertindak secara independen.
2. Permainan Sempurna vs Tidak Sempurna: Dalam permainan dengan informasi sempurna, semua informasi tentang permainan dan tindakan pemain lainnya adalah diketahui, sedangkan dalam permainan dengan informasi tidak sempurna, beberapa informasi disembunyikan.
3. Permainan Berulang vs Sekali Main: Permainan bisa berlangsung dalam satu putaran atau berulang kali.

Teori Permainan digunakan sebagai dasar dalam algoritma pencarian adversarial seperti Minimax dan Alpha-Beta Pruning. Selain itu, ia juga digunakan dalam pelatihan model kecerdasan buatan untuk situasi kompetitif atau antagonis, seperti dalam pembelajaran penguatan (Sutton and Barto, 2018).

Memahami Teori Permainan adalah kunci untuk memahami dan mengimplementasikan algoritma dalam pencarian adversarial. Ini memberikan kerangka kerja matematis untuk memodelkan dan menganalisis interaksi antara agen yang berkompetisi, yang adalah inti dari banyak aplikasi kecerdasan buatan.

5.2.2 Fungsi Evaluasi

Fungsi Evaluasi adalah salah satu komponen kunci dalam algoritma pencarian adversarial. Fungsi ini digunakan untuk mengestimasi kegunaan atau nilai dari sebuah keadaan permainan, yang selanjutnya digunakan untuk membuat keputusan dalam permainan atau situasi kompetitif (Russell and Norvig, 2009).

Fungsi evaluasi, seringkali dinotasikan sebagai $f(x)$, adalah sebuah fungsi yang mengambil sebuah keadaan x sebagai input dan mengembalikan sebuah nilai numerik sebagai output. Nilai ini biasanya merepresentasikan "kebaikan" dari keadaan tersebut untuk sebuah agen.

Deterministik vs Stokastik: Fungsi evaluasi bisa deterministik atau stokastik. Fungsi deterministik memberikan hasil yang sama setiap kali untuk keadaan yang sama, sementara fungsi stokastik bisa memberikan hasil yang berbeda-beda.

Statik vs Dinamik: Fungsi evaluasi statik tidak berubah seiring waktu, sementara fungsi dinamik bisa beradaptasi berdasarkan data atau pengalaman baru (Mitchell, 1997).

Kriteria Desain

- Efisiensi: Fungsi evaluasi harus cepat dihitung agar bisa digunakan dalam algoritma pencarian yang memerlukan evaluasi berulang-ulang.
- Akurasi: Fungsi evaluasi harus akurat dalam memprediksi keuntungan atau kerugian dari sebuah keadaan.
- Simpel vs Kompleks: Kadang-kadang, sebuah fungsi evaluasi yang lebih simpel lebih disukai karena lebih mudah diinterpretasi dan diimplementasikan, meskipun mungkin kurang akurat.

Fungsi evaluasi digunakan dalam berbagai algoritma pencarian adversarial, termasuk Minimax, Alpha-Beta Pruning, dan Monte Carlo Tree Search. Selain itu, fungsi evaluasi juga digunakan dalam algoritma pembelajaran mesin, seperti jaringan saraf tiruan, untuk mengukur kualitas dari model yang sedang dilatih (Goodfellow, Bengio and Courville, 2016)..

Fungsi evaluasi adalah komponen penting dalam pencarian adversarial dan kecerdasan buatan pada umumnya. Memilih atau merancang fungsi evaluasi yang tepat adalah kunci untuk membangun sistem yang efisien dan efektif.

5.2.3 Pohon Keputusan

Pohon Keputusan adalah struktur data hierarkis yang digunakan untuk memodelkan keputusan dan kemungkinan keadaan yang mungkin terjadi sebagai hasil dari keputusan tersebut. Dalam konteks pencarian adversarial, pohon keputusan digunakan untuk memvisualisasikan semua langkah yang mungkin diambil oleh agen yang berkompetisi (Russell and Norvig, 2009).

Gambar 5.1 Pohon Keputusan
(Sumber : pemrogramanmatlab.com)

Definisi dan Komponen

- Node: Setiap titik dalam pohon keputusan disebut node, yang merepresentasikan sebuah keadaan dalam permainan atau situasi.
- Edge: Garis yang menghubungkan dua node disebut edge, yang merepresentasikan aksi atau transisi dari satu keadaan ke keadaan lain.
- Root: Node awal dari pohon, biasanya merepresentasikan keadaan awal dari permainan.
- Leaf: Node akhir dari pohon, biasanya merepresentasikan keadaan akhir dari permainan, di mana hasil (menang, kalah, seri) sudah ditentukan.

Jenis-Jenis Pohon Keputusan

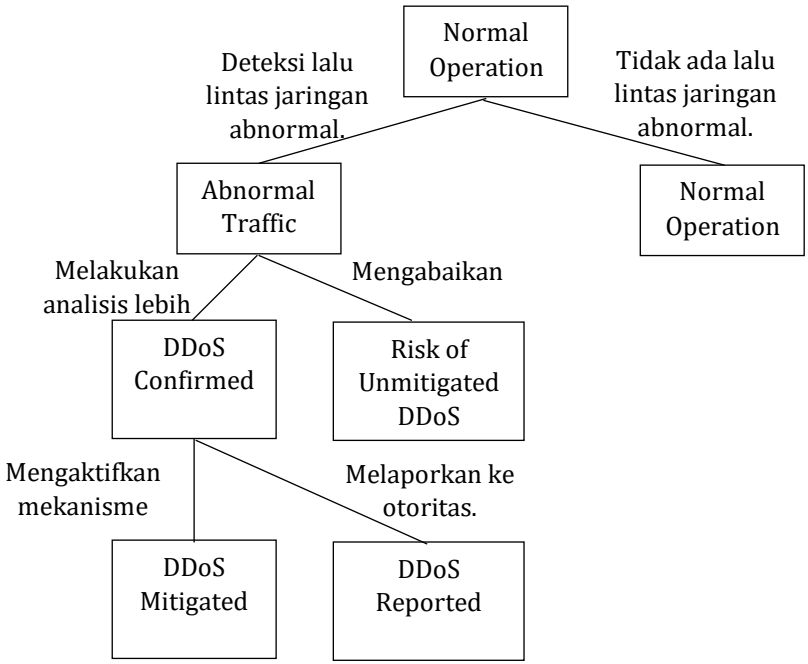
- Pohon Keputusan Eksplisit: Semua kemungkinan keadaan dan aksi dijelaskan secara eksplisit. Biasanya digunakan untuk permainan atau situasi yang sederhana.
- Pohon Keputusan Implisit: Hanya sebagian dari semua kemungkinan keadaan dan aksi yang dijelaskan, biasanya digunakan untuk permainan atau situasi yang kompleks untuk menghemat memori dan waktu komputasi (Browne et al., 2012).

Algoritma dan Pohon Keputusan

- Minimax: Menggunakan pohon keputusan untuk mencari langkah terbaik dengan meminimalkan keuntungan maksimum yang bisa dicapai oleh lawan.
- Alpha-Beta Pruning: Varian dari Minimax yang lebih efisien, memotong cabang pohon yang tidak perlu dievaluasi.

Pohon keputusan digunakan dalam berbagai algoritma dan aplikasi dalam kecerdasan buatan, termasuk permainan papan, keamanan siber, dan bahkan dalam pengambilan keputusan medis.

Dalam konteks keamanan siber, pohon keputusan bisa digunakan untuk memodelkan berbagai skenario serangan dan respons yang mungkin terjadi. Berikut adalah contoh sederhana bagaimana pohon keputusan bisa diaplikasikan dalam keamanan siber, khususnya untuk memitigasi serangan DDoS (Distributed Denial of Service).



Gambar 5.2 Pohon Keputusan untuk Keamanan Siber
(Sumber : dokumentasi pribadi)

Dalam implementasi nyata, pohon keputusan ini bisa diintegrasikan ke dalam sistem IDS (Intrusion Detection System) atau IPS (Intrusion Prevention System). Algoritma bisa dikembangkan untuk berjalan setiap kali ada deteksi lalu lintas jaringan yang abnormal, dan kemudian mengikuti pohon keputusan untuk menentukan tindakan apa yang harus diambil. Berikut contoh pseudocodenya.

```
if detect_abnormal_traffic():
   analyze_traffic()
   if is_ddos_attack():
      mitigation_response = decide_mitigation()  # Menggunakan
pohon keputusan
      if mitigation_response == "Activate_Mechanism":
         activate_ddos_mitigation()
      elif mitigation_response == "Report":
```

```
        report_to_authorities()
else:
   continue_normal_operation()
```

Dengan menggunakan pohon keputusan, tim keamanan siber bisa lebih mudah memahami dan memodelkan berbagai skenario yang mungkin terjadi, sehingga memungkinkan mereka untuk merespons lebih cepat dan lebih efektif terhadap ancaman.

Keuntungan yang dapat diperoleh:

- Mempermudah proses pengambilan keputusan dalam keadaan darurat.
- Membantu dalam merancang dan menguji strategi mitigasi serangan.

Namun, pohon keputusan ini tidak selalu efektif untuk serangan yang kompleks atau yang menggunakan teknik yang belum diketahui.

Pohon Keputusan adalah alat yang sangat berguna dalam memodelkan dan memahami keputusan dalam konteks adversarial. Mereka adalah komponen kunci dalam banyak algoritma pencarian adversarial dan memiliki aplikasi yang luas dalam kecerdasan buatan.

5.3 Algoritma Dasar

5.3.1 Minimax

Algoritma Minimax adalah algoritma pencarian yang digunakan dalam teori permainan untuk menemukan langkah terbaik bagi pemain, asumsi lawan juga bermain optimal. Algoritma ini biasanya digunakan dalam permainan dua pemain seperti catur, tic-tac-toe, dan Go.

Minimax bekerja dengan cara membangun pohon keputusan yang merepresentasikan semua kemungkinan langkah yang bisa

diambil oleh kedua pemain. Pada setiap level pohon, algoritma akan mencoba memaksimalkan atau meminimalkan nilai dari keadaan permainan, tergantung pada pemain mana yang sedang bermain.

Berikut model algoritma minimax:

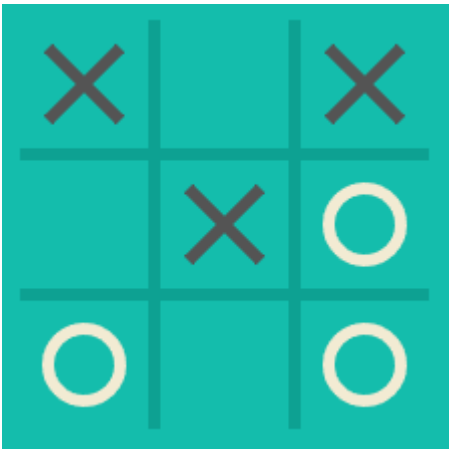
1. Inisialisasi: Mulai dari keadaan awal permainan.
2. Ekspansi Pohon: Bangun pohon keputusan yang merepresentasikan semua kemungkinan langkah.
3. Evaluasi: Gunakan fungsi evaluasi untuk menilai keadaan akhir permainan pada node daun.
4. Propagasi Nilai: Mulai dari node daun, propagasi nilai ke atas menuju node induk.
5. Pilihan Langkah: Pilih langkah yang memberikan nilai maksimum untuk pemain yang sedang bermain.

Bentuk pseudocodenya sebagai berikut

```
def minimax(node, depth, maximizing):
  if depth == 0 or is_terminal(node):
    return evaluate(node)

  if maximizing:
    max_value = -infinity
    for child in children(node):
      value = minimax(child, depth-1, False)
      max_value = max(max_value, value)
    return max_value
  else:
    min_value = +infinity
    for child in children(node):
      value = minimax(child, depth-1, True)
      min_value = min(min_value, value)
    return min_value
```

Misalkan kita memiliki papan tic-tac-toe sebagai berikut:



Gambar 5.3 Permainan tic-tac-toe
(Sumber : dokumentasi pribadi)

Kita ingin mengetahui langkah terbaik untuk pemain 'O'.

1. Inisialisasi: Papan di atas adalah keadaan awal.
2. Ekspansi Pohon: Kita akan membangun pohon keputusan untuk semua kemungkinan langkah dari keadaan ini.
3. Evaluasi: Misalkan kita menggunakan fungsi evaluasi sederhana: +1 untuk menang, -1 untuk kalah, 0 untuk seri atau keadaan lainnya.
4. Propagasi Nilai: Kita akan menggunakan algoritma Minimax untuk mengevaluasi setiap langkah dan propagasi nilai ke atas.
5. Pilihan Langkah: Langkah yang memberikan nilai maksimum untuk pemain 'O' akan dipilih.

Dengan menggunakan algoritma Minimax, kita bisa menentukan langkah terbaik untuk pemain 'O' dalam situasi ini.

Kelebihan

1. Solusi Optimal: Salah satu kelebihan utama dari algoritma Minimax adalah kemampuannya untuk selalu menemukan

solusi optimal asalkan seluruh pohon keputusan bisa dieksplorasi. Ini berarti dalam permainan seperti catur atau tic-tac-toe, algoritma ini akan selalu menemukan langkah terbaik jika diberikan cukup waktu dan sumber daya.

2. Mudah Diimplementasikan: Algoritma ini relatif sederhana dan mudah diimplementasikan. Ini membuatnya menjadi pilihan yang baik untuk permainan atau aplikasi yang tidak terlalu kompleks.
3. Deterministik: Hasil dari algoritma Minimax adalah deterministik, asalkan fungsi evaluasi juga deterministik. Ini berarti Anda akan mendapatkan hasil yang konsisten setiap kali algoritma dijalankan.
4. Berguna dalam Berbagai Aplikasi: Selain permainan, Minimax juga bisa digunakan dalam berbagai aplikasi lain seperti keamanan siber, robotika, dan optimasi industri.

Kekurangan

1. Kompleksitas Waktu: Salah satu kekurangan terbesar dari algoritma Minimax adalah kompleksitas waktu yang bisa menjadi sangat besar untuk permainan atau aplikasi dengan banyak kemungkinan langkah atau keadaan. Ini bisa membuat algoritma ini tidak praktis untuk beberapa kasus.
2. Ketergantungan pada Fungsi Evaluasi: Efektivitas algoritma ini sangat bergantung pada kualitas dari fungsi evaluasi. Jika fungsi evaluasi tidak akurat atau tidak mencerminkan keadaan permainan dengan baik, algoritma ini bisa menghasilkan keputusan yang buruk.
3. Kurang Efisien untuk Beberapa Jenis Permainan: Dalam permainan yang informasinya tidak sempurna atau yang melibatkan unsur kebetulan, algoritma Minimax menjadi kurang efisien atau bahkan tidak dapat digunakan.\Kurang Fleksibel: Algoritma ini mengasumsikan bahwa lawan selalu bermain optimal, yang mungkin tidak selalu terjadi dalam

situasi nyata. Ini bisa menjadi kelemahan jika lawan bermain secara tidak terduga atau irasional.

5.3.2 Alpha-Beta Pruning

Alpha-Beta Pruning adalah teknik optimasi untuk algoritma Minimax yang digunakan dalam teori permainan. Teknik ini mengurangi jumlah node yang dievaluasi oleh algoritma Minimax dengan cara memotong cabang yang tidak perlu dievaluasi karena sudah diketahui bahwa cabang tersebut tidak akan dipilih.

Algoritmanya sebagai berikut:

1. Inisialisasi: Mulai dengan nilai alpha (nilai terbaik untuk pemain Maksimisasi) dan beta (nilai terbaik untuk pemain Minimisasi). Awalnya, alpha adalah -infiniti dan beta adalah +infiniti.
2. Pengecekan Pruning: Selama pencarian, jika nilai alpha lebih besar dari atau sama dengan beta, maka cabang tersebut dapat dipotong.
3. Pembaruan Alpha dan Beta: Alpha diperbarui saat kita berada pada level Maksimisasi dan menemukan nilai yang lebih baik. Beta diperbarui saat kita berada pada level Minimisasi dan menemukan nilai yang lebih baik.

Berikut pseudocodenya:

```
def alpha_beta(node, depth, alpha, beta, maximizing):
  if depth == 0 or is_terminal(node):
    return evaluate(node)

  if maximizing:
    max_value = -infinity
    for child in children(node):
      value = alpha_beta(child, depth-1, alpha, beta, False)
      max_value = max(max_value, value)
```

```
            alpha = max(alpha, value)
            if alpha >= beta:
                break
        return max_value
    else:
        min_value = +infinity
        for child in children(node):
            value = alpha_beta(child, depth-1, alpha, beta, True)
            min_value = min(min_value, value)
            beta = min(beta, value)
            if alpha >= beta:
                break
        return min_value
```

Untuk contoh kasusnya sama seperti pada algoritma minimax untuk permainan tic-tac-toe, dimana kita ingin mengetahui langkah terbaik untuk pemain 'O'. Dengan menggunakan Alpha-Beta Pruning, kita bisa memotong beberapa cabang dari pohon keputusan yang tidak perlu dievaluasi, sehingga proses pencarian menjadi lebih cepat.

Kelebihan dan Kekurangan dari algoritma Alpa-beta Prunning sebagai berikut:

Kelebihan

1. Efisiensi: Alpha-Beta Pruning secara signifikan mengurangi jumlah node yang perlu dievaluasi. Dalam kasus terbaik, algoritma ini bisa mengurangi kompleksitas waktu dari O(b^d) menjadi O(b^(d/2)), di mana b adalah faktor percabangan dan d adalah kedalaman pohon.
2. Optimalitas: Meskipun lebih efisien, Alpha-Beta Pruning tetap memberikan solusi yang sama dengan algoritma Minimax tradisional.
3. Fleksibilitas: Teknik ini bisa digunakan dalam kombinasi dengan teknik pencarian lainnya untuk meningkatkan efisiensi.

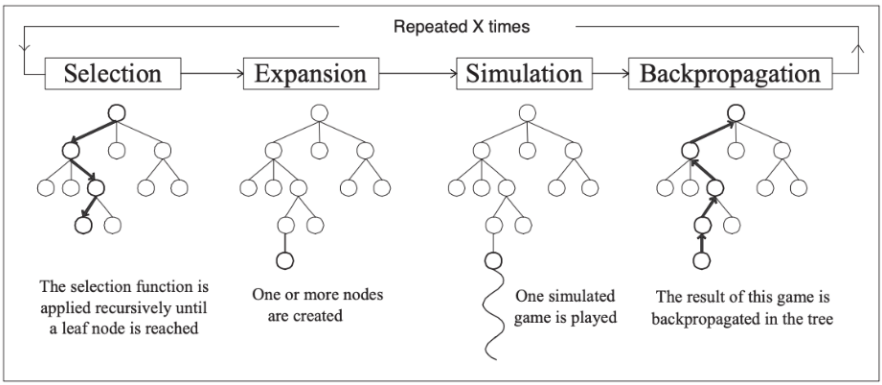
Kekurangan

1. Ketergantungan pada Urutan: Efektivitas Alpha-Beta Pruning sangat bergantung pada urutan node yang dievaluasi. Jika node dengan nilai terbaik dievaluasi lebih awal, maka lebih banyak cabang yang bisa dipotong.
2. Kompleksitas Implementasi: Meskipun pseudocode-nya terlihat sederhana, implementasi yang efisien dari Alpha-Beta Pruning bisa menjadi kompleks, terutama dalam permainan dengan banyak kemungkinan langkah.
3. Masih Ada Potensi untuk Optimasi: Meskipun Alpha-Beta Pruning meningkatkan efisiensi Minimax, masih ada teknik lain seperti pencarian berbasis heuristik yang bisa meningkatkan efisiensi lebih jauh.

5.4 Algoritma Lanjutan

5.4.1 Monte Carlo Tree Search (MCTS)

Monte Carlo Tree Search (MCTS) adalah algoritma pencarian berbasis simulasi yang digunakan dalam kecerdasan buatan untuk menemukan langkah terbaik dari keadaan tertentu. Algoritma ini telah mendapatkan popularitas khususnya dalam permainan seperti Go, di mana ia digunakan oleh program seperti AlphaGo dari DeepMind.



Gambar 5.4 Outline Monte-Carlo Tree Search
(Sumber : paperswithcode.com)

MCTS mengkombinasikan pencarian berbasis pohon dengan simulasi Monte Carlo. Algoritma ini terdiri dari empat langkah utama yang diulang-ulang sampai batas waktu atau jumlah iterasi tertentu tercapai:

1. Seleksi: Mulai dari root dan pilih child nodes secara berurutan hingga mencapai leaf node.
2. Ekspansi: Tambahkan satu atau lebih child nodes ke leaf node yang telah dipilih.
3. Simulasi: Lakukan simulasi dari child node yang baru hingga akhir permainan.
4. Backpropagation: Sebarkan hasil simulasi kembali ke node-node sebelumnya untuk memperbarui statistik mereka.

Contoh Kasus pada permainan Tic-Tac-Toe. Misalkan kita memiliki papan tic-tac-toe yang kosong dan ingin mengetahui langkah terbaik untuk pemain 'X'. Dengan MCTS:

1. Seleksi: Mulai dari keadaan awal papan yang kosong, pilih child node berdasarkan statistik yang ada (misalnya dengan UCB1).
2. Ekspansi: Tambahkan beberapa kemungkinan langkah ke papan.
3. Simulasi: Misalkan kita menambahkan langkah di tengah papan. Dari sini, lakukan simulasi permainan dengan langkah acak untuk kedua pemain hingga papan penuh atau salah satu pemain menang.
4. Backpropagation: Sebarkan hasil simulasi kembali ke node sebelumnya.

Proses ini diulang-ulang hingga batas waktu atau iterasi tercapai, dan langkah dengan statistik terbaik dipilih.

Adapun kelebihan dari algoritma ini adalah:

1. Fleksibilitas: MCTS tidak memerlukan pengetahuan domain khusus, menjadikannya cocok untuk berbagai jenis permainan dan aplikasi.
2. Adaptif: MCTS secara adaptif fokus pada bagian pohon yang paling menjanjikan, berbeda dengan pencarian berbasis pohon lain yang mungkin memeriksa pohon secara merata.
3. Anytime: MCTS adalah algoritma "anytime", yang berarti ia dapat dihentikan kapan saja dan akan memberikan solusi terbaik berdasarkan simulasi yang telah dilakukan.
4. Efektif untuk Ruang Keadaan Besar: MCTS sangat efektif untuk permainan dengan ruang keadaan yang sangat besar, seperti Go.

Sedangkan kekurangan seperti:

1. Komputasi Intensif: Meskipun efektif, MCTS memerlukan banyak simulasi untuk mendapatkan hasil yang baik, yang bisa memerlukan sumber daya komputasi yang signifikan.
2. Tidak Selalu Optimal: Terutama dalam kasus dengan batas waktu atau iterasi, MCTS mungkin tidak selalu menemukan solusi optimal.
3. Ketergantungan pada Fungsi Simulasi: Kualitas hasil MCTS sangat bergantung pada kualitas simulasi. Jika simulasi tidak mencerminkan realitas dengan baik, hasilnya mungkin kurang akurat.

5.4.2 Negamax

Negamax adalah variasi dari algoritma Minimax yang memanfaatkan simetri antara nilai maksimum untuk pemain saat ini dan nilai minimum untuk lawan. Dalam banyak permainan dua pemain dengan aturan yang simetris, setiap keadaan permainan yang baik untuk satu pemain memiliki nilai negatif yang setara untuk pemain lain. Dengan memahami hal ini, kita dapat menyederhanakan pencarian.

Dalam algoritma Minimax, kita memeriksa apakah pemain saat ini adalah pemain yang memaksimalkan atau meminimalkan dan kemudian mencari nilai maksimum atau minimum dari langkah-langkah anak. Dalam Negamax, kita hanya perlu mencari nilai maksimum karena kita selalu mengganti giliran pemain dan mengubah tanda dari evaluasi.

Berikut algoritmanya:

1. Jika kedalaman mencapai 0 atau node adalah node terminal, kembalikan fungsi evaluasi dari node tersebut.
2. Tetapkan nilai awal untuk nilai terbaik ke -infiniti.
3. Untuk setiap langkah anak yang mungkin:
4. Rekursif memanggil fungsi Negamax untuk anak tersebut dengan kedalaman berkurang dan ganti giliran pemain.
5. Ambil negatif dari nilai yang dikembalikan.
6. Jika nilai tersebut lebih besar dari nilai terbaik saat ini, perbarui nilai terbaik.
7. Kembalikan nilai terbaik.

Untuk pseudocodenya:

```
def negamax(node, depth):
    if depth == 0 or is_terminal(node):
        return evaluate(node)

    best_value = -infinity
    for child in children(node):
        value = -negamax(child, depth-1)
        best_value = max(best_value, value)

    return best_value
```

Contoh Kasus dalam permainan tic-tac-toe, jika kita menggunakan 1 untuk kemenangan 'X', -1 untuk kemenangan 'O', dan 0 untuk seri, kita dapat menggunakan Negamax untuk mengevaluasi papan. Setiap kali 'X' bermain, kita mencari nilai maksimum, dan setiap kali 'O' bermain, kita mencari nilai

minimum, tetapi dengan Negamax, kita selalu mencari nilai maksimum dengan mengganti tanda dari evaluasi.

Kelebihan Negamax

1. Sederhana: Negamax lebih sederhana dan lebih elegan daripada Minimax karena tidak perlu membedakan antara pemain yang memaksimalkan dan meminimalkan.
2. Efisien: Karena sederhananya, implementasi Negamax seringkali sedikit lebih cepat daripada Minimax.

Kekurangan Negamax

1. Tidak Selalu Lebih Baik: Meskipun lebih sederhana, Negamax tidak selalu lebih baik daripada Minimax dalam hal efisiensi, terutama jika ada optimisasi khusus yang dapat diterapkan pada Minimax.
2. Ketergantungan pada Fungsi Evaluasi: Seperti Minimax, efektivitas Negamax sangat bergantung pada kualitas dari fungsi evaluasi.

5.4.3 Iterative Deepening

Iterative Deepening adalah teknik pencarian yang menggabungkan kelebihan dari Breadth-First Search (BFS) dan Depth-First Search (DFS). Algoritma ini bekerja dengan cara menjalankan DFS berulang-ulang dengan kedalaman yang meningkat secara bertahap.

Pada setiap iterasi, IDDFS membatasi kedalaman yang diperbolehkan untuk DFS. Jika solusi tidak ditemukan dalam batas kedalaman saat ini, kedalaman ditingkatkan dan pencarian diulang. Proses ini berlanjut hingga solusi ditemukan atau semua node telah dieksplorasi.

Algoritma

1. Tetapkan kedalaman awal d ke 0.

2. Ulangi hingga solusi ditemukan atau semua node telah dieksplorasi:
   - Jalankan DFS dengan batas kedalaman d.
   - Jika solusi ditemukan, kembalikan solusi tersebut.
   - Tingkatkan d dengan 1.

Pseudocode:

```
def IDDFS(root):
    depth = 0
    while True:
        result = DLS(root, depth)
        if result is not None:
            return result
        depth += 1

def DLS(node, depth):
    if depth == 0 and is_goal(node):
        return node
    elif depth > 0:
        for child in children(node):
            result = DLS(child, depth-1)
            if result is not None:
                return result
    return None
```

Contoh Kasus dalam permainan papan seperti catur atau tic-tac-toe, kita mungkin ingin mengeksplorasi semua kemungkinan langkah hingga kedalaman tertentu untuk mengevaluasi posisi terbaik. Menggunakan IDDFS memungkinkan kita untuk mengeksplorasi langkah-langkah dalam urutan yang mendekati urutan BFS tetapi dengan penggunaan memori yang mirip dengan DFS.

Kelebihan Iterative Deepening

1. Optimalitas: Dalam ruang pencarian dengan biaya langkah yang sama, IDDFS menjamin akan menemukan solusi yang optimal, mirip dengan BFS.
2. Penggunaan Memori: IDDFS menggunakan memori yang jauh lebih sedikit dibandingkan BFS karena ia tidak menyimpan semua node pada level tertentu dalam memori sekaligus, mirip dengan DFS.
3. Fleksibel: IDDFS dapat dihentikan kapan saja dan akan memberikan solusi terbaik hingga kedalaman tertentu.

Kekurangan Iterative Deepening

1. Overhead: Meskipun efisien dalam hal ruang, IDDFS mungkin memerlukan lebih banyak waktu dibandingkan dengan BFS atau DFS murni karena ia menjelajahi node yang sama berulang kali.
2. Tidak Efisien untuk Beberapa Jenis Pohon: Pada pohon dengan banyak cabang dan kedalaman yang terbatas, DFS murni mungkin lebih cepat.

5.5 Implementasi

5.5.1 Pemilihan Bahasa Pemrograman

Bahasa pemrograman adalah alat yang digunakan oleh pengembang untuk menulis kode dan menciptakan aplikasi atau sistem. Pemilihan bahasa pemrograman yang tepat dapat mempengaruhi keberhasilan, efisiensi, dan skalabilitas dari solusi yang diimplementasikan.

Faktor-faktor dalam Pemilihan Bahasa Pemrograman

1. Kebutuhan Proyek: Apa yang ingin Anda capai? Misalnya, jika Anda ingin mengembangkan aplikasi web, Anda mungkin akan memilih JavaScript, Python (Django, Flask),

atau Ruby on Rails. Untuk pengembangan aplikasi mobile, Anda mungkin memilih Swift (iOS) atau Kotlin (Android).

2. Efisiensi: Beberapa bahasa, seperti C atau C++, dikenal memiliki kinerja yang tinggi dan sering digunakan dalam aplikasi yang memerlukan efisiensi maksimal, seperti permainan atau sistem real-time.
3. Ketersediaan Pustaka: Bahasa seperti Python memiliki ekosistem pustaka yang luas yang dapat mempercepat pengembangan dengan menyediakan solusi siap pakai untuk berbagai masalah.
4. Komunitas: Bahasa dengan komunitas yang aktif dan besar biasanya memiliki lebih banyak dukungan, tutorial, dan pustaka yang tersedia.
5. Portabilitas: Jika Anda ingin aplikasi Anda berjalan di berbagai platform, Anda mungkin memilih bahasa yang mendukung multi-platform, seperti Java atau Python.
6. Pengalaman Tim: Jika tim Anda sudah familiar dengan bahasa tertentu, mungkin lebih masuk akal untuk tetap menggunakan bahasa tersebut daripada mempelajari yang baru.
7. Tren dan Masa Depan: Memilih bahasa yang sedang naik daun atau memiliki prospek masa depan yang cerah dapat memastikan bahwa aplikasi Anda tetap relevan dan mudah dikelola dalam jangka panjang.

Python adalah salah satu bahasa pemrograman yang populer dan sering digunakan dalam berbagai bidang, termasuk dalam kecerdasan buatan dan pencarian adversarial. Mari kita bahas kelebihan, kekurangan, dan penerapan Python dalam konteks pencarian adversarial

Kelebihan Python dalam Pencarian Adversarial:

1. Pustaka yang Kaya: Python memiliki berbagai pustaka dan kerangka kerja yang mendukung kecerdasan buatan dan

pencarian adversarial, seperti TensorFlow, PyTorch, dan scikit-learn.

2. Sintaks yang Mudah Dibaca: Python dikenal dengan sintaksnya yang jelas dan mudah dibaca, yang memudahkan pengembangan dan pemeliharaan kode.
3. Komunitas yang Besar: Dengan komunitas yang besar dan aktif, Anda dapat dengan mudah menemukan sumber daya, tutorial, dan dukungan untuk masalah yang berkaitan dengan pencarian adversarial.
4. Portabilitas: Python adalah bahasa pemrograman yang multi-platform, yang berarti kode yang Anda tulis dapat dijalankan di berbagai sistem operasi dengan sedikit atau tanpa modifikasi.
5. Integrasi dengan Bahasa Lain: Python dapat dengan mudah diintegrasikan dengan bahasa pemrograman lain seperti C atau C++, memungkinkan pengembang untuk memaksimalkan kinerja di bagian-bagian kritis dari kode.

Kekurangan Python dalam Pencarian Adversarial:

1. Kinerja: Meskipun Python mudah digunakan dan memiliki pustaka yang kaya, ia mungkin tidak secepat bahasa pemrograman lain seperti C++ dalam hal kinerja mentah. Namun, ini seringkali dapat diatasi dengan menggunakan pustaka yang dioptimalkan atau integrasi dengan kode C/C++.
2. Penggunaan Memori: Python mungkin memerlukan lebih banyak memori dibandingkan dengan bahasa pemrograman lain, yang bisa menjadi pertimbangan jika Anda bekerja dengan keterbatasan sumber daya.

Penerapan Python dalam Pencarian Adversarial:

Python sering digunakan dalam implementasi algoritma pencarian adversarial seperti Minimax, Alpha-Beta Pruning, dan Monte Carlo Tree Search. Misalnya, dalam pengembangan bot

permainan catur atau Go, Python dapat digunakan untuk mengimplementasikan logika permainan, evaluasi papan, dan algoritma pencarian.

Sebagai contoh, jika Anda adalah seorang peneliti di bidang keamanan siber yang tertarik dengan pencarian adversarial, Anda mungkin menggunakan Python untuk mengembangkan model pembelajaran mesin yang dapat menangani serangan adversarial. Pustaka seperti TensorFlow atau PyTorch memiliki alat dan teknik untuk membantu dalam pengembangan dan evaluasi model dalam konteks adversarial.

Python adalah pilihan yang baik untuk pengembangan dalam konteks pencarian adversarial karena kemudahannya, pustaka yang mendukung, dan komunitas yang aktif. Namun, penting untuk mempertimbangkan kebutuhan spesifik proyek Anda, terutama jika kinerja dan efisiensi adalah prioritas utama.

5.5.2 Library yang Relevan

Dalam konteks pencarian adversarial, terutama yang berkaitan dengan kecerdasan buatan dan pembelajaran mesin, ada beberapa pustaka Python yang relevan dan sering digunakan. Berikut adalah beberapa di antaranya:

1. TensorFlow dan TensorFlow Adversarial:
   Deskripsi: TensorFlow adalah pustaka pembelajaran mesin open-source yang dikembangkan oleh Google. TensorFlow Adversarial adalah ekstensi dari TensorFlow yang menyediakan alat untuk pembelajaran mesin adversarial.
   Relevansi: Dengan TensorFlow Adversarial, Anda dapat melatih model yang tahan terhadap serangan adversarial dan juga dapat menghasilkan input adversarial untuk menguji model lain.
2. Keras:

Deskripsi: Keras adalah API pembelajaran mesin tingkat tinggi yang ditulis dalam Python, berjalan di atas TensorFlow, CNTK, atau Theano.

Relevansi: Meskipun Keras sendiri bukan khusus untuk pencarian adversarial, ia sering digunakan dalam kombinasi dengan pustaka lain untuk membangun dan melatih model yang tahan terhadap serangan adversarial.

3. CleverHans:

Deskripsi: CleverHans adalah pustaka untuk menguji kinerja model pembelajaran mesin dalam menghadapi serangan adversarial.

Relevansi: CleverHans menyediakan implementasi dari berbagai serangan adversarial dan metode pertahanan, memungkinkan peneliti untuk dengan mudah membandingkan keefektifan berbagai teknik.

4. Adversarial Robustness Toolbox (ART):

Deskripsi: ART adalah pustaka Python yang mendukung pembuatan model pembelajaran mesin yang tahan terhadap serangan adversarial dan membantu peneliti dalam evaluasi keamanan model.

Relevansi: ART menyediakan alat untuk serangan dan pertahanan, serta metode untuk evaluasi keamanan model pembelajaran mesin.

5. PyTorch:

Deskripsi: PyTorch adalah pustaka pembelajaran mesin open-source yang dikembangkan oleh Facebook's AI Research lab.

Relevansi: Seperti TensorFlow, PyTorch juga sering digunakan dalam konteks pembelajaran mesin adversarial. Meskipun tidak memiliki ekstensi khusus untuk adversarial seperti TensorFlow, PyTorch memiliki fleksibilitas yang memungkinkan peneliti untuk dengan mudah mengimplementasikan teknik adversarial.

6. Foolbox:

Deskripsi: Foolbox adalah pustaka Python untuk menghasilkan serangan adversarial terhadap model pembelajaran mesin.
Relevansi: Foolbox menyediakan berbagai metode serangan yang dapat digunakan untuk menguji ketahanan model terhadap serangan adversarial.

5.5.3 Contoh Kasus: Implementasi pada Permainan Tic-Tac-Toe dengan Minimax

```
class TicTacToe:
    def __init__(self):
        self.board = [' ' for _ in range(9)]  # inisialisasi papan
kosong
        self.current_winner = None

    def print_board(self):
        for row in [self.board[i*3:(i+1)*3] for i in range(3)]:
            print('| ' + ' | '.join(row) + ' |')

    def available_moves(self):
        return [i for i, x in enumerate(self.board) if x == " "]

    def empty_squares(self):
        return ' ' in self.board

    def make_move(self, square, letter):
        if self.board[square] == ' ':
            self.board[square] = letter
            if self.winner(square, letter):
                self.current_winner = letter
            return True
        return False

    def winner(self, square, letter):
        row_index = square // 3
```

```
        row = self.board[row_index*3:(row_index+1)*3]
        if all([s == letter for s in row]):
            return True
        col_index = square % 3
        col = [self.board[col_index+i*3] for i in range(3)]
        if all([s == letter for s in col]):
            return True
        if square % 2 == 0:
            diagonal1 = [self.board[i] for i in [0, 4, 8]]
            if all([s == letter for s in diagonal1]):
                return True
            diagonal2 = [self.board[i] for i in [2, 4, 6]]
            if all([s == letter for s in diagonal2]):
                return True
        return False

def minimax(board, depth, is_maximizing):
    if board.current_winner == 'O':
        return {'position': None, 'score': 1 * (depth + 1)}
    elif board.current_winner == 'X':
        return {'position': None, 'score': -1 * (depth + 1)}
    elif not board.empty_squares():
        return {'position': None, 'score': 0}

    if is_maximizing:
        best = {'position': None, 'score': float('-inf')}
        for possible_move in board.available_moves():
            board.make_move(possible_move, 'O')
            sim_score = minimax(board, depth-1, False)
            board.board[possible_move] = ' '
            board.current_winner = None
            sim_score['position'] = possible_move
            if sim_score['score'] > best['score']:
                best = sim_score
        return best
```

```
    else:
        best = {'position': None, 'score': float('inf')}
        for possible_move in board.available_moves():
            board.make_move(possible_move, 'X')
            sim_score = minimax(board, depth-1, True)
            board.board[possible_move] = ' '
            board.current_winner = None
            sim_score['position'] = possible_move
            if sim_score['score'] < best['score']:
                best = sim_score
        return best

# Main game loop
board = TicTacToe()
while board.empty_squares():
    board.print_board()
    move = int(input('Enter your move (0-8): '))
    board.make_move(move, 'X')
    if board.current_winner:
        print('X wins!')
        break
    best_move = minimax(board, len(board.available_moves()),
True)
    board.make_move(best_move['position'], 'O')
    if board.current_winner:
        print('O wins!')
        break
```

Jika Anda menjalankan program tic-tac-toe dengan algoritma Minimax yang telah diberikan, berikut adalah skenario yang mungkin terjadi:

- Papan permainan kosong akan ditampilkan.
- Anda (sebagai pemain 'X') akan diminta untuk memasukkan langkah Anda dengan memilih angka antara 0-8 (mewakili posisi pada papan).

- Setelah Anda memasukkan langkah Anda, komputer (sebagai pemain 'O') akan memikirkan langkah terbaiknya menggunakan algoritma Minimax dan membuat langkahnya.
- Papan permainan akan diperbarui dan ditampilkan lagi.
- Proses ini akan berlanjut sampai salah satu pemain menang atau semua kotak di papan telah diisi (seri).

Analisis Performa:

1. Kecepatan: Algoritma Minimax akan mengevaluasi setiap kemungkinan langkah, yang berarti dalam kasus terburuk (pada awal permainan) ia akan mengevaluasi 9! (362,880) kemungkinan. Namun, dengan sifat permainan tic-tac-toe, banyak dari langkah-langkah ini tidak perlu dievaluasi karena permainan mungkin sudah berakhir sebelum papan penuh.
2. Optimalitas: Algoritma Minimax akan selalu menemukan langkah terbaik, sehingga komputer (dalam hal ini 'O') akan selalu bermain optimal dan tidak mungkin dikalahkan.
3. Efisiensi: Meskipun Minimax adalah algoritma yang optimal, ia mungkin tidak efisien untuk permainan dengan ruang keadaan yang lebih besar. Untuk tic-tac-toe, ini dapat diterima karena ruang keadaan terbatas.
4. Peningkatan: Dengan menggunakan teknik seperti Alpha-Beta Pruning, kita dapat meningkatkan efisiensi algoritma dengan memotong beberapa cabang yang tidak perlu dievaluasi.

Dalam konteks permainan sederhana seperti tic-tac-toe, Python dengan algoritma Minimax mampu memberikan performa yang sangat baik. Namun, untuk permainan yang lebih kompleks dengan ruang keadaan yang lebih besar, pertimbangan lain mungkin diperlukan untuk memastikan efisiensi dan respons waktu yang cepat. Analisis Performa

5.6 Optimasi

5.6.1 Paralelisasi

Paralelisasi adalah teknik yang digunakan untuk meningkatkan kinerja dan efisiensi komputasi dengan membagi tugas menjadi sub-tugas yang lebih kecil yang dapat dijalankan secara bersamaan. Ini memanfaatkan kemampuan komputer modern yang memiliki banyak inti (cores) untuk menjalankan operasi secara paralel.

Dalam konteks komputasi, paralelisasi mengacu pada pemrosesan simultan dari berbagai bagian dari suatu tugas. Tujuannya adalah untuk mempercepat komputasi, memanfaatkan sumber daya perangkat keras sepenuhnya, dan meningkatkan throughput.

Dasar Teori

- Data Parallelism: Memecah dataset menjadi subset yang lebih kecil dan memproses setiap subset secara paralel. Misalnya, saat melakukan operasi pada array besar, setiap inti mungkin bertanggung jawab untuk mengolah bagian dari array tersebut.
- Task Parallelism: Memecah tugas menjadi fungsi atau sub-tugas yang berbeda yang dapat dijalankan secara paralel. Misalnya, dalam aplikasi grafik, satu tugas mungkin bertanggung jawab untuk rendering sementara tugas lain mungkin bertanggung jawab untuk deteksi kolisi.

Algoritma

1. Pembagian Tugas: Tentukan apakah tugas Anda lebih cocok untuk data paralelisme, tugas paralelisme, atau kombinasi keduanya. Bagilah tugas atau data Anda menjadi unit yang lebih kecil yang dapat dijalankan secara paralel.
2. Pengalokasian Sumber Daya: Tentukan berapa banyak inti atau thread yang Anda miliki dan alokasikan sub-tugas atau subset data ke setiap inti atau thread.

3. Sikronisasi: Pastikan bahwa semua thread atau proses berbagi sumber daya dengan benar dan tidak mengakses data secara bersamaan yang dapat menyebabkan kondisi balapan (race condition). Ini mungkin memerlukan penggunaan semaphores, locks, atau mekanisme sikronisasi lainnya.
4. Aggregasi Hasil: Setelah semua thread atau proses selesai, kumpulkan dan gabungkan hasil mereka menjadi solusi akhir.

Kelebihan Paralelisasi

- Peningkatan Kinerja: Dengan menjalankan tugas secara paralel, Anda dapat mempercepat waktu eksekusi dan meningkatkan throughput.
- Efisiensi Sumber Daya: Paralelisasi memungkinkan Anda memanfaatkan semua inti dan sumber daya perangkat keras yang tersedia.

Kekurangan Paralelisasi

- Kompleksitas: Mengimplementasikan algoritma paralel mungkin lebih kompleks daripada versi sekuensialnya.
- Overhead: Membuat, mengelola, dan sikronisasi thread atau proses dapat menambah overhead.
- Kondisi Balapan: Akses bersamaan ke sumber daya bersama dapat menyebabkan kondisi balapan, yang dapat menyebabkan perilaku yang tidak terduga atau kesalahan.

Contoh kasusnya, misalkan Anda ingin menghitung jumlah dari elemen dalam array besar. Daripada menjumlahkan elemen satu per satu (sekuensial), Anda dapat membagi array menjadi bagian yang lebih kecil dan menjumlahkan setiap bagian secara paralel. Kemudian, Anda dapat menggabungkan hasil dari setiap bagian untuk mendapatkan jumlah total.

5.6.2 Heuristik

Heuristik adalah suatu pendekatan yang digunakan untuk mencari solusi yang memadai atau mendekati optimal untuk

suatu masalah, terutama ketika metode eksak memerlukan waktu yang terlalu lama. Heuristik sering digunakan dalam konteks pencarian, optimasi, dan pengambilan keputusan.

Dalam banyak situasi, mencari solusi yang sempurna atau optimal mungkin tidak praktis karena keterbatasan waktu atau sumber daya. Dalam kasus-kasus seperti itu, heuristik dapat memberikan solusi yang "cukup baik" dalam waktu yang jauh lebih singkat.

Dasar Teori

Definisi: Heuristik adalah teknik yang dirancang untuk memecahkan masalah lebih cepat daripada metode tradisional, dengan menerima solusi yang mendekati optimal atau memadai.

Trade-off: Dengan menggunakan heuristik, kita sering kali mengorbankan kepastian mendapatkan solusi optimal demi mendapatkan solusi yang cepat dan memadai.

Aplikasi: Heuristik digunakan dalam berbagai bidang, termasuk kecerdasan buatan, psikologi, ekonomi, dan banyak bidang lainnya.

Jenis-Jenis Heuristik

- Heuristik Kesederhanaan: Menggunakan informasi yang mudah diakses atau yang paling menonjol untuk membuat keputusan, tanpa mempertimbangkan semua informasi yang tersedia.
- Heuristik Ketersediaan: Mengandalkan informasi atau peristiwa yang mudah diingat atau tampak "segar" dalam pikiran saat membuat keputusan.
- Heuristik Representatif: Membuat keputusan berdasarkan seberapa mirip suatu situasi dengan prototipe atau contoh yang ada dalam pikiran seseorang.

- Heuristik Jangkar dan Penyesuaian: Memulai dari suatu titik referensi atau "jangkar" dan kemudian membuat penyesuaian berdasarkan informasi tambahan.

Kelebihan Heuristik

- Kecepatan: Heuristik sering kali memungkinkan solusi untuk ditemukan dalam waktu yang jauh lebih singkat dibandingkan dengan metode eksak.
- Fleksibilitas: Heuristik dapat disesuaikan dengan berbagai masalah dan situasi.
- Efektivitas: Dalam banyak kasus, heuristik dapat menghasilkan solusi yang memadai atau bahkan mendekati optimal.

Kekurangan Heuristik

- Ketidakpastian: Tidak ada jaminan bahwa solusi yang dihasilkan oleh heuristik adalah yang terbaik atau optimal.
- Bias: Beberapa heuristik dapat mengarahkan pengambil keputusan ke solusi yang bias atau suboptimal.
- Ketergantungan pada Pengalaman: Efektivitas heuristik sering kali bergantung pada pengalaman dan pengetahuan individu yang menggunakannya.

Contoh kasus dalam bidang keamanan siber, heuristik mungkin digunakan dalam sistem deteksi intrusi untuk mengidentifikasi aktivitas mencurigakan. Mengingat volume data yang besar dan kebutuhan untuk deteksi real-time, pendekatan heuristik dapat membantu dalam mengidentifikasi potensi ancaman dengan cepat, meskipun mungkin ada risiko positif palsu atau negatif palsu.

Dalam konteks pencarian dan optimasi, heuristik sering digunakan dalam algoritma seperti pencarian berpandu (guided search), algoritma genetika, dan optimasi koloni semut untuk mencari solusi yang mendekati optimal dalam waktu yang lebih singkat.

5.6.3 Pembelajaran Mesin dalam Pencarian Adversarial

Pembelajaran mesin dalam konteks pencarian adversarial merujuk pada situasi di mana model pembelajaran mesin dihadapkan pada upaya bermusuhan yang bertujuan untuk menipu atau mengelabui model tersebut. Konsep ini telah menjadi area riset yang penting, terutama dengan meningkatnya penerapan pembelajaran mesin dalam berbagai sektor dan potensi risiko keamanan yang terkait.

Dalam banyak aplikasi pembelajaran mesin, model dilatih untuk membuat prediksi atau klasifikasi berdasarkan data input. Namun, apa yang terjadi jika data input sengaja dimanipulasi dengan cara yang sulit dideteksi untuk mengelabui model? Inilah inti dari pencarian adversarial dalam pembelajaran mesin.

Serangan adversarial adalah upaya untuk memanipulasi input ke model pembelajaran mesin dengan tujuan membuat model membuat kesalahan, sementara perubahan pada input tetap tidak terdeteksi oleh manusia. Sebagai contoh, gambar kucing yang dimodifikasi sedikit mungkin diklasifikasikan sebagai truk oleh model pembelajaran mesin, meskipun perubahan tersebut tidak terlihat oleh mata manusia.

Ada beberapa metode untuk menghasilkan contoh adversarial:

- Metode Gradient-based: Menggunakan gradien dari model untuk menentukan bagaimana mengubah input agar menghasilkan kesalahan klasifikasi.
- Serangan Transfer: Contoh adversarial yang dibuat untuk satu model terkadang efektif terhadap model lain, meskipun model tersebut memiliki arsitektur atau pelatihan yang berbeda.

Meskipun ada banyak teknik serangan, ada juga berbagai strategi pertahanan:

- Pelatihan Adversarial: Salah satu pendekatan paling umum adalah melatih model dengan contoh adversarial, sehingga model menjadi tahan terhadap serangan tersebut.

- Regularisasi dan Modifikasi Model: Mengubah arsitektur atau fungsi kerugian model untuk membuatnya lebih tahan terhadap serangan.
- Deteksi Contoh Adversarial: Mencoba mendeteksi dan menolak input yang tampaknya telah dimodifikasi.

Pencarian adversarial memiliki implikasi besar dalam keamanan siber:

- Keamanan Sistem: Model pembelajaran mesin yang digunakan dalam sistem keamanan, seperti deteksi intrusi atau analisis malware, dapat diserang untuk menghindari deteksi.
- Keandalan Sistem Otomatis: Sistem otomatis, seperti kendaraan otonom, yang bergantung pada pembelajaran mesin untuk pengambilan keputusan dapat menjadi target serangan adversarial untuk mengakibatkan perilaku yang tidak diinginkan.

Pembelajaran mesin dalam konteks pencarian adversarial menyoroti pentingnya memahami dan memitigasi potensi kelemahan dalam model pembelajaran mesin. Seiring dengan pertumbuhan dan penerapan teknologi pembelajaran mesin, penting bagi peneliti dan praktisi untuk mempertimbangkan risiko dan tantangan yang terkait dengan serangan adversarial dan mengembangkan strategi untuk menghadapinya.

5.7 Kesimpulan

Pencarian adversarial dalam konteks pembelajaran mesin merujuk pada upaya bermusuhan untuk mengelabui model dengan memanipulasi data input. Tujuannya adalah untuk membuat model membuat kesalahan prediksi atau klasifikasi tanpa mengubah esensi sebenarnya dari data tersebut, setidaknya menurut persepsi manusia. Dalam banyak kasus, perubahan kecil pada input, yang mungkin tidak terlihat oleh mata manusia, dapat menyebabkan model membuat kesalahan

besar. Hal ini menimbulkan kekhawatiran serius tentang keandalan dan keamanan model pembelajaran mesin, terutama dalam aplikasi yang kritis.

Tantangan Kedepan:

- Keamanan Model: Seiring dengan meningkatnya serangan adversarial, ada kebutuhan mendesak untuk mengembangkan model yang tahan terhadap serangan ini. Meskipun ada beberapa teknik pertahanan yang telah diajukan, menemukan solusi yang efektif dan efisien tetap menjadi tantangan.
- Deteksi Serangan: Mengidentifikasi apakah suatu input telah dimodifikasi sebagai bagian dari serangan adversarial adalah langkah kunci dalam memitigasi dampaknya. Namun, mendeteksi perubahan mikro yang dirancang untuk mengelabui model tetap menjadi tantangan besar.
- Generalisasi Pertahanan: Meskipun mungkin untuk melatih model untuk tahan terhadap serangan adversarial tertentu, menjamin bahwa model tersebut tahan terhadap semua jenis serangan adalah masalah yang lebih kompleks.
- Penerapan Real-World: Banyak penelitian tentang pencarian adversarial dilakukan dalam pengaturan eksperimental. Menerapkan temuan ini ke aplikasi dunia nyata, dengan semua variabilitas dan kompleksitasnya, adalah tantangan tersendiri.
- Pengertian Manusia vs. Mesin: Salah satu aspek unik dari serangan adversarial adalah bahwa perubahan yang dibuat pada data seringkali tidak terlihat oleh manusia. Memahami perbedaan persepsi antara manusia dan mesin, dan bagaimana hal itu dapat dieksploitasi, adalah area riset yang penting.
- Evaluasi Biaya: Dalam beberapa kasus, biaya untuk melindungi model dari serangan adversarial mungkin melebihi potensi kerugian dari serangan tersebut. Memahami trade-off ini dan kapan harus menerapkan pertahanan adalah tantangan strategis.

Dengan meningkatnya penerapan pembelajaran mesin dalam berbagai sektor industri dan kehidupan sehari-hari, pentingnya memahami dan mengatasi tantangan-tantangan yang terkait dengan pencarian adversarial akan terus meningkat.

DAFTAR PUSTAKA

Browne, C.B. *et al.* (2012) 'A Survey of Monte Carlo Tree Search Methods', *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1), pp. 1–43. Available at: https://doi.org/10.1109/TCIAIG.2012.2186810.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. The MIT Press.

Kochenderfer, M.J. (2015) *Decision Making Under Uncertainty*. The MIT Press. Available at: https://doi.org/10.7551/mitpress/10187.001.0001.

Lye, K. and Wing, J.M. (2005) 'Game strategies in network security', *International Journal of Information Security*, 4(1–2), pp. 71–86. Available at: https://doi.org/10.1007/s10207-004-0060-x.

Mitchell, T.M. (1997) *Machine Learning*. 1st Ed. McGraw-Hill Education.

Osborne, M.J. and Rubinstein, A. (1994) *A Course in Game Theory*. The MIT Press.

Papernot, N. *et al.* (2016) 'Practical Black-Box Attacks against Machine Learning', in *Proceedings of the 2016 ACM on Asia Conference on Computer and Communications Security*. Available at: https://doi.org/10.48550/arXiv.1602.02697.

Russell, S. and Norvig, P. (2009) *Artificial Intelligence: A Modern Approach*. 3rd edn. Pearson.

Sutton, R.S. and Barto, A.G. (2018) *Reinforcement Learning: An Introduction*. 2nd edn. MIT Press, Cambridge.

BAB 6
Representasi Pengetahuan

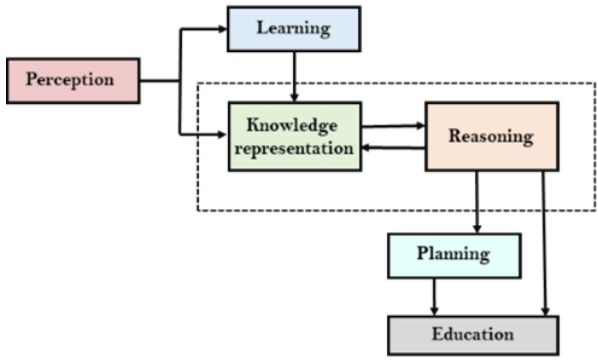
Oleh Rudi Sutomo

6.1 Teori Representasi Pengetahuan

6.1.1 Konsep Dasar Representasi Pengetahuan

1. Definisi Representasi Pengetahuan

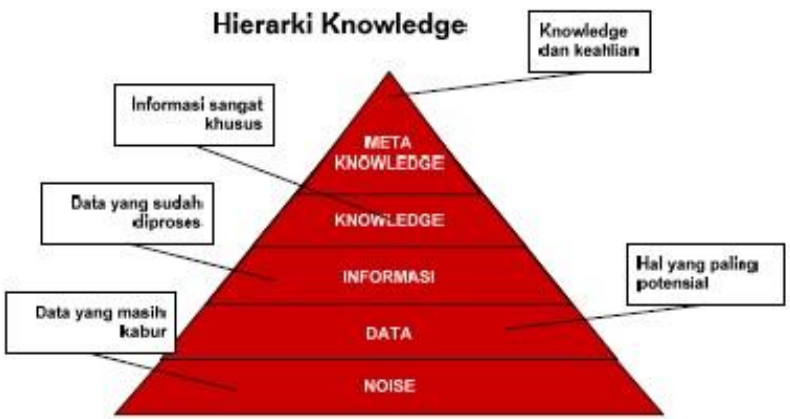
   Pengertian Dasar Representasi Pengetahuan adalah konsep fundamental yang mengacu pada cara informasi, pengetahuan, atau data direpresentasikan dalam bentuk yang dapat dipahami dan digunakan oleh komputer atau sistem kecerdasan buatan. Representasi Pengetahuan melibatkan pengkodean pengetahuan dalam struktur yang memungkinkan komputer untuk memproses, menganalisis, dan menggunakan informasi tersebut untuk berbagai tujuan.



Gambar 6.1 Hierarki Pengetahuan
(Sumber : https://www.javatpoint.com/knowledge-representation-in-ai)

Dalam Representasi Pengetahuan, informasi seringkali diubah menjadi simbol-simbol atau struktur data yang memungkinkan komputer untuk mengidentifikasi relasi, makna, dan konteks dari informasi tersebut. Representasi Pengetahuan menjadi dasar bagi sistem kecerdasan buatan untuk melakukan tugas seperti pemrosesan bahasa alami, pengambilan keputusan, perencanaan, dan pemecahan masalah.

Pentingnya Representasi Pengetahuan adalah bahwa tanpa cara yang tepat untuk merepresentasikan pengetahuan, komputer akan kesulitan dalam memahami dunia sekitarnya. Representasi Pengetahuan adalah suatu teknik untuk merepresentasikan basis pengetahuan yang diperoleh ke dalam suatu skema/diagram tertentu sehingga dapat diketahui relasi/keterhubungan antara suatu data dengan data yang lain(Sutojo, T; Mulyanto, Edi; Suhartono, 2011).



Gambar 6.2 Hierarki Pengetahuan
(Sumber :Mubarak, 1991)

2. Peran Representasi Pengetahuan dalam Kecerdasan Buatan

Peran Representasi Pengetahuan dalam Kecerdasan Buatan sangat penting karena merupakan salah satu aspek kunci yang memungkinkan sistem kecerdasan buatan untuk berfungsi dengan cerdas dan efektif. Berikut adalah beberapa peran penting Representasi Pengetahuan dalam Kecerdasan Buatan:

- Pemahaman Dunia: Representasi Pengetahuan memungkinkan komputer untuk memahami dunia di sekitarnya. Dengan mewakili pengetahuan tentang objek, konsep, hubungan, dan konteks dalam bentuk yang dapat diproses, komputer dapat mengenali pola, makna, dan relasi antara informasi yang diberikan.
- Pemrosesan Bahasa Alami: Dalam pemrosesan bahasa alami, Representasi Pengetahuan membantu dalam memahami teks dan ucapan manusia. Ini melibatkan penerjemahan kata-kata, frasa, dan kalimat menjadi representasi yang dapat digunakan untuk merumuskan pertanyaan, menjawab pertanyaan, atau melakukan tugas seperti penerjemahan bahasa.
- Pengambilan Keputusan: Sistem kecerdasan buatan sering kali harus membuat keputusan berdasarkan informasi yang tersedia. Representasi Pengetahuan membantu dalam memodelkan pengetahuan yang dibutuhkan untuk mengambil keputusan yang cerdas. Ini dapat mencakup pengetahuan aturan, pengetahuan statistik, atau pengetahuan domain yang khusus.
- Pemecahan Masalah: Dalam pemecahan masalah, Representasi Pengetahuan digunakan untuk menggambarkan masalah yang dihadapi dan solusi yang mungkin. Ini membantu sistem kecerdasan buatan dalam mengidentifikasi solusi yang paling sesuai dengan masalah yang dihadapi.
- Perencanaan dan Pengaturan: Dalam perencanaan tindakan atau pengaturan tugas, Representasi Pengetahuan digunakan untuk merencanakan urutan

langkah-langkah yang diperlukan dan memahami konsekuensi dari tindakan-tindakan tersebut.

- Pengenalan Pola: Dalam pengenalan pola, Representasi Pengetahuan digunakan untuk menggambarkan ciri-ciri atau pola yang perlu diidentifikasi dalam data, seperti pengenalan wajah, suara, atau objek dalam gambar.
- Pengolahan Data: Representasi Pengetahuan memungkinkan komputer untuk mengelola data dengan lebih efisien dan mengidentifikasi informasi yang relevan dalam set data yang besar.
- Rekomendasi: Dalam sistem rekomendasi, Representasi Pengetahuan membantu dalam memahami preferensi dan profil pengguna, yang kemudian digunakan untuk memberikan rekomendasi yang sesuai.
- Penelitian dan Pembelajaran: Dalam konteks penelitian dan pembelajaran, Representasi Pengetahuan memungkinkan sistem kecerdasan buatan untuk menyimpan dan mengakses pengetahuan yang diperlukan untuk pengembangan model dan algoritma yang lebih canggih.
- Interaksi Manusia-Mesin: Dalam interaksi antara manusia dan mesin, Representasi Pengetahuan memungkinkan komputer untuk merespons pertanyaan, instruksi, dan permintaan manusia dengan pemahaman yang lebih baik.

Secara keseluruhan, Representasi Pengetahuan adalah fondasi yang kuat untuk pengembangan sistem kecerdasan buatan yang mampu berinteraksi dengan dunia nyata, memecahkan masalah, dan membuat keputusan yang cerdas berdasarkan pengetahuan yang mereka miliki. Tanpa representasi yang tepat, komputer akan kesulitan dalam memahami, menyimpan, dan memanfaatkan pengetahuan untuk tugas-tugas berbasis intelektual.

Gambar 6.3 Tipe Pengetahuan
(Sumber :https://www.javatpoint.com/knowledge-representation-in-ai)

6.1.2 Hubungan antara Bahasa Natural dan Representasi Pengetahuan

Hubungan antara Bahasa Natural dan Representasi Pengetahuan sangat erat dalam konteks kecerdasan buatan dan pemrosesan bahasa alami. Bahasa Natural adalah alat komunikasi utama manusia, dan Representasi Pengetahuan adalah cara sistem kecerdasan buatan memahami dan mengelola informasi dari bahasa manusia. Berikut adalah beberapa aspek penting dari hubungan ini:

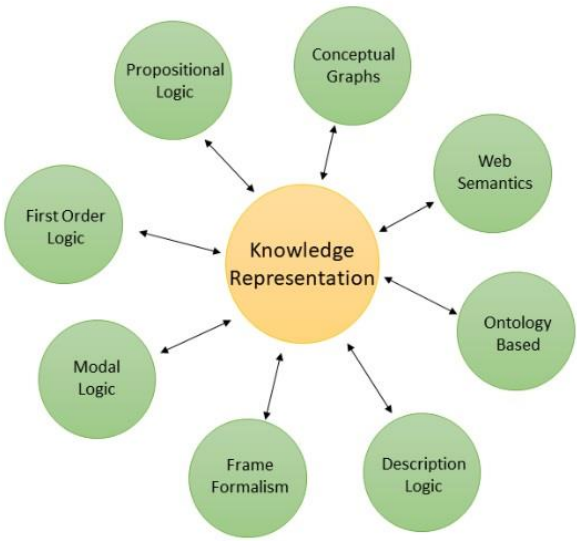
1. Pemrosesan Bahasa Natural: Bahasa Natural adalah cara utama manusia berkomunikasi, dan sistem kecerdasan buatan perlu memahami bahasa manusia agar dapat berkomunikasi dan bekerja sama dengan manusia. Representasi Pengetahuan berperan dalam menerjemahkan,

mengurai, dan memproses bahasa manusia, sehingga komputer dapat mengerti apa yang manusia katakan atau tulis.

2. Representasi Pengetahuan dalam Bahasa: Bahasa manusia mengandung banyak pengetahuan dan makna yang terkandung dalam kata-kata dan kalimat. Representasi Pengetahuan membantu dalam mengubah bahasa manusia menjadi representasi internal yang dapat dimengerti oleh komputer. Ini mencakup pengenalan entitas, hubungan, dan konteks dalam bahasa.
3. Pengenalan Entitas dan Konsep: Representasi Pengetahuan membantu dalam mengidentifikasi entitas (seperti nama orang, tempat, dan objek) serta konsep yang disebutkan dalam bahasa. Hal ini membantu dalam membangun pengetahuan yang lebih kaya tentang informasi yang diberikan.
4. Pertanyaan dan Jawaban: Dalam pemrosesan bahasa alami, orang sering mengajukan pertanyaan dan mengharapkan jawaban. Representasi Pengetahuan digunakan untuk memahami pertanyaan dan mencari jawaban yang sesuai dalam pengetahuan yang tersedia.
5. Pemahaman Konteks: Bahasa seringkali ambigu, dan makna kata atau kalimat dapat berubah tergantung pada konteks. Representasi Pengetahuan membantu dalam memahami konteks di mana bahasa digunakan untuk menghindari kesalahan pemahaman.
6. Pembelajaran dari Bahasa: Sistem kecerdasan buatan dapat menggunakan bahasa manusia untuk memperbarui dan memperkaya pengetahuan mereka. Ini melibatkan ekstraksi pengetahuan dari teks yang luas, termasuk artikel berita, buku, atau sumber-sumber online.
7. Sistem Pemrosesan Bahasa Alami (NLP): Sistem NLP, seperti chatbot atau asisten virtual, menggunakan

Representasi Pengetahuan untuk merespons pertanyaan dan perintah bahasa manusia. Representasi Pengetahuan membantu sistem ini untuk menghasilkan tanggapan yang tepat dan relevan.

Dengan demikian, Bahasa Natural dan Representasi Pengetahuan adalah dua komponen kunci yang saling melengkapi dalam pemrosesan bahasa alami dan dalam pengembangan sistem kecerdasan buatan yang mampu berkomunikasi dengan manusia dengan lebih baik. Representasi Pengetahuan membantu menerjemahkan bahasa manusia menjadi pemahaman komputer, yang kemudian dapat digunakan untuk berbagai tugas yang melibatkan bahasa(Pustejovsky & Stubbs, 2013).



Gambar 6.4 Kategori Pengetahuan
(Sumber :https://www.javatpoint.com/knowledge-representation-in-ai)

6.1.3 Jenis-Jenis Representasi Pengetahuan

1. Representasi Pengetahuan Semantik

Representasi Pengetahuan Semantik adalah salah satu jenis representasi pengetahuan yang fokus pada makna atau semantik dari informasi yang direpresentasikan. Representasi semantik mencoba untuk menggambarkan pengetahuan dengan cara yang lebih abstrak dan berfokus pada konsep-konsep dan relasi antara mereka. Berikut adalah beberapa jenis representasi pengetahuan semantik yang umum:

a. Jaringan Semantik: Jaringan semantik adalah representasi pengetahuan yang menggunakan struktur grafik untuk menggambarkan konsep-konsep dan hubungan antara mereka. Dalam jaringan semantik, simpul (*node*) mewakili konsep atau kata, sementara tepi (*edge*) mewakili relasi antara konsep-konsep tersebut. Misalnya, dalam jaringan semantik, kata "kucing" dapat terhubung dengan kata "hewan" melalui tepi yang menunjukkan hubungan hierarkis.
b. Model Distribusi: Model distribusi (atau vektor distribusi) adalah pendekatan dalam representasi pengetahuan yang memodelkan makna kata berdasarkan konteks di mana kata tersebut muncul dalam teks. Dalam model ini, setiap kata direpresentasikan sebagai vektor numerik dalam ruang multidimensional, dan kata-kata yang sering muncul bersama-sama memiliki vektor yang mendekati satu sama lain dalam ruang tersebut. Model distribusi seperti Word2Vec dan GloVe adalah contoh pendekatan ini.
c. Ontologi: merupakan representasi formal yang digunakan untuk menggambarkan pengetahuan dalam domain tertentu. Ontologi melibatkan definisi kelas-kelas objek, konsep, dan hubungan antara mereka. Ini membantu dalam pemodelan pengetahuan yang lebih mendalam dan kompleks dalam domain khusus. Contoh ontologi adalah Ontologi Web Semantik (Semantic Web

Ontology) yang digunakan untuk menggambarkan pengetahuan dalam konteks World Wide Web.

d. Representasi Berbasis Atribut: Dalam representasi pengetahuan semantik, informasi tentang objek atau konsep dapat dijelaskan melalui atribut-atribut yang dimiliki oleh objek tersebut. Misalnya, dalam representasi pengetahuan tentang mobil, atribut-atribut seperti warna, merek, dan model mobil digunakan untuk mendeskripsikan mobil tersebut.
e. Model Semantik dalam Pemrosesan Bahasa Alami: Model semantik digunakan dalam pemrosesan bahasa alami untuk memahami makna kalimat dan teks. Ini melibatkan penerjemahan kata-kata dan kalimat menjadi representasi semantik yang dapat digunakan untuk menganalisis dan memproses teks dengan cara yang cerdas.
f. Penggalian Pengetahuan: Representasi pengetahuan semantik juga digunakan dalam penggalian pengetahuan (knowledge mining) untuk mengekstrak informasi yang bermakna dari teks atau data. Ini mencakup identifikasi entitas, pengenalan hubungan, dan pemahaman konteks.
g. Jenis-jenis representasi pengetahuan semantik ini membantu komputer dan sistem kecerdasan buatan untuk mengolah informasi dengan cara yang lebih mendalam dan berarti, karena mereka memperhitungkan makna dan konteks dalam pengetahuan yang direpresentasikan. Representasi semantik berperan penting dalam pemahaman dan pemanfaatan pengetahuan dalam berbagai aplikasi, termasuk pemrosesan bahasa alami, pengambilan keputusan, dan pemecahan masalah(Zarri, 2011)

2. Representasi Pengetahuan Berbasis Graf

Representasi Pengetahuan Berbasis Graf adalah salah satu pendekatan dalam pemodelan pengetahuan yang

menggunakan struktur grafis untuk menggambarkan konsep-konsep, objek, dan hubungan antara mereka. Dalam representasi ini, informasi direpresentasikan sebagai jaringan atau grafik, di mana simpul (*node*) mewakili konsep atau objek, dan tepi (*edge*) mewakili hubungan antara konsep-konsep tersebut. Berikut adalah beberapa poin penting tentang Representasi Pengetahuan Berbasis Graf:

a. Struktur Grafik: Representasi Pengetahuan Berbasis Graf menggunakan struktur grafis yang terdiri dari simpul dan tepi untuk menggambarkan elemen-elemen pengetahuan. Simpul mewakili konsep atau objek, sementara tepi mewakili hubungan atau relasi antara simpul-simpul tersebut.
b. Sifat Hierarkis: Grafik semantik sering memiliki sifat hierarkis di mana konsep-konsep dikelompokkan dalam tingkatan atau lapisan. Ini menciptakan struktur hirarkis yang memungkinkan untuk mengorganisir pengetahuan dalam cara yang terstruktur.
c. Contoh Jaringan Semantik: Jaringan Semantik adalah salah satu bentuk Representasi Pengetahuan Berbasis Graf yang umum. Dalam jaringan semantik, simpul mewakili kata, konsep, atau entitas, sementara tepi menghubungkan kata-kata atau konsep yang memiliki hubungan semantik seperti sinonim, antonim, atau hubungan hierarkis.
d. Aplikasi dalam Pemrosesan Bahasa Alami: Representasi Pengetahuan Berbasis Graf sering digunakan dalam pemrosesan bahasa alami. Misalnya, dalam pemahaman teks, kata-kata dalam kalimat dapat direpresentasikan sebagai simpul-simpul dalam jaringan semantik, dan hubungan antara kata-kata (misalnya, hubungan subjek-objek dalam kalimat) direpresentasikan sebagai tepi.
e. Pencarian Informasi: Representasi Pengetahuan Berbasis Graf digunakan dalam mesin pencari untuk mengindeks dan mencari dokumen atau informasi berdasarkan

hubungan antar kata kunci atau konsep. Ini memungkinkan mesin pencari untuk memberikan hasil yang lebih relevan.

f. Pemahaman Ontologi: Ontologi merupakan bentuk Representasi Pengetahuan Berbasis Graf yang lebih kompleks. Ontologi adalah representasi formal tentang kelas-kelas objek, konsep, dan hubungan antara mereka dalam suatu domain. Ontologi digunakan dalam berbagai aplikasi, termasuk pemodelan pengetahuan dalam ilmu pengetahuan, pengolahan informasi medis, dan semantik web.
g. Pengenalan Pola: Dalam pengenalan pola, Representasi Pengetahuan Berbasis Graf dapat digunakan untuk memodelkan pola atau hubungan antara objek dalam gambar atau data visual.

Keuntungan dari Representasi Pengetahuan Berbasis Graf adalah kemampuannya untuk menggambarkan hubungan dan struktur pengetahuan dengan jelas. Ini memungkinkan komputer untuk memahami konsep-konsep yang saling terkait dan melakukan tugas-tugas seperti pemrosesan bahasa alami dan pengenalan pola dengan lebih efisien(Susatyono, 2021).

3. Representasi Pengetahuan Berbasis Logika

Representasi Pengetahuan Berbasis Logika adalah pendekatan dalam pemodelan pengetahuan yang menggunakan logika formal untuk merepresentasikan fakta, aturan, dan hubungan dalam suatu domain. Representasi Pengetahuan Berbasis Logika bergantung pada logika formal, seperti logika proposisional atau logika predikat, untuk menggambarkan pengetahuan dengan cara yang lebih terstruktur dan formal. Berikut adalah beberapa poin penting tentang Representasi Pengetahuan Berbasis Logika:

a. Logika Formal: Representasi Pengetahuan Berbasis Logika menggunakan aturan logika formal untuk merepresentasikan pengetahuan. Ini termasuk logika

proposisional, yang berkaitan dengan pernyataan logis (benar atau salah), dan logika predikat, yang memungkinkan untuk merepresentasikan hubungan kompleks antara entitas dalam domain.

b. Pernyataan Logis: Fakta atau pengetahuan direpresentasikan sebagai pernyataan logis. Pernyataan ini dapat berbentuk proposisi logis yang sederhana atau pernyataan yang lebih kompleks dengan variabel, predikat, dan kuantor.
c. Aturan Inferensi: Representasi Pengetahuan Berbasis Logika sering melibatkan aturan inferensi, yang digunakan untuk menarik kesimpulan dari pengetahuan yang ada. Aturan inferensi dapat digunakan untuk memproses informasi, menjawab pertanyaan, atau memecahkan masalah.
d. Ekspresif: Representasi Pengetahuan Berbasis Logika sangat ekspresif dan dapat menggambarkan pengetahuan dalam tingkat detail yang tinggi. Ini memungkinkan untuk merepresentasikan pengetahuan yang kompleks dan hubungan yang lebih rumit dalam domain.
e. Semantik yang Jelas: Logika formal memberikan semantik yang jelas dan terdefinisi dengan baik untuk representasi pengetahuan. Ini memungkinkan komputer untuk secara akurat memahami makna dan implikasi dari pernyataan logis.
f. Ontologi: Ontologi, yang merupakan representasi formal tentang kelas-kelas objek, konsep, dan hubungan antara mereka, seringkali digunakan dalam Representasi Pengetahuan Berbasis Logika. Ontologi membantu dalam memodelkan pengetahuan dalam domain tertentu dengan lebih terstruktur.
g. Pemrosesan Bahasa Alami: Representasi Pengetahuan Berbasis Logika dapat digunakan dalam pemrosesan bahasa alami untuk memahami dan menjawab pertanyaan dalam bahasa manusia. Ini melibatkan

transformasi kalimat dalam bahasa manusia menjadi pernyataan logis yang dapat dianalisis oleh komputer.

h. Aplikasi dalam Kecerdasan Buatan: Representasi Pengetahuan Berbasis Logika digunakan dalam berbagai aplikasi kecerdasan buatan, seperti sistem ekspert, perencanaan otomatis, pemecahan masalah, dan pengambilan keputusan.

Keuntungan dari Representasi Pengetahuan Berbasis Logika adalah kemampuannya untuk merepresentasikan pengetahuan dengan tingkat formalitas dan kejelasan yang tinggi. Ini memungkinkan sistem kecerdasan buatan untuk mengambil keputusan yang logis dan dapat dipertanggungjawabkan berdasarkan pengetahuan yang mereka miliki. Namun, penggunaan logika formal juga dapat membatasi ekspresivitas dalam menggambarkan pengetahuan yang ambigu atau tidak pasti(Rosnelly, 2012).

4. Representasi Pengetahuan Berbasis Vektor

Representasi Pengetahuan Berbasis Vektor adalah pendekatan dalam pemodelan pengetahuan yang menggunakan vektor numerik untuk merepresentasikan konsep, kata, atau objek dalam bentuk yang dapat dimengerti oleh komputer. Representasi ini memanfaatkan teknik-teknik pembelajaran mesin, terutama dalam konteks pemrosesan bahasa alami dan pengolahan data. Berikut adalah beberapa poin penting tentang Representasi Pengetahuan Berbasis Vektor:

- Pendekatan Distribusi: Representasi Pengetahuan Berbasis Vektor berfokus pada pendekatan distribusi, yang mengasumsikan bahwa makna konsep atau kata-kata dapat dipahami melalui konteks di mana mereka muncul. Ini berarti bahwa konsep yang sering muncul bersama-sama dalam teks akan memiliki representasi vektor yang lebih mirip.

- Vektor Numerik: Dalam representasi ini, setiap kata, konsep, atau entitas direpresentasikan sebagai vektor numerik dalam ruang multidimensional. Setiap dimensi dalam ruang ini merepresentasikan atribut atau karakteristik tertentu dari entitas tersebut.
- Model Distribusi: Model distribusi, seperti Word2Vec, GloVe, atau FastText, adalah contoh representasi pengetahuan berbasis vektor yang sangat populer. Model-model ini menggunakan teks yang besar sebagai data pelatihan untuk mempelajari vektor representasi kata-kata.
- Kemiripan Semantik: Dalam ruang vektor, kemiripan semantik antara kata-kata atau konsep-konsep dapat diukur dengan mengukur kedekatan vektor mereka. Kata-kata atau konsep yang memiliki representasi vektor yang serupa cenderung memiliki makna yang mirip atau berkaitan.
- Penggunaan dalam Pemrosesan Bahasa Alami: Representasi Pengetahuan Berbasis Vektor digunakan secara luas dalam pemrosesan bahasa alami untuk tugas-tugas seperti pemahaman makna kata, penerjemahan mesin, analisis sentimen, dan klasifikasi teks. Dengan menerjemahkan kata-kata menjadi vektor, komputer dapat memahami dan memproses bahasa manusia lebih efektif.
- Pencarian Informasi: Representasi vektor juga digunakan dalam mesin pencari untuk mengindeks dan mencari dokumen berdasarkan konten teks. Ini memungkinkan mesin pencari untuk memberikan hasil yang lebih relevan.
- Pengenalan Pola: Dalam pengenalan pola, representasi berbasis vektor digunakan untuk merepresentasikan objek atau gambaran visual, memungkinkan komputer untuk mengenali pola dalam data visual.

Keuntungan utama dari Representasi Pengetahuan Berbasis Vektor adalah kemampuannya untuk menggambarkan pengetahuan dalam format yang dapat diproses oleh algoritma pembelajaran mesin. Dengan menggunakan vektor, komputer dapat mengukur kemiripan dan relasi antara entitas dengan cara yang efisien. Representasi berbasis vektor juga memiliki fleksibilitas yang tinggi dan dapat diterapkan dalam berbagai tugas pemrosesan data dan bahasa alami(Muttaqin et al., 2023).

6.2 Algoritma Representasi Pengetahuan

Algoritma Representasi Pengetahuan adalah rangkaian metode atau prosedur yang digunakan untuk menghasilkan representasi formal atau struktural dari pengetahuan dalam suatu domain. Tujuannya adalah untuk menyederhanakan, mengorganisir, dan membuat pengetahuan dapat dipahami oleh komputer atau sistem kecerdasan buatan. Berikut adalah beberapa poin penting tentang Algoritma Representasi Pengetahuan:

- Pengumpulan Data: Algoritma Representasi Pengetahuan biasanya dimulai dengan mengumpulkan data atau informasi yang relevan dalam domain tertentu. Ini bisa berupa teks, data numerik, gambar, atau jenis data lainnya yang relevan.
- Pemrosesan Data: Data yang dikumpulkan kemudian diolah untuk mengidentifikasi entitas, atribut, dan hubungan antara entitas tersebut. Algoritma ini dapat melibatkan teknik-teknik pemrosesan bahasa alami, pengenalan pola, atau pemrosesan data lainnya tergantung pada jenis data yang digunakan.
- Representasi Struktural: Setelah data diolah, algoritma akan membuat representasi struktural yang mencerminkan pengetahuan dalam domain tersebut. Ini dapat berupa jaringan semantik, grafik, ontologi, atau representasi berbasis vektor, tergantung pada kebutuhan.

- Pemilihan Atribut: Dalam beberapa kasus, algoritma dapat mengidentifikasi atribut-atribut yang penting dari entitas dalam domain. Ini membantu dalam merepresentasikan entitas dengan cara yang lebih khusus.
- Pemahaman Konteks: Algoritma Representasi Pengetahuan berusaha untuk memahami konteks di mana pengetahuan diterapkan. Ini bisa mencakup identifikasi hubungan hierarkis, makna relatif, dan konteks domain yang relevan.
- Ekstraksi Aturan: Beberapa algoritma mungkin juga mengekstrak aturan atau hubungan yang ditemukan dalam data. Ini dapat digunakan untuk menginformasikan sistem kecerdasan buatan dalam pengambilan keputusan atau pemecahan masalah.
- Validasi dan Evaluasi: Setelah representasi pengetahuan dibuat, algoritma dapat melalui tahap validasi dan evaluasi untuk memastikan bahwa representasi tersebut akurat dan efektif dalam mewakili pengetahuan dalam domain yang dimaksud.
- Aplikasi: Representasi pengetahuan yang dihasilkan dapat digunakan dalam berbagai aplikasi kecerdasan buatan, seperti sistem ekspert, pemrosesan bahasa alami, pengenalan pola, pemecahan masalah, dan pengambilan keputusan.

Beberapa algoritma yang umum digunakan dalam Representasi Pengetahuan melibatkan teknik-teknik pembelajaran mesin, termasuk penggunaan model distribusi (seperti Word2Vec), ontologi (seperti OWL), dan logika formal (seperti logika predikat). Pemilihan algoritma tergantung pada jenis data yang digunakan dan tujuan representasi pengetahuan yang ingin dicapai.

Algoritma Representasi Pengetahuan adalah langkah kunci dalam memungkinkan komputer untuk memahami dan memanfaatkan pengetahuan manusia dalam berbagai konteks,

dan mereka berperan penting dalam pengembangan sistem kecerdasan buatan yang efektif(Ogheneovo & Nlerum, 2020).

6.2.1 Algoritma Representasi Pengetahuan Graf

Algoritma Representasi Pengetahuan Graf adalah sekelompok metode atau prosedur yang digunakan untuk menghasilkan representasi struktural dari pengetahuan dalam bentuk grafik atau jaringan. Representasi ini menggambarkan hubungan antara konsep, entitas, atau objek dalam suatu domain dengan menggunakan simpul (*node*) dan tepi (*edge*) dalam grafik. Berikut adalah beberapa poin penting tentang Algoritma Representasi Pengetahuan Graf:

- Pembuatan Struktur Grafis: Algoritma ini menghasilkan struktur grafis yang terdiri dari simpul dan tepi untuk merepresentasikan entitas dan hubungan antara mereka. Simpul mewakili konsep, objek, atau entitas, sedangkan tepi menghubungkan simpul yang memiliki hubungan atau keterkaitan dalam pengetahuan.
- Ekstraksi dan Pemrosesan Data: Algoritma Representasi Pengetahuan Graf umumnya memulai dengan ekstraksi data dari berbagai sumber, seperti teks, database, atau sumber-sumber lainnya. Data ini kemudian diproses untuk mengidentifikasi entitas dan relasi antara entitas.
- Model Entitas-Hubungan: Representasi ini sering mengadopsi model entitas-hubungan, di mana simpul-simpul mewakili entitas atau konsep, sedangkan tepi-tipe mewakili relasi antara entitas tersebut. Contoh sederhana adalah grafik berbasis entitas-hubungan yang digunakan dalam representasi pengetahuan dalam domain sosial, di mana simpul-simpul mewakili orang dan tepi-tipe mewakili hubungan seperti "teman" atau "keluarga".
- Ontologi: Dalam beberapa kasus, Algoritma Representasi Pengetahuan Graf menggunakan ontologi, yang merupakan representasi formal tentang kelas-kelas objek, konsep, dan

hubungan antara mereka dalam domain tertentu. Ontologi membantu dalam memodelkan pengetahuan dalam domain yang lebih terstruktur.

- Pemahaman Konteks: Algoritma ini berusaha untuk memahami konteks di mana pengetahuan diterapkan. Ini mencakup pengidentifikasian hierarki konsep, makna relatif, dan konteks domain yang relevan.
- Aplikasi: Representasi Pengetahuan Graf yang dihasilkan dapat digunakan dalam berbagai aplikasi, seperti sistem pemrosesan bahasa alami, pengenalan pola, pemodelan pengetahuan dalam ilmu pengetahuan, pemrosesan data geografis, dan berbagai sistem berbasis grafik lainnya.
- Visualisasi: Grafik atau jaringan yang dihasilkan dari algoritma ini sering dapat divisualisasikan dengan baik. Visualisasi ini memungkinkan manusia untuk memahami dan mengeksplorasi pengetahuan dengan lebih baik.

Keuntungan dari Algoritma Representasi Pengetahuan Graf adalah kemampuannya untuk merepresentasikan hubungan dan struktur pengetahuan dengan jelas dalam bentuk yang dapat dipahami oleh manusia dan komputer. Ini memungkinkan untuk mengorganisir dan mengakses pengetahuan dalam cara yang lebih terstruktur, yang sangat berguna dalam berbagai aplikasi kecerdasan buatan dan analisis data(Priandoyo, 2004).

a. Model Graf sebagai Representasi Pengetahuan

Model Graf sebagai Representasi Pengetahuan merupakan pendekatan yang menggunakan struktur grafik atau jaringan untuk menggambarkan pengetahuan dalam suatu domain. Dalam model ini, konsep, objek, atau entitas direpresentasikan sebagai simpul (node), sementara hubungan atau relasi antara konsep-konsep tersebut direpresentasikan sebagai tepi (edge) dalam grafik. Berikut adalah detail lebih lanjut tentang Model Graf sebagai Representasi Pengetahuan:

1. Simpul (*Node*)

Mewakili Konsep atau Entitas: Setiap simpul dalam grafik mewakili konsep, objek, atau entitas dalam domain yang sedang diwakili. Contoh simpul dapat mencakup kata, nama orang, tempat, produk, atau konsep lainnya yang relevan.

2. Tepi (Edge)

   Mewakili Hubungan atau Relasi: Tepi atau edge dalam grafik menghubungkan dua simpul dan mewakili hubungan atau relasi antara konsep-konsep tersebut. Contoh hubungan dapat mencakup "adalah saudara dari," "merupakan bagian dari," "sama dengan," atau relasi lainnya yang relevan dalam domain.

3. Atribut pada Simpul

   Menyimpan Informasi Tambahan: Selain merepresentasikan konsep atau entitas, simpul dalam model grafik juga dapat memiliki atribut atau properti yang menyimpan informasi tambahan tentang konsep tersebut. Misalnya, simpul yang mewakili orang dapat memiliki atribut seperti nama, usia, atau alamat.

4. Berorientasi atau Tidak Berorientasi

   Berorientasi: Dalam grafik berorientasi, tepi memiliki arah yang menunjukkan hubungan yang bersifat unidireksional. Misalnya, dalam grafik berorientasi, tepi "orang A adalah saudara dari orang B" akan memiliki arah dari A ke B.

   Tidak Berorientasi: Dalam grafik tidak berorientasi, tepi tidak memiliki arah, sehingga hubungan antara simpul-simpul adalah dua arah. Misalnya, dalam grafik tidak berorientasi, hubungan "teman dari" adalah saling menghubungkan antara dua simpul.

5. Hierarki dan Struktur

   Grafik semantik sering memiliki struktur hirarkis di mana konsep-konsep dikelompokkan dalam tingkatan atau lapisan yang mencerminkan hubungan hierarkis.

Ini memungkinkan untuk mengorganisir pengetahuan dalam cara yang terstruktur.

6. Ontologi
   Dalam beberapa kasus, representasi grafik dapat menggunakan ontologi, yang merupakan representasi formal tentang kelas-kelas objek, konsep, dan hubungan antara mereka dalam domain tertentu. Ontologi membantu dalam memodelkan pengetahuan dalam domain yang lebih terstruktur dan dapat diterapkan secara luas.
7. Aplikasi
   Model Graf sebagai Representasi Pengetahuan digunakan dalam berbagai aplikasi seperti pemrosesan bahasa alami, analisis data, pemodelan pengetahuan dalam ilmu pengetahuan, pemrosesan data geografis, pengenalan pola, dan berbagai sistem berbasis grafik lainnya.
8. Visualisasi
   Salah satu keuntungan utama dari model grafik adalah kemampuannya untuk divisualisasikan dengan baik. Visualisasi grafik memungkinkan manusia untuk memahami dan mengeksplorasi pengetahuan dengan lebih baik.

Model Graf sebagai Representasi Pengetahuan adalah alat yang kuat dalam pemodelan pengetahuan, terutama ketika ada hubungan kompleks antara entitas dalam domain yang ingin diwakili. Ini membantu dalam menggambarkan pengetahuan dengan cara yang lebih terstruktur dan dapat digunakan oleh sistem kecerdasan buatan untuk analisis, pemrosesan, dan pengambilan keputusan yang lebih cerdas(Lewandowski et al., 2015).

b. Algoritma Pembentukan Graf

Algoritma Pembentukan Graf adalah rangkaian metode atau prosedur yang digunakan untuk membangun struktur grafik

atau jaringan yang merepresentasikan pengetahuan dalam suatu domain.

Proses pembentukan grafik ini melibatkan penentuan simpul (node) dan tepi (edge) serta hubungan antara mereka. Berikut adalah beberapa detail lebih lanjut tentang Algoritma Pembentukan Graf:

- Pendefinisian Tujuan

  Proses dimulai dengan mendefinisikan tujuan atau sasaran representasi grafik. Tujuan ini mungkin berkaitan dengan pengorganisasian pengetahuan dalam domain tertentu, pemodelan hubungan antara konsep, atau penyederhanaan struktur data yang rumit.
- Pengumpulan Data

  Algoritma ini biasanya memulai dengan mengumpulkan data atau informasi yang relevan dalam domain. Data ini dapat berasal dari berbagai sumber, seperti teks, basis data, sumber-sumber online, atau data lain yang relevan.
- Ekstraksi Entitas

  Setelah data dikumpulkan, langkah berikutnya adalah mengidentifikasi entitas atau konsep yang akan direpresentasikan sebagai simpul-simpul dalam grafik. Ini melibatkan ekstraksi entitas yang mungkin termasuk objek, kata, atau konsep penting dalam domain.
- Pengidentifikasian Hubungan

  Selanjutnya, algoritma mengidentifikasi hubungan atau relasi antara entitas atau konsep yang diidentifikasi. Hubungan ini akan direpresentasikan sebagai tepi atau edge dalam grafik. Pengidentifikasian hubungan melibatkan analisis data dan pemahaman tentang bagaimana entitas saling terkait dalam domain.
- Pembentukan Grafik

Proses pembentukan grafik dimulai dengan membuat simpul-simpul yang mewakili entitas atau konsep yang telah diidentifikasi. Selanjutnya, tepi atau edge ditambahkan untuk menghubungkan simpul-simpul yang memiliki hubungan atau relasi dalam domain.

- Hierarki dan Struktur
  Algoritma ini dapat mengorganisir simpul-simpul dalam tingkatan atau lapisan untuk menciptakan struktur hirarkis yang mencerminkan hubungan hierarkis dalam domain. Ini membantu dalam memodelkan pengetahuan dalam cara yang terstruktur.
- Validasi dan Evaluasi
  Setelah grafik dibentuk, langkah selanjutnya adalah melakukan validasi dan evaluasi untuk memastikan bahwa representasi grafik mencerminkan pengetahuan dengan benar dan sesuai dengan tujuan yang ditetapkan.
- Aplikasi
  Representasi grafik yang dihasilkan dapat digunakan dalam berbagai aplikasi, seperti pemrosesan bahasa alami, pemodelan pengetahuan dalam ilmu pengetahuan, analisis data, dan sistem berbasis grafik lainnya.

Algoritma Pembentukan Graf dapat bervariasi tergantung pada tujuan representasi grafik dan sifat data yang digunakan. Beberapa algoritma mungkin lebih kompleks daripada yang lain, tergantung pada kompleksitas pengetahuan dalam domain yang diwakili. Namun, algoritma ini adalah langkah kunci dalam mengubah pengetahuan menjadi representasi grafik yang dapat dimanfaatkan oleh sistem kecerdasan buatan untuk analisis, pemrosesan, dan pengambilan keputusan(Disertasi, 2021).

c. Algoritma Pencarian dalam Graf

Algoritma Pencarian dalam Graf adalah serangkaian metode atau prosedur yang digunakan untuk mencari atau menjelajahi grafik atau jaringan guna menemukan solusi atau informasi tertentu. Pencarian dalam grafik sering digunakan dalam berbagai konteks, termasuk pemodelan pengetahuan, pemrosesan bahasa alami, pengenalan pola, pemecahan masalah, dan berbagai aplikasi lainnya. Berikut adalah beberapa detail lebih lanjut tentang Algoritma Pencarian dalam Graf:

1. Grafik sebagai Representasi Pengetahuan

   Pencarian dalam grafik biasanya dilakukan pada representasi pengetahuan yang diwakili dalam bentuk grafik. Grafik ini terdiri dari simpul (*node*) yang mewakili entitas atau konsep, dan tepi (*edge*) yang mewakili hubungan antara simpul-simpul tersebut.

2. Titik Awal (*Start Node*)

   Pencarian dimulai dari simpul awal, yang sering disebut sebagai titik awal atau simpul awal. Ini adalah titik di mana pencarian dimulai.

3. Tujuan (*Goal Node*)

   Pencarian dalam grafik memiliki tujuan tertentu, yaitu mencapai simpul tujuan atau simpul yang memenuhi kriteria tertentu. Tujuan ini harus didefinisikan dengan jelas sebelum pencarian dimulai.

4. Metode Pencarian

   Terdapat berbagai metode pencarian dalam grafik, dan pemilihan metode tergantung pada jenis masalah yang ingin diselesaikan dan sifat grafik. Beberapa metode pencarian yang umum meliputi:

   - Pencarian dalam Lebar (*Breadth-First Search*, BFS): Melibatkan menjelajahi simpul-simpul pada tingkat yang sama sebelum mengeksplorasi tingkat selanjutnya.
   - Pencarian dalam Kedalaman (*Depth-First Search*, DFS): Melibatkan penjelajahan sepanjang satu jalur

atau cabang hingga mencapai titik ujung sebelum kembali ke simpul sebelumnya.

- Pencarian Terbaik Pertama (*Best-First Search*): Melibatkan pemilihan simpul berdasarkan heuristik tertentu yang mencoba mendekati tujuan dengan efisien.
- Pencarian A (*A-Star*): Menggabungkan biaya aktual dan heuristik untuk memilih simpul yang memiliki perkiraan biaya terendah menuju tujuan.

5. Penandaan Simpul (*Node Marking*)
   Selama pencarian, simpul-simpul yang telah dijelajahi sering ditandai atau dicatat untuk menghindari pengulangan atau siklus dalam grafik. Ini juga membantu dalam melacak jalur yang telah ditemukan.
6. Kriteria Berhenti (Termination Criteria)
   Algoritma pencarian harus memiliki kriteria berhenti yang didefinisikan dengan jelas. Kriteria ini menentukan kondisi di mana pencarian berakhir, misalnya, ketika simpul tujuan ditemukan.
7. Analisis Kinerja
   Evaluasi kinerja algoritma pencarian adalah langkah penting. Ini melibatkan pengukuran waktu yang diperlukan untuk menyelesaikan pencarian, jumlah simpul yang dieksplorasi, dan efisiensi dalam mencapai tujuan.
8. Aplikasi
   Algoritma Pencarian dalam Graf digunakan dalam berbagai aplikasi, termasuk pemrosesan bahasa alami (misalnya, pencarian jalur dalam pohon sintaksis), pengenalan pola (misalnya, pencarian objek dalam citra), pemodelan pengetahuan (misalnya, pemrosesan ontologi), dan banyak lagi.

Algoritma Pencarian dalam Graf merupakan komponen penting dalam berbagai sistem kecerdasan buatan dan aplikasi berbasis grafik. Pemahaman tentang berbagai metode pencarian dan cara

mereka beroperasi membantu dalam pemecahan masalah dan analisis data yang efisien dalam konteks yang berbeda(Disertasi, 2021).

6.2.2 Algoritma Representasi Pengetahuan Logika

Algoritma Representasi Pengetahuan Logika merupakan metode atau prosedur yang digunakan untuk mewakili pengetahuan dalam suatu domain menggunakan logika formal. Pendekatan ini memungkinkan pemodelan pengetahuan dengan cara yang lebih terstruktur dan dapat dipahami oleh komputer. Berikut adalah beberapa detail lebih lanjut tentang Algoritma Representasi Pengetahuan Logika:

1. Logika Formal

   Algoritma ini menggunakan logika formal, seperti logika proposisional atau logika predikat, untuk merepresentasikan pengetahuan. Logika proposisional menggambarkan pernyataan sebagai proposisi logis (benar atau salah), sementara logika predikat memungkinkan untuk merepresentasikan hubungan kompleks antara entitas dalam domain
2. Pernyataan Logis

   Fakta atau pengetahuan dalam domain direpresentasikan sebagai pernyataan logis. Pernyataan ini dapat berbentuk proposisi logis yang sederhana atau pernyataan yang lebih kompleks dengan variabel, predikat, dan kuantor.
3. Aturan Inferensi

   Algoritma Representasi Pengetahuan Logika sering melibatkan aturan inferensi, yang digunakan untuk menarik kesimpulan dari pengetahuan yang ada. Aturan inferensi dapat digunakan untuk memproses informasi, menjawab pertanyaan, atau memecahkan masalah.
4. Ekspresif

Representasi Pengetahuan Logika sangat ekspresif dan dapat menggambarkan pengetahuan dalam tingkat detail yang tinggi. Ini memungkinkan untuk merepresentasikan pengetahuan yang kompleks dan hubungan yang lebih rumit dalam domain.

5. Semantik yang jelas
   Logika formal memberikan semantik yang jelas dan terdefinisi dengan baik untuk representasi pengetahuan. Ini memungkinkan komputer untuk secara akurat memahami makna dan implikasi dari pernyataan logis.
6. Ontologi
   Ontologi, yang merupakan representasi formal tentang kelas-kelas objek, konsep, dan hubungan antara mereka, seringkali digunakan dalam Representasi Pengetahuan Logika. Ontologi membantu dalam memodelkan pengetahuan dalam domain tertentu dengan lebih terstruktur.
7. Pemrosesan Bahsa Alami
   Representasi Pengetahuan Logika dapat digunakan dalam pemrosesan bahasa alami untuk memahami dan menjawab pertanyaan dalam bahasa manusia. Ini melibatkan transformasi kalimat dalam bahasa manusia menjadi pernyataan logis yang dapat dianalisis oleh komputer.
8. Aplikasi dalam Kecerdasan Buatan
   Representasi Pengetahuan Logika digunakan dalam berbagai aplikasi kecerdasan buatan, seperti sistem ekspert, perencanaan otomatis, pemecahan masalah, dan pengambilan keputusan.

Keuntungan dari Algoritma Representasi Pengetahuan Logika adalah kemampuannya untuk merepresentasikan pengetahuan dengan tingkat formalitas dan kejelasan yang tinggi. Ini memungkinkan sistem kecerdasan buatan untuk mengambil keputusan yang logis dan dapat dipertanggungjawabkan berdasarkan pengetahuan yang mereka miliki. Namun,

penggunaan logika formal juga dapat membatasi ekspresivitas dalam menggambarkan pengetahuan yang ambigu atau tidak pasti(Ogheneovo & Nlerum, 2020)

1. Representasi Pengetahuan dalam Logika Predikat

   Representasi Pengetahuan dalam Logika Predikat adalah pendekatan untuk mewakili pengetahuan dalam suatu domain menggunakan logika predikat. Logika predikat adalah cabang dari logika yang lebih kuat dan ekspresif daripada logika proposisional, yang memungkinkan untuk merepresentasikan hubungan kompleks antara entitas dalam domain. Berikut adalah penjelasan lebih lanjut tentang Representasi Pengetahuan dalam Logika Predikat:

   a. Predikat: Predikat adalah unsur dasar dalam logika predikat. Mereka adalah pernyataan yang menggambarkan hubungan antara objek atau entitas dalam domain. Predikat dapat berupa sederhana, seperti "adalah saudara dari," atau lebih kompleks, seperti "merupakan orang tua dari" atau "bertempat di."
   b. Variabel: Variabel digunakan dalam logika predikat untuk menggantikan entitas yang spesifik. Mereka memungkinkan untuk membuat pernyataan yang bersifat umum. Variabel sering dilambangkan dengan huruf-huruf seperti x, y, atau z.
   c. Konstanta: Konstanta adalah objek atau entitas spesifik dalam domain yang direpresentasikan secara langsung dalam logika predikat. Misalnya, "*John*" atau "*New York*" adalah konstanta.
   d. Kuantor: Kuantor adalah elemen penting dalam logika predikat yang digunakan untuk menggambarkan cakupan pernyataan. Terdapat dua jenis kuantor utama dalam logika predikat:

      Kuantor Universal (∀): Digunakan untuk menyatakan bahwa pernyataan berlaku untuk semua entitas dalam domain yang sesuai.

Kuantor Eksistensial (∃): Digunakan untuk menyatakan bahwa ada setidaknya satu entitas dalam domain yang sesuai dengan pernyataan.

e. Aturan Inferensi: Representasi Pengetahuan dalam Logika Predikat sering melibatkan aturan inferensi yang digunakan untuk menarik kesimpulan dari pengetahuan yang ada. Aturan inferensi ini dapat berupa aturan logika, seperti modus ponens atau modus tollens, atau aturan domain yang spesifik.
f. Pernyataan Fakta: Fakta-fakta dalam domain direpresentasikan sebagai pernyataan logis dalam logika predikat. Sebagai contoh, "*John* adalah saudara dari Mary" dapat direpresentasikan sebagai pernyataan predikat dengan variabel, seperti "Saudara (*John, Mary*)."
g. Aturan atau Hukum: Aturan atau hukum dalam domain juga dapat direpresentasikan dalam logika predikat. Ini memungkinkan sistem untuk melakukan penalaran logis dan pemecahan masalah berdasarkan pengetahuan yang ada.
h. Ontologi: Ontologi, yang merupakan representasi formal tentang kelas-kelas objek, konsep, dan hubungan antara mereka dalam domain tertentu, sering digunakan dalam Representasi Pengetahuan dalam Logika Predikat. Ontologi membantu dalam memodelkan pengetahuan dalam domain dengan lebih terstruktur.
i. Penggunaan dalam Kecerdasan Buatan: Representasi Pengetahuan dalam Logika Predikat digunakan dalam berbagai aplikasi kecerdasan buatan, seperti sistem ekspert, perencanaan otomatis, pengambilan keputusan, dan pemrosesan bahasa alami.

Keuntungan dari Representasi Pengetahuan dalam Logika Predikat adalah kemampuannya untuk merepresentasikan pengetahuan dengan tingkat formalitas dan kejelasan yang tinggi. Ini memungkinkan sistem kecerdasan buatan untuk

mengambil keputusan yang logis dan dapat dipertanggungjawabkan berdasarkan pengetahuan yang mereka miliki. Representasi ini juga sangat berguna untuk pemodelan pengetahuan dalam domain yang kompleks dan hubungan yang rumit antara entitas(Ogheneovo & Nlerum, 2020).

2. Konsep Unifikasi dalam Representasi Pengetahuan Logika

   Konsep Unifikasi dalam Representasi Pengetahuan Logika adalah teknik penting yang digunakan untuk menemukan penggantian variabel dengan ekspresi lain dalam suatu pernyataan logis. Teknik ini memiliki peran kunci dalam logika predikat dan digunakan dalam berbagai konteks dalam pemrosesan bahasa alami, pemecahan masalah, inferensi logis, dan berbagai aplikasi kecerdasan buatan lainnya. Berikut adalah penjelasan lebih lanjut tentang konsep unifikasi dalam logika:

   a. Variabel

      Variabel adalah simbol yang digunakan dalam logika predikat untuk menggantikan entitas yang spesifik. Variabel biasanya dilambangkan dengan huruf-huruf seperti x, y, atau z. Misalnya, dalam pernyataan "Saudara (*John, Mary*)," "*John*" dan "*Mary*" adalah konstanta, sedangkan dalam pernyataan "Saudara (X, Y)," "X" dan "Y" adalah variabel yang mewakili entitas yang belum ditentukan.

   b. Pernyataan Logis

      Pernyataan logis dalam logika predikat sering mengandung variabel. Misalnya, pernyataan "Saudara (X, Y)" adalah pernyataan logis dengan dua variabel yang menggambarkan hubungan saudara antara dua entitas yang belum ditentukan.

   c. Substitusi

      Substitusi adalah penggantian variabel dengan ekspresi lain dalam pernyataan logis. Substitusi ini

memungkinkan untuk mengganti variabel dengan nilai konkret atau ekspresi yang memiliki variabel lain. Misalnya, jika kita mengganti "X" dengan "*John*" dalam pernyataan "Saudara (X, Y)," kita akan mendapatkan "Saudara (*John*, Y)."

d. Unifikasi
   Unifikasi adalah proses yang digunakan untuk menemukan substitusi yang memungkinkan dua pernyataan logis untuk menjadi identik atau memiliki kesamaan. Dalam konteks logika predikat, unifikasi digunakan untuk menemukan nilai variabel yang memadai sehingga dua pernyataan logis dapat dianggap setara.
e. Tujuan Unifikasi
   Tujuan dari unifikasi adalah untuk menyatukan pernyataan logis yang berisi variabel dan ekspresi sehingga kita dapat melakukan inferensi logis atau pemecahan masalah. Misalnya, jika kita ingin mencari tahu apakah ada entitas yang menjadi saudara *Mary* dalam pernyataan "Saudara (X, *Mary*)," kita dapat melakukan unifikasi untuk menemukan nilai X yang memungkinkan pernyataan ini menjadi benar.
   Contoh sederhana unifikasi adalah:
   Pernyataan 1: "Saudara (X, Y)"
   Pernyataan 2: "Saudara (*John, Mary*)"
   Dalam kasus ini, unifikasi akan menghasilkan substitusi X = *John* dan Y = *Mary* sehingga kedua pernyataan logis menjadi identik.
f. Unifikasi dalam Pemrosesan Bahasa Alami
   Unifikasi sering digunakan dalam pemrosesan bahasa alami untuk mencocokkan pola teks dengan aturan tata bahasa atau basis pengetahuan. Misalnya, unifikasi dapat digunakan untuk menentukan bahwa kalimat "*John* adalah saudara *Mary*" cocok dengan aturan "Saudara (X, Y)" dalam basis pengetahuan.

g. Peran Penting dalam Inferensi Logis
Unifikasi juga berperan penting dalam inferensi logis, di mana sistem menggunakan unifikasi untuk menemukan fakta-fakta yang relevan atau menyatukan aturan-aturan yang berlaku untuk mencapai kesimpulan.

Konsep unifikasi merupakan alat yang kuat dalam logika predikat dan berkontribusi pada kemampuan pemrosesan bahasa alami, pemecahan masalah, dan inferensi logis dalam berbagai aplikasi kecerdasan buatan. Itu memungkinkan sistem untuk bekerja dengan pernyataan logis yang mengandung variabel dan mencari nilai yang memungkinkan pernyataan-penyataan tersebut menjadi benar atau relevan(Ogheneovo & Nlerum, 2020).

6.2.3 Algoritma Representasi Pengetahuan Berbasis Vektor

Algoritma Representasi Pengetahuan Berbasis Vektor adalah pendekatan untuk mewakili pengetahuan dengan menggunakan representasi vektor dalam ruang multidimensional. Teknik ini banyak digunakan dalam berbagai aplikasi kecerdasan buatan, terutama dalam pengolahan bahasa alami, pemodelan topik, rekomendasi, pengenalan pola, dan pemahaman konsep.

a. Konsep Representasi Vektor
Representasi vektor adalah cara untuk mengubah entitas, konsep, atau informasi dalam bentuk vektor numerik. Dalam representasi berbasis vektor, setiap entitas atau konsep dalam domain diberikan representasi numerik yang unik dalam ruang vektor multidimensional.

b. *Word Embeddings*
Salah satu aplikasi paling terkenal dari representasi berbasis vektor adalah representasi kata (word embeddings) dalam pemrosesan bahasa alami. Dalam representasi ini, setiap kata dalam bahasa diwakili oleh vektor numerik yang

menggambarkan maknanya. Misalnya, dalam Word2Vec, kata "king" dan "queen" dapat direpresentasikan oleh vektor numerik yang memiliki hubungan semantik.

c. Pemrosesan Bahasa Alami

Algoritma Representasi Pengetahuan Berbasis Vektor digunakan secara luas dalam pemrosesan bahasa alami (NLP). Representasi vektor kata digunakan untuk menjalankan tugas seperti pemahaman konteks kata, penerjemahan mesin, analisis sentimen, dan banyak lagi. Metode seperti Word2Vec, GloVe, dan FastText adalah contoh populer dari algoritma ini.

d. *Embeddings* Berbasis Dokumen

Selain representasi kata, algoritma berbasis vektor juga digunakan untuk mewakili dokumen atau teks secara keseluruhan. Dokumen-dokumen dapat diwakili dalam ruang vektor sehingga mereka dapat dibandingkan, diklasifikasikan, atau digunakan dalam rekomendasi konten.

e. Konsep Semantik

Representasi berbasis vektor juga digunakan untuk memahami konsep semantik dalam berbagai domain. Misalnya, dalam pemodelan topik, dokumen atau kata-kata dapat direpresentasikan sebagai vektor dalam ruang topik, yang memungkinkan pemahaman tentang hubungan antar topik.

f. Rekomendasi dan Pengenalan Pola

Algoritma representasi berbasis vektor digunakan dalam sistem rekomendasi untuk memahami preferensi pengguna dan mencocokkan mereka dengan item yang sesuai. Mereka juga digunakan dalam pengenalan pola, seperti pengenalan wajah dan pengenalan objek dalam pengolahan citra.

g. *Latent Semantic Analysis* (LSA)

LSA adalah salah satu teknik representasi pengetahuan berbasis vektor yang digunakan dalam pemodelan topik dan analisis dokumen. Ini mengidentifikasi pola-pola dalam teks dan menciptakan representasi vektor untuk kata-kata dan dokumen.

h. *Machine Learning dan Deep Learning*

Representasi berbasis vektor sering digunakan sebagai tahap awal dalam berbagai tugas dalam *machine learning dan deep learning*, seperti klasifikasi teks, pengenalan suara, dan pemrosesan citra.

Algoritma Representasi Pengetahuan Berbasis Vektor memungkinkan komputer untuk mengolah dan memahami pengetahuan dalam bentuk yang lebih terstruktur dan semantik. Ini memfasilitasi berbagai tugas kecerdasan buatan dengan cara yang lebih efisien dan efektif, terutama dalam konteks pemrosesan bahasa alami dan analisis data(Muttaqin et al., 2023) .

1. Representasi Distribusi Kata (*Word Embeddings*)

   Representasi Distribusi Kata, atau yang lebih dikenal sebagai *Word Embeddings*, adalah pendekatan dalam pemrosesan bahasa alami (NLP) yang digunakan untuk mewakili kata-kata dalam bentuk vektor numerik dalam ruang multidimensional. Konsep ini telah membawa perubahan signifikan dalam cara kita memahami dan memproses bahasa dalam konteks komputasi.

   a. Makna Distribusi

      Representasi distribusi kata didasarkan pada asumsi bahwa makna kata-kata dapat dipahami melalui konteks di mana kata-kata tersebut muncul dalam teks. Dengan kata lain, kata-kata yang sering muncul dalam konteks yang sama atau memiliki pola distribusi yang mirip cenderung memiliki makna yang serupa atau terkait.

b. Vektor Numerik

   Setiap kata dalam bahasa diberikan representasi dalam bentuk vektor numerik. Vektor ini memiliki banyak dimensi, dan setiap dimensi menggambarkan atribut atau karakteristik tertentu dari kata tersebut. Misalnya, sebuah vektor dapat menggambarkan bagaimana kata tersebut berkaitan dengan konsep-konsep tertentu, seperti genre musik, rasa makanan, atau tempat geografis.

c. Word2Vec

   Merupakan salah satu teknik *Word Embeddings* yang paling terkenal, yang dikembangkan oleh Google. Word2Vec menggunakan jaringan saraf tiruan untuk melatih model yang dapat menghasilkan vektor kata. Model ini memahami kata dengan cara yang memungkinkan untuk mengekspresikan hubungan semantik antara kata-kata dalam ruang vektor. Misalnya, vektor untuk "*king*" dan "*queen*" cenderung memiliki arah yang sama dalam ruang vektor.

d. GloVe

   Global Vectors for Word Representation (GloVe) adalah metode lain untuk *Word Embeddings*. Ini memanfaatkan statistik kumpulan data besar untuk memahami hubungan antara kata-kata. Dengan GloVe, kata-kata direpresentasikan berdasarkan probabilitas kemunculan kata-kata tersebut bersama-sama dalam teks.

e. Penerjemahan Bahasa dan Analisis Sentimen

   Word Embeddings digunakan dalam berbagai aplikasi NLP, termasuk penerjemahan bahasa, analisis sentimen, dan pemahaman bahasa alami. Mereka memungkinkan komputer untuk mengenali makna kata-kata dalam konteks tertentu, yang memudahkan dalam

menerjemahkan teks atau menentukan sentimen positif atau negatif dari sebuah kalimat.

f. Reduksi Dimensi
   Vektor kata dalam *Word Embeddings* sering memiliki dimensi yang sangat tinggi. Untuk beberapa aplikasi, seperti pemodelan topik, dimensi ini dapat direduksi untuk mewakili kata-kata dalam ruang vektor yang lebih kompak.

g. *Transfer Learning*
   Representasi distribusi kata yang telah dilatih sebelumnya dapat digunakan dalam tugas-tugas NLP yang berbeda. Dengan demikian, *Word Embeddings* juga memungkinkan *transfer learning*, di mana model yang telah dilatih dalam bahasa tertentu atau tugas tertentu dapat digunakan kembali untuk tugas-tugas serupa.

h. Evaluasi
   Kualitas *Word Embeddings* sering dievaluasi dengan mengukur sejauh mana vektor kata dapat merepresentasikan hubungan semantik dan sintaktik antara kata-kata. Contoh pengujian termasuk tes analogi (misalnya, "*king*" - "*man*" + "*woman*" ≈ "*queen*") dan pengujian kemiripan kata.

Word Embeddings telah menjadi alat penting dalam pemrosesan bahasa alami dan berkontribusi signifikan pada kemajuan dalam banyak aplikasi NLP, memungkinkan komputer untuk lebih memahami dan mengolah bahasa manusia dengan cara yang lebih kontekstual dan semantik(Muttaqin et al., 2023).

2. Algoritma Word2Vec dan GloVe

Algoritma Word2Vec dan GloVe adalah dua metode yang paling populer digunakan untuk membuat representasi distribusi kata

(*Word Embeddings*) dalam pemrosesan bahasa alami (NLP). Kedua algoritma ini memiliki cara kerja yang berbeda tetapi bertujuan untuk menghasilkan vektor kata yang dapat menggambarkan hubungan semantik antara kata-kata dalam teks.

a. Word2Vec (Word to Vector)

Word2Vec adalah algoritma yang mengubah kata-kata dalam teks menjadi vektor numerik dalam ruang vektor multidimensional. Ide dasarnya adalah bahwa kata-kata yang sering muncul dalam konteks yang sama cenderung memiliki makna yang mirip atau terkait satu sama lain. Word2Vec memiliki dua model utama yaitu *Continuous Bag of Words* (CBOW) dan *Skip-gram*.

- *Continuous Bag of Words* (CBOW)

Model ini mencoba memprediksi kata target berdasarkan kata-kata di sekitarnya dalam sebuah kalimat. Ini cocok untuk dataset yang besar dengan banyak kata.

- *Skip-gram*

Model ini sebaliknya mencoba memprediksi kata-kata di sekitar kata target. Ini lebih baik untuk dataset yang lebih kecil atau dengan kata-kata langka.

Proses pelatihan Word2Vec melibatkan langkah-langkah berikut:

- Membuat kamus kata dalam dataset.
- Menginisialisasi vektor kata secara acak.
- Melakukan iterasi melalui dataset dan memperbarui vektor kata berdasarkan kesalahan prediksi terhadap kata-kata konteks.
- Setelah pelatihan selesai, vektor kata menggambarkan makna kata dalam ruang vektor.

Word2Vec menghasilkan representasi kata yang mencerminkan hubungan semantik antara kata-kata. Misalnya, vektor "*king*" cenderung memiliki arah dan jarak yang serupa dengan vektor "*queen*" dalam ruang vektor.

b. GloVe (*Global Vectors for Word Representation*)

GloVe adalah algoritma yang didasarkan pada statistik dan probabilitas distribusi kata dalam teks. Ini berfokus pada matriks frekuensi kemunculan kata (*co-occurrence matrix*) dan menggunakan metode faktorisasi matriks untuk menghasilkan vektor kata.

Langkah awal dalam GloVe adalah membangun matriks kemunculan bersama yang mencatat seberapa sering dua kata muncul bersama-sama dalam teks. Ini menghasilkan matriks simetris di mana entri (i, j) mencerminkan seberapa sering kata i dan kata j muncul bersama.

Proses pelatihan GloVe melibatkan faktorisasi matriks kemunculan bersama menjadi dua matriks: satu untuk kata-kata dan satu untuk kata-kata konteks. Faktorisasi ini dilakukan dengan tujuan menghasilkan vektor kata yang mewakili makna kata. Algoritma mengoptimalkan fungsi tujuan yang meminimalkan kesalahan prediksi pada perbandingan kemunculan kata-kata.

GloVe juga menghasilkan vektor kata yang mencerminkan hubungan semantik antara kata-kata. Seperti Word2Vec, vektor "*king*" dan "*queen*" dalam GloVe juga akan memiliki hubungan yang erat dalam ruang vektor.

Perbandingan antara Word2Vec dan GloVe

- Word2Vec lebih fokus pada prediksi kata-kata berdasarkan konteksnya, sementara GloVe berfokus pada matriks kemunculan bersama dan faktorisasi matriks.

- Word2Vec lebih cocok untuk dataset besar dengan banyak kata, sedangkan GloVe dapat memberikan hasil yang baik bahkan dengan dataset yang lebih kecil.

Kedua algoritma ini menghasilkan representasi vektor kata yang efisien dan berguna dalam berbagai tugas NLP seperti penerjemahan mesin, analisis sentimen, dan pemodelan topik.

Pilihan antara Word2Vec dan GloVe sering tergantung pada tugas dan dataset yang Anda miliki, dan keduanya telah membawa perubahan signifikan dalam pengolahan bahasa alami dan pemahaman konten teks(Muttaqin et al., 2023).

3. Penggunaan Representasi Vektor dalam Pemrosesan Bahasa Alami

Penggunaan representasi vektor dalam pemrosesan bahasa alami (NLP) memiliki dampak signifikan pada kemampuan sistem komputer untuk memahami, mengolah, dan berinteraksi dengan bahasa manusia. Berikut adalah beberapa penggunaan utama representasi vektor dalam NLP:

a. *Word Embeddings* (Representasi Kata)

Representasi vektor kata, seperti Word2Vec, GloVe, dan FastText, digunakan untuk menggambarkan makna kata-kata dalam ruang vektor multidimensional. Ini membantu dalam pemahaman makna kata dalam konteks dan memungkinkan berbagai aplikasi, termasuk analisis sentimen, penerjemahan mesin, dan klasifikasi teks.

b. Analisis Sentimen

Representasi vektor kata digunakan dalam analisis sentimen untuk menentukan apakah suatu teks mengandung sentimen positif, negatif, atau netral. Ini

digunakan dalam pemantauan media sosial, tinjauan produk, dan analisis opini publik.

c. Penerjemahan Mesin

Dalam penerjemahan mesin, representasi vektor kata digunakan untuk memetakan kata-kata dari bahasa sumber ke bahasa target. Ini memungkinkan sistem penerjemahan untuk memahami makna dan konteks teks dengan lebih baik.

d. Pengenalan Entitas Bernama (NER)

Representasi vektor kata digunakan dalam tugas pengenalan entitas bernama untuk mengidentifikasi dan mengekstrak informasi seperti nama orang, tempat, tanggal, dan organisasi dari teks.

e. Klasifikasi Teks

Dalam klasifikasi teks, representasi vektor digunakan untuk mengubah dokumen atau teks menjadi vektor numerik yang dapat digunakan dalam pemodelan klasifikasi. Ini digunakan dalam berbagai aplikasi, seperti deteksi spam, klasifikasi berita, dan analisis teks medis.

f. Pemrosesan Bahasa Alami Multibahasa

Representasi vektor kata dapat digunakan untuk pemrosesan bahasa alami dalam berbagai bahasa. Ini memungkinkan *transfer learning*, di mana model yang telah dilatih dalam satu bahasa dapat digunakan untuk tugas yang sama dalam bahasa lain.

g. Pemrosesan Bahasa Alami Generatif

Representasi vektor digunakan dalam generasi teks, seperti pembuatan ringkasan otomatis, pembuatan cerita, atau generasi teks berdasarkan permintaan pengguna.

h. Rekomendasi Konten

Representasi vektor digunakan dalam sistem rekomendasi konten, seperti rekomendasi produk, rekomendasi film, dan rekomendasi musik, untuk memahami preferensi pengguna dan mencocokkannya dengan konten yang sesuai.

i. Analisis Tren dan Topik

Dalam analisis tren dan pemodelan topik, representasi vektor digunakan untuk mengelompokkan dokumen atau teks berdasarkan topik yang serupa, memungkinkan pemahaman konsep dalam teks.

j. Kecerdasan Buatan (AI) Berbasis Teks

Representasi vektor adalah komponen kunci dalam pengembangan asisten virtual, chatbot, dan sistem AI yang berinteraksi dengan manusia melalui bahasa alami.

Penggunaan representasi vektor dalam NLP terus berkembang dengan peningkatan teknik dan model yang lebih canggih. Hal ini memungkinkan sistem NLP untuk semakin mendekati pemahaman bahasa manusia dan mendukung berbagai aplikasi yang berkaitan dengan pemrosesan dan pemahaman teks(Pustejovsky & Stubbs, 2013).

6.3 Implementasi Representasi Pengetahuan

Implementasi Representasi Pengetahuan mengacu pada penerapan konsep dan teknik yang digunakan untuk mengatur, menyimpan, dan mengakses pengetahuan dalam sistem komputer atau aplikasi kecerdasan buatan. Ini melibatkan konversi pengetahuan manusia yang kompleks dan beragam menjadi bentuk yang dapat dimengerti dan dimanfaatkan oleh mesin. Beberapa aspek penting dalam implementasi Representasi Pengetahuan adalah sebagai berikut:

1. Bentuk Representasi

Pada tahap awal, perlu diputuskan bentuk representasi yang paling sesuai untuk jenis pengetahuan yang akan diwakili. Ini bisa berupa aturan logika, graf pengetahuan, vektor kata (*word embeddings*), atau model pembelajaran mesin.

2. Akuisisi Pengetahuan

   Pengetahuan manusia perlu dikumpulkan dan diorganisasi dalam bentuk yang sesuai dengan bentuk representasi yang dipilih. Ini bisa melibatkan interaksi dengan ahli domain, ekstraksi dari sumber-sumber teks, atau pengumpulan data.

3. Pemodelan Struktur Pengetahuan

   Dalam beberapa kasus, pengetahuan perlu dimodelkan dalam bentuk struktural yang menggambarkan entitas, atribut, dan hubungan antar entitas. Ini umumnya diterapkan dalam graf pengetahuan dan basis pengetahuan berbasis ontologi.

4. Pengorganisasian Pengetahuan

   Pengetahuan perlu diorganisasi dalam struktur yang memungkinkan akses yang efisien. Ini dapat mencakup pengelompokan, pengindeksan, dan penataan hierarkis atau jaringan hubungan.

5. Pengembangan Algoritma dan Logika

   Algoritma dan logika yang sesuai perlu diimplementasikan untuk memungkinkan mesin mengakses dan menggunakan pengetahuan. Ini mencakup pengembangan aturan inferensi, strategi pencarian, atau teknik pemrosesan bahasa alami jika diperlukan.

6. Integrasi dengan Aplikasi

   Representasi Pengetahuan harus diintegrasikan ke dalam aplikasi atau sistem yang membutuhkan pengetahuan tersebut. Misalnya, dalam sistem pakar, pengetahuan harus diintegrasikan ke dalam aliran kerja pengambilan keputusan.

7. Pembaruan dan Perbaikan
   Pengetahuan tidak statis dan akan berubah seiring waktu. Oleh karena itu, sistem perlu dirancang untuk menerima pembaruan dan perbaikan dalam pengetahuan, serta untuk belajar dari pengalaman.
8. Evaluasi dan Pengujian
   Implementasi Representasi Pengetahuan perlu dievaluasi dan diuji untuk memastikan bahwa sistem mampu menggunakan pengetahuan dengan benar dan menghasilkan hasil yang sesuai.

Dalam berbagai aplikasi AI, implementasi Representasi Pengetahuan adalah tahap kritis dalam mengembangkan sistem yang cerdas dan responsif. Kemampuan untuk mengakses dan memanfaatkan pengetahuan dengan cara yang efisien adalah kunci untuk pengambilan keputusan yang cerdas, pemecahan masalah yang efektif, dan interaksi yang bermakna antara mesin dan manusia(Muttaqin et al., 2023).

6.3.1 Representasi Pengetahuan dalam Sistem Kecerdasan Buatan

Representasi Pengetahuan adalah elemen kunci dalam Sistem Kecerdasan Buatan (AI) yang memungkinkan komputer untuk memahami dan memanfaatkan pengetahuan serupa manusia. Dalam AI, Representasi Pengetahuan mengacu pada cara informasi, konsep, atau pengetahuan dari dunia nyata direpresentasikan dalam bentuk yang dapat diolah oleh komputer. Ini memungkinkan mesin untuk menyimpan, mengakses, dan memanfaatkan pengetahuan dengan cara yang cerdas.

Representasi Pengetahuan dapat diwujudkan dalam berbagai bentuk, termasuk aturan logika, graf pengetahuan, representasi vektor kata (*word embeddings*), atau model pembelajaran mesin yang telah dilatih untuk tugas khusus. Kemampuan AI untuk

merepresentasikan pengetahuan dengan baik adalah kunci untuk mengambil keputusan yang cerdas, memecahkan masalah yang kompleks, dan berinteraksi dengan manusia melalui bahasa alami, menjadikannya salah satu aspek paling kritis dalam pengembangan AI yang efektif(Ram, 2012).

1. Sistem Berbasis Aturan (*Rule-Based Systems*)

   Representasi Pengetahuan dalam Sistem Kecerdasan Buatan, terutama melalui Sistem Berbasis Aturan (*Rule-Based Systems*), memainkan peran penting dalam pemahaman dan pengambilan keputusan dalam berbagai domain. Dalam Sistem Berbasis Aturan, pengetahuan ekspert diwakili dalam bentuk aturan-aturan yang terdiri dari pernyataan kondisional "jika... maka...". Aturan-aturan ini merepresentasikan pengetahuan domain yang telah dikumpulkan dan diformulasikan oleh para ahli manusia. Ketika sistem menerima input atau informasi, aturan-aturan ini digunakan untuk melakukan penilaian, pengambilan keputusan, atau tindakan yang sesuai.

   Misalnya, dalam sistem medis berbasis aturan, aturan-aturan dapat menggambarkan hubungan antara gejala, diagnosis, dan tindakan medis yang harus diambil. Saat pasien menyampaikan gejala tertentu, sistem dapat menganalisis aturan-aturan dan memberikan diagnosis atau rekomendasi pengobatan berdasarkan pengetahuan medis yang terkandung dalam aturan tersebut.

   Keunggulan sistem berbasis aturan adalah kemampuannya untuk merepresentasikan pengetahuan dalam bentuk yang mudah dimengerti dan diinterpretasikan oleh manusia. Selain itu, aturan-aturan dapat diperbarui atau dimodifikasi dengan relatif mudah oleh para ahli domain. Namun, sistem berbasis aturan memiliki keterbatasan dalam menangani kompleksitas yang tinggi dan ketidakpastian dalam domain tertentu. Dalam beberapa kasus, representasi pengetahuan

lebih canggih, seperti graf pengetahuan atau teknik pemrosesan bahasa alami (NLP), mungkin diperlukan untuk mengatasi tugas yang lebih kompleks dan dinamis.

Dalam keseluruhan, Representasi Pengetahuan dalam Sistem Berbasis Aturan memberikan landasan yang kuat untuk pengambilan keputusan otomatis dalam berbagai aplikasi AI, dari sistem medis hingga sistem pengelolaan pengetahuan bisnis. Ini memungkinkan komputer untuk memanfaatkan pengetahuan manusia yang ada dan menjadikannya dasar untuk tindakan cerdas dan rekomendasi dalam berbagai konteks(Pustejovsky & Stubbs, 2013).

2. Penggunaan Graf Pengetahuan dalam Sistem Expert

Representasi Pengetahuan dalam Sistem Kecerdasan Buatan, khususnya melalui penggunaan Graf Pengetahuan dalam Sistem Expert, memainkan peran sentral dalam menghadirkan kecerdasan buatan dalam berbagai domain. Graf Pengetahuan adalah bentuk representasi pengetahuan yang mengorganisasi entitas dan hubungan antara entitas dalam bentuk graf atau jaringan. Dalam Sistem Expert, graf pengetahuan menggambarkan pengetahuan domain yang dikumpulkan dan diorganisasi sedemikian rupa sehingga sistem dapat dengan mudah mengakses dan memanfaatkannya.

Sebagai contoh, dalam domain medis, graf pengetahuan dapat digunakan untuk merepresentasikan hubungan antara penyakit, gejala, pengobatan, dan hasil tes. Entitas seperti penyakit dan gejala dihubungkan dengan hubungan semantik yang menggambarkan sebab-akibat atau asosiasi. Ketika seorang ahli medis atau sistem expert menerima informasi dari pasien tentang gejala tertentu, sistem dapat menggunakan graf pengetahuan ini untuk melakukan diagnosa, menghasilkan rekomendasi pengobatan, atau

meramalkan perkembangan penyakit berdasarkan hubungan yang ada dalam graf.

Keuntungan utama dari penggunaan graf pengetahuan dalam Sistem Expert adalah kemampuannya untuk merepresentasikan pengetahuan yang kompleks dan berstruktur dengan jelas. Graf memungkinkan pengorganisasian hierarki atau jaringan pengetahuan yang kompleks, sehingga sistem dapat dengan mudah mengeksplorasi hubungan antara entitas dan mengambil keputusan yang lebih tepat. Selain itu, perubahan atau pembaruan pengetahuan dapat diintegrasikan ke dalam graf dengan relatif mudah, sehingga sistem expert dapat berkembang seiring dengan perkembangan pengetahuan domain.

Namun, perlu diingat bahwa konstruksi graf pengetahuan yang baik memerlukan upaya dan pemahaman yang mendalam tentang domain tertentu. Selain itu, kemampuan pencarian dan inferensi dalam graf pengetahuan juga perlu dikembangkan agar sistem dapat dengan efisien mengeksploitasi pengetahuan yang terkandung dalam graf.

Dalam keseluruhan, penggunaan Graf Pengetahuan dalam Sistem Expert adalah salah satu cara paling efektif untuk mewujudkan kecerdasan buatan dalam tugas yang melibatkan pengetahuan yang kompleks dan kontekstual. Ini memungkinkan sistem untuk memahami, mengelola, dan memanfaatkan pengetahuan dengan lebih baik, yang menjadi dasar bagi pengambilan keputusan yang cerdas dalam berbagai domain(Muttaqin et al., 2023).

3. Implementasi Representasi Pengetahuan dalam Chatbot

Representasi Pengetahuan adalah unsur kunci dalam implementasi chatbot yang efektif. Chatbot adalah aplikasi kecerdasan buatan yang berinteraksi dengan manusia

melalui percakapan teks, dan untuk berfungsi dengan baik, chatbot perlu memahami dan merespons bahasa manusia dengan cara yang kontekstual dan relevan. Representasi Pengetahuan dalam chatbot mencakup beberapa aspek penting:

- Chatbot memanfaatkan Representasi Vektor Kata (*Word Embeddings*) untuk memahami makna kata-kata dalam teks yang diberikan oleh pengguna. Dengan mengubah kata-kata menjadi vektor numerik dalam ruang vektor multidimensional, chatbot dapat mengidentifikasi sinonim, antonim, dan hubungan semantik lainnya antara kata-kata. Hal ini membantu chatbot memahami pertanyaan atau pernyataan pengguna dengan lebih baik dan dapat merespons dengan cara yang tepat.
- Selanjutnya, chatbot sering mengintegrasikan Representasi Pengetahuan dalam bentuk aturan atau skrip yang menggambarkan logika bisnis atau pengetahuan domain tertentu. Ini berarti chatbot memiliki akses ke pengetahuan yang telah dikumpulkan dan diformulasikan sebelumnya, sehingga dapat memberikan jawaban yang tepat berdasarkan aturan-aturan yang ada. Misalnya, dalam chatbot perbankan, aturan-aturan dapat menggambarkan bagaimana chatbot harus merespons permintaan transfer uang atau pertanyaan tentang saldo rekening.
- Selain itu, chatbot juga dapat mengandalkan Representasi Pengetahuan dalam bentuk Graf Pengetahuan. Graf Pengetahuan memungkinkan chatbot untuk mengorganisasi pengetahuan dalam bentuk jaringan entitas dan hubungan. Ini mempermudah chatbot untuk mencari informasi, menjelaskan konsep, atau memberikan rekomendasi berdasarkan pengetahuan yang terkandung dalam graf.

- Dalam implementasi chatbot yang lebih canggih, teknik pemrosesan bahasa alami (NLP) digunakan bersama dengan Representasi Pengetahuan. Ini memungkinkan chatbot untuk memahami pertanyaan yang lebih kompleks, mengidentifikasi entitas bernama, dan mengekstrak informasi dari kalimat dengan konteks yang lebih dalam.

Secara keseluruhan, Representasi Pengetahuan dalam implementasi chatbot memungkinkan chatbot untuk menjadi lebih cerdas dan responsif dalam berinteraksi dengan pengguna. Chatbot dapat menggunakan pengetahuan yang telah ada, memahami makna teks, dan merespons dengan jawaban yang sesuai, menciptakan pengalaman interaksi yang lebih efisien dan memuaskan bagi pengguna(Pustejovsky & Stubbs, 2013)

6.3.2 Representasi Pengetahuan dalam Pemrosesan Bahasa Alami

Representasi Pengetahuan dalam Pemrosesan Bahasa Alami (*Natural Language Processing* atau NLP) adalah cara komputer mengubah pengetahuan yang ada dalam teks atau bahasa manusia menjadi bentuk yang dapat diolah dan dimengerti oleh algoritma komputer. Ini melibatkan berbagai teknik dan metode untuk mengorganisir, menggambarkan, dan memodelkan informasi yang terkandung dalam teks.

Representasi Pengetahuan dalam NLP memungkinkan komputer untuk memahami makna kata-kata, sintaksis kalimat, dan konteks yang lebih luas dalam bahasa manusia. Ini juga mencakup kemampuan untuk mengenali entitas bernama, menggambarkan hubungan antar konsep, dan mengidentifikasi topik atau tema dalam teks.

Dengan menggunakan Representasi Pengetahuan ini, sistem NLP dapat menjalankan berbagai tugas seperti penerjemahan

bahasa, analisis sentimen, ekstraksi informasi, dan komunikasi dengan manusia melalui chatbot atau asisten virtual, menjadikannya salah satu komponen kunci dalam perkembangan teknologi pemrosesan bahasa alami yang semakin canggih.

a. Penggunaan Representasi Vektor untuk Analisis Teks

Penggunaan Representasi Vektor untuk Analisis Teks adalah salah satu terobosan paling signifikan dalam Pemrosesan Bahasa Alami (NLP). Representasi Vektor, seperti *Word Embeddings*, mengubah kata-kata dalam teks menjadi vektor numerik dalam ruang vektor multidimensional. Konsep dasar di baliknya adalah bahwa kata-kata yang serupa dalam konteks akan memiliki representasi vektor yang mendekati satu sama lain, memungkinkan komputer untuk memahami makna dan hubungan semantik antar kata-kata(Pustejovsky & Stubbs, 2013).

Dalam *Word Embeddings*, kata-kata dipelajari dari sejumlah besar teks dan diwakili sebagai vektor numerik. Ini memberikan beberapa manfaat penting dalam analisis teks:

- Pemahaman Makna

  Kata Representasi vektor memungkinkan komputer untuk memahami makna kata-kata dalam konteks yang lebih luas. Sebagai contoh, dalam representasi vektor, kata "kucing" dan "kucing besar" akan memiliki vektor yang serupa, menunjukkan hubungan semantik antara keduanya.

- Deteksi Hubungan Semantik

  Vektor kata juga memungkinkan deteksi hubungan semantik seperti sinonim dan antonim. Komputer dapat mengidentifikasi kata-kata yang serupa dalam vektor dan bahkan melakukan operasi aljabar pada vektor kata,

seperti "raja"-"pria"+"wanita" menghasilkan vektor yang mendekati "ratu."

- Analisis Sentimen

  Dalam analisis sentimen, representasi vektor membantu komputer untuk memahami apakah suatu teks mengandung sentimen positif, negatif, atau netral dengan menganalisis kata-kata yang digunakan dan relasi antara mereka.

- Penerjemahan Mesin

  Dalam penerjemahan mesin, representasi vektor membantu dalam pemahaman kalimat dalam bahasa asli dan penghasilan kalimat yang sesuai dalam bahasa target.

- Rekomendasi dan Pengelompokan

  Dalam sistem rekomendasi dan pengelompokan, representasi vektor digunakan untuk memahami preferensi pengguna dan mengidentifikasi item yang serupa.

- Pencarian Informasi

  Dalam pencarian informasi, vektor kata digunakan untuk memahami relevansi antara *query* pengguna dan dokumen yang dicari.

Penggunaan Representasi Vektor telah membuka pintu bagi kemajuan besar dalam pemrosesan bahasa alami, memungkinkan komputer untuk bekerja dengan teks dalam cara yang lebih semantik dan kontekstual. Ini adalah alat yang sangat penting dalam analisis teks modern dan memainkan peran penting dalam berbagai aplikasi NLP yang semakin cerdas dan responsif.

b. Algoritma Pemahaman Bahasa Alami dengan Representasi Pengetahuan

Algoritma Pemahaman Bahasa Alami dengan Representasi Pengetahuan adalah fondasi dalam memungkinkan komputer untuk memahami dan berinteraksi dengan bahasa manusia secara lebih mendalam dan semantik. Representasi Pengetahuan mencakup penggunaan teknik seperti *Word Embeddings* untuk mengubah kata-kata dalam teks menjadi vektor numerik yang memungkinkan pemahaman makna kata dan hubungan antar kata dalam konteks.

Algoritma NLP juga mencakup analisis sintaktis dan semantik untuk memahami struktur kalimat, serta pengenalan entitas bernama (NER) untuk mengidentifikasi entitas seperti nama orang atau tempat dalam teks. Dalam pemrosesan bahasa alami yang lebih luas, algoritma ini digunakan dalam penerjemahan mesin, analisis sentimen, analisis topik, pencarian informasi, dan banyak lagi.

Dengan menggunakan Representasi Pengetahuan dalam algoritma NLP, komputer dapat berinteraksi dengan bahasa manusia dalam cara yang semakin cerdas, membuka pintu bagi aplikasi yang lebih canggih dan responsif dalam komunikasi manusia-mesin(Pustejovsky & Stubbs, 2013).

6.3.3 Representasi Pengetahuan dalam Vision Komputer

Representasi Pengetahuan dalam Vision Komputer adalah cara komputer mengorganisir, menyimpan, dan memahami informasi visual yang terkandung dalam gambar atau video. Ini mencakup penggunaan model dan algoritma yang memungkinkan komputer untuk mengenali objek, wajah, aksi, dan konteks visual lainnya dalam data gambar dan video.

Representasi Pengetahuan dalam Vision Komputer melibatkan konversi data visual menjadi format yang dapat dimengerti oleh komputer, seperti fitur berbasis vektor atau graf yang merepresentasikan entitas visuais dan hubungan antar mereka.

Teknik-teknik ini membantu dalam tugas seperti deteksi objek, pengenalan pola, analisis wajah, dan bahkan pemahaman konteks dalam video.

Dengan kemajuan dalam representasi pengetahuan dalam Vision Komputer, komputer dapat memberikan hasil yang lebih akurat dan kontekstual dalam analisis visual, mendukung aplikasi seperti kendaraan otonom, pemantauan keamanan, dan pengenalan obyek dalam dunia nyata(Ritha, 2016).

a. Representasi Objek dalam Vision Komputer

   Representasi Objek dalam Vision Komputer adalah cara komputer menggambarkan dan memahami objek yang ada dalam gambar atau video. Objek dalam konteks ini dapat mencakup berbagai entitas visual, seperti orang, mobil, binatang, atau objek lainnya. Representasi Objek melibatkan proses identifikasi, penjelasan, dan pemodelan objek-objek ini dalam data visual. Ini melibatkan penggunaan algoritma dan teknik pemrosesan gambar yang dapat mengenali objek berdasarkan fitur-fitur visual seperti bentuk, tekstur, warna, dan ukuran.

   Hasilnya biasanya berupa representasi vektor atau graf yang mencakup informasi tentang objek tersebut, seperti lokasi, orientasi, atau atribut khusus. Representasi Objek ini merupakan komponen kunci dalam tugas Vision Komputer, seperti deteksi objek dalam citra medis, pengawasan lalu lintas, pengenalan wajah, atau pencarian visual dalam basis data besar-besaran.

   Dengan pemahaman yang lebih baik tentang objek dalam data visual, komputer dapat mendukung berbagai aplikasi yang bergantung pada analisis objek dalam dunia nyata.

b. Pemrosesan Citra dengan Representasi Pengetahuan

   Pemrosesan Citra dengan Representasi Pengetahuan merupakan pendekatan dalam Vision Komputer yang

melibatkan pemanfaatan pengetahuan domain atau aturan yang telah diketahui untuk menganalisis dan memproses citra. Representasi Pengetahuan dalam konteks ini bisa mencakup pemahaman tentang objek, fitur, atau pola visual yang khas dalam gambar. Algoritma pemrosesan citra menggunakan pengetahuan ini untuk mengidentifikasi objek atau fitur tertentu dalam citra, bahkan ketika citra tersebut mengandung gangguan atau variasi.

Pendekatan ini berfokus pada pemodelan objek atau fitur dengan basis pengetahuan, yang dapat mencakup pengetahuan tentang bentuk, warna, tekstur, atau distribusi spasial dalam citra. Pemrosesan Citra dengan Representasi Pengetahuan dapat digunakan dalam berbagai aplikasi, seperti pengenalan wajah, deteksi objek dalam lalu lintas, pemrosesan citra medis, atau pengenalan karakter dalam dokumen. Dengan memadukan pengetahuan domain dengan teknik pemrosesan citra, sistem dapat meningkatkan akurasi dan keandalan dalam analisis visual, mendukung pengambilan keputusan yang lebih baik dalam berbagai bidang(Betalya, 2005)

DAFTAR PUSTAKA

Betalya. 2005. Representasi Citra. *Universitas Gunadarma*.

Disertasi, R. 2021. *TEMPORAL PADA PEMAHAMAN SCENE*. *33214302*.

Lewandowski, C. M., Co-investigator, N., & Lewandowski, C. M. 2015. Representasi Pengetahuan. *The Effects of Brief Mindfulness Intervention on Acute Pain Experience: An Examination of Individual Difference*, *1*, 1689–1699. http://entin.lecturer.pens.ac.id/Kecerdasan Buatan/Buku/Bab 2 Representasi Pengetahuan.pdf

Mubarak, M. 1991. Pengantar Kecerdasan Buatan (Artificial Intelligence. In *Pengantar Kecerdasan Buatan* (Vol. 5, Issue 2. https://doi.org/10.1002/1521-377320010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Muttaqin, Alfaris, L., Muhammad, A. C., Arafah, M., Albinur Limbong, Suryani, Abdal, N. M., Fairuzabadi, M., Pungus, S. R., Nurahman, A., Ruben Cornelius Siagian, & Hazriani. 2023. *Representasi Pengetahuan* (Issue May.

Ogheneovo, E., & Nlerum, P. 2020. Knowledge Representation in Artificial Intelligence and Expert Systems Using Inference Rule. *International Journal of Engineering Research*, *11*(April), 1886–1900.

Priandoyo, A. 2004. Penggunaan Kembali Basis Pengetahuan Dengan Representasi BingkaiDalam Aplikasi Kecerdasan Buatan. *SNATi*, 17–25.

Pustejovsky, J., & Stubbs, A. 2013. *Natural Language Annotation for Machine Learning -- A guide to Corpus-building for applications*.

Ram, C. 2012. Advances in Knowledge Representation. In *Advances in Knowledge Representation*. https://doi.org/10.5772/2351

Ritha, N. 2016. Kecerdasan Buatan Berbasis Problem Based Learning. *Kecerdasan Buatan Berbasis Problem Based*

Learning, 1–85.
Rosnelly, R. 2012. Sistem Pakar: Konsep dan Teori. *Cv Andi Offset*, 122. https://books.google.co.id/books?hl=id&lr=&id=WrOACwAAQBAJ&oi=fnd&pg=PR3&dq=aplikasi+sistem+pakar+psikologi&ots=ObZiop0PNz&sig=z3RnAN03QVVAfRCZtytrDJmWVtc&redir_esc=y#v=onepage&q=aplikasi sistem pakar psikologi&f=false
Susatyono, J. D. 2021. *Sistem Pakar : Kajian Konsep & Penerapannya*.
Sutojo, T; Mulyanto, Edi; Suhartono, V. 2011. *Kecerdasan Buatan*.
Zarri, G. P. 2011. Knowledge Representation. In *Encyclopedia of KnowledgeManagement*. https://doi.org/10.4018/9781591405733.ch061

BAB 7
Logika Samar (Fuzzy logic)

Oleh Rahmat Taufik R.L Bau

7.1 Pendahuluan

Fuzzy logic adalah suatu kerangka kerja matematis yang diciptakan oleh Lotfi A. Zadeh pada tahun 1965. Konsep ini memungkinkan kita untuk menggambarkan dan mengukur ketidakpastian serta ketidakjelasan dengan cara yang lebih mirip dengan cara berpikir manusia. *Fuzzy logic* memperkenalkan konsep himpunan *fuzzy*, variabel *fuzzy*, dan aturan *fuzzy* yang mengizinkan kita untuk mengatasi situasi di mana informasi yang tersedia tidak hanya bersifat biner (benar atau salah), tetapi dapat memiliki tingkat keanggotaan yang bervariasi antara 0 dan 1.

Ketidakpastian dan ketidakjelasan adalah realitas yang sering kita hadapi dalam berbagai aspek kehidupan. Seringkali, kita dihadapkan pada situasi di mana kita tidak dapat membuat keputusan berdasarkan informasi yang hanya benar atau salah. Misalnya, dalam pengambilan keputusan cuaca, kita harus mempertimbangkan sejauh mana cuaca adalah "cerah" atau "hujan" dalam skala yang lebih halus.

Bab ini bertujuan untuk mendalami konsep dan aplikasi *Fuzzy logic*. Prinsip-prinsip dasarnya, algoritma, dan implementasinya akan diuraikan. Konsep-konsep tersebut mencakup himpunan *fuzzy*, variabel *fuzzy*, dan aturan *fuzzy*, yang membentuk dasar dari kerangka kerja ini. Lebih lanjut, bagaimana algoritma *fuzzy*

logic berperan dalam menerapkan konsep ini ke dalam berbagai aplikasi di dunia nyata akan dipelajari.

7.2 Teori *Fuzzy logic*

Fuzzy logic adalah suatu pendekatan matematis yang telah merevolusi cara kita memahami dan mengatasi ketidakpastian serta ketidakjelasan dalam berbagai konteks. Teori ini memungkinkan kita untuk menggambarkan dan mengukur situasi yang tidak selalu dapat dijelaskan dalam terminologi biner sederhana (benar atau salah), melainkan dalam kerangka yang lebih fleksibel dan mirip dengan cara berpikir manusia. Dalam konteks ini, *fuzzy logic* memperkenalkan konsep-konsep utama seperti himpunan *fuzzy*, variabel *fuzzy*, dan aturan *fuzzy* yang memiliki peran penting dalam mewujudkan pendekatan ini.

7.2.1 Konsep Himpunan *Fuzzy*

Fuzzy logic telah menjadi landasan utama dalam penanganan ketidakpastian dan ketidakjelasan dalam berbagai bidang ilmu, teknologi, dan aplikasi. Salah satu elemen kunci dalam *fuzzy logic* adalah konsep himpunan *fuzzy*. Untuk memahami lebih dalam, mari kita eksplorasi bagaimana konsep ini telah mengubah cara kita memandang himpunan dan bagaimana hal itu berperan dalam mengatasi ketidakpastian.

1. Himpunan Konvensional vs. Himpunan *Fuzzy*

Dalam matematika konvensional, himpunan adalah kumpulan elemen yang memiliki karakteristik yang sama. Dengan kata lain, setiap elemen adalah anggota atau bukan anggota himpunan tersebut. Misalnya, himpunan bilangan bulat positif hanya terdiri dari bilangan bulat positif dan tidak ada yang lain. Konsep ini cukup sederhana dan bermanfaat dalam banyak aplikasi.

Namun, dunia nyata sering kali tidak sejelas himpunan konvensional ini. Ada banyak situasi di mana kita tidak dapat mengklasifikasikan sesuatu sebagai benar-benar milik himpunan atau tidak. Contohnya, mari kita pertimbangkan himpunan "tingkat kecerahan" dalam sebuah ruangan. Konvensionalnya, kita mungkin hanya memiliki dua himpunan: "terang" dan "gelap." Namun, *fuzzy logic* memungkinkan kita untuk menggambarkannya dengan cara yang lebih realistis. Kita bisa memiliki himpunan *fuzzy* "cukup terang," "sedikit terang," "agak gelap," dan sebagainya. Setiap himpunan ini memiliki tingkat keanggotaan yang bervariasi, misalnya, "cukup terang" mungkin memiliki tingkat keanggotaan sekitar 0,7, yang berarti ruangan itu cukup terang, tetapi tidak sepenuhnya terang.

2. Tingkat Keanggotaan dalam Himpunan *Fuzzy*

Salah satu konsep penting dalam himpunan *fuzzy* adalah tingkat keanggotaan. Ini adalah nilai numerik yang mengukur sejauh mana suatu elemen termasuk dalam himpunan *Fuzzy*. Tingkat keanggotaan berada dalam rentang antara 0 hingga 1, di mana 0 berarti elemen tidak termasuk dalam himpunan (tidak anggota), dan 1 berarti elemen sepenuhnya termasuk dalam himpunan (anggota penuh).

Mari kita gunakan contoh lain untuk memahami konsep tingkat keanggotaan. Bayangkan kita memiliki himpunan *fuzzy* "tingkat bahaya cuaca" dalam konteks pengambilan keputusan untuk perjalanan. Himpunan ini mungkin memiliki kategori seperti "bahaya tinggi," "bahaya sedang," dan "bahaya rendah." Tingkat keanggotaan akan menggambarkan sejauh mana kondisi cuaca saat ini termasuk dalam setiap kategori ini. Jika cuaca sangat buruk dan berbahaya, maka tingkat keanggotaan "bahaya tinggi" akan mendekati 1, sementara tingkat keanggotaan "bahaya sedang" dan "bahaya rendah" akan mendekati 0.

3. Penggunaan Himpunan *Fuzzy* dalam Pemikiran Manusia

Salah satu kekuatan utama dari himpunan *fuzzy* adalah bahwa konsep ini mencerminkan cara berpikir manusia yang sering kali tidak dapat diungkapkan secara tegas. Manusia cenderung berpikir dalam nuansa dan tidak selalu memandang segala sesuatu sebagai benar atau salah secara mutlak.

Misalnya, ketika seseorang berbicara tentang cuaca, kita sering mendengar pernyataan seperti, "Hari ini cukup hangat," atau "Makanan itu agak pedas." Pernyataan semacam ini mencerminkan pemahaman kita yang tidak selalu berdasarkan kebenaran atau kekeliruan mutlak. *Fuzzy logic* memungkinkan kita untuk menyampaikan pemikiran ini dalam bentuk matematis dan menggunakannya dalam pengambilan keputusan.

7.2.2 Variabel Fuzzzy

Variabel *fuzzy* adalah alat penting dalam *Fuzzy logic* yang memungkinkan kita mengukur dan menggambarkan fenomena yang berubah-ubah dengan cara yang lebih fleksibel. Variabel ini memungkinkan kita untuk menggambarkan hubungan antara berbagai variabel dengan cara yang lebih alami daripada yang bisa dicapai oleh metode konvensional.

1. Variabel Konvensional vs. Variabel *Fuzzy*

Dalam matematika konvensional, variabel adalah simbol yang mewakili nilai yang dapat berubah-ubah. Misalnya, dalam pengukuran suhu, kita menggunakan variabel "T" untuk mewakili suhu dalam derajat Celsius. Namun, dalam *fuzzy logic*, variabel ini bisa lebih fleksibel.

Dalam dunia nyata, seringkali kita dihadapkan pada situasi di mana variabel yang kita amati tidak selalu memiliki nilai yang pasti. Misalnya, jika kita berbicara tentang "kecepatan kendaraan," kita tidak hanya memiliki satu nilai kecepatan, tetapi berbagai tingkat kecepatan. Dengan *fuzzy logic*, kita dapat menggambarkan variabel "kecepatan kendaraan" sebagai

variabel *fuzzy* yang memiliki tingkat keanggotaan, seperti "cepat," "lambat," atau "sedang." Ini memberi kita cara yang lebih baik untuk memodelkan variasi kecepatan dalam situasi yang dinamis.

2. Hubungan Antara Variabel *Fuzzy*

Variabel *fuzzy* memungkinkan kita untuk menyatakan hubungan antara berbagai variabel dengan cara yang lebih kompleks daripada yang bisa dicapai oleh metode konvensional. Dalam pengendalian otomatis, contohnya, hubungan antara variabel seperti "kecepatan kendaraan" dan "jarak pengereman" seringkali tidak bisa diungkapkan dengan aturan sederhana.

Dengan *Fuzzy logic*, kita dapat menggunakan variabel *fuzzy* untuk menggambarkan hubungan ini dengan tingkat keanggotaan. Misalnya, jika kita ingin mengendalikan sistem pengereman berdasarkan "kecepatan kendaraan" dan "jarak pengereman," kita bisa memiliki aturan *fuzzy* seperti "Jika kecepatan kendaraan sangat tinggi dan jarak pengereman sangat dekat, maka aktifkan pengereman maksimal." Dalam aturan ini, "sangat tinggi" dan "sangat dekat" adalah tingkat keanggotaan yang mendeskripsikan kedua variabel tersebut. Dengan cara ini, *Fuzzy logic* memungkinkan kita untuk menggambarkan hubungan yang lebih kompleks antara variabel-variabel ini dalam situasi nyata.

7.2.3 Aturan *Fuzzy*

Aturan *fuzzy* adalah langkah berikutnya dalam pengembangan *Fuzzy logic* yang memungkinkan kita untuk mengambil keputusan berdasarkan variabel-variabel *fuzzy* yang telah kita tentukan sebelumnya. Aturan ini memungkinkan kita menggabungkan beberapa himpunan *fuzzy* dan variabel *fuzzy* dalam suatu sistem yang lebih besar.

1. Pengambilan Keputusan Berdasarkan Aturan *Fuzzy*

Aturan *fuzzy* adalah fondasi dari bagaimana *Fuzzy logic* digunakan dalam pengambilan keputusan. Mereka menggambarkan cara menghubungkan variabel-variabel *fuzzy* dan himpunan *fuzzy* untuk mendapatkan hasil yang bermakna. Aturan ini sering kali mengandung kondisi yang dinyatakan dalam bentuk "Jika [kondisi], maka [tindakan]."

Sebagai contoh, mari kita kembali ke aturan *fuzzy* tentang cuaca. Kondisi dalam aturan ini dapat berbunyi, "Jika tingkat keanggotaan 'bahaya tinggi' dalam himpunan 'tingkat bahaya cuaca' lebih dari 0,7, maka sarankan untuk tidak melakukan perjalanan." Dalam hal ini, tingkat keanggotaan diukur berdasarkan sejauh mana kondisi cuaca termasuk dalam himpunan "bahaya tinggi." Jika tingkat keanggotaan ini melewati ambang batas yang telah ditetapkan (0,7), maka tindakan yang diambil adalah untuk tidak melakukan perjalanan.

2. Kombinasi Aturan *Fuzzy*

Aturan *fuzzy* dapat digunakan bersama-sama dalam sistem yang lebih besar untuk mengambil keputusan yang lebih kompleks. Dalam sistem pengendalian otomatis, misalnya, beberapa aturan *fuzzy* dapat digunakan untuk mengendalikan perangkat berdasarkan kondisi yang berubah-ubah.

Sebagai contoh, dalam kendaraan otonom, aturan-aturan *fuzzy* yang berbeda dapat digunakan untuk menggabungkan informasi dari sensor-sensor seperti lidar, kamera, dan radar untuk mengambil keputusan tentang bagaimana mobil tersebut harus bergerak. Dalam hal ini, aturan-aturan tersebut bekerja bersama untuk menciptakan sistem yang cerdas dalam pengambilan keputusan.

7.3 Algoritma dalam *Fuzzy Logic*

Dalam *fuzzy logic*, algoritma merupakan bagian yang sangat penting untuk mengolah data dan mengambil keputusan

berdasarkan konsep-konsep *fuzzy*. Dengan menggunakan algoritma yang tepat, kita dapat mengimplementasikan konsep himpunan *fuzzy*, variabel *Fuzzy*, dan aturan *fuzzy* dalam berbagai aplikasi. Dalam bagian ini, kita akan menjelaskan beberapa algoritma kunci yang digunakan dalam *Fuzzy logic*, termasuk algoritma inferensi, defuzzifikasi, dan pengoptimalan.

7.3.1. Algoritma Inferensi *Fuzzy*

Algoritma inferensi *fuzzy* adalah inti dari *fuzzy logic*, yang memungkinkan kita untuk mengambil keputusan berdasarkan aturan *fuzzy* yang telah ditentukan sebelumnya. Ini adalah langkah di mana input variabel *fuzzy* digunakan untuk menghasilkan output yang sesuai berdasarkan aturan-aturan tersebut. Ada beberapa metode inferensi *fuzzy* yang umum digunakan, di antaranya adalah Metode Mamdani dan Metode Sugeno.

1. Metode Mamdani

Metode Mamdani adalah salah satu metode inferensi *fuzzy* yang paling umum digunakan. Ini bekerja dengan menggabungkan berbagai aturan *fuzzy* yang berlaku dalam suatu situasi untuk menghasilkan kesimpulan yang lebih akurat.

Metode Mamdani menggabungkan hasil inferensi *fuzzy* dengan menggunakan aturan tertentu. Untuk setiap aturan, terdapat tingkat keanggotaan (μ) yang dihasilkan oleh himpunan *fuzzy* input dan output. Hasil dari setiap aturan digabungkan dengan menggunakan operator yang sesuai (biasanya operasi minimum, disebut juga "AND *Fuzzy*").

Misalnya, jika ada dua aturan:

- Aturan 1: Jika x adalah A1, maka y adalah B1 dengan tingkat keanggotaan μ1.
- Aturan 2: Jika x adalah A2, maka y adalah B2 dengan tingkat keanggotaan μ2.

Maka, tingkat keanggotaan gabungan (μgabungan) untuk variabel output y adalah: μgabungan(y)=min(μ1,μ2)

Selanjutnya, hasil gabungan ini akan digunakan dalam proses defuzzifikasi untuk menghasilkan nilai konkret.

Langkah-langkah utama dalam Metode Mamdani adalah sebagai berikut:

- Fuzzifikasi: Pada tahap ini, input dari sistem dikonversi menjadi variabel *fuzzy* dengan mengaplikasikan fungsi keanggotaan yang sesuai. Setiap variabel input akan memiliki satu atau lebih himpunan *fuzzy* yang mewakili nilai-nilai yang mungkin.
- Evaluasi Aturan: Pada tahap ini, aturan-aturan *fuzzy* yang telah ditentukan sebelumnya diterapkan. Setiap aturan memiliki kondisi yang menghubungkan variabel input dengan variabel output, dan aturan-aturan ini dievaluasi untuk setiap kombinasi input.
- Agregasi Hasil: Hasil dari setiap aturan dievaluasi akan diagregasi menjadi satu himpunan *fuzzy* yang menggambarkan variabel output.
- Defuzzifikasi: Tahap terakhir adalah defuzzifikasi, di mana himpunan *fuzzy* hasil dari tahap sebelumnya diubah menjadi nilai konkret yang dapat digunakan dalam pengambilan keputusan atau tindakan selanjutnya.

2. Metode Sugeno

Metode Sugeno adalah pendekatan lain dalam algoritma inferensi *fuzzy*. Metode ini memungkinkan untuk menggabungkan aturan *fuzzy* dengan lebih sederhana dan langsung ke dalam perhitungan matematis.

Metode Sugeno berfokus pada ekspresi matematis yang lebih langsung. Setiap aturan dalam Metode Sugeno memiliki bentuk berikut:

$f(x)=P0+P1*x1+P2*x2+...+Pn*xn$

Di sini, x1,x2,…,xn adalah variabel input, dan P0,P1,P2,…,Pn adalah koefisien yang sesuai dengan variabel input. Hasil dari setiap aturan adalah nilai numerik y yang langsung dapat digunakan sebagai hasil output.

Langkah-langkah utama dalam Metode Sugeno adalah sebagai berikut:

- Fuzzifikasi: Seperti dalam Metode Mamdani, tahap ini melibatkan konversi variabel input menjadi variabel *fuzzy* dengan mengaplikasikan fungsi keanggotaan yang sesuai.
- Evaluasi Aturan: Pada tahap ini, aturan-aturan *fuzzy* diterapkan seperti dalam Metode Mamdani. Namun, perbedaannya terletak pada ekspresi matematis dari aturan-aturan ini. Sebaliknya, aturan-aturan Sugeno berisi fungsi-fungsi linier yang terhubung dengan variabel input.
- Agregasi Hasil: Hasil dari setiap aturan dievaluasi akan diagregasi menjadi satu nilai output.

7.3.2. Algoritma Defuzzifikasi

Setelah proses inferensi *fuzzy* menghasilkan output dalam bentuk variabel *fuzzy*, langkah selanjutnya adalah mengubah variabel *fuzzy* ini menjadi nilai konkret yang dapat digunakan dalam pengambilan keputusan atau tindakan selanjutnya. Proses ini disebut defuzzifikasi. Ada beberapa metode defuzzifikasi yang umum digunakan, di antaranya adalah Metode Centroid dan Metode Average of Maxima.

1. Metode Centroid

Metode Centroid adalah salah satu teknik defuzzifikasi yang paling umum digunakan. Ini menghitung nilai rata-rata dari tingkat keanggotaan variabel *fuzzy* untuk menghasilkan nilai tengah atau pusat dari distribusi himpunan *fuzzy*. Nilai ini kemudian digunakan sebagai nilai defuzzifikasi.

Metode Centroid menggunakan pusat massa atau centroid dari himpunan *fuzzy* sebagai nilai defuzzifikasi. Rumusnya adalah sebagai berikut:

$$y = \frac{\int \text{semua variabel output}(y \cdot \mu(y))}{\int \text{semua variabel output}^{\mu(y)}}$$

Di sini, μ(y) adalah tingkat keanggotaan pada variabel output y.

Langkah-langkah utama dalam Metode Centroid adalah sebagai berikut:

- Hitung Tingkat Keanggotaan: Untuk setiap himpunan *fuzzy* yang berkontribusi pada variabel output, hitung tingkat keanggotaan berdasarkan hasil inferensi *fuzzy*.
- Hitung Pusat Massa: Hitung pusat massa atau centroid dari tingkat keanggotaan yang dihitung pada langkah sebelumnya. Ini dilakukan dengan cara mengambil nilai rata-rata tertimbang dari tingkat keanggotaan, di mana bobotnya adalah tingkat keanggotaan itu sendiri.
- Nilai Defuzzifikasi: Hasil dari langkah sebelumnya adalah nilai defuzzifikasi yang digunakan sebagai output dari sistem *fuzzy logic*.

2. Metode Average of Maxima

Metode defuzzifikasi lainnya adalah Metode Average of Maxima. Dalam metode ini, kita mengambil nilai maksimum dari tingkat keanggotaan variabel *fuzzy* untuk setiap himpunan *fuzzy* yang relevan, dan kemudian menghitung nilai rata-rata dari nilai-nilai maksimum tersebut.

Langkah-langkah utama dalam Metode Average of Maxima adalah sebagai berikut:

- Hitung Tingkat Keanggotaan: Seperti dalam Metode Centroid, hitung tingkat keanggotaan untuk setiap himpunan *fuzzy* yang berkontribusi pada variabel output.

- Identifikasi Nilai Maksimum: Temukan nilai maksimum dari tingkat keanggotaan yang dihitung pada langkah sebelumnya.
- Nilai Defuzzifikasi: Hasil dari langkah sebelumnya adalah nilai defuzzifikasi yang digunakan sebagai output dari sistem *fuzzy logic*.

7.4 Implementasi *Fuzzy Logic* dalam Kehidupan Sehari-hari

Fuzzy logic, sebuah pendekatan matematis yang memungkinkan kita untuk menggambarkan dan mengukur ketidakpastian dan ketidakjelasan dengan cara yang lebih mirip dengan cara berpikir manusia, telah mencapai kesuksesan besar dalam berbagai aplikasi di berbagai bidang. Konsep dasar *fuzzy logic*, seperti himpunan *fuzzy*, variabel *fuzzy*, dan aturan *fuzzy*, telah memberikan alat yang kuat untuk mengatasi situasi yang tidak selalu dapat diukur dengan benar atau salah, tetapi memiliki tingkat keanggotaan yang bervariasi antara 0 dan 1.

Dalam bab ini, kita akan mengeksplorasi lebih dalam bagaimana *fuzzy logic* telah berhasil diimplementasikan dalam berbagai aplikasi di berbagai bidang. Kita akan membahas contoh-contoh konkret, mulai dari sistem kendali otomatis hingga pengolahan citra dan pengambilan keputusan, yang menunjukkan potensi besar dari pendekatan ini dalam menghadapi dunia yang penuh dengan ketidakpastian.

7.4.1 Sistem Kendali *Fuzzy*: Menghadapi Variabilitas Lingkungan

Salah satu bidang yang paling berhasil menerapkan *fuzzy logic* adalah dalam sistem kendali otomatis. *fuzzy logic* digunakan untuk mengendalikan perangkat elektronik seperti mesin cuci, AC, dan bahkan mobil otonom. Konsep dasar dalam penggunaan *fuzzy logic* dalam sistem kendali adalah kemampuannya untuk

menyesuaikan operasinya berdasarkan kondisi lingkungan yang berubah.

1. Sistem Kendali Mesin Cuci

Misalkan kita pertimbangkan sistem kendali mesin cuci. Ketika Anda memasukkan cucian ke dalam mesin cuci dan memilih siklus pencucian, *fuzzy logic* dapat membantu mesin cuci untuk membuat keputusan yang cerdas tentang berapa lama dan seberapa kuat mencuci berdasarkan berbagai faktor seperti jenis cucian, berat cucian, dan tingkat kotoran.

Sebagai contoh, jika mesin cuci mendeteksi bahwa cucian Anda adalah pakaian yang ringan dan tidak terlalu kotor, maka *fuzzy logic* dapat memberikan sinyal kepada mesin untuk menggunakan siklus pencucian yang lebih ringan dan menghemat energi. Namun, jika mesin cuci mendeteksi cucian yang sangat kotor atau berat, maka siklus pencucian yang lebih intens dapat diaktifkan.

Penggunaan *fuzzy logic* dalam sistem kendali mesin cuci memungkinkan mesin cuci untuk "memahami" lebih baik kondisi sebenarnya cucian dan beroperasi dengan efisiensi yang lebih tinggi daripada sistem kontrol konvensional yang bersifat biner (nyala/mati).

2. Sistem Kendali AC

Dalam konteks sistem kendali AC (Air Conditioning), *fuzzy logic* digunakan untuk mengatur suhu dan kecepatan kipas berdasarkan suhu ruangan dan preferensi pengguna. Sebagai contoh, jika suhu ruangan agak panas, tetapi tidak terlalu panas, maka *fuzzy logic* dapat memutuskan untuk menjalankan AC dengan kipas berkecepatan sedang dan mengatur suhu ke tingkat yang nyaman. Namun, jika suhu ruangan sangat panas, maka AC dapat bekerja dengan kipas berkecepatan tinggi dan suhu yang lebih rendah.

Fuzzy logic memungkinkan AC untuk mengambil keputusan berdasarkan situasi yang kompleks dan berubah-ubah, seperti perubahan suhu luar ruangan, jumlah orang di dalam ruangan, dan preferensi pengguna. Ini menghasilkan kenyamanan yang lebih tinggi dan penggunaan energi yang lebih efisien.

3. Sistem Kendali Mobil Otonom

Sistem kendali mobil otonom adalah contoh yang lebih kompleks tentang bagaimana *fuzzy logic* digunakan dalam dunia nyata. Mobil otonom harus mampu mengambil keputusan yang cepat dan tepat berdasarkan situasi di jalan, seperti lalu lintas yang padat, cuaca yang buruk, dan perilaku pengemudi lainnya.

Dalam konteks ini, *fuzzy logic* digunakan untuk menggabungkan informasi dari berbagai sensor seperti lidar, kamera, dan radar. *fuzzy logic* dapat membantu mobil otonom untuk membuat keputusan tentang bagaimana harus bergerak, seperti kapan harus berhenti, berbelok, atau berakselerasi. Ini juga memungkinkan mobil untuk beradaptasi dengan situasi yang berubah-ubah, seperti situasi lalu lintas yang kompleks.

7.4.2 Pengolahan Citra: Mengenali dan Memahami Gambar

Pengolahan citra adalah bidang lain di mana *fuzzy logic* telah memberikan kontribusi besar. *Fuzzy logic* digunakan dalam pengenalan objek dalam gambar dan memahami konteksnya dengan lebih baik. Ini sangat bermanfaat dalam aplikasi pemrosesan gambar medis dan pengenalan pola.

1. Pengenalan Pola

Dalam pengenalan pola, *fuzzy logic* dapat digunakan untuk mengklasifikasikan objek dalam gambar berdasarkan tingkat kesamaannya dengan berbagai himpunan *fuzzy* yang telah ditentukan sebelumnya. Misalnya, dalam sistem pengenalan wajah, *fuzzy logic* dapat membantu dalam menentukan sejauh mana wajah yang dideteksi sesuai dengan berbagai kriteria seperti bentuk mata, hidung, dan bibir. Ini memungkinkan

sistem untuk mengenali objek dengan lebih akurat meskipun ada variasi dalam penampilan objek tersebut.

2. Pengolahan Gambar Medis

Dalam pengolahan gambar medis, *fuzzy logic* digunakan untuk mengidentifikasi struktur anatomi dalam citra medis seperti MRI (Magnetic Resonance Imaging) atau CT (Computed Tomography). *fuzzy logic* memungkinkan sistem untuk mengenali batas antara organ dan jaringan dengan lebih baik, bahkan dalam kasus di mana batas tersebut tidak jelas.

Misalnya, dalam pengolahan citra MRI otak, *fuzzy logic* dapat membantu dalam mengidentifikasi tepi dari berbagai struktur otak seperti korteks, substansia putih, dan lobus otak. Hal ini sangat penting dalam diagnosis penyakit dan perencanaan perawatan medis.

7.4.3 Sistem Keputusan: Menghadapi Ketidakpastian dalam Data

Fuzzy logic juga telah digunakan dalam pengambilan keputusan di berbagai bidang, termasuk keuangan, manajemen risiko, dan logistik. Ini membantu dalam mengatasi ketidakpastian dalam data yang sering kali menjadi tantangan dalam pengambilan keputusan.

1. Manajemen Risiko

Dalam manajemen risiko, *fuzzy logic* digunakan untuk mengukur tingkat risiko dari berbagai keputusan bisnis. Misalnya, dalam industri asuransi, *fuzzy logic* dapat digunakan untuk menilai risiko yang terkait dengan penawaran polis asuransi kepada individu atau perusahaan.

Fuzzy logic memungkinkan para analis risiko untuk mempertimbangkan sejumlah faktor yang mungkin memiliki tingkat ketidakpastian, seperti usia, jenis pekerjaan, atau kondisi kesehatan. Dengan demikian, keputusan dapat diambil dengan

lebih akurat, dan premi asuransi dapat disesuaikan dengan lebih baik.

2. Keuangan

Dalam keuangan, *fuzzy logic* dapat digunakan dalam pengelolaan portofolio investasi. Ketika investor menghadapi berbagai pilihan investasi dengan tingkat risiko yang berbeda, *fuzzy logic* dapat membantu dalam mengidentifikasi portofolio investasi yang paling sesuai dengan tujuan dan toleransi risiko investor.

Fuzzy logic memungkinkan pengelola portofolio untuk menggabungkan berbagai faktor, seperti kinerja historis saham, volatilitas pasar, dan tujuan investasi, dalam pengambilan keputusan. Ini dapat membantu menghasilkan hasil investasi yang lebih baik dan mengurangi risiko.

3. Logistik

Dalam bidang logistik, *fuzzy logic* digunakan untuk mengoptimalkan pengiriman dan rantai pasokan. Pengelola logistik dapat menggunakan *fuzzy logic* untuk mengambil keputusan tentang rute pengiriman, alokasi sumber daya, dan manajemen persediaan.

Fuzzy logic memungkinkan mereka untuk mengatasi ketidakpastian dalam faktor-faktor seperti lalu lintas, cuaca, dan permintaan pelanggan. Dengan demikian, operasi logistik dapat menjadi lebih efisien dan responsif terhadap perubahan kondisi.

DAFTAR PUSTAKA

Arifin Zainal. 2018. *Evaluasi Pembelajara.* Jakarta: Direktorat Jenderal Pendidikan Islam Kementrian Agama RI.

Arikunto Suharsimi. 2001. *Dasar-Dasar Evaluasi Pendidikan*. Jakarta: PT Bumi Aksara.

Asrul, dkk. 2014. *Evaluasi Pembelajaran*. Medan: Cipta pustaka Media.

Danial, D., Nurjannah, N., & Mirna, M. 2019. *Evaluation of The Learning.*

Program of Mathematics Study Program at Islamic Institute Of Muhammadiyah Sinjai. *Matematika Dan Pembelajaran*, *7*(1), 65. https://doi.org/10.33477/mp.v7i1.1046.

Fitriani. 2019. Kemampuan Pemecahan Masalah Siswa SMP. *JTMT: Journal Tadris Matematika*, *01*(01), 25–30. http://journal.iaimsinjai.ac.id/index.php/Jtm/article/view/393.

Hamalik, Oemar. 2010. *Manajemen Pengembangan Kurikulum*. Bandung : PT Remaja Rosdakarya.

Elis Ratnawulan. 2014. *Evaluasi Pembelajaran*. Bandung: Penerbit pustaka.

Ida Farida. 2017. *Evaluasi Pembelajaran*. Bandung:PT Remaja Rosdakarya.

Muhammad Afandi.2013. *Evaluasi Pembelajaran Sekolah Dasar*. Semarang:UNISSULA Press

Sukardi. 2009. *Evaluasi Pendidikan.* Jakarta: PT Bumi Aksara.

Tim Pengembangan MKDP Kurikulum dan Pembelajaran. (2011).*Kurikulum dan Pembelajaran*. Jakarta Utara: PT Raja Grafindo Persada.

BAB 8
Algoritma Genetika dan Implementasinya

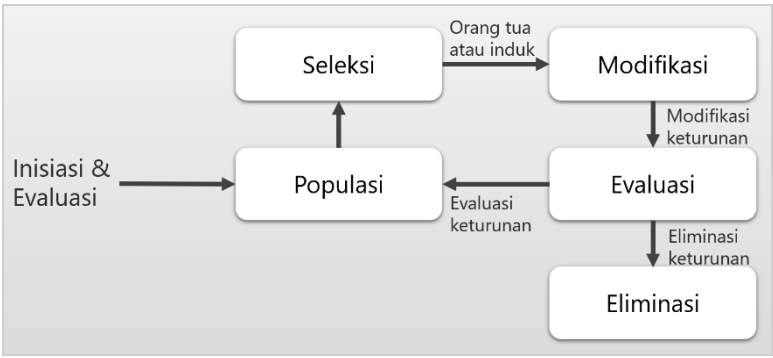
Oleh Jayanti Yusmah Sari

8.1 Pendahuluan

Algoritma merupakan serangkaian instruksi yang diulang-ulang untuk memecahkan suatu masalah. Sedangkan genetika adalah ilmu yang mempelajari tentang gen, variasi genetik, dan keturunan pada organisme hidup. Dengan demikian, definisi Algoritma Genetika (Genetic Algorithms/GAs) yakni program komputer yang meniru proses biologis evolusi untuk memecahkan masalah dan memodelkan sistem evolusi (Mitchell, 1995).

Pada tahun 1960an John Holland pertama kali mengemukakan konsep Algoritma Genetika. Selama kurun waktu 10 tahun, algoritma ini dikembangkan oleh John Holland bersama mahasiswa dan koleganya di Universitas Michigan. Pengembangan Algoritma Genetika bertujuan untuk memahami fenomena "adaptasi" yang terjadi di alam dan mengembangkan cara agar mekanisme adaptasi alami dapat diimplementasikan ke dalam program komputer. Algoritma Genetika ini sebenarnya turunan dari prinsip genetika dan seleksi alam (paradigma Darwin), yang akan menyeleksi individu yang lebih kuat sehingga dapat bertahan dalam lingkungan yang kompetitif dan menghasilkan solusi yang optimal. Gambaran siklus evolusioner ini dapat dilihat pada Gambar 9.1.

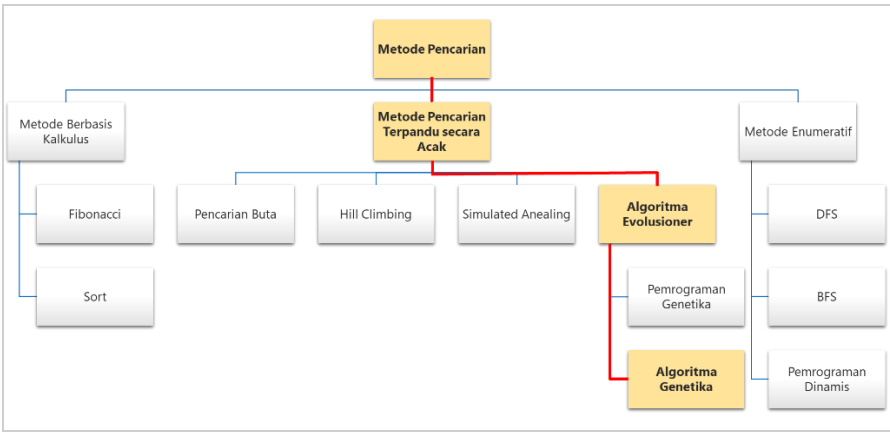
Sekitar tahun 1975, John Holland berhasil menyajikan Algoritma Genetika sebagai abstraksi evolusi biologis dan memberikan kerangka teoritis untuk adaptasi berdasarkan Algoritma Genetika dalam buku yang ditulisnya (Holland, 1992). Dalam buku tersebut, dijelaskan bahwa Algortima Genetika adalah metode untuk berpindah dari satu populasi ”kromosom” (misalnya rangkaian “bit/gen” yang mewakili kandidat solusi untuk suatu masalah) ke populasi kromosom lain yang baru, menggunakan proses “seleksi” genetika dan operator persilangan (crossover), dan mutasi. Setiap kromosom disusun oleh “gen” (misalnya, bit), sedangkan setiap gen merupakan turunan dari “alel” tertentu (misalnya, 0 atau 1). Operator seleksi akan memilih kromosom-kromosom dalam populasi yang akan diizinkan untuk bereproduksi. Kromosom yang dipilih adalah yang memiliki nilai fitness terbaik sehingga akan menghasilkan lebih banyak keturunan dibandingkan kromosom yang memiliki nilai fitness yang kurang baik. Crossover (persilangan) akan menukarkan subbagian dari dua kromosom, yang secara umum meniru rekombinasi biologis antara dua organisme berkromosom tunggal (“haploid”). Mutasi akan secara acak mengubah nilai alel beberapa lokasi di kromosom; dan inversi akan membalikkan urutan yang berdekatan bagian kromosom, sehingga urutan susunan gen akan ditata ulang (Mitchell, 1995).



Gambar 8.1 Siklus Evolusioner

(Sumber : (Daga and Garibaldi, 2020))

Algoritma Genetika juga dikenal sebagai algoritma evolusioner karena menunjukkan pengorganisasian diri dan adaptasi serupa dengan cara organisme biologis terkuat bertahan hidup dan bereproduksi. Algoritma Genetika dikategorikan sebagai metode pencarian global heuristik dan termasuk dalam kelas algoritma optimasi probabilistik. Gambar 9.2 menunjukkan posisi Algoritma Genetika dalam kelas metode pencarian (*classes of search techniques*).



Gambar 8.2 Kelas Metode Pencarian
(Sumber :Kühn *et al.*, 2016)

8.2 Istilah Penting dalam Algoritma Genetika

Seperti yang telah diuraikan sebelumnya, Algoritma Genetika akan menelusuri ruang "kromosom", yang masing-masing mewakili kandidat solusi untuk masalah tertentu (dalam beberapa kasus, solusi terdiri dari atas kumpulan kromosom). Kebanyakan metode yang disebut Algoritma Genetika memiliki setidaknya elemen umum berikut: populasi kromosom, seleksi berdasarkan nilai *fitness*, persilangan untuk menghasilkan keturunan baru, dan mutasi acak pada keturunan baru. Gambar

9.3 berikut menunjukkan beberapa istilah (terminologi) yang akan dijumpai dalam penggunaan Algoritma Genetika.

1. Populasi (population): sekelompok individu yang merupakan alternatif solusi yang mungkin untuk suatu masalah tertentu. Satu siklus proses evolusi akan memproses populasi secara bersama-sama.

Gambar 8.3 Istilah dalam Algoritma Genetika

2. Individu (individual): solusi yang mungkin, kandidat solusi, sekumpulan gen, kadang disamakan dengan kromosom.
3. Generasi (generation): satu siklus proses evolusi.
4. Gen (Genes): solusi untuk masalah direpresentasikan sebagai sekumpulan parameter yang dikenal sebagai gen, variabel dasar yang membentuk suatu kromosom. Gen ini bisa berupa nilai biner, bulat, pecahan, maupun karakter.

5. Kromosom (chromosome): blueprint dari seorang individu, gabungan gen-gen yang membentuk serangkaian nilai.
6. Nilai Fitness (Fitness score/value): setiap kromosom mempunyai nilai fitness yang diperoleh dari kromosom itu sendiri dengan menggunakan fungsi fitness. Nilai fitness menunjukkan seberapa baik nilai dari suatu individu atau solusi yang didapatkan. Nilai inilah yang dijadikan acuan untuk mencapai nilai optimal.
7. Alel (Allele): kemungkinan pengaturan gen (hitam, pirang, dll.), nilai gen yang biasanya berupa nilai biner, pecahan, bilangan bulat serta karakter.
8. Lokus (Locus): posisi suatu gen pada kromosom.
9. Genom (Genome): kumpulan semua kromosom untuk seorang individu.
10. Mutasi (Mutation): operator yang akan membalik beberapa bit dalam kromosom secara acak.
11. Persilangan (Crossover): operator persilangan (pindah silang) dua kromosom untuk menghasilkan dua kromosom keturunan.
12. Seleksi (Selection): operator yang akan memilih kromosom untuk reproduksi dalam suatu populasi.

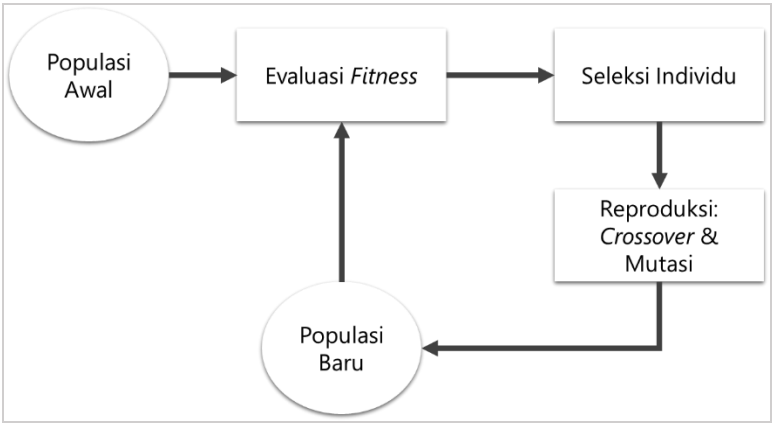
8.3 Implementasi Algoritma Genetika

Implementasi Algoritma Genetika sebagai simulasi komputer akan membangkitkan populasi representasi abstrak (kromosom/genom) dari kandidat solusi (individu) terhadap masalah optimasi yang seterusnya berevolusi menjadi solusi yang paling optimal. Algoritma Genetika sering digunakan untuk meningkatkan kinerja metode kecerdasan buatan lainnya seperti sistem pakar atau jaringan saraf. Algoritma Genetika melakukan pembelajaran dengan menghasilkan keturunan yang

semakin baik yang diukur dengan fungsi fitness, yaitu ukuran tujuan yang ingin dicapai (maksimum atau minimum).

8.3.1 Siklus Algoritma Genetika

David Goldberg mengemukakan konsep siklus Algoritma Genetika yang dimulai dari pembangkitan populasi awal secara acak. Kemudian untuk setiap individu dihitung nilai *fitness*-nya. Selanjutnya, individu terbaik diseleksi, untuk melakukan persilangan dan mutasi hingga menghasilkan populasi baru. Siklus Algoritma Genetika ini akan berulang untuk populasi baru yang dihasilkan hingga generasi ke–*n* seperti ditunjukkan pada Gambar 9.4 dan dapat diimplementasikan dalam bentuk *pseudo code* seperti Gambar 9.5.



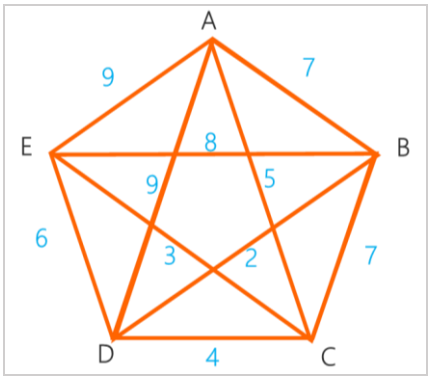
Gambar 8.4 Siklus Algoritma Genetika
(Sumber : Putra, 2018)

```
Genetic_Algorithm() {
                Initialize random population;
                Evaluate the population;
                Generation = 0;
     While termination criterion is not
     satisfied {
            Generation = Generation + 1;
            Select good chromosomes by
            reproduction procedure;
            Perform crossover with probability
            crossover (Pc);
            Perform mutation with probability
            of mutation (Pm);
            Evaluate the population;
     }
}
```

Gambar 8.5 Pseudo Code Algoritma Genetika
(Sumber :bin Peeyee *et al.*, 2019)

8.3.2 Contoh Penyelesaian Studi Kasus dengan Algoritma Genetika

Contoh studi kasus TSP (Traveling Salesman Problem), untuk menentukan rute terpendek dari lintasan berikut (dengan titik asal adalah Kota A) yang akan diselesaikan menggunakan cara manual dan Algoritma Genetika.



Gambar 8.6 Contoh Kasus TSP dengan 5 Titik Kota

Penyelesaian manual menggunakan logika sederhana.

Diketahui terdapat 5 kota yang akan dilalui, maka jumlah titik (n) = 5, sehingga jumlah kemungkinan lintasan = (n-1)!/2 = 4!/2 = 12 lintasan. Jika Kota A merupakan titik asal sekaligus titik akhir, maka kemungkinan lintasan beserta jaraknya masing-masing ditunjukkan pada Gambar 9.6.

Dengan cara manual, diperoleh panjang jalur lintasan paling minimum yaitu **23**, pada **lintasan ke-4 = (A-B-D-E-C-A) atau (A-C-E-D-B-A).** Jumlah tahapan dengan penyelesaian manual dapat dihitung dengan (n - 1)!. Dengan demikian, nilai n akan berbanding lurus dengan waktu yang dibutuhkan, semakin besar nilai n maka waktu yang dibutuhkan untuk mendapatkan panjang lintasan minimumnya juga semakin lama.

1. Lintasan 1 = (a b c d e a) atau (a e d c b a) panjang lintasan = 7 + 7 + 4 + 6 + 9 = 33
2. Lintasan 2 = (a b c e d a) atau (a d e c b a) panjang lintasan = 7 + 7 + 3 + 6 + 9 = 32
3. Lintasan 3 = (a b d c e a) atau (a e c d b a) panjang lintasan = 7 + 2 + 4 + 3 + 9 = 25
4. Lintasan 4 = (a b d e c a) atau (a c e d b a) panjang lintasan = 7 + 2 + 6 + 3 + 5 = 23
5. Lintasan 5 = (a b e c d a) atau (a d c e b a) panjang lintasan = 7 + 8 + 3 + 4 + 9 = 31
6. Lintasan 6 = (a b e d c a) atau (a c d e b a) panjang lintasan = 7 + 8 + 6 + 4 + 5 = 30
7. Lintasan 7 = (a c b d e a) atau (a e d b c a) panjang lintasan = 5 + 7 + 2 + 6 + 9 = 29
8. Lintasan 8 = (a c b e d a) atau (a d e b c a) panjang lintasan = 5 + 7 + 8 + 6 + 9 = 35
9. Lintasan 9 = (a c d b e a) atau (a e b d c a) panjang lintasan = 5 + 4 + 2 + 8 + 9 = 28
10. Lintasan 10 = (a d b c e a) atau (a e c b d a) panjang lintasan = 9 + 2 + 7 + 3 + 9 = 30
11. Lintasan 11 = (a d b e c a) atau (a c e b d a) panjang lintasan = 9 + 2 + 8 + 3 + 5 = 27
12. Lintasan 12 = (a d c b e a) atau (a e b c d a) panjang lintasan = 9 + 4 + 7 + 8 + 9 = 37

Gambar 8.7 Contoh Jalur Lintasan untuk Kasus TSP

Penyelesaian dengan Algoritma Genetika.

Dengan contoh kasus TSP yang sama (Gambar 9.5), berikut tahap-tahap penyelesaian menggunakan Algoritma Genetika.

1. Inisialisasi

Misal terdapat 6 populasi dalam satu generasi yang dibangkitkan secara acak, yaitu :

Kromosom[1] = [B D E C]
Kromosom[2] = [D B E C]
Kromosom[3] = [C B D E]
Kromosom[4] = [E B C D]
Kromosom[5] = [E C B D]
Kromosom[6] = [C D E B]

2. Evaluasi kromosom

   Selanjutnya, dihitung nilai *fitness* untuk masing-masing kromosom dari tahap inisialisasi, dengan cara menghitung bobot untuk masing-masing lintasan (dalam hal ini adalah jarak):

   Fitness[1] = AB+BD+DE+EC+CA = 7 + 2 + 6 + 3 + 5 = 23
   Fitness[2] = AD+DB+BE+EC+CA = 9 + 2 + 8 + 3 + 5 = 27
   Fitness[3] = AC+CB+BD+DE+EA = 5 + 7 + 2 + 6 + 9 = 29
   Fitness[4] = AE+EB+BC+CD+DA = 9 + 8 + 7 + 4 + 9 = 37
   Fitness[5] = AE+EC+CB+BD+DA = 9 + 3 + 7 + 2 + 9 = 30
   Fitness[6] = AC+CD+DE+EB+BA = 5 + 4 + 6 + 8 + 7 = 30

3. Seleksi Kromosom

   Dengan menggunakan persamaan: Q[i] = 1 / Fitness[i]
   maka diperoleh:

   Q[1] = 1 / 23 = 0,043
   Q[2] = 1 / 27 = 0,037
   Q[3] = 1 / 29 = 0,034
   Q[4] = 1 / 37 = 0,027
   Q[5 = 1 / 30 = 0,033
   Q[6] = 1 / 30 = 0,033

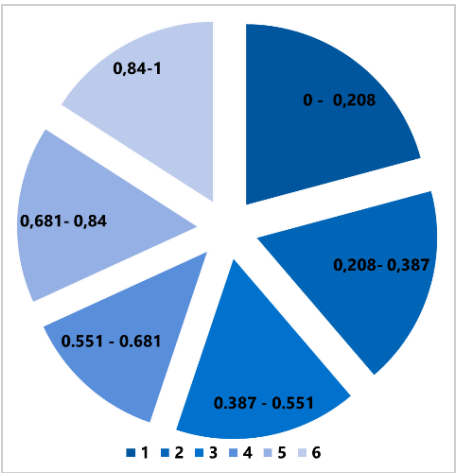
   Total = 0,043 + 0,037 + 0,034 + 0,027 + 0,033 + 0,033 = 0,207

   Kemudian hitung Probabilitas Seleksi menggunakan persamaan: P[i] = Q[i] / Total, sehingga diperoleh:

P[1] = 0,043 / 0,207 = 0,208
P[2] = 0,037 / 0,207 = 0,179
P[3] = 0,034 / 0,207 = 0,164
P[4] = 0,027 / 0,207 = 0,130
P[5] = 0,033 / 0,207 = 0,159
P[6] = 0,033 / 0,207 = 0,159

Selanjutnya, hitung Probabilitas Kumulatif:
C[1] = 0,208
C[2] = 0,208+0,179 = 0,387
C[3] = 0,387+0,164 = 0,551
C[4] = 0,551+0,130 = 0,681
C[5] = 0,681+0,159 = 0,840
C[6] = 0,840+0,159 = 1

Hasil probabilitas kumulatif yang diperoleh dibuat dalam bentuk Roulette Wheel pada Gambar 9.7. Selanjutnya, akan dibangkitkan 6 bilangan random R[i]:
R[1] = 0,314
R[2] = 0,111,
R[3] = 0,392,
R[4] = 0,743
R[5] = 0,571,
R[6] = 0,891



Gambar 8.8 *Roulette Wheel*

maka induk yang dipilih adalah C[2], C[1], C[3], C[5], C[4], dan C[6] sehingga populasi baru akan terbentuk.

Populasi Awal			Populasi Baru	
C[1]	**Kromosom[1]** = [B D E C]	→	C[2]	**Kromosom[2]** = [D B E C]
C[2]	**Kromosom[2]** = [D B E C]	→	C[1]	**Kromosom[1]** = [B D E C]
C[3]	**Kromosom[3]** = [C B D E]	→	C[3]	**Kromosom[3]** = [C B D E]
C[4]	**Kromosom[4]** = [E B C D]	→	C[5]	**Kromosom[5]** = [E C B D]
C[5]	**Kromosom[5]** = [E C B D]	→	C[4]	**Kromosom[4]** = [E B C D]
C[6]	**Kromosom[6]** = [C D E B]	→	C[6]	**Kromosom[6]** = [C D E B]

4. Persilangan (*crossover*)

   Pada tahap ini, misal ditentukan parameter *crossover probability* (ρc) sebesar 25%, apabila bilangan random yang dibangkitkan <0,25 maka bilangan tersebut akan menjadi induk baru. Misal hasil 6 bilangan random yang dibangkitkan yaitu:

   R[1] = 0,451
   R[2] = 0,211
   R[3] = 0,202
   R[4] = 0,877
   R[5] = 0,771
   R[6] = 0,131

   Kromosom ke-*i* yang dipilih menjadi induk adalah kromosom yang memenuhi persamaan *R*[*i*]<ρc = *R*[*i*]<0,25. Maka yang akan dijadikan induk adalah Kromosom[2], Kromosom[3], dan Kromosom[6]. Berikut ilustrasi proses persilangan dari ketiga kromosom ini.

   Kromosom[2] = **Kromosom[2]** >< **Kromosom[3]**
   = [B D E C] >< [C B D E]
   = [B D C E]
   Kromosom[3] = **Kromosom[3]** >< **Kromosom[6]**
   = [C B D E] >< [C D E B]
   = [C D E B]
   Kromosom[6] = **Kromosom[6]** >< **Kromosom[2]**
   = [C D E B] >< [B D E C]
   = [C D B E]

   Hasil dari persilangan (*crossover*) tersebut adalah:

Kromosom[1] = [D B E C] diambil dari kromosom induk 1

Kromosom[2] = [B D C E] diambil dari hasil crossover kromosom 2 dan 3

Kromosom[3] = [C D E B] diambil dari hasil crossover kromosom 3 dan 6

Kromosom[4] = [E C B D] diambil dari kromosom induk 4

Kromosom[5] = [E B C D] diambil dari kromosom induk 5

Kromosom[6] = [C D B E] diambil dari hasil crossover kromosom 6 dan 2

5. Mutasi

   Sebelum melakukan mutasi, terlebih dahulu hitung panjang total gen dengan mengalikan jumlah gen dalam 1 kromosom dengan jumlah kromosom. Sehingga untuk contoh kasus ini, panjang total gen = 4*6 = 24. Jika dimisalkan probabilitas mutasi (ρm) = 20%, maka jumlah gen yang akan dimutasi adalah sebanyak = 0,2*24 = 4,8 (dibulatkan menjadi 5). Selanjutnya, posisi mutasi diperoleh dengan membangkitkan 5 bilangan acak (misal 3, 7, 10, 20, 24) maka berikut proses mutasi yang terjadi:

Kromosom[1] = [D B **E** C] →	**Kromosom[1]** = [D B **C** E]
Kromosom[2] = [B D **E** C] →	**Kromosom[2]** = [B D **C** E]
Kromosom[3]= [C **B** D E] →	**Kromosom[3]** = [C **E** D B]
Kromosom[4] = [E C B D] →	**Kromosom[4]** = [E C B D]
Kromosom[5] = [E B C **D**] →	**Kromosom[5]** = [D B C **E**]
Kromosom[6] = [C D E **B**] →	**Kromosom[6]** = [B D E **C**]

6. Evaluasi

   Kromosom baru hasil proses mutasi selanjutnya dievaluasi dengan menghitung nilai *fitness*-nya. Karena masalah yang akan diselesaikan adalah pencarian rute terpendek, maka dipilih kromosom dengan nilai *fitness* paling minimum (23) yaitu kromosom[3] dan kromosom[6]. Dengan demikian, rute terpendek untuk kasus TSP pada Gambar 9.5 adalah A-C-E-D-B-A atau A-B-D-E-C-A.

Kromosom[1] = [D B C E] = 3+2+7+3+9 = 24

Kromosom[2] = [B D C E] = 7+2+4+3+9 = 25

Kromosom[3] = [C E D B] = 5+3+6+2+7 = 23

Kromosom[4] = [E C B D] = 9+3+7+2+9 = 30

Kromosom[5] = [D B C E] = 9+2+7+3+9 = 30

Kromosom[6] = [B D E C] = 7+2+6+3+5 = 23

8.4 Pemanfaatan Algoritma Genetika

Contoh implementasi Algoritma Genetika yang dijelaskan pada sub bab sebelumnya sangat sederhana, namun sebenarnya Algoritma Genetika bisa digunakan dalam sejumlah besar masalah dan model ilmiah, teknik, serta yang termasuk dalam bidang:

1. Optimasi: Algoritma Genetika telah diterapkan dalam berbagai macam tugas optimasi numerik dan optimasi kombinatorial seperti rangkaian tata letak dan penjadwalan shift kerja.
2. Automatic Programming: Algoritma Genetika juga digunakan untuk mengembangkan program komputer, dan untuk merancang struktur komputasi lainnya, seperti automata seluler dan menyortir jaringan.
3. Machine learning: Pemanfaatan Algoritma Genetika untuk banyak aplikasi machine learning termasuk tugas klasifikasi dan prediksi seperti prediksi cuaca atau struktur protein. Algoritma Genetika juga telah digunakan untuk mengembangkan sistem machine learning tertentu, seperti penentuan bobot untuk jaringan saraf tiruan, aturan untuk mempelajari sistem pengklasifikasi, atau sistem produksi simbolik, dan sensor untuk robot.

4. Model ekonomi: Algoritma Genetika telah digunakan untuk memodelkan proses inovasi, pengembangan strategi penawaran, dan kemunculan pasar ekonomi.
5. Model sistem imun: Algoritma Genetika telah digunakan untuk memodelkan berbagai sistem kekebalan alami termasuk mutasi somatik selama hidup seseorang dan penemuan keluarga multi-gen selama masa evolusi.
6. Model ekologi: Pemanfaatan Algoritma Genetika untuk memodelkan fenomena ekologi seperti perlombaan senjata evolusioner, ko-evolusi inang-parasit, simbiosis, dan aliran sumber daya dalam ekologi.

8.5 Ringkasan

Berikut beberapa hal penting terkait dengan Algoritma Genetika.

1. Algoritma Genetika adalah prosedur berulang yang merepresentasikan kandidat solusinya sebagai rangkaian gen yang disebut kromosom.
2. Algoritma Genetika umumnya diterapkan pada ruang masalah yang sangat besar.
3. Evolusi dimulai dari pembangkitan populasi individu secara acak dan terus berulang dari generasi ke generasi. Dalam setiap generasi, dilakukan evaluasi nilai fitness dari individu dalam populasi. Beberapa individu dipilih dari populasi saat ini (berdasarkan nilai fitness-nya), dan dimodifikasi (dengan persilangan dan mutasi) untuk membentuk populasi baru. Populasi baru kemudian digunakan pada iterasi selanjutnya.
4. Umumnya, iterasi Algoritma Genetika berhenti ketika telah mencapai jumlah generasi maksimum atau nilai fitness yang optimal untuk populasi.
5. Jika iterasi berhenti karena jumlah generasi maksimum, solusi yang optimal mungkin tercapai atau mungkin juga tidak tercapai.

6. Umumnya, solusi direpresentasikan dalam string biner 0 dan 1, namun pengkodean lain juga dimungkinkan.
7. Solusi pada Algoritma Genetika dapat diperoleh kapanpun karena solusi dihasilkan pada generasi ke-n.
8. Algoritma Genetika membutuhkan waktu yang relatif lebih efisien karena tidak semua kemungkinan solusi diuji, sesuai dengan kriteria yang ingin dicapai.
9. Metode pencarian non-evolusioner dapat digunakan jika jumlah n kecil, namun jika n-nya besar maka akan penyelesaiannya akan membutuhkan waktu yang lama jika dibandingkan dengan Algoritma Genetika.

DAFTAR PUSTAKA

Daga, A.P. and Garibaldi, L. (2020) 'GA-adaptive template matching for offline shape motion tracking based on edge detection: IAS estimation from the SURVISHNO 2019 challenge video for machine diagnostics purposes', Algorithms, 13(2), p. 33.

Holland, J.H. (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press.

Kühn, M. et al. (2016) 'Investigation Of Genetic Operators And Priority Heuristics for Simulation Based Optimization Of Multi-Mode Resource Constrained Multi-Project Scheduling Problems (MMRCMPSP).', in ECMS, pp. 481–487.

Mitchell, M. (1995) 'Genetic algorithms: An overview.', in Complex. Citeseer, pp. 31–39.

bin Peeyee, M.W. et al. (2019) 'Heuristic based model for groceries shopping navigator', Indonesian Journal of Electrical Engineering and Computer Science, 16(2), pp. 932–940.

Putra, I.M.S. (2018) 'Penerapan Algoritma Genetika Dan Implementasi Dalam MATLAB', vol, 53, pp. 1689–1699.

BAB 9
JARINGAN SARAF TIRUAN (*ARTIFICIAL NEURAL NETWORK*) TEORI, ALGORITMA, DAN IMPLEMENTASINYA

Oleh Fahmy Rinanda Saputri

9.1 Konsep Dasar Jaringan Syaraf Tiruan (Artificial Neural Network) dan Pemodelannya

Jaringan syaraf tiruan (artificial neural network) adalah model matematika yang terinspirasi dari struktur dan fungsi jaringan syaraf biologis manusia. Tujuan utama dari jaringan syaraf tiruan adalah untuk memproses informasi seperti otak manusia, dengan kemampuan untuk mempelajari pola-pola kompleks dari data, mengenali pola, dan membuat keputusan berdasarkan data yang diberikan. Berikut ini adalah beberapa konsep utama terkait dengan jaringan syaraf tiruan:

1. Neuron (Neuron Tiruan): Neuron adalah unit dasar dalam jaringan syaraf tiruan. Neuron tiruan memiliki beberapa input, menerapkan fungsi matematis pada input tersebut, dan menghasilkan output. Neuron ini mirip dengan neuron biologis dalam hal menerima, memproses, dan mengirimkan sinyal.
2. Bobot (Weight): Setiap koneksi antara neuron memiliki bobot yang mengindikasikan kekuatan hubungan antara neuron tersebut. Bobot ini disesuaikan selama proses pembelajaran jaringan untuk mengoptimalkan kinerja jaringan.

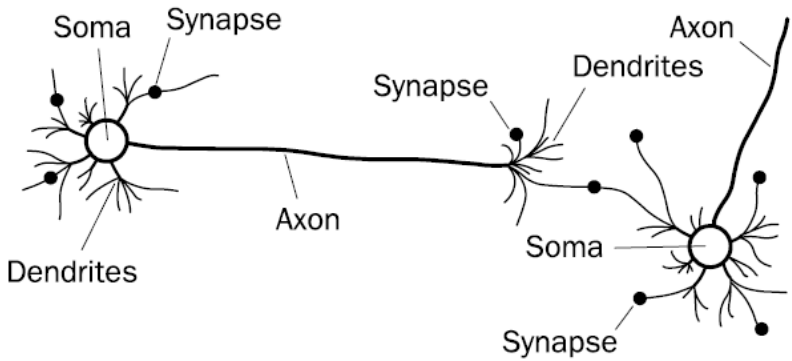
3. Fungsi Aktivasi: Neuron dalam jaringan syaraf tiruan menggunakan fungsi aktivasi untuk mengatur output mereka. Beberapa fungsi aktivasi umum meliputi sigmoid, ReLU (Rectified Linear Unit), dan tangen hiperbolik. Fungsi aktivasi ini membantu mengenali pola dalam data.
4. Arsitektur Jaringan: Jaringan syaraf tiruan terdiri dari lapisan-lapisan neuron yang saling terhubung. Lapisan input menerima data masukan, lapisan tersembunyi (hidden layer) digunakan untuk memproses informasi, dan lapisan output menghasilkan hasil prediksi atau keputusan.
5. Pembelajaran: Jaringan syaraf tiruan belajar dari data dengan mengoptimalkan bobot-bobotnya. Proses ini dapat dilakukan dengan berbagai algoritma pembelajaran, seperti backpropagation, yang menggunakan gradien turunan untuk mengurangi kesalahan prediksi jaringan terhadap data pelatihan.
6. Deep Learning: Jika jaringan syaraf tiruan memiliki beberapa lapisan tersembunyi, disebut sebagai jaringan syaraf tiruan dalam kedalaman (deep neural network atau deep learning). Deep learning telah menjadi populer dalam berbagai aplikasi seperti pengenalan wajah, pemrosesan bahasa alami, dan pengenalan pola.
7. Aplikasi: Jaringan syaraf tiruan digunakan dalam berbagai bidang, termasuk pengenalan gambar, pemrosesan bahasa alami, prediksi keuangan, permainan komputer, kendaraan otonom, pengenalan suara, dan banyak lagi.

Apa itu jaringan syaraf?

Jaringan saraf adalah sebuah model penalaran yang bersumber dari struktur otak manusia. Otak terdiri dari sekelompok sel saraf yang erat terhubung satu sama lain, yang disebut neuron, yang merupakan unit dasar pemrosesan informasi. Otak manusia memiliki hampir 10 miliar neuron dan 60 triliun koneksi sinapsis yang menghubungkannya. Dengan memanfaatkan banyak neuron secara bersamaan, otak dapat menjalankan tugasnya dengan jauh lebih cepat dibandingkan

dengan komputer tercepat yang ada saat ini (Negnevitsky, 2005).

Walaupun struktur setiap neuron sangat sederhana, kolaborasi elemen-elemen ini menghasilkan kapasitas pemrosesan yang luar biasa. Neuron itu sendiri terdiri dari tiga komponen utama, yaitu tubuh sel (soma), serat-serat bercabang yang dikenal sebagai dendrit, dan serat panjang yang disebut akson. Sementara dendrit menyebar dan membentuk jaringan di sekitar soma, akson meluas menuju dendrit dan soma dari neuron lain. Gambar 10.1 menggambarkan secara visual struktur jaringan saraf ini.



Gambar 9.1 Jaringan Syaraf Biologis
(Sumber : Negnevitsky, 2005)

Sinyal dalam jaringan saraf mengalir dari satu neuron ke neuron lainnya melalui proses elektrokimia yang sangat kompleks. Sinapsis, tempat di mana dua neuron berhubungan, memainkan peran penting dalam proses ini dengan melepaskan zat kimia yang memicu perubahan dalam potensial listrik tubuh sel. Ketika potensial listrik mencapai ambang batas tertentu, suatu impuls listrik, yang disebut potensial aksi, dipicu dan dihantarkan melalui akson.

Denyut nadi ini dapat menyebar ke seluruh jaringan dan akhirnya mencapai sinapsis, yang kemudian dapat menguatkan atau melemahkan koneksi-koneksi di antara neuron. Hal yang menarik adalah jaringan saraf juga menunjukkan plastisitas, yang berarti bahwa neuron dapat mengalami perubahan jangka panjang dalam kekuatan koneksi mereka sebagai respons terhadap pola rangsangan yang berubah. Selain itu, neuron memiliki kemampuan untuk membentuk koneksi baru dengan neuron lain dan bahkan seluruh kelompok neuron kadang-kadang dapat bergerak dari satu tempat ke tempat lain. Semua ini menjadi dasar dari proses pembelajaran di otak.

Otak kita bisa dianggap sebagai sistem pemrosesan informasi yang sangat kompleks, nonlinier, dan berjalan secara paralel. Informasi tidak hanya disimpan dan diproses di satu lokasi tetapi tersebar dan diproses bersamaan di seluruh jaringan saraf. Dengan kata lain, dalam jaringan saraf, data dan pemrosesannya bersifat global, bukan lokal.

Berkat plastisitasnya, hubungan antar neuron yang mengarah ke "jawaban benar" diperkuat sementara yang mengarah ke "jawaban salah" melemah. Hasilnya, jaringan saraf memiliki kemampuan untuk belajar melalui pengalaman. Kemampuan untuk belajar adalah salah satu karakteristik mendasar dan esensial dari jaringan saraf biologis, yang telah menjadi sumber inspirasi dalam upaya-upaya untuk meniru jaringan saraf biologis dalam komputer.

Meskipun saat ini jaringan syaraf tiruan (JST) masih jauh dari meniru otak manusia dengan sempurna, perbandingannya dapat diibaratkan seperti perbandingan antara pesawat kertas dengan pesawat jet supersonik. Ini adalah langkah besar menuju perkembangan teknologi yang menjanjikan. JST memiliki kemampuan untuk 'belajar', yaitu menggunakan pengalaman yang diperoleh untuk meningkatkan kinerjanya. Ketika diberikan jumlah sampel data yang cukup, JST mampu

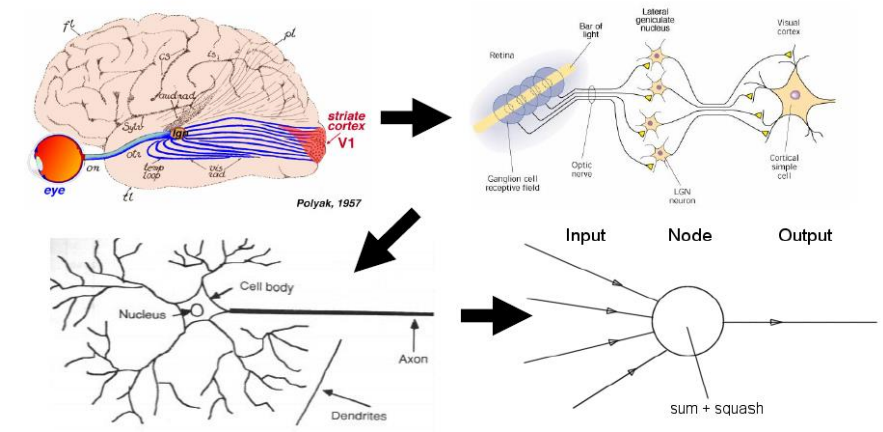
menggeneralisasi dan menerapkan pengetahuannya pada sampel-sampel lain yang belum pernah mereka temui sebelumnya. Mereka dapat digunakan untuk mengidentifikasi tulisan tangan, mengenali kata-kata dalam percakapan manusia, dan bahkan mendeteksi substansi peledak di bandara. Selain itu, JST memiliki potensi untuk mengamati pola-pola yang mungkin tidak dapat dikenali oleh para ahli. Sebagai contoh, Chase Manhattan Bank menggunakan jaringan saraf untuk memeriksa data terkait penggunaan kartu kredit yang mencurigakan, dan hasilnya menunjukkan bahwa penjualan yang paling mencurigakan adalah sepatu wanita dengan harga berkisar antara $40 dan $80 (Negnevitsky, 2005).

Bagaimana JST memodelkan otak?

Jaringan saraf tiruan (artificial neural networks) adalah model matematis yang berupaya memodelkan beberapa aspek dasar dari cara otak manusia bekerja. Ini dilakukan dengan merinci beberapa konsep utama dari sistem saraf biologis ke dalam konteks komputasi. Salah satu konsep utama adalah neuron tiruan, yang berfungsi mirip dengan neuron biologis. Neuron-neuron ini menerima sinyal input, mengalikannya dengan bobot yang dapat disesuaikan, dan menerapkan fungsi aktivasi untuk menghasilkan output. Konsep ini mencoba meniru bagaimana neuron biologis dalam otak manusia menerima, memproses, dan mengirimkan sinyal.

Selanjutnya, jaringan syaraf tiruan mengadopsi ide hubungan dan bobot. Seperti sinapsis antara neuron dalam otak manusia, jaringan syaraf tiruan memiliki koneksi-koneksi antara neuron-neuron dengan bobot yang mengatur sejauh mana sinyal dari satu neuron memengaruhi neuron lain. Dengan mengontrol bobot-bobot ini, jaringan syaraf tiruan mencoba mereplikasi cara otak manusia mengatur kekuatan koneksi antar-neuron, yang sangat penting dalam pemrosesan informasi yang kompleks.

Selain itu, jaringan syaraf tiruan menggunakan arsitektur lapisan. Ini mencakup lapisan-lapisan seperti lapisan input, lapisan tersembunyi, dan lapisan output, yang menyerupai hierarki lapisan fungsi yang ada dalam otak manusia. Lapisan-lapisan ini berkontribusi pada pemrosesan dan pemahaman data yang semakin kompleks. Melalui pembelajaran dan penyesuaian bobot, jaringan syaraf tiruan belajar dari data pelatihan, mencoba meniru kemampuan otak manusia dalam mempelajari pola dan beradaptasi dengan pengalaman. Meskipun jaringan syaraf tiruan mencoba memodelkan beberapa aspek otak, perlu dicatat bahwa mereka masih jauh lebih sederhana daripada otak manusia yang sangat kompleks dalam pemahaman, kreativitas, dan kemampuan adaptasi yang luar biasa.



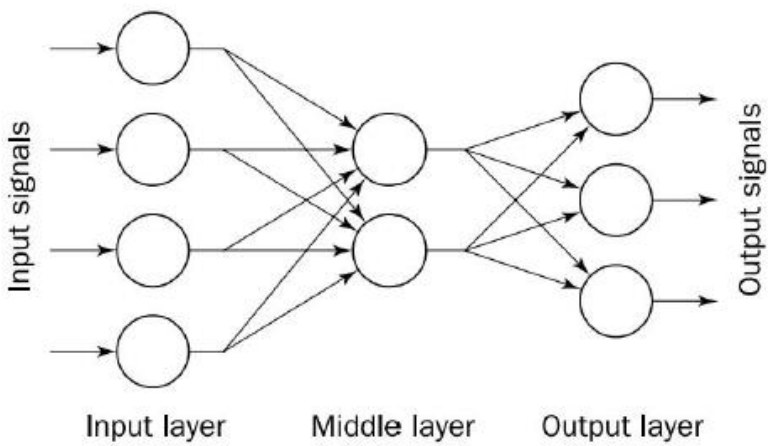
Gambar 9.2 Struktur Dasar Jaringan Syaraf Tiruan

Gambar 10.2 adalah struktur jaringan syaraf tiruan. Struktur jaringan syaraf tiruan (JST) didasarkan pada konsep dasar dari sistem saraf biologis. JST terdiri dari unit-unit dasar yang disebut neuron tiruan, yang bekerja serupa dengan neuron biologis. Setiap neuron tiruan menerima sejumlah sinyal input, mengalikannya dengan bobot tertentu, dan mengaplikasikan

fungsi aktivasi untuk menghasilkan sinyal output. Neuron-neuron ini dihubungkan satu sama lain melalui koneksi-koneksi dengan bobot yang mengatur kekuatan pengaruh satu neuron terhadap neuron lainnya. JST sering memiliki beberapa lapisan, termasuk lapisan input, lapisan tersembunyi, dan lapisan output, yang mencerminkan hierarki lapisan fungsi yang ada dalam otak manusia. Dengan kombinasi kompleksitas ini, JST dapat memproses informasi, mengidentifikasi pola, dan belajar dari data pelatihan, menciptakan sistem yang dapat digunakan untuk berbagai aplikasi di bidang kecerdasan buatan dan pemrosesan data.

Bagaimana syaraf tiruan 'belajar'?

Neuron saling terhubung melalui koneksi, dan setiap koneksi memiliki nilai bobot numerik yang terkait dengannya. Bobot ini memiliki peran kunci dalam menyimpan informasi jangka panjang dalam jaringan syaraf tiruan (ANN). Mereka mencerminkan seberapa pentingnya setiap input yang diterima oleh neuron tersebut. Proses pembelajaran dalam jaringan saraf terjadi melalui penyesuaian berulang-ulang dari bobot-bobot ini.



Gambar 9.3 Arsitektur Jaringan Syaraf Tiruan pada Umumnya
(Sumber : Negnevitsky, 2005)

Penyesuaian bobot dalam jaringan syaraf tiruan adalah inti dari kemampuan pembelajaran mereka. Ketika jaringan menerima data pelatihan dan membuat prediksi atau keputusan, bobot-bobot ini diperbarui untuk mengurangi kesalahan antara hasil yang diharapkan dan hasil yang sebenarnya. Ini adalah iterasi berulang yang memungkinkan jaringan untuk semakin mendekati pemahaman yang akurat tentang pola data yang diberikan.

Tabel 9.1 Analogi Antara Jaringan Saraf Biologis dan Jaringan Saraf Tiruan

Biological neural network	Artificial neural network
Soma	Neuron
Dendrite	Input
Axon	Output
Synapse	Weight

Sumber: Negnevitsky, 2005

Dengan kata lain, bobot-bobot ini berperan sebagai mekanisme penyesuaian yang memungkinkan jaringan syaraf tiruan untuk belajar dari pengalaman dan menjadi lebih baik dalam tugas-tugas tertentu seiring waktu. Itu sebabnya bobot-bobot ini sering disebut sebagai "memori jangka panjang" jaringan syaraf tiruan karena mereka membantu jaringan untuk menyimpan dan mengakumulasi pengetahuan yang diperlukan untuk tugas yang lebih kompleks dan akurat.

Namun, apakah jaringan saraf memahami bagaimana melakukan penyesuaian bobot?

Seperti yang terlihat pada Gambar 10.3, jaringan syaraf tiruan (JST) biasanya terstruktur dalam hierarki lapisan, dan neuron-neuron dalam jaringan ditempatkan sepanjang lapisan-lapisan

ini. Neuron-neuron yang berhubungan langsung dengan lingkungan eksternal membentuk lapisan input dan output. Bobot-bobot diatur ulang untuk menyesuaikan respons input/output jaringan dengan respons lingkungan.

Setiap neuron adalah unit pemrosesan informasi dasar yang memiliki kemampuan untuk menghitung tingkat aktivitasnya berdasarkan input yang diterima dan bobot numerik yang terkait. Untuk membangun sebuah jaringan syaraf tiruan, langkah awal adalah menentukan berapa banyak neuron yang akan digunakan dan bagaimana mereka akan dihubungkan untuk membentuk struktur jaringan. Dalam kata lain, pemilihan arsitektur jaringan adalah langkah pertama yang penting. Kemudian, kita harus memutuskan algoritma pembelajaran yang akan digunakan. Selanjutnya, kita melatih jaringan saraf, yang melibatkan inisialisasi bobot jaringan dan penyesuaian bobot-bobot ini berdasarkan serangkaian contoh pelatihan.

Kita akan memulai dengan memahami neuron sebagai elemen dasar yang membentuk dasar dari jaringan saraf tiruan.

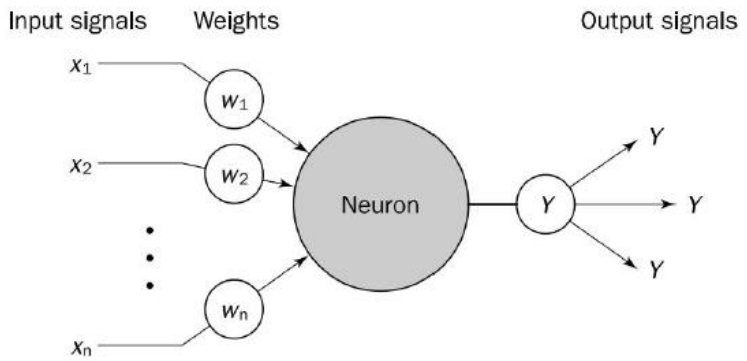
9.2 Metode Pelatihan dan Model Jaringan Syaraf Tiruan

9.2.1 Konsep Dasar Pemodelan Jaringan Syaraf Tiruan

Menurut Haykin (2009), neuron berfungsi sebagai entitas pemrosesan informasi yang menjadi fondasi utama dalam operasi Jaringan Saraf Tiruan (Riadi, 2016). Dalam konteks ini, Haykin juga menjelaskan bahwa terdapat tiga komponen dasar dalam model saraf, yaitu:

1. Seperangkat sinapsis, yang berperan sebagai penghubung antara neuron dan diklasifikasikan berdasarkan bobot atau intensitas koneksi masing-masing.

2. Sebuah summation unit untuk menggabungkan sinyal-sinyal masukan, yang dinilai dengan mempertimbangkan intensitas sinapsis dari setiap neuron.
3. Suatu fungsi aktivasi yang berfungsi untuk membatasi amplitudo keluaran dari neuron. Fungsi ini bertujuan untuk membatasi rentang amplitudo yang diizinkan oleh keluaran sinyal menjadi suatu nilai yang terbatas.



Gambar 9.4 Diagram Neuron
(Sumber : Negnevitsky, 2005)

Bagaimana neuron menentukan keluarannya?

Pada tahun 1943, Warren McCulloch dan Walter Pitts mengusulkan ide yang sangat sederhana yang masih menjadi dasar bagi sebagian besar jaringan saraf tiruan. Neuron menghitung jumlah tertimbang dari sinyal masukan dan membandingkannya dengan nilai ambang, θ. Jika masukan bersih kurang dari nilai ambang, keluaran neuron adalah -1. Namun, jika masukan bersih lebih besar dari atau sama dengan nilai ambang, neuron menjadi diaktifkan dan keluarannya mencapai nilai +1 (McCulloch dan Pitts, 1943). Dengan kata lain, neuron menggunakan fungsi transfer atau aktivasi berikut ini:

$$X = \sum_{i=1}^{n} x_i w_i$$

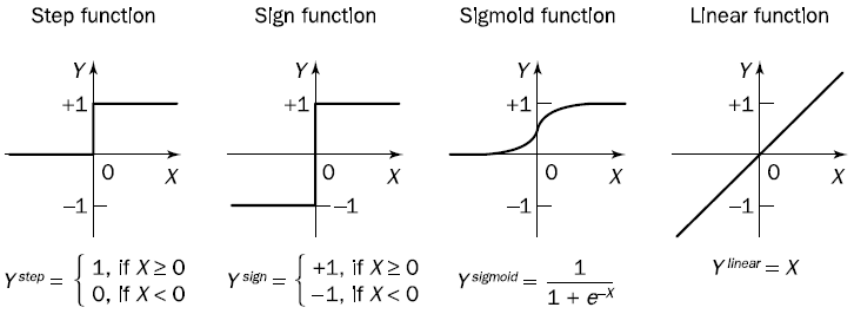
$$Y = \begin{cases} +1 & \text{if } X \geqslant \theta \\ -1 & \text{if } X < \theta \end{cases}$$

Di mana X adalah masukan bersih tertimbang ke neuron, xi adalah nilai masukan i, wi adalah bobot masukan i, n adalah jumlah masukan neuron, dan Y adalah keluaran dari neuron. Tipe fungsi aktivasi ini disebut sebagai fungsi tanda. Dengan demikian, keluaran aktual dari neuron dengan fungsi aktivasi tanda dapat direpresentasikan sebagai:

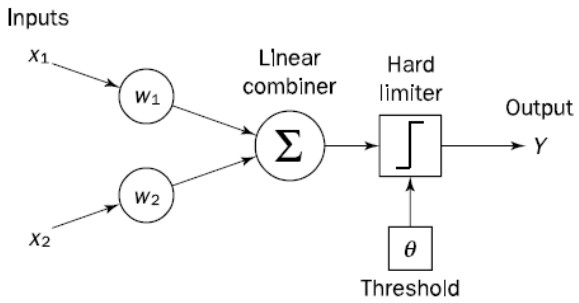
$$Y = sign\left[\sum_{i=1}^{n} x_i w_i - \theta\right]$$

Apakah fungsi tanda merupakan satu-satunya fungsi aktivasi yang digunakan oleh neuron?

Banyak fungsi aktivasi yang telah diuji, tetapi hanya beberapa yang telah ditemukan memiliki aplikasi praktis. Empat pilihan umumnya - fungsi langkah, fungsi tanda, fungsi linear, dan fungsi sigmoid - diilustrasikan dalam Gambar 10.5. Fungsi aktivasi langkah dan tanda, yang juga disebut sebagai fungsi batas keras, sering digunakan dalam neuron pengambilan keputusan untuk tugas klasifikasi dan pengenalan pola.



Gambar 9.5 Fungsi Aktivasi Neuron
(Sumber : Negnevitsky, 2005)



Gambar 9.6 Perceptron Dua Masukan Lapisan Tunggal
(Sumber : Negnevitsky, 2005)

Fungsi sigmoid mengubah masukan, yang dapat memiliki nilai antara plus dan minus tak terbatas, menjadi nilai yang wajar dalam rentang antara 0 dan 1. Neuron dengan fungsi ini digunakan dalam jaringan back-propagation. Fungsi aktivasi linear memberikan keluaran yang sama dengan masukan tertimbang dari neuron. Neuron dengan fungsi linear sering digunakan untuk pendekatan linear.

Bisakah satu neuron mempelajari suatu tugas?

Pada tahun 1958, Frank Rosenblatt memperkenalkan algoritma pelatihan yang memberikan prosedur pertama untuk melatih sebuah ANN sederhana: perceptron (Rosenblatt, 1958). Perceptron adalah bentuk paling sederhana dari jaringan saraf.

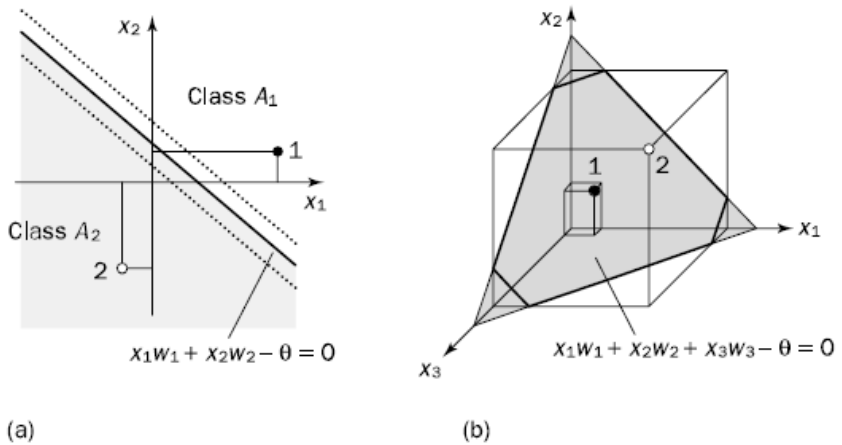
Ini terdiri dari satu neuron dengan bobot sinaptik yang dapat disesuaikan dan pengatur keras. Perceptron dengan dua masukan dalam satu lapisan ditampilkan dalam Gambar 10.6.

9.2.2 *Perceptron*

Operasi *perceptron* Rosenblatt didasarkan pada model neuron McCulloch dan Pitts. Model ini terdiri dari penggabung linear yang diikuti oleh pengatur keras. Jumlah tertimbang dari masukan diterapkan pada pengatur keras, yang menghasilkan keluaran yang sama dengan +1 jika masukannya positif dan -1 jika negatif. Tujuan dari perceptron adalah mengklasifikasikan masukan, atau dengan kata lain, rangsangan yang diterapkan secara eksternal x1; x2; . . . ; xn, ke salah satu dari dua kelas, misalnya A1 dan A2. Dengan demikian, dalam kasus perceptron yang sederhana, ruang berdimensi n dibagi oleh sebuah hiperplane menjadi dua wilayah keputusan. Hiperplane ini didefinisikan oleh fungsi yang dapat dipisahkan secara linear.

$$\sum_{i=1}^{n} x_i w_i - \theta = 0$$

Dalam kasus dua masukan, x1 dan x2, batas keputusan mengambil bentuk garis lurus yang ditunjukkan dengan tebal pada Gambar 10.7(a). Titik 1, yang berada di atas garis batas, termasuk ke dalam kelas A1; dan titik 2, yang berada di bawah garis tersebut, termasuk dalam kelas A2. Ambang θ dapat digunakan untuk menggeser batas keputusan.



Gambar 9. 7 Keterpisahan Linier pada Perceptron: (a) perceptron dua masukan; (b) perceptron tiga masukan
(Sumber : Negnevitsky, 2005)

Dengan tiga masukan, hiperplane masih dapat divisualisasikan. Gambar 10.7(b) menunjukkan tiga dimensi untuk perceptron dengan tiga masukan. Hiperplane pemisah di sini didefinisikan oleh persamaan berikut:

$$x_1w_1 + x_2w_2 + x_3w_3 - \theta = 0$$

Namun, bagaimana perceptron mempelajari tugas klasifikasinya?

Ini dilakukan dengan melakukan penyesuaian kecil pada bobot-bobot untuk mengurangi perbedaan antara keluaran aktual dan keluaran yang diinginkan dari perceptron. Bobot awal secara acak ditentukan, biasanya dalam rentang [-0.5 , 0.5], dan kemudian diperbarui untuk mendapatkan keluaran yang konsisten dengan contoh-contoh pelatihan. Proses pembaruan bobot pada perceptron sangat sederhana. Jika pada iterasi ke-p, keluaran aktual adalah Y(p) dan keluaran yang diinginkan

adalah Yd(p), maka kesalahan dihitung dengan menggunakan rumus berikut:

$$e(p) = Y_d(p) - Y(p) \qquad \text{where } p = 1, 2, 3, \ldots$$

Iterasi p di sini mengacu pada contoh pelatihan ke-p yang disajikan kepada perceptron. Jika kesalahan, e(p), adalah positif, kita perlu meningkatkan keluaran perceptron Y(p), tetapi jika negatif, kita perlu mengurangi Y(p). Dengan mempertimbangkan bahwa setiap masukan perceptron berkontribusi pada xi(p) × wi(p) untuk total masukan X(p), kita dapat menyimpulkan bahwa jika nilai masukan xi(p) adalah positif, peningkatan pada bobot wi(p) cenderung meningkatkan keluaran perceptron Y(p), sedangkan jika xi(p) adalah negatif, peningkatan wi(p) cenderung mengurangi Y(p). Dengan demikian, dapat dibentuk aturan pembelajaran perceptron berikut:

$$w_i(p+1) = w_i(p) + \alpha \times x_i(p) \times e(p),$$

di mana α adalah tingkat pembelajaran, sebuah konstanta positif yang lebih kecil dari satu. Aturan pembelajaran perceptron pertama kali diusulkan oleh Rosenblatt pada tahun 1960 (Rosenblatt, 1960). Dengan menggunakan aturan ini, kita dapat mendapatkan algoritma pelatihan perceptron untuk tugas klasifikasi.

Langkah 1: Inisialisasi

Tetapkan bobot awal w1; w2; . . . ; wn dan ambang θ ke angka acak dalam rentang [-0.5 , 0.5]

Langkah 2: Aktivasi

Aktifkan perceptron dengan menerapkan masukan x1(p); x2(p); . . . ; xn(p) dan keluaran yang diinginkan Yd(p). Hitung keluaran aktual pada iterasi p = 1.

$$Y(p) = step\left[\sum_{i=1}^{n} x_i(p)w_i(p) - \theta\right],$$

di mana n adalah jumlah masukan perceptron, dan step adalah fungsi aktivasi langkah.

Langkah 3: Latihan pembobotan

Perbarui bobot perceptron

$$w_i(p+1) = w_i(p) + \Delta w_i(p),$$

di mana Δwi(p) adalah koreksi bobot pada iterasi p. Koreksi bobot dihitung menggunakan aturan delta:

$$\Delta w_i(p) = \alpha \times x_i(p) \times e(p)$$

Langkah 4: Iterasi

Tingkatkan iterasi p dengan satu, kembali ke Langkah 2, dan ulangi proses ini hingga konvergensi.

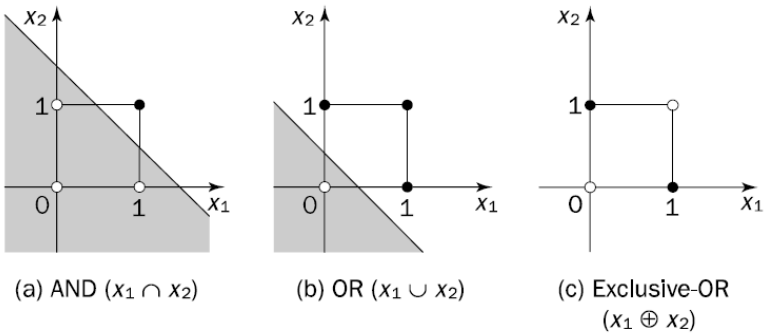
Tabel 9.2 Tabel Kebenaran untuk Operasi Logika Dasar

Input variables		AND	OR	Exclusive-OR
x_1	x_2	$x_1 \cap x_2$	$x_1 \cup x_2$	$x_1 \oplus x_2$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Sumber: Negnevitsky, 2005

Dengan cara yang serupa, perceptron dapat mempelajari operasi OR. Namun, perceptron satu lapisan tidak dapat dilatih untuk melakukan operasi Exclusive-OR. Sedikit geometri dapat membantu kita memahami mengapa hal ini terjadi. Gambar 10.8 menggambarkan fungsi AND, OR, dan Exclusive-OR sebagai plot dua dimensi berdasarkan nilai dua masukan. Titik-titik dalam

ruang masukan di mana keluaran fungsi adalah 1 ditunjukkan dengan titik hitam, dan titik-titik di mana keluaran adalah 0 ditunjukkan dengan titik putih.



Gambar 9.8 Plot Dua Dimensi dari Operasi Logika Dasar
(Sumber : Negnevitsky, 2005)

Tabel 9.3 Contoh Pembelajaran Perceptron: Operasi Logika AND

	Inputs		Desired output	Initial weights		Actual output	Error	Final weights	
Epoch	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	−0.1	0	0	0.3	−0.1
	0	1	0	0.3	−0.1	0	0	0.3	−0.1
	1	0	0	0.3	−0.1	1	−1	0.2	−0.1
	1	1	1	0.2	−0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	−1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	−1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	−1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

Threshold: $\theta = 0.2$; learning rate: $\alpha = 0.1$.

Sumber: Negnevitsky, 2005

Dalam Gambar 9.8(a) dan (b), kita dapat menggambar garis sehingga titik-titik hitam berada di satu sisi dan titik-titik putih di sisi lain, tetapi titik-titik yang ditunjukkan dalam Gambar 10.8(c) tidak dapat dipisahkan oleh satu garis. Sebuah perceptron dapat merepresentasikan sebuah fungsi hanya jika ada suatu garis yang memisahkan semua titik hitam dari semua titik putih. Fungsi-fungsi seperti itu disebut linearly separable. Oleh karena itu, perceptron dapat mempelajari operasi AND dan OR, tetapi tidak Exclusive-OR.

Mengapa perceptron hanya dapat mempelajari fungsi linearly separable?

Fakta bahwa perceptron hanya dapat mempelajari fungsi linearly separable langsung mengikuti dari persamaan:

$$X = \sum_{i=1}^{n} x_i w_i$$

$$Y = \begin{cases} +1 & \text{if } X \geqslant \theta \\ -1 & \text{if } X < \theta \end{cases}$$

Keluaran perceptron Y hanya sama dengan 1 jika total masukan tertimbang X lebih besar atau sama dengan nilai ambang . Ini berarti seluruh ruang masukan dibagi menjadi dua dengan batas yang didefinisikan oleh X = θ. Misalnya, garis pemisah untuk operasi AND didefinisikan oleh persamaan

$$x_1w_1 + x_2w_2 = \theta$$

Jika kita menggantikan nilai-nilai bobot w1 dan w2 serta ambang θ yang diberikan dalam Tabel 10.3, kita mendapatkan salah satu garis pemisah yang mungkin sebagai berikut:

$$0{,}1x_1 + 0{,}1x_2 = 0{,}2$$

atau

$$x_1 + x_2 = 2$$

Dengan demikian, wilayah di bawah garis batas, di mana keluarannya adalah 0, diberikan oleh

$$x_1 + x_2 \leq 2 < 0,$$

dan wilayah di atas garis ini, di mana keluarannya adalah 1, diberikan oleh

$$x_1 + x_2 - 2 \geq 0$$

Kenyataan bahwa perceptron hanya dapat mempelajari fungsi linearly separable adalah berita yang kurang baik, karena tidak banyak fungsi yang seperti itu.

Dapatkah kita melakukan lebih baik dengan menggunakan unsur sigmoidal atau linear sebagai pengganti pengatur keras?

Perseptron satu lapisan membuat keputusan dengan cara yang sama, terlepas dari fungsi aktivasi yang digunakan oleh perceptron (Shynk, 1990; Shynk dan Bershad, 1992). Ini berarti bahwa perceptron satu lapisan hanya dapat mengklasifikasikan pola-pola yang linearly separable, terlepas dari apakah kita menggunakan fungsi aktivasi hard-limit atau soft-limit. Keterbatasan komputasi perceptron dianalisis secara matematis dalam buku terkenal Minsky dan Papert, "Perceptrons" (Minsky dan Papert, 1969). Mereka membuktikan bahwa perceptron Rosenblatt tidak dapat membuat generalisasi global berdasarkan contoh yang dipelajari secara lokal. Selain itu, Minsky dan Papert menyimpulkan bahwa keterbatasan perceptron satu lapisan juga akan berlaku untuk jaringan saraf multilayer. Kesimpulan ini tentu saja tidak mendorong penelitian lebih lanjut tentang jaringan saraf tiruan.

Bagaimana kita mengatasi masalah yang tidak linearly separable?

Untuk mengatasi masalah tersebut, kita memerlukan jaringan saraf multilayer. Sebenarnya, sejarah telah membuktikan bahwa keterbatasan perceptron Rosenblatt dapat diatasi dengan

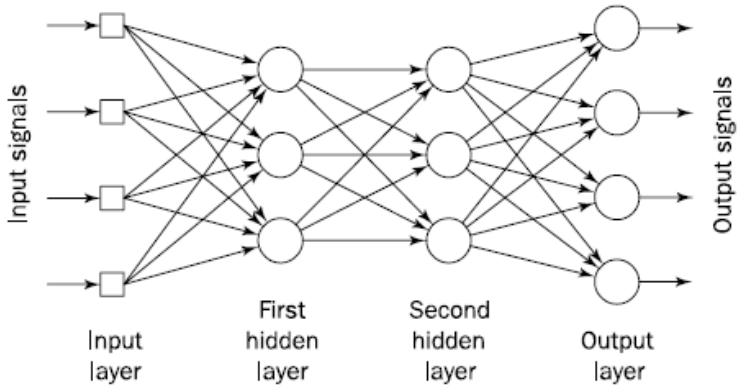
bentuk-bentuk lanjutan dari jaringan saraf, misalnya perceptron multilayer yang dilatih dengan algoritma back-propagation.

9.2.3 Jaringan Syaraf *Multilayer*

Sebuah multilayer perceptron adalah jaringan saraf *feedforward* dengan satu atau lebih lapisan tersembunyi. Biasanya, jaringan terdiri dari lapisan input dari neuron sumber, setidaknya satu lapisan tengah atau tersembunyi dari neuron komputasi, dan lapisan output dari neuron komputasi. Sinyal-sinyal input dipropagasi secara maju secara berurutan dari satu lapisan ke lapisan berikutnya. Sebuah multilayer perceptron dengan dua lapisan tersembunyi ditunjukkan dalam Gambar 10.9.

Tapi mengapa kita memerlukan lapisan tersembunyi?

Setiap lapisan dalam jaringan saraf multilayer memiliki fungsi khususnya. Lapisan input menerima sinyal input dari dunia luar dan mendistribusikan kembali sinyal-sinyal ini ke semua neuron dalam lapisan tersembunyi. Sebenarnya, lapisan input jarang memiliki neuron komputasi, dan oleh karena itu tidak memproses pola input. Lapisan output menerima sinyal keluaran, atau dengan kata lain, pola stimulus, dari lapisan tersembunyi dan menentukan pola keluaran dari seluruh jaringan. Neuron dalam lapisan tersembunyi mendeteksi fitur-fitur; bobot dari neuron mewakili fitur-fitur yang tersembunyi dalam pola input. Fitur-fitur ini kemudian digunakan oleh lapisan output dalam menentukan pola keluaran. Dengan satu lapisan tersembunyi, kita dapat merepresentasikan fungsi berkelanjutan apa pun dari sinyal-sinyal input, dan dengan dua lapisan tersembunyi bahkan fungsi-fungsi yang tidak berkelanjutan dapat direpresentasikan.



Gambar 9.9 Perceptron multilayer dengan dua lapisan tersembunyi
(Sumber : Negnevitsky, 2005)

Mengapa lapisan tengah dalam jaringan multilayer disebut sebagai lapisan 'tersembunyi'?

Apa yang lapisan ini sembunyikan? Lapisan tersembunyi 'menyembunyikan' keluaran yang diinginkan. Neuron dalam lapisan tersembunyi tidak dapat diamati melalui perilaku input/output dari jaringan. Tidak ada cara yang jelas untuk mengetahui apa yang seharusnya menjadi keluaran yang diinginkan dari lapisan tersembunyi. Dengan kata lain, keluaran yang diinginkan dari lapisan tersembunyi ditentukan oleh lapisan itu sendiri.

Apakah jaringan saraf dapat mencakup lebih dari dua lapisan tersembunyi?

Jaringan saraf buatan (ANN) komersial biasanya mencakup tiga atau empat lapisan, termasuk satu atau dua lapisan tersembunyi. Setiap lapisan dapat berisi 10 hingga 1000 neuron. Jaringan saraf eksperimental mungkin memiliki lima atau bahkan enam lapisan, termasuk tiga atau empat lapisan tersembunyi, dan menggunakan jutaan neuron, tetapi sebagian besar aplikasi praktis hanya menggunakan tiga lapisan, karena setiap lapisan tambahan meningkatkan beban komputasi secara eksponensial.

Bagaimana jaringan saraf multilayer belajar?

Lebih dari seratus algoritma pembelajaran yang berbeda tersedia, tetapi metode paling populer adalah back-propagation. Metode ini pertama kali diusulkan pada tahun 1969 (Bryson dan Ho, 1969), tetapi diabaikan karena komputasi yang rumit. Baru pada pertengahan tahun 1980-an algoritma pembelajaran back-propagation ditemukan kembali. Pembelajaran dalam jaringan multilayer berlangsung dengan cara yang sama seperti perceptron. Sebuah set pelatihan pola input disajikan ke jaringan. Jaringan menghitung pola keluarannya, dan jika terdapat kesalahan - atau dengan kata lain perbedaan antara pola keluaran yang sebenarnya dan yang diinginkan - bobot disesuaikan untuk mengurangi kesalahan ini.

Pada perceptron, hanya ada satu bobot untuk setiap input dan hanya satu keluaran. Tetapi dalam jaringan multilayer, terdapat banyak bobot, masing-masing berkontribusi pada lebih dari satu keluaran.

Bagaimana kita dapat menilai kesalahan dan membaginya di antara bobot-bobot yang berkontribusi?

Dalam jaringan saraf back-propagation, algoritma pembelajaran memiliki dua fase. Pertama, pola input pelatihan disajikan ke lapisan input jaringan. Jaringan kemudian menyebarkan pola input dari lapisan ke lapisan sampai pola keluaran dihasilkan oleh lapisan output. Jika pola ini berbeda dari keluaran yang diinginkan, kesalahan dihitung dan kemudian disebarkan kembali ke belakang melalui jaringan dari lapisan output ke lapisan input. Bobot disesuaikan saat kesalahan disebarkan. Seperti halnya dengan jaringan saraf lainnya, jaringan back-propagation ditentukan oleh koneksi antara neuron (arsitektur jaringan), fungsi aktivasi yang digunakan oleh neuron, dan algoritma pembelajaran yang menentukan prosedur penyesuaian bobot. Biasanya, jaringan back-propagation adalah jaringan multilayer yang memiliki tiga atau empat lapisan.

Lapisan-lapisan tersebut terhubung sepenuhnya, artinya setiap neuron dalam setiap lapisan terhubung ke setiap neuron lain dalam lapisan maju yang berdekatan.

Sebuah neuron menentukan keluarannya dengan cara yang mirip dengan perceptron Rosenblatt. Pertama, neuron menghitung input tertimbang bersih seperti sebelumnya:

$$X = \sum_{i=1}^{n} x_i w_i - \theta,$$

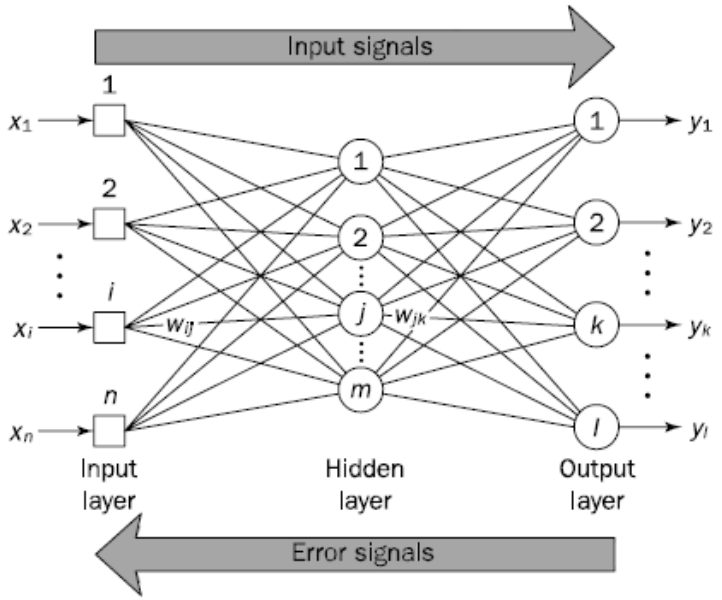
di mana n adalah jumlah input, dan θ adalah ambang yang diterapkan pada neuron. Selanjutnya, nilai input ini dilewatkan melalui fungsi aktivasi. Namun, berbeda dengan perceptron, neuron dalam jaringan back-propagation menggunakan fungsi aktivasi sigmoid:

$$Y^{sigmoid} = \frac{1}{1 + e^{-X}}$$

Turunan dari fungsi ini mudah dihitung. Ini juga menjamin bahwa keluaran neuron dibatasi antara 0 dan 1.

Untuk mendapatkan hukum pembelajaran back-propagation, mari pertimbangkan jaringan tiga lapisan yang ditunjukkan dalam Gambar 9.10. Indeks i, j, dan k di sini merujuk pada neuron dalam lapisan input, lapisan tersembunyi, dan lapisan keluaran, masing-masing.

Sinyal masukan, x1; x2; . . . ; xn, dipropagasi melalui jaringan dari kiri ke kanan, dan sinyal kesalahan, e1; e2; . . . ; el, dari kanan ke kiri. Simbol w_{ij} menunjukkan bobot untuk koneksi antara neuron i dalam lapisan input dan neuron j dalam lapisan tersembunyi, dan simbol w_{jk} bobot antara neuron j dalam lapisan tersembunyi dan neuron k dalam lapisan keluaran.



Gambar 9.10 Jaringan Saraf Propagasi Balik Tiga Lapis
(Sumber : Negnevitsky, 2005)

Untuk mengirimkan sinyal kesalahan, kita memulainya dari lapisan keluaran dan bekerja mundur ke lapisan tersembunyi. Sinyal kesalahan pada keluaran neuron k pada iterasi p didefinisikan sebagai:

$$e_k(p) = y_{d,k}(p) - y_k(p),$$

dimana $y_{d,k}(p)$ adalah keluaran yang diinginkan dari neuron k pada iterasi p. Neuron k, yang terletak di lapisan keluaran, memiliki keluaran yang diinginkan sendiri. Kita dapat menggunakan prosedur sederhana untuk memperbarui bobot wjk. Sebenarnya, aturan untuk memperbarui bobot di lapisan keluaran mirip dengan aturan pembelajaran perceptron pada Persamaan $w_i(p+1) = w_i(p) + \Delta w_i(p)$, :

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p),$$

di mana $\Delta w_{jk}(p)$ adalah koreksi bobot.

Ketika kita menentukan koreksi bobot untuk perceptron, kita menggunakan sinyal masukan xi. Tetapi dalam jaringan multilayer, masukan dari neuron di lapisan keluaran berbeda dari masukan neuron di lapisan masukan.

Karena kita tidak dapat menggunakan sinyal masukan xi, apa yang seharusnya kita gunakan sebagai penggantinya?

Kita menggunakan keluaran neuron j di lapisan tersembunyi, yj, sebagai pengganti input xi. Koreksi bobot dalam jaringan multilayer dihitung dengan (Fu, 1994):

$$\Delta w_{jk}(p) = \alpha \times y_j(p) \times \delta_k(p),$$

di mana $\delta_k(p)$ adalah gradien kesalahan di neuron k di lapisan keluaran pada iterasi p.

Apa itu gradien kesalahan?

Gradien kesalahan ditentukan sebagai turunan dari fungsi aktivasi dikali dengan kesalahan pada keluaran neuron.

Dengan demikian, untuk neuron k di lapisan keluaran, kita memiliki

$$\delta_k(p) = \frac{\partial y_k(p)}{\partial X_k(p)} \times e_k(p),$$

di mana $y_k(p)$ adalah keluaran dari neuron k pada iterasi p, dan $x_k(p)$ adalah masukan terbobot bersih ke neuron k pada iterasi yang sama.

Untuk fungsi aktivasi sigmoid, Persamaan sebelumnya dapat direpresentasikan sebagai

$$\delta_k(p) = \frac{\partial \left\{ \dfrac{1}{1 + \exp[-X_k(p)]} \right\}}{\partial X_k(p)} \times e_k(p) = \frac{\exp[-X_k(p)]}{\{1 + \exp[-X_k(p)]\}^2} \times e_k(p)$$

Dengan demikian, kita mendapatkan:

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p),$$

Dimana

$$y_k(p) = \frac{1}{1 + \exp[-X_k(p)]}.$$

Bagaimana kita dapat menentukan koreksi bobot untuk neuron dalam lapisan tersembunyi?
Untuk menghitung koreksi bobot untuk lapisan tersembunyi, kita dapat menggunakan persamaan yang sama dengan lapisan keluaran:

$$\Delta w_{ij}(p) = \alpha \times x_i(p) \times \delta_j(p),$$

di mana δ_k(p) adalah gradien kesalahan pada neuron j di lapisan tersembunyi

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^{l} \delta_k(p) w_{jk}(p),$$

Dimana l adalah jumlah neuron di lapisan keluaran

$$y_j(p) = \frac{1}{1 + e^{-X_j(p)}};$$

$$X_j(p) = \sum_{i=1}^{n} x_i(p) \times w_{ij}(p) - \theta_j;$$

Langkah 1: Inisialisasi
Atur semua bobot dan tingkat ambang jaringan ke angka acak yang terdistribusi secara merata dalam rentang kecil (Haykin, 1999):

$$\left(-\frac{2.4}{F_i}, +\frac{2.4}{F_i}\right),$$

di mana Fi adalah jumlah total input dari neuron i dalam jaringan. Inisialisasi bobot dilakukan berdasarkan neuron masing-masing.

Langkah 2: Aktivasi

Aktifkan jaringan saraf back-propagation dengan mengaplikasikan input x1(p), x2(p), . . . ; xn(p) dan keluaran yang diinginkan yd,1(p), yd,2(p), . . . , yd,n(p).

(a) Hitung keluaran aktual dari neuron di lapisan tersembunyi:

$$y_j(p) = sigmoid\left[\sum_{i=1}^{n} x_i(p) \times w_{ij}(p) - \theta_j\right],$$

dimana n adalah jumlah input dari neuron j di lapisan tersembunyi, dan sigmoid adalah fungsi aktivasi sigmoid.

(b) Hitung keluaran aktual dari neuron di lapisan output:

$$y_k(p) = sigmoid\left[\sum_{j=1}^{m} x_{jk}(p) \times w_{jk}(p) - \theta_k\right],$$

dimana m adalah jumlah input dari neuron k di lapisan output.

Langkah 3: Latihan pembobotan

Perbarui bobot dalam jaringan back-propagation dengan mengirimkan mundur kesalahan yang terkait dengan neuron output.

(a) Hitung gradien kesalahan untuk neuron di lapisan output:

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p)$$

Dimana

$$e_k(p) = y_{d,k}(p) - y_k(p)$$

Hitung koreksi bobot:

$$\Delta w_{jk}(p) = \alpha \times y_j(p) \times \delta_k(p)$$

Perbaharui bobot pada neuron output/keluaran:

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p)$$

(b) Hitung gradien kesalahan untuk neuron di lapisan tersembunyi:

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^{l} \delta_k(p) \times w_{jk}(p)$$

Hitung koreksi bobot:

$$\Delta w_{ij}(p) = \alpha \times x_i(p) \times \delta_j(p)$$

Perbarui bobot pada neuron tersembunyi:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

Langkah 4: Iterasi
Tingkatkan iterasi p sebanyak satu, kembali ke Langkah 2, dan ulangi proses ini hingga kriteria kesalahan yang dipilih terpenuhi.

9.3 Implementasi Jaringn Syaraf Tiruan

Implementasi jaringan saraf tiruan melibatkan beberapa langkah penting. Berikut adalah langkah-langkah umum yang terlibat dalam mengimplementasikan jaringan saraf tiruan:

1. Pemilihan Framework atau Library: Pilih framework atau library yang sesuai untuk bahasa pemrograman yang Anda gunakan. Beberapa pilihan populer termasuk TensorFlow, Keras, PyTorch, scikit-learn (Python), dan Deeplearning4j (Java).
2. Pengumpulan dan Persiapan Data: Kumpulkan data pelatihan yang sesuai untuk tugas yang ingin Anda selesaikan dengan jaringan saraf tiruan. Bersihkan dan persiapkan data ini agar siap digunakan dalam pelatihan.
3. Arahan Jaringan: Tentukan arsitektur jaringan saraf tiruan yang sesuai untuk tugas Anda. Ini melibatkan

pemilihan jumlah lapisan (layer), jenis neuron, jumlah neuron dalam setiap lapisan, dan fungsi aktivasi yang digunakan.

4. Pelatihan Jaringan: Gunakan data pelatihan yang telah Anda persiapkan untuk melatih jaringan saraf tiruan. Ini melibatkan proses iteratif di mana bobot dan bias jaringan diperbarui untuk mengurangi kesalahan (error) antara hasil prediksi dan hasil yang sebenarnya.
5. Validasi dan Evaluasi: Setelah melatih jaringan, gunakan data validasi untuk mengukur kinerja jaringan. Evaluasi performa jaringan Anda menggunakan metrik yang sesuai, seperti akurasi, presisi, recall, atau metrik lainnya yang relevan untuk tugas Anda.
6. Optimisasi: Jika hasil validasi tidak memenuhi harapan, Anda dapat mengoptimalkan jaringan dengan mencoba berbagai hyperparameter, seperti tingkat pembelajaran (learning rate), jumlah iterasi, dan jumlah neuron dalam lapisan tersembunyi.
7. Implementasi dalam Aplikasi: Setelah Anda puas dengan kinerja jaringan Anda, Anda dapat mengimplementasikannya dalam aplikasi atau sistem yang sesuai. Ini bisa menjadi bagian dari aplikasi berbasis web, perangkat lunak desktop, atau bahkan perangkat keras jika diperlukan.
8. Pemeliharaan: Perhatikan bahwa jaringan saraf tiruan juga memerlukan pemeliharaan. Anda perlu memantau kinerjanya, memperbarui model jika diperlukan dengan data yang lebih baru, dan menyesuaikan dengan perubahan tugas atau lingkungan.
9. Scaling: Jika diperlukan, Anda dapat mempertimbangkan untuk menskalakan jaringan saraf tiruan Anda dengan menggunakan infrastruktur komputasi yang lebih kuat, seperti GPU atau TPU, untuk mengatasi dataset yang lebih besar atau model yang lebih kompleks.

10. Dokumentasi dan Pelaporan: Pastikan untuk mendokumentasikan semua langkah-langkah dan hasil dari implementasi jaringan saraf tiruan Anda. Ini akan membantu dalam pemeliharaan, replikasi, dan pelaporan hasil kepada pemangku kepentingan.

Beberapa implementasi yang sudah dilakukan menggunakan jaringan syaraf tiruan seperti memprediksi nilai ujian nasional menggunakan metode backpropagation (Agus, Samuel Haryanto et al., 2015), mengenali pola tanda tangan menggunakan metode backpropagation (Vyan Martha et al., 2020), memprediksi prediksi cuaca (Agus, Samuel Haryanto et al., 2015), dan lain sebagainya.

DAFTAR PUSTAKA

Agus, Samuel Haryanto, F., Puspitaningrum, D., & Ernawati. 2015. Implementasi Jaringan Syaraf Tiruan Backpropagation untuk Memprediksi Cuaca (Studi Kasus : Kota Bengkulu). *Rekursif*, *3*(2), 82–94.

Negnevitsky, M. 2005. Artificial Intelligence: A Guide to Intelligent Systems. In *Pearson Education Limited* (Vol. 2).

Riadi, M. 2016. *Jaringan Saraf Tiruan (JST)*. Kajianpustaka. https://www.kajianpustaka.com/2016/11/jaringan-saraf-tiruan-jst.html

Vyan Martha, A., Hanafi, M., & Burhanuddin, A. 2020. Implementasi Jaringan Syaraf Tiruan (JST) untuk Mengenali Pola Tanda Tangan dengan Metode Backpropagation. *Jurnal Komtika (Komputasi Dan Informatika)*, *3*(2), 51–57. https://doi.org/10.31603/komtika.v3i2.3472

BAB 10
DATA MINING

Oleh Melissa Indah Fianty

10.1 Apa itu Data Mining?

”We are living in the information age” adalah pepatah yang populer; namun, sebenarnya kita hidup di zaman data. Terabytes atau petabytes data mengalir ke dalam jaringan komputer kita, dan berbagai jenis perangkat setiap hari dari bisnis, lembaga berita, masyarakat, ilmu pengetahuan, teknik, kedokteran, dan hampir setiap aspek kehidupan sehari-hari lainnya. Pertumbuhan eksplosif dalam volume data yang tersedia ini adalah hasil dari komputerisasi masyarakat kita dan perkembangan alat-alat komputasi yang begitu cepat, sensor, pengumpulan data, penyimpanan, dan publikasi data yang kuat."

Perusahaan di seluruh dunia menghasilkan kumpulan data raksasa, termasuk transaksi penjualan, catatan perdagangan saham, deskripsi produk, promosi penjualan, profil dan kinerja perusahaan, serta umpan balik pelanggan. Praktik ilmiah dan teknik menghasilkan jumlah petabyte data yang sangat besar secara berkelanjutan, mulai dari penginderaan jarak jauh, pengukuran proses, eksperimen ilmiah, kinerja sistem, observasi teknik, hingga pemantauan lingkungan. Penelitian biomedis dan industri kesehatan menghasilkan jumlah data yang luar biasa besar dari mesin urutan gen, eksperimen biomedis, dan laporan penelitian, catatan medis, pemantauan pasien, serta pencitraan medis. Miliaran pencarian web yang didukung oleh mesin pencari memproses puluhan petabyte data setiap harinya. Alat media sosial semakin populer, menghasilkan sejumlah besar

teks, gambar, dan video, serta menciptakan berbagai jenis komunitas web dan jaringan sosial. Daftar sumber yang menghasilkan jumlah data yang besar ini tak terbatas.

Data yang berkembang secara eksplosif, luas tersedia, dan berskala besar ini membuat zaman kita menjadi zaman data yang sejati. Alat-alat yang kuat dan serbaguna sangat dibutuhkan untuk secara otomatis mengungkapkan informasi berharga dari jumlah data yang luar biasa besar dan mengubah data tersebut menjadi pengetahuan yang terorganisir. Kebutuhan ini telah mengarah pada lahirnya data mining.

Secara esensial, data mining adalah proses menemukan pola menarik, model, dan jenis pengetahuan lainnya dalam kumpulan data besar. Istilah data mining, sebagai pandangan hidup yang mencari potongan emas dari data, muncul pada tahun 1990-an. Namun, untuk merujuk pada mining emas dari batu atau pasir, kita mengatakan mining emas daripada mining batu atau pasir. Secara analog, data mining seharusnya lebih tepatnya dinamai " “knowledge mining from data". Namun, istilah yang lebih pendek, knowledge mining, mungkin tidak mencerminkan penekanan pada mining dari jumlah data yang besar. Meskipun demikian, mining adalah istilah yang jelas menggambarkan proses yang menemukan seperangkat kecil potongan berharga dari bahan mentah yang sangat banyak. Oleh karena itu, istilah yang keliru ini yang mengandung baik "data" maupun "mining" menjadi pilihan yang populer. Selain itu, banyak istilah lain memiliki makna yang serupa dengan data mining, misalnya knowledge mining from data, KDD (i.e., Knowledge Discovery from Data), pattern discovery, knowledge extraction, data archaeology, data analytics, and information harvesting (Flores et al., 2023).

Data Mining adalah bidang yang masih muda, dinamis, dan penuh potensi. Ini telah membuat dan akan terus membuat

langkah besar dalam perjalanan kita dari zaman data menuju zaman informasi yang akan datang (Han et al., 2023).

Data mining mengubah kumpulan data besar menjadi pengetahuan. Mesin pencari (misalnya, Google) menerima miliaran permintaan setiap hari. Pengetahuan baru dan bermanfaat apa yang dapat diperoleh mesin pencari dari kumpulan permintaan yang begitu besar yang dikumpulkan dari pengguna dari waktu ke waktu? Menariknya, beberapa pola yang ditemukan dalam permintaan pencarian pengguna dapat mengungkap pengetahuan yang sangat berharga yang tidak dapat diperoleh dengan membaca item data individu secara sendirian. Sebagai contoh, Flu Trends Google menggunakan istilah pencarian tertentu sebagai indikator aktivitas flu. Ini menemukan hubungan yang erat antara jumlah orang yang mencari informasi terkait flu dan jumlah orang yang benar-benar memiliki gejala flu. Sebuah pola muncul ketika semua permintaan pencarian terkait flu diagregat. Dengan menggunakan data pencarian Google yang diagregat, Flu Trends dapat memperkirakan aktivitas flu hingga dua minggu lebih cepat daripada yang dapat dilakukan oleh sistem tradisional. Contoh ini menunjukkan bagaimana data mining dapat mengubah kumpulan data besar menjadi pengetahuan yang dapat membantu mengatasi tantangan global saat ini.

10.2 Data Mining: Langkah Penting Dalam Penemuan Pengetahuan

Data Banyak orang menganggap data mining sebagai sinonim untuk istilah lain yang populer digunakan, yaitu knowledge discovery from data, atau KDD, sementara yang lain menganggap data mining hanya sebagai langkah penting dalam proses keseluruhan knowledge discovery (Duque et al., 2023). Proses knowledge discovery keseluruhan ditunjukkan dalam Gambar 1.1 sebagai urutan iteratif dari langkah-langkah berikut:

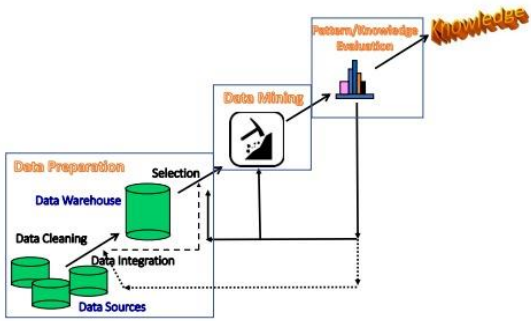
1. Data preparation

a. Data cleaning (untuk menghilangkan noise dan data yang tidak konsisten)

b. Data integration (di mana beberapa sumber data dapat digabungkan)

c. Data transformation (di mana data diubah dan digabungkan ke dalam bentuk yang sesuai)

d. Data selection (di mana data yang relevan untuk tugas analisis diambil dari basis data)



Gambar 10.1 Data Mining: Tahapan yang Tidak Dapat Dihindari Dalam Proses *Knowledge Discovery*
(Sumber: Han et al., 2023)

2. Data mining (proses penting di mana metode intelligent diterapkan untuk mengekstrak pola atau membangun model)
3. Pattern/model evaluation (untuk mengidentifikasi pola atau model yang benar-benar menarik yang mewakili pengetahuan berdasarkan ukuran kepentingan)
4. Knowledge presentation (di mana teknik visualisasi dan representasi pengetahuan digunakan untuk menyajikan pengetahuan yang ditambang kepada pengguna)

Langkah-langkah 1(a) hingga 1(d) adalah bentuk berbeda dari pra-pemrosesan data, di mana data harus dipersiapkan terlebih

dahulu. Langkah data mining dapat berinteraksi dengan pengguna atau knowledge base. Pola-pola menarik disajikan kepada pengguna dan dapat disimpan sebagai new knowledge in the knowledge base (Duque et al., 2023).

Pandangan sebelumnya menunjukkan data mining sebagai satu langkah dalam proses knowledge discovery, meskipun merupakan langkah penting karena mengungkapkan pola atau model yang tersembunyi untuk evaluasi. Namun, dalam industri, media, dan lingkungan penelitian, istilah data mining sering digunakan untuk merujuk kepada seluruh proses knowledge discovery (mungkin karena istilah tersebut lebih singkat daripada knowledge discovery from data). Oleh karena itu, mengadopsi pandangan luas tentang fungsionalitas data mining: Data mining adalah proses menemukan pola menarik dan pengetahuan dari jumlah data yang besar. Sumber data dapat mencakup basis data, gudang data, Web, repositori informasi lain, atau data yang disalurkan ke dalam sistem secara dinamis.

10.3 Keanekaragaman Jenis Data untuk Data Mining

Sebagai teknologi umum, data mining dapat diterapkan pada jenis data apa pun selama data tersebut memiliki makna untuk aplikasi yang dituju. Namun, berbagai jenis data mungkin memerlukan metodologi data mining yang berbeda, mulai dari yang sederhana hingga yang cukup canggih, menjadikan data mining sebagai bidang yang kaya dan beragam.

10.3.1 Data Terstruktur vs. Data Tak Terstruktur

Apakah data memiliki struktur yang jelas, kita dapat mengkategorikan data sebagai data terstruktur vs. data tak terstruktur.

Data yang tersimpan dalam relational databases, data cubes, data matrices, dan many data warehouses memiliki struktur yang seragam, mirip dengan record- or table-like structures,

yang didefinisikan oleh kamus data, dengan satu set atribut (atau kolom) yang tetap, masing-masing dengan satu set rentang nilai dan makna semantik yang tetap. Data-data ini adalah contoh tipikal dari data yang sangat terstruktur. Dalam banyak aplikasi nyata, persyaratan struktural yang ketat ini dapat diubah dengan berbagai cara untuk menyesuaikan sifat semi-terstruktur dari data tersebut, misalnya, untuk memungkinkan objek data mengandung satu nilai, satu set kecil nilai tipe yang beragam, atau struktur bertingkat, atau untuk memungkinkan struktur objek atau subobjek didefinisikan dengan fleksibel dan dinamis (misalnya, struktur XML) (Krishnamoorthy & Karthikeyan, 2022a).

Ada banyak kumpulan data yang mungkin tidak sestruktur tabel relasional atau matriks data. Namun, mereka memiliki struktur tertentu dengan makna semantik yang jelas. Sebagai contoh, sebuah kumpulan data transaksional mungkin berisi satu set besar transaksi, masing-masing berisi satu set item. Sebuah kumpulan data urutan mungkin berisi satu set besar urutan, masing-masing berisi satu set elemen yang diurutkan yang pada gilirannya dapat berisi satu set item. Banyak kumpulan data aplikasi, seperti shopping transaction data, time-series data, gene or protein data, or Weblog data, belong to this category.

Jenis kumpulan data semi-terstruktur yang lebih canggih adalah data grafik atau jaringan, di mana satu set node terhubung oleh satu set tepi (juga disebut sebagai tautan); dan setiap node/tautan dapat memiliki deskripsi semantiknya sendiri atau struktur bawahan.

Setiap kategori kumpulan data terstruktur dan semi-terstruktur seperti itu dapat memiliki jenis pola atau pengetahuan khusus yang akan diolah dan banyak metode data mining yang didedikasikan, seperti such as sequential pattern mining, graph pattern mining, and information network mining methods, telah

dikembangkan untuk menganalisis kumpulan data semacam itu (Krishnamoorthy & Karthikeyan, 2022b).

Di luar data terstruktur atau semi-terstruktur seperti itu, ada juga banyak data tak terstruktur, seperti data teks dan data multimedia (misalnya, audio, gambar, video). Meskipun beberapa penelitian menganggapnya sebagai one-dimensional or multidimensional byte streams, data tersebut membawa banyak semantik menarik. Metode yang spesifik untuk domain telah dikembangkan untuk menganalisis data semacam itu di bidang natural language understanding, text mining, computer vision, and pattern recognition. Selain itu, kemajuan terbaru dalam pembelajaran mendalam telah membuat kemajuan besar dalam pengolahan data teks, gambar, dan video. Namun, mining struktur tersembunyi dari data tak terstruktur dapat sangat membantu memahami dan memanfaatkan data tersebut.

Data dunia nyata seringkali merupakan campuran dari data terstruktur, data semi-terstruktur, dan data tak terstruktur. Sebagai contoh, sebuah situs web belanja online mungkin memiliki informasi untuk satu set besar produk, yang pada dasarnya adalah data terstruktur yang disimpan dalam basis data relasional, dengan satu set lapangan yang tetap pada nama produk, harga, spesifikasi, dan sebagainya. Namun, beberapa lapangan mungkin pada dasarnya adalah data teks, gambar, dan video, seperti pengenalan produk, ulasan dari ahli atau pengguna, gambar produk, dan video iklan. Metode data mining seringkali dikembangkan untuk menemukan jenis data tertentu, dan hasilnya dapat diintegrasikan dan disinkronkan untuk melayani tujuan keseluruhan.

10.3.2 Data yang terkait dengan aplikasi yang berbeda

Aplikasi yang berbeda mungkin menghasilkan atau perlu mengatasi kumpulan data yang sangat berbeda dan memerlukan metode analisis data yang cukup berbeda. Oleh karena itu, saat

mengkategorikan kumpulan data untuk data mining, kita harus mempertimbangkan aplikasi tertentu.

Ambil data urutan sebagai contoh. Urutan biologis seperti urutan DNA atau protein mungkin memiliki makna semantik yang sangat berbeda dari urutan transaksi belanja atau aliran klik web, yang memerlukan metode menemuan urutan yang cukup berbeda. Jenis khusus dari data urutan adalah data seri waktu di mana seri waktu dapat berisi satu set nilai numerik yang diurutkan dengan selang waktu yang sama, yang juga berbeda dari urutan transaksi belanja, yang mungkin tidak memiliki selang waktu yang tetap (pelanggan dapat berbelanja kapan saja sesuai keinginannya).

Data dalam beberapa aplikasi dapat berhubungan dengan informasi spasial, informasi waktu, atau keduanya, membentuk data spasial, data temporal, dan data spasial-temporal, secara berturut-turut. Metode menemuan data khusus, seperti spatial data mining, temporal data mining, spatiotemporal data mining, or trajectory pattern mining, harus dikembangkan untuk menemukan kumpulan data semacam itu.

Untuk data grafik dan jaringan, aplikasi yang berbeda juga mungkin memerlukan metode data mining yang cukup berbeda. Sebagai contoh, jaringan sosial (misalnya, data Facebook atau LinkedIn), jaringan komunikasi komputer, jaringan biologis, dan jaringan informasi (misalnya, penulis yang terhubung dengan kata kunci) mungkin membawa semantik yang cukup berbeda dan memerlukan metode mining yang berbeda pula.

Bahkan untuk kumpulan data yang sama, menemukan berbagai jenis pola atau pengetahuan mungkin memerlukan metode data mining yang berbeda. Sebagai contoh, untuk satu set program perangkat lunak (sumber), menemukan modul subprogram yang diplagiat atau menemukan bug copy-and-paste mungkin memerlukan teknik data mining yang cukup berbeda.

Jenis data yang beragam dan persyaratan aplikasi yang beragam memerlukan metode data mining yang sangat berbeda. Oleh karena itu, data mining adalah domain penelitian yang kaya dan menarik, dengan banyak metode baru yang menunggu untuk dipelajari dan dikembangkan.

10.3.3 Data yang disimpan vs. data aliran

Biasanya, data mining mengelola kumpulan data yang terbatas dan disimpan, seperti yang tersimpan dalam berbagai jenis repositori data besar. Namun, dalam beberapa aplikasi seperti video surveillance atau remote sensing, data dapat mengalir secara dinamis dan terus-menerus, sebagai aliran data tak terbatas. Mining stream data akan memerlukan metode yang cukup berbeda dibandingkan dengan data yang disimpan, yang dapat membentuk tema menarik lainnya dalam penelitian kita.

10.4 Mining Various Kinds of Knowledge

Berbagai jenis pola dan pengetahuan dapat ditemukan melalui data mining. Secara umum, tugas data mining dapat dikelompokkan menjadi dua kategori: descriptive data mining dan predictive data mining. Deskriptif mining menggambarkan properti dari kumpulan data yang diminati, sementara predictive mining melakukan induksi pada kumpulan data untuk membuat prediksi.

10.4.1 Multidimensional data summarization

Seringkali melelahkan bagi pengguna untuk memeriksa detail dari sekumpulan data yang besar. Oleh karena itu, diinginkan untuk secara otomatis merangkum kumpulan data yang diminati dan membandingkannya dengan kumpulan data kontras pada beberapa tingkat yang tinggi. Deskripsi ringkas seperti itu dari satu set data yang diminati disebut data summarization. Data summarization seringkali dapat dilakukan dalam multidimensional space. Jika multidimensional space telah didefinisikan dengan baik dan sering digunakan, seperti kategori

produk, produsen, lokasi, atau waktu, jumlah data besar dapat diagregasi dalam bentuk data cubes untuk memudahkan pengguna melakukan drill-down atau roll-up dari ruang ringkasan dengan mengklik mouse. Hasil dari ringkasan multidimensi seperti itu dapat disajikan dalam berbagai pie charts, bar charts, curves, multidimensional data cubes, dan multidimensional tables.

10.4.2 Mining frequent patterns, associations, and correlations

Frequent patterns, seperti namanya, adalah pola yang sering muncul dalam data. Ada banyak jenis pola sering terjadi, termasuk set item sering terjadi, subsekuen sering terjadi (juga dikenal sebagai pola sekuensial), dan substruktur sering terjadi. Sebuah frequent itemset sering terjadi biasanya merujuk pada satu set item yang sering muncul bersama dalam kumpulan data transaksional, misalnya susu dan roti, yang sering dibeli bersama di toko kelontong oleh banyak pelanggan. Subsekuen yang sering muncul, seperti pola bahwa pelanggan cenderung membeli laptop terlebih dahulu, diikuti oleh tas laptop, dan kemudian aksesori lainnya, adalah pola sekuensial yang sering muncul. Substruktur dapat merujuk kepada bentuk struktural yang berbeda (misalnya, graphs, trees, atau lattices) yang dapat digabungkan dengan (frequent) structured pattern. Jika suatu substruktur muncul sering, itu disebut sebagai pola terstruktur yang sering terjadi. Menemukan pola sering terjadi mengarah pada penemuan asosiasi dan korelasi menarik dalam data.

Contoh Analisis Asosiasi. Bayangkan bahwa seorang manajer toko online ingin tahu item-item apa yang sering dibeli bersama (yaitu, dalam transaksi yang sama). Sebuah contoh aturan seperti itu, yang ditemukan dari basis data transaksional, adalah:

buys (X, “computer”) ⇒ buys(X, “webcam”) [support = 1%, confidence = 50%],

di mana X adalah variabel yang mewakili pelanggan. Confidence sebesar 50% berarti jika seorang pelanggan membeli komputer, ada 50% kemungkinan bahwa dia juga akan membeli webcam. Support sebesar 1% berarti bahwa 1% dari semua transaksi yang sedang dianalisis menunjukkan bahwa komputer dan webcam dibeli bersama. Aturan asosiasi ini melibatkan satu atribut atau predikat tunggal (yaitu, buys) yang berulang. Aturan asosiasi yang mengandung satu predikat disebut sebagai aturan single-dimensional association rules. Dengan menghilangkan notasi predikat, aturan tersebut dapat ditulis dengan sederhana sebagai "computer ⇒ webcam [1%, 50%]."

Bayangkan, mining the same database menghasilkan aturan asosiasi lainnya:

age (X, "20..29") ∧ income(X, "40K..49K") ⇒ buys(X, "laptop")

[support = 0.5%, confidence = 60%].

Aturan ini menunjukkan bahwa dari semua pelanggannya yang diteliti, 0,5% berusia 20 hingga 29 tahun dengan pendapatan antara $40.000 hingga $49.000 dan telah membeli laptop (komputer). Ada probabilitas 60% bahwa seorang pelanggan dalam kelompok usia dan pendapatan ini akan membeli laptop. Perhatikan bahwa ini adalah asosiasi yang melibatkan lebih dari satu atribut atau predikat (yaitu, usia, pendapatan, dan membeli). Dengan mengadopsi istilah yang digunakan dalam basis data multidimensi, di mana setiap atribut disebut sebagai dimensi, aturan di atas dapat disebut sebagai aturan multidimensional association rule.

Biasanya, aturan asosiasi dianggap tidak menarik jika mereka tidak memenuhi kedua ambang batas dukungan minimum dan keyakinan minimum. Analisis tambahan dapat dilakukan untuk menemukan korelasi statistik yang menarik antara pasangan atribut-nilai yang terkait.

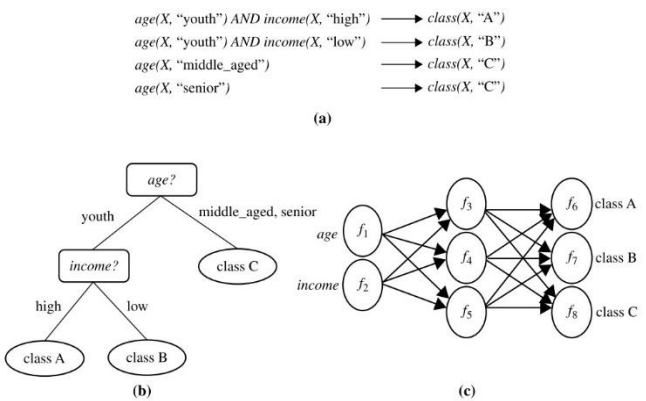
10.4.3 Classification and regression for predictive analysis

Terkait classification adalah proses untuk menemukan model (atau fungsi) yang menggambarkan dan membedakan kelas atau konsep data. Model ini didasarkan pada analisis sekumpulan data pelatihan (yaitu, objek data yang memiliki label kelas yang diketahui). Model ini digunakan untuk memprediksi label kelas objek yang label kelasnya tidak diketahui.

Bergantung pada metode classification, model yang dihasilkan dapat berbentuk beragam, seperti kumpulan aturan klasifikasi (yaitu, aturan IF-THEN), a decision tree, a mathematical formula, or a learned neural network yang bisa dilihat pada gambar 11.2. Decision tree adalah struktur pohon mirip bagan alur, di mana setiap node menunjukkan pengujian pada nilai atribut, setiap cabang mewakili hasil dari pengujian tersebut, dan daun-daun pohon mewakili kelas atau distribusi kelas. Decision tree dengan mudah dapat diubah menjadi aturan classification. Sebuah neural network, ketika digunakan untuk classification, biasanya terdiri dari unit pemrosesan mirip neuron dengan koneksi terbobot antara unit-unit tersebut. Ada banyak metode lain untuk membangun model classification, seperti naïve Bayesian classification, support vector machines, dan k-nearest-neighbor classification.

Sementara classification memprediksi label kategoris (discrete, tidak terurut), regression menghasilkan fungsi berkebalikan yang bernilai kontinu. Artinya, regresi digunakan untuk memprediksi nilai data numerik yang hilang atau tidak tersedia daripada label kelas (discrete). Istilah prediksi merujuk pada prediksi numerik dan prediksi label kelas. Regression analysis adalah metodologi statistik yang paling sering digunakan untuk prediksi numerik, meskipun metode lain juga ada. Regresi juga mencakup identifikasi tren distribusi berdasarkan data yang tersedia.

Klasifikasi dan regresi mungkin perlu diawali dengan feature selection or relevance analysis yang mencoba mengidentifikasi atribut (sering disebut features) yang signifikan relevan dengan proses klasifikasi dan regresi. Atribut-atribut ini akan dipilih untuk proses klasifikasi dan regresi. Atribut lain yang tidak relevan kemudian dapat dikecualikan dari pertimbangan.



Gambar 10.2 Sebuah Classification Model Dapat Diwakili dalam Berbagai Bentuk: (a) aturan IF-THEN, (b) decision treen, atau (c) neural network
(Sumber: Han et al., 2023)

Contoh Classification and regression. Misalkan seorang manajer penjualan toko online ingin mengklasifikasikan sekumpulan besar barang di toko berdasarkan tiga kinds of responses terhadap sales campaign: good response, mild response, dan no response. Anda ingin menghasilkan model untuk masing-masing dari tiga kelas ini berdasarkan fitur-fitur deskriptif barang-barang tersebut, seperti harga, merek, tempat pembuatan, jenis, dan kategori. Klasifikasi yang dihasilkan harus secara maksimal membedakan setiap kelas dari yang lain, menyajikan gambaran terorganisir tentang kumpulan data tersebut.

Misalkan hasil klasifikasi tersebut diungkapkan sebagai decision tree. Decision tree, misalnya, dapat mengidentifikasi harga sebagai faktor penting pertama yang paling baik membedakan tiga kelas tersebut. Fitur lain yang membantu lebih membedakan objek dari masing-masing kelas termasuk brand dan place_made. Decision tree seperti ini dapat membantu manajer memahami dampak sales campaign yang diberikan dan merancang campaign yang lebih efektif di masa depan.

Misalkan, sebagai gantinya, daripada memprediksi response labels untuk setiap item toko, Anda ingin memprediksi jumlah pendapatan yang akan dihasilkan oleh setiap item selama penjualan mendatang, berdasarkan data penjualan sebelumnya. Ini adalah contoh dari analisis regresi karena model regresi yang dibangun akan memprediksi fungsi kontinu (or ordered value).t

10.4.4 *Cluster analysis*

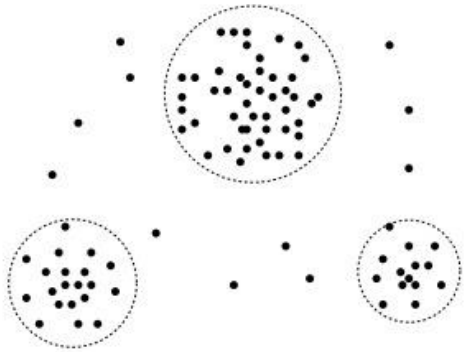
Berbeda dengan klasifikasi dan regresi, yang menganalisis data pelatihan berlabel kelas, *cluster analysis* (juga disebut clustering) mengelompokkan objek data tanpa memperhatikan label kelas. Dalam banyak kasus, data berlabel kelas mungkin belum ada pada awalnya. Pengelompokan dapat digunakan untuk menghasilkan label kelas untuk sekelompok data. Objek-objek dikelompokkan berdasarkan prinsip memaksimalkan kemiripan dalam kelas dan meminimalkan kemiripan antarkelas. Artinya, objek-objek dalam kluster dibentuk sehingga objek dalam kluster memiliki kemiripan tinggi satu sama lain, tetapi cukup berbeda dari objek dalam kluster lain. Setiap kluster yang terbentuk dapat dilihat sebagai sebuah kelas objek, dari mana aturan dapat ditarik. Pengelompokan juga dapat memudahkan *taxonomy formation*, yaitu pengorganisasian pengamatan ke dalam hierarki kelas yang mengelompokkan peristiwa serupa Bersama (Dol & Jawandhiya, 2023).

Contoh 1.4. *cluster analysis*. *Cluster analysis* dapat dilakukan pada data pelanggan toko online untuk mengidentifikasi

subpopulasi pelanggan yang homogen. Kluster-kluster ini mungkin mewakili kelompok target individual untuk pemasaran. Gambar 11.3 menunjukkan plot 2-D pelanggan berdasarkan lokasi pelanggan dalam sebuah kota. Tiga kluster titik data terlihat jelas.

10.4.5 *Deep learning*

Untuk banyak task data mining, seperti klasifikasi dan pengelompokan, langkah kunci seringkali terletak pada menemukan "good features" yaitu representasi vektor dari setiap tuple data masukan. Sebagai contoh, untuk memprediksi apakah wabah penyakit regional akan terjadi, seseorang mungkin telah mengumpulkan sejumlah besar fitur dari data pemantauan kesehatan, termasuk jumlah kasus positif harian, jumlah tes harian, jumlah rawat inap harian, dan sebagainya. Secara tradisional, langkah ini (yang disebut feature engineering) sering sangat bergantung pada pengetahuan dalam bidang tertentu. Teknik *Deep learning* menyediakan cara otomatis untuk feature engineering, yang mampu menghasilkan fitur-fitur yang memiliki makna semantik (misalnya, tingkat positif mingguan) dari fitur-fitur masukan awal. Fitur-fitur yang dihasilkan seringkali secara signifikan meningkatkan kinerja mining (misalnya, classification accuracy).



Gambar 10.3 Sebuah Plot 2-D Data Pelanggan Berdasarkan Lokasi Pelanggan Di Sebuah Kota, Menampilkan Tiga Kluster Data
(Sumber: Han et al., 2023)

Deep learning didasarkan *neural networks*. *Neural networks* adalah kumpulan unit masukan-keluaran yang terhubung di mana setiap koneksi memiliki bobot yang terkait dengannya. Selama e learning phase, network learns dengan mengatur bobot-bobot agar mampu memprediksi nilai target yang benar (misalnya, class labels) dari tuple masukan. Algoritma inti untuk mempelajari bobot-bobot tersebut disebut backpropagation, yang mencari himpunan bobot dan nilai bias yang dapat memodelkan data untuk meminimalkan fungsi kerugian antara prediksi jaringan dan output target sebenarnya dari tuple data. Berbagai bentuk (disebut architectures) dari *neural networks* telah dikembangkan, termasuk feed-forward *neural networks*, convolutional *neural networks*, recurrent *neural networks*, graph *neural networks*, dan banyak lagi. *Deep learning* memiliki berbagai aplikasi dalam pengolahan computer vision, natural language processing, machine translation, social network analysis dan sebagainya. Ini telah digunakan dalam berbagai tugas data mining, termasuk klasifikasi, classification, clustering, outlier detection, and reinforcement learning (Aslani et al., 2023).

10.4.6 *Outlier analysis*

Sebuah himpunan data dapat berisi objek yang tidak sesuai dengan perilaku umum atau model data tersebut. Objek data ini disebut sebagai outliers. Banyak metode data mining mengabaikan outlier sebagai noise atau pengecualian. Namun, dalam beberapa aplikasi (misalnya, fraud detection), peristiwa langka bisa lebih menarik daripada yang terjadi secara reguler. Analisis data outlier disebut sebagai outlier analysis or anomaly mining.

Outlier dapat dideteksi menggunakan uji statistik yang mengasumsikan distribusi atau model probabilitas untuk data, atau menggunakan ukuran jarak di mana objek yang jauh dari setiap kluster lain dianggap sebagai outlier. Dalam menggunakan metode statistical or distance measures, metode berbasis density-based dapat mengidentifikasi outlier dalam wilayah lokal, meskipun mereka terlihat normal dari pandangan distribusi statistik global.

Contoh Outlier analysis. Outlier analysis dapat mengungkap penggunaan kartu kredit yang curang dengan mendeteksi pembelian dengan jumlah yang tidak biasa besar untuk nomor rekening tertentu dibandingkan dengan biaya reguler yang dikeluarkan oleh akun yang sama. Nilai outlier juga dapat dideteksi dengan mengenai lokasi dan jenis pembelian, atau frekuensi pembelian.

10.5 Data Mining: Confluence of Multiple Disciplines

Sebagai disiplin yang mempelajari metode yang efisien dan efektif untuk mengungkap pola dan pengetahuan dari berbagai jenis set data massif untuk banyak aplikasi, data mining secara alami melayani sebagai perpaduan dari beberapa disiplin, termasuk pembelajaran mesin, statistik, pengenalan pola,

pemrosesan bahasa alami, teknologi basis data, visualisasi dan interaksi manusia-komputer (HCI), algoritma, komputasi berkinerja tinggi, ilmu sosial, dan banyak domain aplikasi (Gambar 1.4).

Gambar 10.4 Data Mining: Confluence of Multiple Disciplines
(Sumber: Han et al., 2023)

Sifat interdisipliner dari penelitian dan pengembangan data mining memberikan kontribusi yang signifikan terhadap kesuksesan data mining dan aplikasi yang luas. Di sisi lain, data mining tidak hanya diperoleh dari pengetahuan dan perkembangan disiplin-disiplin ini, penelitian, pengembangan, dan aplikasi data mining yang didedikasikan pada berbagai jenis big data mungkin telah berdampak besar pada perkembangan disiplin-disiplin ini dalam beberapa tahun terakhir. Dalam bagian ini, kami akan membahas beberapa disiplin yang berpengaruh kuat dan berinteraksi aktif dengan penelitian, pengembangan, dan aplikasi data mining.

10.5.1 Statistics and data mining

Statistics mempelajari pengumpulan, analisis, interpretasi atau penjelasan, dan penyajian data. Data mining memiliki koneksi yang inheren dengan statistika.

A statistical model adalah seperangkat fungsi matematika yang menggambarkan perilaku objek dalam kelas target dalam hal variabel acak dan distribusi probabilitas terkait mereka. Model statistik banyak digunakan untuk memodelkan data dan kelas

data. Misalnya, dalam tugas data mining seperti karakterisasi data dan klasifikasi, model statistik dari kelas target dapat dibangun. Dengan kata lain, model statistik semacam ini dapat menjadi hasil dari tugas data mining. Sebaliknya, tugas data mining dapat dibangun di atas model statistik. Misalnya, kita dapat menggunakan statistik untuk memodelkan noise dan nilai data yang hilang. Kemudian, saat mining patterns dalam set data besar, proses data mining dapat menggunakan model tersebut untuk membantu mengidentifikasi dan menangani nilai yang bising atau hilang dalam data.

Penelitian statistik mengembangkan alat-alat untuk prediction dan forecasting menggunakan data dan model statistik. Metode statistik dapat digunakan untuk merangkum atau menggambarkan kumpulan data. Statistik berguna untuk mining various patterns berbagai pola dari data dan untuk memahami mekanisme yang mendasari pembentukan dan pengaruh pola. Inferential statistics (atau predictive statistics) memodelkan data dengan cara yang memperhitungkan keacakan dan ketidakpastian dalam pengamatan dan digunakan untuk membuat inferensi tentang proses atau populasi yang sedang diselidiki.

Metode statistik juga dapat digunakan untuk memverifikasi hasil data mining. Misalnya, setelah model klasifikasi atau prediksi ditambang, model tersebut harus diverifikasi dengan pengujian hipotesis statistik. A statistical hypothesis test (kadang disebut confirmatory data analysis) membuat keputusan statistik menggunakan data eksperimental. Hasilnya disebut signifikan secara statistik jika tidak mungkin terjadi secara kebetulan. Jika model klasifikasi atau prediksi berlaku, maka statistik deskriptif dari model tersebut meningkatkan kekuatan model.

Menerapkan metode statistik dalam data mining jauh dari yang mudah. Seringkali, tantangan serius adalah bagaimana mengukur metode statistik di atas kumpulan data besar. Banyak

metode statistik memiliki kompleksitas tinggi dalam komputasi. Ketika metode tersebut diterapkan pada set data besar yang juga didistribusikan di beberapa situs logis atau fisik, algoritma harus dirancang dan disesuaikan dengan hati-hati untuk mengurangi biaya komputasi. Tantangan ini menjadi lebih sulit untuk aplikasi online, seperti saran kueri online dalam mesin pencari, di mana data mining diperlukan untuk terus mengelola aliran data cepat dan real-time.

Penelitian data mining telah mengembangkan banyak solusi yang dapat diukur dan efektif untuk analisis set data besar dan aliran data. Selain itu, jenis-jenis set data yang berbeda dan aplikasi yang berbeda mungkin memerlukan metode analisis yang agak berbeda. Solusi-solusi efektif telah diajukan dan diuji, yang menghasilkan banyak metode analisis statistik berbasis data mining yang baru dan dapat diukur.

10.5.2 Machine learning and data mining

Machine learning adalah bidang yang menginvestigasi bagaimana komputer dapat belajar (atau improve their performance) berdasarkan data (Macuácua et al., 2023).

Machine learning adalah disiplin yang berkembang pesat, dengan banyak metodologi dan aplikasi baru yang dikembangkan dalam beberapa tahun terakhir, mulai dari mesin vektor pendukung hingga model grafik probabilitas dan deep learning, yang akan kami bahas dalam buku ini.

Secara umum, machine learning mengatasi dua masalah klasik: supervised learning (pembelajaran berpengawasan) dan unsupervised learning (pembelajaran tanpa pengawasan) (Mistry et al., 2023).

1. Supervised learning (pembelajaran berpengawasan): Contoh klasik dari supervised learning adalah klasifikasi. Pengawasan dalam pembelajaran berasal dari contoh-contoh training data set. Misalnya, untuk secara otomatis mengenali

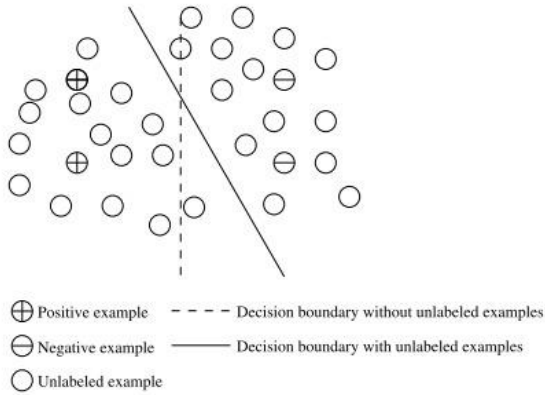
kode pos yang ditulis tangan pada surat, sistem pembelajaran menggunakan kumpulan gambar kode pos yang ditulis tangan dan terjemahan mesin yang sesuai sebagai contoh pelatihan, dan mempelajari (yaitu, menghitung) model klasifikasi.

2. Unsupervised learning (pembelajaran tanpa pengawasan): Contoh klasik dari unsupervised learning adalah pengelompokan. Proses pembelajaran adalah tanpa pengawasan karena contoh not class-labeled. Biasanya, kita dapat menggunakan pengelompokan untuk menemukan kelompok dalam data. Misalnya, metode pembelajaran tanpa pengawasan dapat mengambil, sebagai masukan, sekelompok gambar digit yang ditulis tangan. Misalkan itu menemukan 10 kelompok data. Kelompok-kelompok ini mungkin seharusnya sesuai dengan 10-digit yang berbeda dari 0 hingga 9, masing-masingnya. Namun, karena training data are not labeled, model yang telah dipelajari tidak dapat memberi tahu kita makna semantik dari kelompok-kelompok yang ditemukan.

Meskipun ada banyak kesamaan antara data mining dan machine learning dalam kedua masalah dasar ini, data mining berbeda dari machine learning dalam beberapa aspek utama. Pertama, bahkan pada tugas yang mirip seperti klasifikasi dan pengelompokan, data mining sering bekerja pada kumpulan data yang sangat besar, atau bahkan pada aliran data tak terbatas, skalabilitas bisa menjadi perhatian penting, dan banyak algoritma data mining yang efisien dan sangat scalable atau algoritma pertambangan aliran harus dikembangkan untuk menyelesaikan tugas-tugas tersebut.

Kedua, dalam banyak masalah data mining, kumpulan data biasanya besar, tetapi training data masih bisa cukup kecil karena mahal bagi para ahli untuk memberikan label kualitas tinggi untuk banyak contoh. Oleh karena itu, data mining harus

menghabiskan banyak usaha untuk mengembangkan metode berbasis supervised yang lemah. Ini termasuk metodologi seperti pembelajaran semi-supervised dengan sekelompok kecil data berlabel tetapi sekelompok besar data unlabeled data (dengan gagasan dalam Gambar 11.5), integrasi atau ensemble dari beberapa model lemah yang diperoleh dari nonahli (misalnya, yang diperoleh dari crowd-sourcing), pengawasan jarak jauh, seperti menggunakan pengetahuan yang tersedia dan umum (tetapi jauh berhubungan dengan masalah yang akan diselesaikan) basis data (misalnya, wikipedia, DBPedia), pembelajaran aktif dengan teliti memilih contoh untuk ditanyakan kepada para ahli, atau pembelajaran transfer dengan mengintegrasikan model yang dipelajari dari domain masalah yang mirip. Data mining telah memperluas metode berbasis supervised yang lemah untuk membangun model klasifikasi berkualitas pada kumpulan data besar dengan sekelompok kecil data pelatihan berkualitas tinggi yang sangat terbatas.



Gambar 10.5 Semisupervised Learning
(Sumber: Han et al., 2023)

Ketiga, metode machine learning mungkin tidak mampu menangani banyak jenis masalah penemuan pengetahuan pada big data. Di sisi lain, data mining, yang mengembangkan solusi efektif untuk masalah-masalah aplikasi konkret, menggali lebih

dalam dalam domain masalah, dan meluas jauh di luar cakupan yang dicakup oleh pembelajaran mesin. Sebagai contoh, banyak masalah aplikasi, seperti analisis data transaksi bisnis, analisis urutan eksekusi program perangkat lunak, dan analisis struktur kimia dan biologi, memerlukan metode efektif untuk menambang pola-pola yang sering muncul, pola-pola sekuensial, dan pola-pola terstruktur. Penelitian data mining telah menghasilkan banyak metode mining yang dapat diskalakan, efektif, dan beragam untuk tugas-tugas seperti itu. Sebagai contoh lain, analysis of large-scale social dan information networks memunculkan banyak masalah menantang yang mungkin tidak sesuai dengan cakupan tipikal dari banyak metode machine learning karena information interaction across links dan nodes in such networks. Data mining telah mengembangkan banyak solusi menarik untuk masalah-masalah seperti itu.

10.6 Data Mining and Applications

Untuk menunjukkan pentingnya aplikasi data mining, kita secara singkat membahas beberapa contoh aplikasi data mining yang sangat sukses dan popular data mining: business intelligence; search engines; social media and social networks; dan biology, medical science, and health care (Schuh et al., 2019).

11.6.1 Business intelligence

Penting bagi bisnis untuk memahami konteks komersial organisasi dengan lebih baik, seperti pelanggan, pasar, pasokan dan sumber daya, dan pesaing. Teknologi Business intelligence (BI) menyediakan provide historical, current, dan predictive views of business operations. Contohnya termasuk pelaporan, pemrosesan analitis online, manajemen kinerja bisnis, inteligensia kompetitif, pembandingan, dan analisis prediktif.

"Seberapa penting data mining dalam Business intelligence?" Tanpa data mining, banyak bisnis mungkin tidak dapat melakukan analisis pasar yang efektif, membandingkan umpan balik pelanggan tentang produk serupa, menemukan kelebihan dan kelemahan pesaing mereka, mempertahankan pelanggan yang sangat berharga, dan membuat keputusan bisnis cerdas.

Jelas, data mining adalah inti dari Business intelligence. Alat pemrosesan analitis online dalam bisnis inteligensia bergantung pada data warehousing dan multidimensional data mining. Teknik klasifikasi dan prediksi adalah inti dari analisis prediktif dalam bisnis inteligensia, yang memiliki banyak aplikasi dalam menganalisis pasar, pasokan, dan penjualan. Selain itu, pengelompokan memainkan peran sentral dalam manajemen hubungan pelanggan, yang mengelompokkan pelanggan berdasarkan kesamaan mereka. Dengan menggunakan teknik multidimensional summarization, kita dapat lebih memahami fitur dari setiap kelompok pelanggan dan mengembangkan program reward pelanggan yang disesuaikan (Chen et al., 2023).

10.6.2 Web search engines

Web search engines adalah server komputer khusus yang mencari informasi di Web. Hasil pencarian dari pertanyaan pengguna seringkali dikembalikan dalam bentuk daftar (disebut hits). Hits dapat terdiri dari halaman web, gambar, dan jenis file lainnya. Beberapa mesin pencari juga mencari dan mengembalikan data yang tersedia dalam basis data publik atau direktori terbuka. Mesin pencari berbeda dari web directories karena direktori web dikelola oleh editor, sedangkan mesin pencari beroperasi secara algoritmik atau dengan campuran input algoritmik dan human (Tran et al., 2023).

Mesin pencari menimbulkan tantangan besar bagi data mining. Pertama, mereka harus menangani jumlah data yang sangat besar dan terus bertambah. Biasanya, data seperti itu tidak dapat diproses menggunakan satu atau beberapa mesin.

Sebaliknya, mesin pencari sering perlu menggunakan cloud komputer, yang terdiri dari ribuan atau bahkan ratusan ribu komputer yang bekerja sama untuk menambang jumlah data yang besar tersebut. Mengembangkan metode data mining yang dapat diukur di cloud komputer dan kumpulan data yang besar adalah area penelitian dan pengembangan yang aktif.

Kedua, mesin pencari web sering harus berurusan dengan data online. Sebuah mesin pencari mungkin mampu membuat model secara offline pada kumpulan data besar. Untuk melakukannya, mesin pencari dapat membuat pengklasifikasi query yang menugaskan sebuah query pencarian ke kategori yang telah ditentukan berdasarkan topik query (yaitu, apakah query pencarian "apel" dimaksudkan untuk mendapatkan informasi tentang buah atau merek komputer). Bahkan jika sebuah model dibuat secara offline, adaptasi model secara online harus cukup cepat untuk menjawab pertanyaan pengguna secara real time.

10.6.3 Social media and social networks

Kemunculan media sosial dan jaringan sosial telah secara mendasar mengubah kehidupan kita dan cara kita bertukar informasi serta bersosialisasi saat ini (Du et al., 2023). Dengan jumlah data media sosial dan jaringan sosial yang luar biasa banyaknya, sangat penting untuk menganalisis data tersebut guna mengidentifikasi pola dan tren yang dapat diambil tindakan dari data media sosial dan jaringan sosial.

Social media mining adalah proses penyaringan melalui jumlah data media sosial yang besar (misalnya, penggunaan media sosial, perilaku sosial online, koneksi antar individu, perilaku belanja online, pertukaran konten, dll.) untuk mengidentifikasi pola dan tren. Pola dan tren tersebut telah digunakan untuk mendeteksi peristiwa sosial, pemantauan dan pengawasan kesehatan masyarakat, analisis sentimen dalam media sosial, rekomendasi dalam media sosial, provenans informasi, analisis kepercayaan media sosial, dan deteksi spammer sosial.

Social network mining adalah upaya untuk menyelidiki struktur jaringan sosial dan informasi yang terkait dengan jaringan tersebut melalui penggunaan teori jaringan dan grafik serta metode data mining. Social network structures ditandai dalam hal nodes (aktor individu, orang, atau objek dalam jaringan) dan ikatan, tepian, atau tautan (hubungan atau interaksi) yang menghubungkannya. Contoh struktur sosial yang biasa divisualisasikan melalui analisis jaringan sosial meliputi social media networks, memes spread, friendship and acquaintance networks, collaboration graphs, kinship, disease transmission, dan sexual relationships. Jaringan-jaringan ini sering divisualisasikan melalui sociogram di mana nodes direpresentasikan sebagai titik-titik dan ikatan direpresentasikan sebagai garis-garis.

Social network mining telah digunakan untuk mendeteksi komunitas tersembunyi, mengungkapkan evolusi dan dinamika jaringan sosial, menghitung ukuran jaringan (misalnya, sentralitas, transitivitas, rekorisitas, keseimbangan, status, dan kemiripan), menganalisis bagaimana informasi menyebar di situs media sosial, mengukur dan memodelkan pengaruh simpul/struktur dan homofili, serta melakukan analisis jaringan sosial berbasis lokasi.

10.6.4 Biology, medical science, and health care

Bidang biologi, ilmu kedokteran, dan perawatan kesehatan juga telah menghasilkan data besar dalam skala eksponensial. Data biomedis memiliki berbagai bentuk, mulai dari "omics" hingga imaging, mobile health, and electronic health records. Dengan ketersediaan metode pengumpulan digital yang lebih efisien, ilmuwan biomedis dan dokter kini menemukan diri mereka menghadapi set data yang semakin besar dan mencoba mencari cara kreatif untuk menyaring data ini dan memberikan makna. Memang, data yang dulunya dianggap besar sekarang terasa kecil karena jumlah data yang dikumpulkan dalam satu hari oleh

seorang peneliti dapat melampaui apa yang mungkin dihasilkan selama karirnya bahkan satu dekade yang lalu. Banjir informasi biomedis ini memerlukan pemikiran baru tentang bagaimana data dapat dikelola dan dianalisis untuk meningkatkan pemahaman ilmiah dan meningkatkan pelayanan kesehatan.

10.7 Data Mining and Society

Dengan data mining dalam kehidupan sehari-hari kita, penting untuk mempelajari dampak dari teknologi data minig terhadap masyarakat. Bagaimana kita dapat menggunakan teknologi data mining untuk kepentingan masyarakat? Bagaimana kita dapat melindungi diri dari penyalahgunaannya? Pengungkapan atau penggunaan data yang tidak tepat dan potensi pelanggaran hak privasi individu serta hak perlindungan data adalah bidang yang perlu mendapat perhatian.

Data mining akan membantu dalam penemuan ilmiah, pengelolaan bisnis, pemulihan ekonomi, dan perlindungan keamanan (misalnya, penemuan penyusup dan serangan siber secara real-time). Namun, ini juga membawa risiko pengungkapan tidak sengaja terhadap beberapa informasi bisnis atau pemerintah yang bersifat rahasia dan pengungkapan informasi pribadi individu. Studi tentang keamanan data dalam data mining dan penerbitan data yang menjaga privasi dan privacy-preserving data publishing dan data mining are important, ongoing research theme.

10.8 Kesimpulan

1. Necessity is the mother of invention. Dengan pertumbuhan data yang semakin meningkat dalam setiap aplikasi, data mining memenuhi kebutuhan untuk analisis data yang efektif, scalable, dan fleksibel dalam masyarakat kita. Data mining dapat dianggap sebagai evolusi dari teknologi informasi dan sebagai pertemuan beberapa disiplin terkait dan domain aplikasi.

2. Data mining adalah proses penemuan pola dan pengetahuan menarik dari jumlah data yang sangat besar. As a knowledge discovery process, biasanya melibatkan data cleaning, data integration, data selection, data transformation, pattern and model discovery, pattern or model evaluation, and knowledge presentation.
3. Sebuah pola atau model dianggap menarik jika valid pada data uji dengan tingkat kepastian tertentu, baru, berpotensi berguna (misalnya, dapat diambil tindakan atau memvalidasi kecurigaan tentang yang pengguna ingin tahu), dan mudah dimengerti oleh manusia. Pola menarik mewakili pengetahuan. Pattern interestingness, either objective or subjective, dapat digunakan untuk memandu proses penemuan.
4. Data mining dapat dilakukan pada berbagai jenis data selama data tersebut memiliki makna untuk aplikasi tertentu, seperti data terstruktur (misalnya, basis data relasional, data transaksi) dan data tak terstruktur (misalnya, data teks dan multimedia), serta data yang terkait dengan berbagai aplikasi. Data juga dapat dikategorikan sebagai data yang disimpan vs. data aliran, sedangkan yang terakhir mungkin perlu menjelajahi stream mining algorithms.
5. Data mining functionalities digunakan untuk menentukan jenis pola atau knowledge yang akan ditemukan dalam tugas data mining. Fungsi-fungsi tersebut termasuk karakterisasi dan diskriminasi; the mining of frequent patterns, associations, and correlations; classification and regression; deep learning; cluster analysis; and outlier detection. Seiring munculnya jenis data baru, aplikasi baru, dan permintaan analisis baru, tidak diragukan lagi kita akan melihat lebih banyak tugas data mining yang baru di masa depan.
6. Data mining adalah perpaduan dari beberapa disiplin tetapi memiliki fokus penelitian yang unik, yang didedikasikan

untuk banyak aplikasi canggih. Dapat mempelajari hubungan erat antara data mining dengan statistik, pembelajaran mesin, teknologi basis data, dan banyak disiplin lainnya.

7. Data mining memiliki banyak aplikasi sukses, seperti business intelligence, pencarian web, bioinformatika, informatika kesehatan, keuangan, perpustakaan digital, dan pemerintahan digital.
8. Data mining mungkin memiliki dampak yang kuat pada masyarakat dan kajian tentang dampak tersebut, seperti bagaimana memastikan efektivitas data mining dan sekaligus memastikan privasi dan keamanan data, telah menjadi isu penting dalam penelitian.

DAFTAR PUSTAKA

Aslani, N., Galehdar, N., & Garavand, A. (2023). A systematic review of data mining applications in kidney transplantation. *Informatics in Medicine Unlocked*, *37*, 101165. https://doi.org/https://doi.org/10.1016/j.imu.2023.101165

Chen, K., Abtahi, F., Carrero, J.-J., Fernandez-Llatas, C., & Seoane, F. (2023). Process mining and data mining applications in the domain of chronic diseases: A systematic review. *Artificial Intelligence in Medicine*, *144*, 102645. https://doi.org/https://doi.org/10.1016/j.artmed.2023.102645

Dol, S. M., & Jawandhiya, P. M. (2023). Classification Technique and its Combination with Clustering and Association Rule Mining in Educational Data Mining — A survey. *Engineering Applications of Artificial Intelligence*, *122*, 106071. https://doi.org/https://doi.org/10.1016/j.engappai.2023.106071

Du, Q., Li, Y., Li, Y., Zhou, J., & Cui, X. (2023). Data mining of social media for urban resilience study: A case of rainstorm in Xi'an. *International Journal of Disaster Risk Reduction*, *95*, 103836. https://doi.org/https://doi.org/10.1016/j.ijdrr.2023.103836

Duque, J., Silva, F., & Godinho, A. (2023). Data Mining applied to Knowledge Management. *Procedia Computer Science*, *219*, 455–461. https://doi.org/https://doi.org/10.1016/j.procs.2023.01.312

Flores, K. R., de Carvalho, L. V. F. M., Reading, B. J., Fahrenholz, A., Ferket, P. R., & Grimes, J. L. (2023). Machine learning and

data mining methodology to predict nominal and numeric performance body weight values using Large White male turkey datasets. *Journal of Applied Poultry Research*, *32*(4), 100366. https://doi.org/https://doi.org/10.1016/j.japr.2023.100366

Han, J., Pei, J., & Tong, H. (2023). *Data Mining: Concepts and Techniques*.

Krishnamoorthy, M., & Karthikeyan, R. (2022a). Competence of medicinal plant database using data mining algorithms for large biological databases. *Measurement: Sensors*, *24*, 100420. https://doi.org/https://doi.org/10.1016/j.measen.2022.100420

Krishnamoorthy, M., & Karthikeyan, R. (2022b). Pattern mining algorithms for data streams using itemset. *Measurement: Sensors*, *24*, 100421. https://doi.org/https://doi.org/10.1016/j.measen.2022.100421

Macuácua, J. C., Centeno, J. A. S., & Amisse, C. (2023). Data mining approach for dry bean seeds classification. *Smart Agricultural Technology*, *5*, 100240. https://doi.org/https://doi.org/10.1016/j.atech.2023.100240

Mistry, S., Riches, N. O., Gouripeddi, R., & Facelli, J. C. (2023). Environmental exposures in machine learning and data mining approaches to diabetes etiology: A scoping review. *Artificial Intelligence in Medicine*, *135*, 102461. https://doi.org/https://doi.org/10.1016/j.artmed.2022.102461

Schuh, G., Reinhart, G., Prote, J.-P., Sauermann, F., Horsthofer, J., Oppolzer, F., & Knoll, D. (2019). Data Mining Definitions and Applications for the Management of Production Complexity. *Procedia CIRP*, *81*, 874–879.

https://doi.org/https://doi.org/10.1016/j.procir.2019.03.217

Tran, T., Ruppert, T., Eigner, G., & Abonyi, J. (2023). Assessing human worker performance by pattern mining of Kinect sensor skeleton data. *Journal of Manufacturing Systems*, *70*, 538–556. https://doi.org/https://doi.org/10.1016/j.jmsy.2023.08.010

BAB 11
Machine Learning (Teori, Algoritma dan Implementasinya)

Oleh Erick Fernando

11.1 Sejarah *Machine Learning*

Awal *machine learning* dapat ditelusuri kembali ke tahun 1943 ketika Walter Pitts dan Warren McCulloch memperkenalkan kerangka matematika awal untuk jaringan saraf dalam publikasi ilmiah mereka yang berjudul " *A logical calculus of the ideas immanent in nervous activity."* Selanjutnya, karya Donald Hebb, *The Organization of Behavior*, dirilis pada tahun 1949. Buku ini menyajikan kerangka teoritis yang mengeksplorasi hubungan antara perilaku, jaringan saraf, dan aktivitas otak. Diharapkan bahwa penelitian ini akan memberikan kontribusi yang signifikan terhadap kemajuan *machine learning*(Fradkov, 2020; Cohen, 2021; Foote, 2022).

Pada tahun 1950, Tes Turing dirancang oleh Alan Turing sebagai sarana untuk menilai keberadaan kecerdasan asli dalam sistem komputer. Agar berhasil menavigasi ujian, sistem komputer harus memiliki kemampuan untuk secara meyakinkan menipu pengamat manusia agar menganggapnya sebagai sesama manusia. Ide tersebut diamati dalam artikel ilmiah berjudul "*Computing Machinery and Intelligence"* yang ditulis oleh individu selama masa jabatan mereka di University of Manchester. Perceptron, yang dikembangkan oleh psikolog Amerika Frank Rosenblatt pada tahun 1957, diakui sebagai model jaringan saraf perdana yang meniru proses kognitif otak

manusia. Penemuan ini memiliki signifikansi kontemporer. *The nearest neighbour*, yang pertama kali diperkenalkan pada tahun 1967, adalah algoritma terkemuka yang menangani *'travelling salesman problem'*. Tantangan ini memerlukan seorang salesman yang memulai di kota yang sewenang-wenang dan berulang kali mengunjungi kota-kota yang berdekatan sampai semua kota telah dikunjungi.

Konsep *backpropagation* muncul pada paruh kedua abad ke-20, awalnya didirikan pada 1960-an dan diperkenalkan kembali pada 1980-an. Tujuannya adalah untuk mengidentifikasi lapisan tersembunyi di dalam jaringan saraf, yang terletak di antara lapisan input dan output, sehingga menjadikannya layak untuk aplikasi komersial. Pada tahun 1981, Gerald Dejong membuat penemuan penting yang disebut *Explanation Based Learning* (EBL), yang melibatkan pemanfaatan algoritma komputer untuk memfasilitasi proses penjelasan dan generalisasi data. Jaringan saraf NetTalk, dikembangkan pada tahun 1985 oleh Terry Sejnowski, adalah sebuah algoritma yang menunjukkan kemampuan untuk menghasilkan ucapan seperti bayi. Ini dicapai dengan memanfaatkan data tekstual dan transkrip fonetik yang sesuai sebagai masukan. Algoritma *Q-learning*, yang dibuat oleh Christopher Watkins pada tahun 1989, telah meningkatkan implementasi praktis dari pembelajaran penguatan secara signifikan. Pendekatan statistik untuk algoritma memperoleh popularitas pada 1990-an karena keterbatasan yang dirasakan dari jaringan saraf, yang dianggap kurang dapat ditafsirkan dan membutuhkan sumber daya pemrosesan yang lebih besar. Metode tersebut mencakup *support vector machines* dan *random forest algorithms*, yang keduanya pertama kali diterbitkan pada tahun 1995. Selanjutnya, pencapaian signifikan di bidang kecerdasan buatan terjadi pada tahun 1997, di mana IBM Deep Blue muncul sebagai pemenang atas juara catur dunia yang berkuasa (Fradkov, 2020; Cohen, 2021; Foote, 2022).

Selama tahun 2000-an, ada lonjakan popularitas *support vector clustering*, *unsupervised learning*, dan pendekatan kernel dalam bidang *Machine Learning*. Munculnya convolutional neural networks pada tahun 2009 merupakan tonggak penting. Pada tahun 2009, Fei-Fei Li, seorang profesor ilmu komputer terkemuka di Universitas Stanford, mengembangkan kumpulan data ekstensif yang secara akurat mewakili skenario dunia nyata. Kumpulan data ini berfungsi sebagai dasar untuk karya terobosan Alex Krizhevsky, yang kemudian memperkenalkan *Convolutional Neural Network* (CNN) pertama yang dikenal sebagai AlexNet pada tahun 2012. Pada tahun 2011, Watson dari IBM meraih kemenangan atas kontestan manusia dalam acara game Jeopardy, sementara Google Brain meluncurkan komputer yang mampu mengkategorikan item, bahkan kucing. Selanjutnya, organisasi mengembangkan algoritma yang menggunakan video YouTube dan foto tanpa label untuk mendeteksi dan mengklasifikasikan entitas kucing, per tahun 2012. Algoritma Word2vec, yang diperkenalkan pada 2013, menggunakan jaringan saraf untuk memperoleh pengetahuan tentang asosiasi kata, sehingga berfungsi sebagai dasar untuk *large language models*. Pada tahun 2014, Facebook memperkenalkan DeepFace, sebuah algoritma yang mencapai tonggak penting dengan melampaui tolok ukur sebelumnya di bidang algoritma komputer untuk pengenalan wajah. Pada tahun 2014, Ian Goodfellow memperkenalkan *generative adversarial network* (GANs). (Simeone, 2018; Alpaydın, 2020).

Saat ini, arsitektur algoritmik berfungsi sebagai kerangka fundamental untuk pembuatan gambar, video, dan audio, yang banyak digunakan dalam pembuatan deepfake. Pada tahun 2018, DeepMind mencetak pencapaian penting melalui pengembangan AlphaFold, model komputasi yang terkenal karena kemampuannya memprediksi konfigurasi struktural protein secara akurat. Rentang waktu termasuk 2015 hingga saat ini, sering dikenal sebagai Dekade sejak 2015, telah

menyaksikan pembuatan algoritme yang sangat canggih. Namun demikian, penerapan, pengamanan, dan inklusivitas algoritme ini telah memicu pertanyaan dan kekhawatiran yang substansial.

11.2 *Machine Learning*

Mechine Learning adalah pendekatan komputasi yang memungkinkan komputer memperoleh pengetahuan secara mandiri dan meningkatkan kinerja dengan menganalisis dan memproses data yang sudah ada (Burger, 2018; Simeone, 2018; Kubat, 2021; Chatterjee *et al.*, 2023; Chopra and Khurana, 2023). Konsep ML pertama kali berkisar pada gagasan yang memungkinkan mesin memperoleh kemampuan untuk melakukan berbagai tugas, seperti menghasilkan wawasan dari data tekstual atau terlibat dalam permainan strategis, dan lain-lain. Secara umum, pendekatan Machine Learning melibatkan inisialisasi model dengan berbagai parameter. Model umum ini memiliki kemampuan untuk melakukan berbagai tugas, bergantung pada definisi spesifik yang ditetapkan pada parameternya. Proses pembelajaran melibatkan optimalisasi nilai parameter untuk mendapatkan kesesuaian optimal antara model dan data observasi selama fase pelatihan(Carbonell *et al.*, 2014; Cohen, 2020). Berdasarkan data pelatihan yang diberikan, model generik dapat disesuaikan untuk tugas tertentu dengan menyesuaikan parameter utama. Selanjutnya, model tersebut menjalani pelatihan, menghasilkan iterasi spesifik dari template umum, yang kemudian berfungsi sebagai algoritme untuk tugas yang diberikan.

Intinya, ML menggunakan serangkaian algoritme untuk membuat model matematika dan memberikan prediksi berdasarkan data atau informasi masa lalu. ML digunakan dalam beragam aplikasi, termasuk namun tidak terbatas pada

pengenalan gambar, pengenalan ucapan, pemfilteran email, penandaan otomatis Facebook, dan sistem rekomendasi.

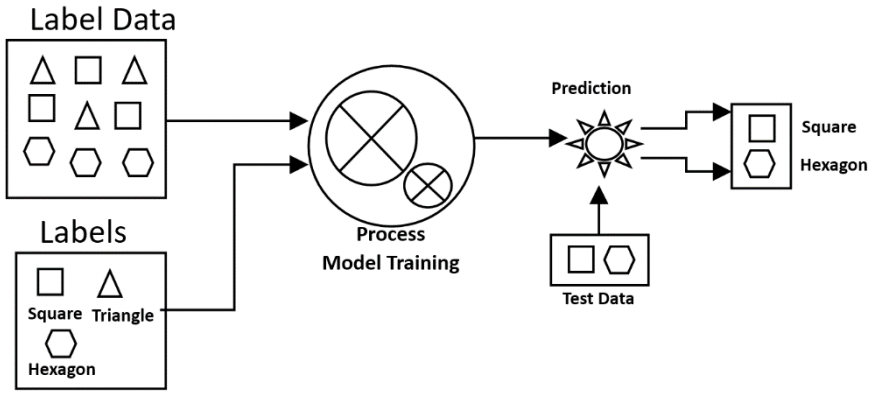
11.3 Tipe-Tipe *Machine Learning*

11.3.1 Supervised Learning

Supervised Learning adalah jenis *Machine Learning* di mana mesin dilatih menggunakan data pelatihan yang "berlabel" dengan baik, dan berdasarkan data tersebut, mesin memprediksi keluarannya. Data berlabel berarti beberapa data masukan sudah ditandai dengan keluaran yang benar (Burger, 2018; Kubat, 2021; Chopra and Khurana, 2023). Dalam *Supervised Learning*, data pelatihan yang diberikan kepada mesin berfungsi sebagai pengawas yang mengajarkan mesin untuk memprediksi keluaran dengan benar. *Supervised Learning* adalah proses penyediaan data masukan serta data keluaran yang benar ke model ML. Tujuan dari algoritma Supervised Learning adalah menemukan fungsi pemetaan untuk memetakan variabel masukan (x) dengan variabel keluaran (y).

Supervised Learning, model dilatih menggunakan kumpulan data berlabel, tempat model mempelajari setiap jenis data. Setelah proses pelatihan selesai, model diuji berdasarkan data pengujian (bagian dari set pelatihan), dan kemudian memprediksi keluarannya.

Cara kerja *Supervised Learning* dapat dengan mudah dipahami melalui contoh dan diagram di bawah ini:



Gambar 11. 1 Simulasi *Supervised Learning*

Algoritma *Supervised Learning* mengambil sekumpulan data masukan yang diketahui (kumpulan pembelajaran) dan respons yang diketahui terhadap data tersebut (output), dan membentuk model untuk menghasilkan prediksi yang masuk akal untuk respons terhadap data masukan baru. Gunakan *Supervised Learning* jika anda memiliki data untuk keluaran yang ingin Anda prediksi.

11.3.2 Semi-supervised learning

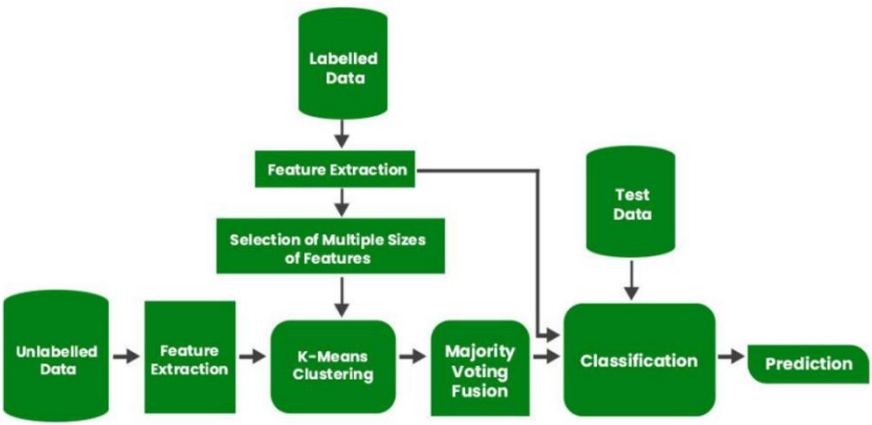
Semi-supervised learning merupakan teknik ML metode yang menggunakan sejumlah kecil data berlabel dan sejumlah besar data tidak berlabel untuk melatih suatu model(Nedal and Morgan, 2018; Simeone, 2018; Alpaydın, 2020; Chatterjee *et al.*, 2023). *Semi-supervised learning* dapat diartikan juga sebagai teknik algoritma Machine Learning yang mewakili perantara antara algoritma *Semi-supervised learning* dan *unsupervised learning*.

Tujuan *Semi-supervised learning* untuk mempelajari suatu fungsi yang dapat secara akurat memprediksi variabel keluaran berdasarkan variabel masukan, serupa dengan *supervised learning*. Dengan demikian, *Semi-supervised learning* sangat berguna ketika tersedia sejumlah besar data tak berlabel, namun terlalu mahal atau sulit untuk memberi label pada semuanya.

Semi-supervised learning menggunakan pelabelan semu untuk melatih model dengan data pelatihan yang diberi label lebih sedikit dibandingkan *Supervised Learning*. Prosesnya dapat menggabungkan berbagai model jaringan saraf dan cara pelatihan. Keseluruhan cara kerja *Semi-supervised learning* dijelaskan pada poin-poin di bawah ini:

Pertama, model ini dilatih dengan jumlah data pelatihan yang lebih sedikit, serupa dengan model *supervised learning*. Pelatihan berlanjut hingga model memberikan hasil yang akurat. Algoritma menggunakan kumpulan data yang tidak berlabel dengan label semu pada langkah berikutnya, dan sekarang hasilnya mungkin tidak akurat.

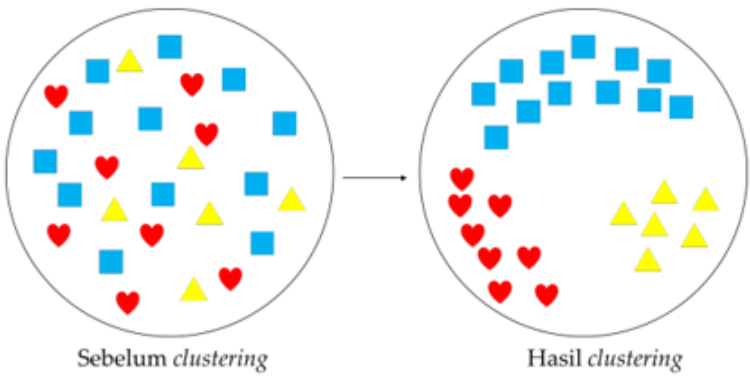
Kedua, label dari data pelatihan berlabel dan data label semu dihubungkan bersama. Data masukan dalam data pelatihan berlabel dan data pelatihan tidak berlabel juga dihubungkan. Pada akhirnya, latih kembali model dengan masukan gabungan baru seperti yang dilakukan pada langkah pertama. Ini akan mengurangi kesalahan dan meningkatkan akurasi model.



Gambar 11.2 Simulasi *Semi Supervised Learning*

11.3.3 Unsupervised learning

Unsupervised learning melakukan pembelajaran secara langsung dari data mentah dan tanpa kehadiran supervisor dalam loop pelatihan dalam bentuk label(Simeone, 2018; Yang, 2019; Kubat, 2021; Chatterjee *et al.*, 2023). Algoritma *Unsupervised learning* bebas mengeksplorasi distribusi data yang mendasarinya dan menghasilkan model yang paling menggambarkan keseluruhan data yang disajikan. Proses pelatihan tidak dipandu dengan pemberian label dan dengan demikian algoritma *Unsupervised learning* lebih kuat karena mereka dapat menemukan model pola yang mungkin tidak terpikirkan oleh seorang pakar. Namun tugas para pakar domain adalah untuk memahami pola yang ditemukan dan menjelaskannya. Algoritma *Unsupervised learning* hanya menggunakan distribusi data untuk mengeksplor wawasan yang mungkin dalam bentuk cluster, grup, atau distribusi. Ada banyak penerapan algoritma *Unsupervised learning* seperti *clustering, dimensionality detection, generative models*, dll. *Clustering* adalah salah satu implementasi *Unsupervised learning* yang populer yang melibatkan penemuan otomatis cluster dari data mentah.



Gambar 11.3 Ilustrasi *Clustering*

Selanjutnya, dimensionality reduction merupakan teknik ML yang mengurangi jumlah atribut (fitur) untuk mendapatkan atribut yang paling relevan dalam meningkatkan prediksi. Banyak studi empiris menyatakan bahwa model dengan jumlah fitur atau dimensi yang lebih banyak memiliki kinerja generalisasi yang lebih buruk. Model dapat belajar dengan baik dengan fitur yang lebih sedikit. Selain itu Fitur informatif, model lebih mengembangkan representasi valid tentang data. Salah satu contoh, *Principal component Analysis* (PCA) merupakan teknik reduksi dimensi yang paling populer. PCA mengurangi dimensi data dengan mengidentifikasi sumbu yang paling banyak mengandung variabilitas. PCA dapat menemukan komponen-komponen utama yang menawarkan ciri-ciri paling diskriminatif dari data.

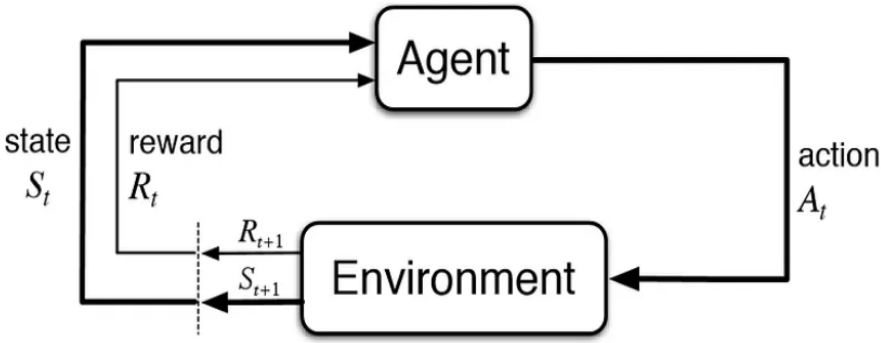
Terakhir, *Generative models* merupakan suatu teknik yang mencakup distribusi data, dan memberi tahu hasil dengan seberapa besar kemungkinan suatu contoh diberikan. Misalnya, *Generative models* yang memprediksi kata berikutnya dalam suatu rangkaian biasanya.

11.3.4 Reinforcement Learning

Reinforcement Learning dapat berupa agen pengambilan keputusan yang mengambil tindakan dalam lingkungan dan menerima imbalan (atau penalti) atas tindakannya dalam mencoba memecahkan masalah(Nedal and Morgan, 2018; Simeone, 2018; Yang, 2019; Alpaydın, 2020; Kubat, 2021; Chatterjee *et al.*, 2023). Setelah serangkaian coba-coba dijalankan, itu harus mempelajari kebijakan terbaik, yaitu urutannya tindakan yang memaksimalkan total *reward*.

Reinforcement Learning melakukan penyelesaian tugas dengan menggunakan serangkaian tindakan. Penyelesaian akan mendapat umpan balik dalam bentuk imbalan. Imbalan mendefinisikan masalah dan diperlukan jika kita menginginkan agen pembelajaran. Agen pembelajaran mempelajari rangkaian

tindakan terbaik untuk memecahkan masalah di mana "terbaik" diukur sebagai rangkaian tindakan yang memiliki imbalan kumulatif maksimum.



Gambar 11.4 *Reinforcement Learning*

Keterangan:
Environment — lingkungan fisik tempat agen beroperasi
State — Situasi agen saat ini
Reward — Umpan balik dari lingkungan
Policy — Metode untuk memetakan status agen ke tindakan
Value — Hadiah masa depan yang akan diterima agen dengan mengambil tindakan dalam keadaan tertentu

11.4 Algoritma *Machine Learning*

11.4.1 Naive Bayes

Naive Bayes adalah teknik yang dikenal luas dalam klasifikasi statistik yang memanfaatkan pendekatan Teorema Bayes. Metode ini dianggap sebagai salah satu bentuk paling dasar dari algoritma *supervised learning.* Algoritma Naive Bayes dikenal karena efisiensi, presisi, dan keandalannya ketika diterapkan pada kumpulan data yang luas. Metode ini beroperasi dengan asumsi bahwa dampak karakteristik dalam suatu kelas tidak bergantung pada pengaruh fitur lainnya. Asumsi ini terlalu

menyederhanakan perhitungan sehingga dianggap naif. Asumsi ini disebut independensi bersyarat kelas.

Formula dasar

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Keterangan :

P(h): probabilitas hipotesis h benar (apa pun datanya). Ini dikenal sebagai probabilitas sebelumnya dari h.

P(D): probabilitas data (terlepas dari hipotesisnya). Ini dikenal sebagai probabilitas sebelumnya.

P(h|D): probabilitas hipotesis h jika diberi data D. Ini dikenal sebagai probabilitas posterior.

P(D|h): probabilitas data d jika hipotesis h benar. Ini dikenal sebagai probabilitas posterior.

11.4.2 K-means

Teknik K-means adalah prosedur berulang yang dirancang untuk membagi kumpulan data tertentu menjadi subkelompok yang berbeda dan tidak tumpang tindih, yang biasa disebut *cluster*. Setiap titik data dalam kumpulan data ditetapkan ke satu klaster, memastikan tidak ada tumpang tindih yang terjadi di seluruh *cluster*. Tujuannya adalah untuk meminimalkan ketidaksamaan titik data dalam setiap *cluster*, sekaligus memaksimalkan ketidaksamaan antar *cluster*. Prosesnya melibatkan pengalokasian titik data ke sebuah cluster dengan cara yang meminimalkan jumlah jarak kuadrat antara titik data dan pusat massa *cluster*, yang mewakili rata-rata aritmatika dari semua titik data yang ditugaskan ke cluster tersebut. Tingkat homogenitas yang lebih tinggi diamati di dalam sebuah *cluster*

ketika jumlah variasi di antara titik-titik data yang termasuk dalam *cluster* tersebut berkurang.

Formula umum:

$$J(V) = \sum_{i=1}^{C} \sum_{J=1}^{Ci} \left(||Xi - Vi|| \right)^2$$

'||xi - vj||' adalah jarak Euclidean antara xi dan vj.

'ci' adalah jumlah titik data di cluster ke-i.

'c' adalah jumlah pusat cluster.

Langkah-langkah algoritma untuk pengelompokan k-means

Misalkan X = {x1,x2,x3,……..,xn} adalah himpunan titik data dan V = {v1,v2,…….,vc} adalah himpunan pusat.

1. Pilih pusat cluster 'c' secara acak.
2. Hitung jarak antara setiap titik data dan pusat cluster.
3. Tetapkan titik data ke pusat cluster yang jaraknya dari pusat cluster minimal dari semua pusat cluster.
4. Hitung ulang pusat cluster baru menggunakan:
   dimana, 'ci' mewakili jumlah titik data di cluster ke-i.

$$Vi = (1/Ci) \sum_{j=1}^{Ci} Xi$$

5. Menghitung ulang jarak antara setiap titik data dengan pusat cluster yang baru diperoleh.
6. Jika tidak ada titik data yang ditetapkan ulang maka hentikan, jika tidak, ulangi dari langkah 3).

11.4.3 K-Nearest Neighbors

K-Nearest Neighbor (KNN) merupakan salah satu teknik ML populer berfungsi dalam tugas klasifikasi dan regresi. Hal ini bergantung pada gagasan bahwa titik data yang serupa cenderung memiliki label atau nilai yang serupa. Algoritma KNN dalam proses pelatihan akan selalu menyimpan seluruh dataset pelatihan sebagai referensi. Prediksi yang dibuat menggunakan asumsi menghitung jarak antara titik data masukan dan semua contoh pelatihan, menggunakan metrik jarak yang dipilih seperti jarak Euclidean.

KNN menetapkan label kelas yang paling umum di antara K Neighbor sebagai label prediksi untuk titik data masukan. Selanjutnya mengguna regresi untuk menghitung rata-rata atau rata-rata tertimbang dari nilai target K Neighbor untuk memprediksi nilai titik data masukan.

Langkah-langkah KNN adalah:

1. Menerima data yang tidak diklasifikasikan;
2. Ukur jarak (Euclidian, Manhattan, Minkowski atau Weighted) dari data baru ke semua data lain yang sudah diklasifikasikan;
3. Mendapatkan K(K adalah parameter yang Anda tentukan) jarak yang lebih kecil;
4. Periksa daftar kelas yang memiliki jarak terpendek dan hitung jumlah setiap kelas yang muncul;
5. Mengambil kelas yang benar, kelas yang paling sering muncul;
6. Mengklasifikasikan data baru dengan kelas yang Anda ambil pada langkah 5;

Formula pengukuran :

Euclidean :

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

Manhattan :

$$\sum_{i=1}^{k}|x_i - y_i|$$

Minkowski :

$$\left(\sqrt{\sum_{i=1}^{k}(|x_i - y_i|)^2}\right)^{1/q}$$

11.4.4 Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah teknik *Supervised Learning* di bidang ML. SVM yang canggih memiliki kemampuan untuk menggunakan teknik klasifikasi linier dan nonlinier, analisis regresi, dan bahkan metodologi deteksi outlier.

Tujuan utama dari teknik SVM adalah untuk mengidentifikasi *hyperplane* terbaik dalam ruang berdimensi N yang secara efektif mempartisi titik data milik kelas berbeda dalam ruang fitur. Hal ini dicapai dengan memaksimalkan margin antara titik-titik terdekat yang termasuk dalam berbagai kelas. Dimensi hyperplane bergantung pada kuantitas fitur. Dalam skenario ketika jumlah fitur masukan adalah dua, hyperplane dapat direpresentasikan sebagai persamaan linier, sehingga menghasilkan garis lurus. Jika jumlah fitur masukan adalah tiga, bidang hiper direduksi menjadi bidang dua dimensi, dan jika

jumlah fitur melebihi tiga, maka bidang tersebut melampaui dimensi tersebut.

SVM menunjukkan keserbagunaan dalam penerapannya pada berbagai tugas, termasuk tugas klasifikasi menggunakan data teks, gambar, dan suara, serta tugas deteksi seperti identifikasi spam dan deteksi anomali wajah. Selain itu, SVM dapat digunakan untuk identifikasi tulisan tangan dan analisis ekspresi gen. SVM menunjukkan kemampuan beradaptasi dan efisiensi di berbagai aplikasi karena kemampuannya menangani masukan berdimensi tinggi secara efektif dan menangkap korelasi nonlinier.

Intuisi matematis dari SVM

Pertimbangkan masalah klasifikasi biner dengan dua kelas, diberi label +1 dan -1. Kami memiliki kumpulan data pelatihan yang terdiri dari vektor fitur masukan X dan label kelasnya yang sesuai Y.

Persamaan hyperplane linier dapat ditulis sebagai:

$$w^T x + b = 0$$

Vektor W mewakili vektor normal ke hyperplane. yaitu arah tegak lurus terhadap hyperplane. Parameter b dalam persamaan mewakili offset atau jarak hyperplane dari titik asal sepanjang vektor normal w.

Jarak antara titik data x_i dan batas keputusan dapat dihitung sebagai:

$$d_i = \frac{w^T x_i + b}{\|w\|}$$

dimana ||w|| mewakili norma Euclidean dari vektor bobot w. Norma Euclidean dari vektor normal W

Untuk pengklasifikasi Linear SVM :

$$\hat{y} = \begin{cases} 1 : w^T x + b \geq 0 \\ 0 : w^T x + b < 0 \end{cases}$$

Pengoptimalan:

Untuk pengklasifikasi SVM linier margin lebih tinggi:

$$\underset{w,b}{minimize}\ \frac{1}{2} w^T w = \underset{w,b}{minimize}\ \frac{1}{2} \|w\|^2$$

$$subject\ to\ y_i(w^T x_i + b) \geq 1\ for\ i = 1,2,3,\ldots,m$$

Variabel target atau label untuk instance pelatihan ke-i dilambangkan dengan simbol ti dalam pernyataan ini. Dan ti=-1 untuk kejadian negatif (saat yi= 0) dan ti=1 *positive instances* (saat yi = 1). Karena kita memerlukan batasan keputusan yang memenuhi batasan: $t_i(w^T x_i + b) \geq 1$

11.5 *Training*, *Validation*, dan *Testing Set*

Pada ML memerlukan proses training dan testing dalam proses mengembangan model. Training merupakan konstruksi atau bentuk model. Testing merupakan proses pengujian dari kinerja model ML yang terbentuk dari proses training. Selain itu, Proses ML memerlukan Dataset yang merupakan kumpulan data yang merupakan sampel. Sampel itu sendiri merupakan data yang terkumpul dan digunakan untuk membentuk model ML. Data dapat berupa beragam antara lain : gambar, text, suara, video, dan lain-lain. Terakhir, Ml sering menggunakan proses validation set dimana suatu proses yang dilakukan dengan himpunan data yang digunakan untuk mengoptimasi pada proses latihan model. Hal ini berfungsi untuk mendapatkan generalisasi model (model dapat membaca/mengenali pola secara generik).

Training set adalah bagian dataset yang kita latih untuk membuat prediksi atau menjalankan fungsi dari sebuah algoritma ML lainnya sesuai tujuannya masing-masing. Test set adalah bagian dataset yang digunakan dalam tes untuk melihat keakuratannya, atau dengan kata lain melihat performanya.

Komposisi atau rasio suatu dataset (training, Validation, testing) sangat beragam dapat (80%: 10%: 10%) atau (90%: 5%: 5%). Apabila proses validation set tidak digunakan dalam proses maka dapat dibagi menjadi dua bagian saja (training, testing). Hal ini terjadi pada dataset berukuran sangat kecil.

Sedangkan pada umumnya dataset dapat dibagi menjadi 2 bagian, yaitu data training dan data testing. Data training nantinya akan digunakan untuk melatih algoritma dalam mencari model yang sesuai, sedangkan data testing akan dipakai untuk menguji dan mengetahui performa model yang didapatkan pada tahapan testing.

Selain itu, pemanfaatan sampling komposit digunakan untuk membuat set pelatihan dan pengujian, dengan rasio 70% untuk pelatihan dan 30% untuk pengujian. Namun, rasio alternatif sebesar 80% untuk pelatihan dan 20% untuk pengujian juga dipertimbangkan. Pendekatan yang disebutkan di atas digunakan untuk meniru performa model secara efisien dengan terlebih dahulu mengeksekusinya pada data yang sudah ada, sebelum memasukkan seluruh data baru ke dalam sistem. Saat melakukan pemisahan pelatihan/ujian, kita sering kali mengandalkan dua asumsi utama. Data harus mencerminkan populasi secara akurat dan memberikan gambaran yang masuk akal tentang proses aktual yang memerlukan model tersebut. Proses yang sedang dipertimbangkan menunjukkan tingkat stabilitas yang tinggi sepanjang waktu, sehingga model yang dibangun menggunakan data dari bulan sebelumnya harus secara akurat menangkap pola dan dinamika data untuk bulan berikutnya.

11.6 Pengukuran Kinerja Model

Kinerja model pada ML diukur secara kuantitatif. Pengukuran model dilakukan pada saat proses latihan. Dalam proses tersebut model ML dapat dilakukan optimalisasi utility function yang bermanfaat meningkatkan performa model ML (tergantung algoritma). Optimalisasi dilakukan dengan cara mengurangi nilai error atau entropy proses pembentukan model. Evaluasi model secara eksternal dengan melewatkan data pada model (validation dan testing data) sehingga evaluasi final model dengan pengukuran performa model tersebut(setiap algoritma berbeda). Pengukuran performa dapat juga berdasarkan domain permasalahan nantinya. Contoh ukuran BLEU pada peringkasan dokumen dapat menggunakan ROUGE. Selain itu, pengukuran pada information retrival dapat menggunakan presisi, recall. F1-measure, atau mean average precision (MAP). Serta, pengukuran lain juga dapat berbeda. Pengukuran performa masing-masing miliki karakteristik nilai optimal berbeda. Kasus contoh lain pada klasifikasi selalu menggunakan akurasi.

11.6.1 Akurasi

Skor akurasi dalam ML adalah metrik evaluasi yang mengukur jumlah prediksi benar yang dibuat oleh suatu model dibandingkan dengan jumlah total prediksi yang dibuat. Akurasi dapat memberikan informasi keakuratan model yang berguna memudahkan pelaporan nilai model kepada seluruh pemangku kepentingan, sehingga meningkatkan peluang keberhasilan inisiatif ML. Akurasi menghitungnya dengan membagi jumlah prediksi yang benar dengan jumlah total prediksi.

11.6.2 F1 Score

F1 Score adalah metrik evaluasi ML yang mengukur akurasi model menggabungkan skor precision and recall model. Evaluasi

metrik dengan menghitung berapa kali model membuat prediksi yang benar di seluruh kumpulan data. Hal ini merupakan metrik yang andal hanya jika kumpulan datanya seimbang kelasnya; artinya, setiap kelas dari dataset memiliki jumlah sampel yang sama.

Precision mengukur berapa banyak prediksi "positif" yang dibuat oleh model yang benar.

Recall, mengukur berapa banyak sampel kelas positif yang ada dalam kumpulan data yang diidentifikasi dengan benar oleh model.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Pada proses perhitungan F1 Score dengan menggabungkan hasil nilai *precision* dan Recall, dan memaksimalkan skor F1 berarti memaksimalkan *precision* dan Recall secara bersamaan. F1 Score banyak menjadi pilihan peneliti untuk mengevaluasi model mereka dalam kaitannya dengan akurasi.

$$F1\ Score = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Keterangan :

True Positives (TP): Jumlah sampel diprediksi dengan benar sebagai "positive."

False Positives (FP): Jumlah sampel diprediksi dengan tidak benar sebagai"positive."

True Negatives (TN): Jumlah sampel diprediksi dengan benar sebagai"negative."

False Negatives (FN): Jumlah sampel diprediksi dengan tidak benar sebagai "negative."

11.7 *Issues in Machine Learning*

Ada banyak masalah yang dihadapi penelitian ML saat mengembangkan aplikasi berbasis ML.

11.7.1 Kualitas Data

Performa sistem Machine Learning bergantung pada kualitas data yang diberikan sebagai masukan ke sistem ML. Keberadaan data yang berisik berpotensi menyebabkan algoritma ML menghasilkan hasil yang tidak akurat. Diperlukan pemrosesan awal data, yang mencakup penghapusan outlier dan mengatasi angka-angka yang hilang. Sangat penting untuk menyelesaikan tugas ini sebelum mengirimkan data untuk tujuan pelatihan.

12.7.2 Meningkatkan Kualitas Pelatihan

Produksi keluaran berkualitas tinggi oleh sistem ML bergantung pada pelaksanaan proses pelatihan yang benar. Ada beberapa faktor yang berkontribusi terhadap peningkatan kualitas pelatihan. Disarankan untuk menambah durasi sesi pelatihan. Untuk meningkatkan kompleksitas model, fitur tambahan harus dimasukkan ke dalam kumpulan data.

11.7.3 Overfitting dan Underfitting Data Pelatihan

Overfitting mengacu pada fenomena di mana proses pelatihan dilakukan menggunakan kumpulan data yang terlalu besar, sehingga berdampak buruk pada kinerja secara keseluruhan. Salah satu pendekatan potensial untuk memitigasi masalah overfitting data adalah melakukan analisis data secara menyeluruh dan komprehensif. Untuk meningkatkan kualitas dan keandalan data, disarankan untuk menggunakan teknik tertentu. Pertama, disarankan untuk mengurangi jumlah fitur

yang digunakan dalam analisis. Ini dapat membantu menyederhanakan kumpulan data dan meningkatkan interpretasi hasil. Selain itu, sangat penting untuk mengidentifikasi dan menghilangkan outlier yang ada dalam data. Penyimpangan ini dapat berdampak signifikan terhadap keakuratan analisis dan harus ditangani dengan tepat. Terakhir, teknik augmentasi data mungkin bisa dilakukan

Underfitting dianggap sebagai antitesis dari *overfitting*. *Underfitting* diamati ketika model ML dicirikan oleh kesederhanaan yang berlebihan. Ada kebutuhan untuk memasukkan fungsionalitas tambahan khusus ke dalam model ini.

12.7.4 Machine Learning Melibatkan Proses yang Kompleks

ML adalah proses kompleks yang terdiri dari analisis data, pemrosesan awal data, pelatihan data, melakukan perhitungan matematis yang kompleks, dll.

11.7.5 Data pelatihan tidak mencukupi

Model ML perlu dilatih dengan data dalam jumlah besar untuk mendapatkan hasil yang akurat.

11.7.6 Kesalahan Dalam Sampel dan Kesalahan Keluar dari Sampel

Titik data yang digunakan dalam membangun model biasanya disebut sebagai data sampel. Istilah "kesalahan dalam sampel" mengacu pada kesalahan yang muncul saat mengevaluasi model ML pada kumpulan data yang sama yang digunakan untuk melatihnya. Terjadinya kesalahan dalam sampel biasanya disebut sebagai kesalahan penggantian. Data di luar sampel mengacu pada poin data yang baru dan tidak berkaitan dengan set data pelatihan mana pun. Konsep kesalahan di luar sampel mengacu pada perbedaan atau ketidakakuratan yang diamati saat menerapkan model prediktif pada kumpulan data baru atau

yang belum pernah dilihat sebelumnya. Fenomena ini biasanya disebut sebagai kesalahan generalisasi.

11.8 Langkah-Langkah Proses ML

ML melibatkan proses yang kompleks dalam membangun model, langkah-langkah berikut harus dilakukan:

1. Pengumpulan Data

Langkah awal dalam proyek ML kami memerlukan perolehan data yang relevan. Data dapat diperoleh dari banyak sumber, termasuk namun tidak terbatas pada file, sensor, database, dan lain-lain.

2. Pra-pemrosesan Data

Pemanfaatan data yang diperoleh dari berbagai sumber untuk pembuatan model ML terhambat oleh ketidaksesuaiannya untuk tujuan analisis langsung. Hal ini disebabkan oleh adanya sejumlah besar data yang berisik, teks yang tidak terorganisir, nilai yang hilang, nilai yang sangat besar, atau teks yang tidak relevan dalam kumpulan data.

Penghapusan data Noice diperlukan untuk mendapatkan dataset yang bebas dari informasi yang tidak diinginkan. Saat terlibat dalam pengembangan model ML, aturan 80/20 harus dipatuhi. Berdasarkan pedoman ini, disarankan agar sekitar 80% waktu yang dialokasikan digunakan untuk pra-pemrosesan data, sedangkan 20% sisanya harus digunakan untuk tahap analisis.

3. Menemukan Model Yang Terbaik Untuk Data

Identifikasi model ML yang menunjukkan kinerja optimal dalam mengatasi masalah yang ada. Topik yang menarik berkaitan dengan proses pelatihan dan pengujian. Penilaian model yang telah dikembangkan Persiapan data meliputi pengolahan data awal, sedangkan prediksi berkaitan dengan tahap evaluasi model ML yang dibangun. Ketimpangan Hoeffding juga dapat

digunakan untuk mewakili kelangsungan ML. Berdasarkan Pertidaksamaan Hoeffding, ketika mempertimbangkan ukuran sampel M dan tingkat toleransi, terdapat kemungkinan bahwa perbedaan antara estimasi dalam sampel (μ) dan hasil yang diprediksi (v) kurang dari konstanta tetap.

11.9 Implementasi *Machine Learning*

11.9.1 Asisten Pribadi Virtual

Asisten pribadi virtual ini memfasilitasi pengambilan informasi menggunakan perintah suara berdasarkan permintaan pengguna. Melalui pemanfaatan asisten pribadi virtual ini, seseorang memiliki kemampuan untuk menanyakan pertanyaan terkait. Asisten pribadi virtual memiliki kemampuan untuk mengumpulkan informasi, mengambil pertanyaan yang diajukan sebelumnya, dan mengakses data melalui aplikasi seluler. *Machine learning* diakui secara luas sebagai komponen fundamental asisten pribadi virtual, karena mereka menggunakan teknik pengumpulan dan penyempurnaan data untuk memberikan hasil yang disesuaikan tergantung pada preferensi pengguna dan informasi sebelumnya. Asisten pribadi virtual biasanya dimasukkan ke dalam berbagai platform, termasuk aplikasi seluler seperti Google Allo, smartphone seperti perangkat Samsung, dan smart speakers seperti Amazon Echo dan Google Home. Asisten pribadi virtual adalah perangkat yang ringkas, portabel dan mobile.

11.9.2 Analisis sentimen

Analisis sentimen, juga dikenal sebagai penambangan opini, adalah metode komputasi dalam bidang pemrosesan bahasa alami (NLP) yang digunakan untuk memastikan polaritas data, mengklasifikasikannya menjadi positif, negatif, atau netral. Analisis sentimen adalah proses menentukan emosi atau pendapat seseorang teks yang diberikan. Ini juga disebut sebagai penambangan opini atau klasifikasi sentimen. Penerapan

analisis sentimen terdapat pada aplikasi pengambilan keputusan, website berbasis review, dll

11.9.3 Rekomendasi Produk

Tindakan melakukan pembelian online di situs web tertentu telah mendorong kesadaran bahwa ada kebutuhan untuk memberikan rekomendasi yang berkaitan dengan keberlanjutan kepada klien. Aplikasi atau platform e-commerce memberikan rekomendasi yang dipersonalisasi berdasarkan preferensi pengguna. Rekomendasi produk ditentukan dengan menganalisis banyak faktor seperti pembelian sebelumnya, barang yang telah dilihat atau ditambahkan ke keranjang, dan preferensi merek individu. Untuk memfasilitasi rekomendasi pelanggan yang akurat, pengalaman berbelanja memerlukan pemanfaatan ML untuk analisis komprehensif.

11.9.4 Kesehatan

Penerapan *Machine learning* memiliki pengaruh yang signifikan terhadap perkembangan teknologi dalam domain kesehatan medis. Pemanfaatan sensor dan perangkat yang dapat dikenakan telah memungkinkan penyediaan wawasan seketika pada kondisi kesehatan komprehensif individu, yang mencakup parameter seperti detak jantung, tekanan darah, dan metrik relevan lainnya. Profesional medis dapat menggunakan data ini untuk menilai status kesehatan seseorang, mengidentifikasi tren melalui catatan pasien, dan membuat prediksi terkait potensi penyakit di masa mendatang. Dimungkinkan untuk melatih mesin untuk menunjukkan kecerdasan yang sebanding dengan manusia. Mesin memiliki kemampuan untuk meniru peran praktisi medis dengan memanfaatkan catatan kesehatan historis untuk mengantisipasi penyakit dan merekomendasikan obat yang tepat.

DAFTAR PUSTAKA

Alpaydın, E. 2020. *Introduction to Machine Learning*. 4th edn. Cambridge, Massachusetts: MIT Press.

Burger, S. V. 2018. *Introducing to machine learning with R*. First Edit. United States of America: O'Reilly Media, Inc.

Carbonell, J.G. *et al.* 2014. 'Machine Learning : A Historical and Methodological', (November).

Chatterjee, P. *et al.* 2023. *Machine Learning Algorithms and Applications in Engineering*, *Machine Learning Algorithms and Applications*. Boca Raton: CRC Press. Available at: https://doi.org/10.1201/9781003104858.

Chopra, D. and Khurana, R. 2023. *Introduction to Machine Learning with Python*, *Introduction to Machine Learning with Python*. Singapore: Bentham Science.

Cohen, S. 2020. *The evolution of machine learning: past, present, and future*, *Artificial Intelligence and Deep Learning in Pathology*. Elsevier Inc. Available at: https://doi.org/10.1016/B978-0-323-67538-3.00001-4.

Cohen, S. 2021. 'The evolution of machine learning: past, present, and future', in *Artificial Intelligence and Deep Learning in Pathology*. Elsevier, pp. 1–12. Available at: https://doi.org/10.1016/B978-0-323-67538-3.00001-4.

Foote, K.D. 2022) 'The History of Machine Learning and Its Convergent Trajectory Towards AI', in *Machine Learning and the City*. Wiley, pp. 129–142. Available at: https://doi.org/10.1002/9781119815075.ch9.

Fradkov, A.L. 2020. 'Early History of Machine Learning', *IFAC-PapersOnLine*, 53(2), pp. 1385–1390. Available at: https://doi.org/10.1016/j.ifacol.2020.12.1888.

Kubat, M. 2021. *An Introduction to Machine Learning*, *An Introduction to Machine Learning*. Available at: https://doi.org/10.1007/978-3-030-81935-4.

Nedal, D. and Morgan, P. 2018. *Introduction To Machine Learning*

With Python. AI Sciences LLC. Available at: https://medium.com/@arifwicaksanaa/pengertian-use-case-a7e576e1b6bf.

Simeone, O. 2018. 'A brief introduction to machine learning for engineers', *Foundations and Trends in Signal Processing*, 12(3–4), pp. 200–431. Available at: https://doi.org/10.1561/2000000102.

Yang, X.-S. 2019. *Introduction to Algorithms for Data Mining and Machine Learning*. Candice Janco.

BAB 12
Penalaran Berbasis Kasus (Teori, Algoritma dan Implementasinya)

Oleh Irmawati

12.1 Pengertian Penalaran Berbasis Kasus

Penalaran berbasis kasus merupakan suatu pendekatan dalam kecerdasan buatan yang menggunakan pengetahuan yang tersimpan dalam kasus-kasus yang telah terjadi sebelumnya untuk menyelesaikan masalah baru (Main, Dillon and Shiu, 2001). Dalam pendekatan ini, sistem mengidentifikasi masalah yang mirip dengan kasus-kasus yang ada dan menerapkan pengetahuan dari kasus tersebut untuk menghasilkan solusi atau rekomendasi untuk masalah baru. Pendekatan ini mirip dengan cara manusia belajar dari pengalaman dan menerapkannya dalam situasi yang serupa. Penalaran Berbasis Kasus sering digunakan dalam berbagai aplikasi seperti sistem pakar, diagnosa medis, manajemen pengetahuan, dan lainnya.

Komponen utama dalam penalaran berbasis kasus terdiri dari beberapa elemen yang berperan penting dalam memungkinkan sistem penalaran berbasis kasus untuk bekerja secara efektif. Berikut komponen-komponen utama dalam penalaran berbasis kasus:

1. Basis Kasus: Basis kasus adalah penyimpanan sentral dari pengalaman atau kasus-kasus yang telah terjadi sebelumnya. Setiap kasus dalam basis kasus mencakup deskripsi masalah, solusi yang diberikan, serta konteks atau informasi

tambahan yang relevan. Basis kasus ini merupakan sumber utama pengetahuan dalam sistem penalaran berbasis kasus.

2. Mesin Inferensi: Mesin inferensi adalah komponen yang bertanggung jawab untuk mengevaluasi masalah baru dengan mencari dan memilih kasus-kasus yang relevan dari basis kasus. Setelah kasus-kasus tersebut ditemukan, mesin inferensi menerapkan pengetahuan dari kasus-kasus tersebut ke masalah yang sedang dihadapi untuk menghasilkan solusi atau rekomendasi. Mesin inferensi ini dapat menggunakan berbagai metode seperti pencocokan pola, penalaran analogi, atau teknik lainnya.
3. Siklus Berulang: Siklus berulang adalah proses yang mengatur bagaimana pengetahuan baru diperoleh dari pengalaman dan dimasukkan kembali ke basis kasus untuk digunakan dalam penalaran di masa depan. Siklus ini terdiri dari empat tahap:
    - Retrieval: Sistem mengidentifikasi kasus-kasus yang relevan dari basis kasus.
    - Reuse: Pengetahuan dari kasus-kasus tersebut diterapkan ke masalah baru.
    - Revise: Jika diperlukan, solusi yang diberikan dapat disesuaikan dengan masalah yang sedang dihadapi.
    - Retain: Kasus baru yang dihasilkan dari solusi yang direvisi dapat dimasukkan kembali ke basis kasus.
4. Antarmuka Pengguna (User Interface): Antarmuka pengguna memungkinkan interaksi antara sistem penalaran berbasis kasus dan pengguna manusia. Ini mencakup penggunaan sistem untuk memasukkan masalah baru, melihat rekomendasi yang dihasilkan, dan mengkonfirmasi atau memvalidasi solusi yang diberikan oleh sistem. Antarmuka pengguna harus dirancang untuk membuat interaksi dengan sistem menjadi intuitif dan efisien.

Berikut adalah beberapa manfaat dari penalaran berbasis kasus:

1. Penyelesaian Masalah Berbasis Pengalaman: Dengan menggunakan kasus-kasus yang telah terjadi sebelumnya, sistem penalaran berbasis kasus dapat memecahkan masalah baru dengan cara yang lebih efisien dan akurat karena ia mengambil inspirasi dari pengalaman sebelumnya.
2. Fleksibilitas: penalaran berbasis kasus memiliki kemampuan untuk menangani berbagai jenis masalah dan tugas, karena sistem ini tidak terbatas pada algoritma atau aturan tertentu. Ini membuatnya cocok untuk berbagai bidang, seperti pengenalan pola, diagnosis medis, perencanaan tugas, dan lainnya.
3. Pemeliharaan Pengetahuan: penalaran berbasis kasus memungkinkan sistem untuk terus memperbarui dan memelihara pengetahuan dengan mudah. Ketika kasus baru muncul atau informasi baru tersedia, sistem dapat memperbarui basis pengetahuannya.
4. Pelajaran dari Kesalahan: Sistem penalaran berbasis kasus dapat mempelajari dari kesalahan yang terjadi dalam kasus sebelumnya. Hal ini memungkinkan untuk menghindari kesalahan yang sama dalam kasus yang serupa di masa depan.
5. Skalabilitas: Dalam kasus peningkatan jumlah data, sistem penalaran berbasis kasus dapat dengan mudah diperluas untuk mengakomodasi kasus-kasus baru tanpa perlu mengubah struktur fundamentalnya.
6. Interpretabilitas: Sistem penalaran berbasis kasus dapat memberikan alasan yang jelas dan dapat dipahami mengenai mengapa suatu keputusan diambil. Ini memungkinkan pengguna atau profesional untuk memahami dan mempercayai keputusan yang dihasilkan oleh sistem.
7. Adaptasi terhadap Perubahan Konteks: penalaran berbasis kasus dapat menyesuaikan solusi berdasarkan perubahan dalam konteks atau lingkungan, karena ia

mempertimbangkan kasus-kasus yang relevan dalam situasi tertentu.

8. Penghematan Waktu dan Sumber Daya: Sistem penalaran berbasis kasus dapat menghemat waktu dan sumber daya dengan menghindari pengembangan solusi dari awal untuk setiap masalah baru, karena ia dapat menggunakan pengetahuan yang sudah ada.

12.2 Proses Penalaran Berbasis Kasus

Proses penalaran berbasis kasus merupakan paradigma dalam kecerdasan buatan dengan inspirasi dari cara manusia belajar berdasarkan pengalaman sebelumnya. Dalam penalaran berbasis kasus, sistem komputer atau agen belajar dan mengambil keputusan dengan merujuk pada kasus-kasus yang telah ada sebelumnya. Proses ini terdiri dari beberapa tahapan kunci. Pertama, sistem mencari dan mengambil kasus yang relevan dari basis kasus berdasarkan kesamaan dengan masalah saat ini. Kemudian, sistem memahami kasus-kasus tersebut untuk mengidentifikasi informasi penting dan solusi yang mungkin relevan. Setelah pemahaman, sistem menyesuaikan solusi dari kasus-kasus tersebut agar sesuai dengan masalah yang sedang dihadapi. Terakhir, solusi yang disesuaikan dievaluasi untuk menentukan kecocokan dan efektivitasnya.

Proses penalaran berbasis kasus telah diterapkan dalam berbagai bidang, termasuk sistem pakar, bantuan keputusan, perawatan kesehatan, dan manufaktur. Hal ini memungkinkan penggunaan pengalaman masa lalu untuk memecahkan masalah yang kompleks dan dapat menghemat waktu dan sumber daya.

Tahapan-tahapan dalam proses penalaran merupakan langkah-langkah yang sistematis dalam pemikiran kritis dan analitis. Membantu individu untuk memecahkan masalah, membuat keputusan yang informasional, dan mengembangkan argumen

yang kuat. Berikut adalah tahapan-tahapan dalam proses penalaran, disertai dengan contoh kasus:

1. Identifikasi Masalah:
   Mengidentifikasi masalah atau pertanyaan yang perlu dipecahkan. Melibatkan pemahaman yang jelas tentang masalah yang dihadapi.
   Contoh Kasus:
   Seorang manajer proyek mengidentifikasi bahwa proyeknya akan melebihi anggaran dan harus mencari cara untuk mengatasi masalah ini.
2. Kumpulkan Informasi:
   Setelah masalah diidentifikasi, langkah berikutnya adalah mengumpulkan informasi yang relevan tentang masalah tersebut.
   Contoh Kasus:
   Seorang siswa harus melakukan penelitian tentang efek perubahan iklim untuk mengumpulkan data dan fakta yang relevan.
3. Evaluasi Informasi:
   Tahap ini melibatkan penilaian terhadap kebenaran, keandalan, dan relevansi informasi yang telah dikumpulkan.
   Contoh Kasus:
   Seorang jurnalis harus memeriksa sumber-sumber berita untuk memastikan kebenaran informasi yang akan ditulis dalam artikelnya.
4. Pengembangan Pendekatan atau Argumen:
   Berdasarkan analisis informasi, langkah selanjutnya adalah mengembangkan pendekatan atau argumen yang logis dan terstruktur untuk menanggapi masalah.
   Contoh Kasus:
   Seorang pengacara harus merumuskan argumen yang kuat untuk membela kliennya dalam persidangan.

5. Evaluasi Argumen atau Pendekatan:
   Setelah pembuatan argumen atau pendekatan, tahap ini melibatkan penilaian kritis terhadap kekuatan dan kelemahan dari argumen yang dibangun.
   Contoh Kasus:
   Seorang guru harus mengevaluasi argumen siswanya dalam sebuah debat untuk memberikan penilaian yang adil.
6. Pengambilan Keputusan:
   Pada tahap ini, individu harus mengambil keputusan berdasarkan argumen dan analisis yang telah mereka lakukan selama proses penalaran.
   Contoh Kasus:
   Seorang manajer harus memutuskan apakah akan mempekerjakan seorang pelamar berdasarkan analisis berbagai kualifikasi dan pengalaman pelamar.
7. Implementasi dan Tindak Lanjut:
   Setelah keputusan diambil, tahap terakhir adalah mengimplementasikan keputusan tersebut dan melakukan tindak lanjut sesuai dengan solusi atau keputusan yang telah diambil.
   Contoh Kasus:
   Seorang pemimpin proyek harus mengkoordinasikan tindakan timnya sesuai dengan rencana yang telah dibuat untuk menyelesaikan proyek.

Contoh kasus-kasus di atas memberikan gambaran konkret tentang bagaimana tahapan-tahapan dalam proses penalaran dapat diterapkan dalam berbagai konteks. Proses ini membantu individu untuk berpikir secara sistematis, menghindari kesalahan pemikiran, dan menghasilkan keputusan yang lebih baik.

12.2.1 Pengguna dalam Proses Penalaran Berbasis Kasus

Dalam konteks penalaran berbasis kasus, "pengguna" merujuk kepada individu atau entitas yang menggunakan metode ini untuk memecahkan masalah atau mengambil keputusan. Pengguna adalah orang atau kelompok yang menghadapi suatu masalah atau situasi tertentu dan menggunakan pengetahuan dan pengalaman dari kasus-kasus yang telah ada untuk membantu mereka dalam proses pengambilan keputusan atau pemecahan masalah.

Peran pengguna dalam penalaran berbasis kasus adalah sebagai subjek yang aktif dalam seluruh proses. Mereka memiliki tanggung jawab untuk mengidentifikasi masalah yang mereka hadapi, mencari kasus-kasus yang relevan, menganalisis kasus-kasus tersebut, menyesuaikan solusi dari kasus-kasus tersebut dengan situasi mereka, mengambil keputusan, dan mengevaluasi hasilnya (Recio-García, González-Calero and Díaz-Agudo, 2014).

Pengguna dapat berupa individu, seperti seorang profesional di bidang tertentu yang menggunakan penalaran berbasis kasus untuk mengatasi masalah pekerjaan mereka, atau kelompok, seperti tim yang menggunakan metode ini untuk pengambilan keputusan kelompok (López, 2022). Pengguna juga bisa merupakan sistem komputer atau entitas otomatis lainnya yang dirancang untuk mengambil keputusan berdasarkan penalaran berbasis kasus.

Jadi, secara umum, pengguna dalam penalaran berbasis kasus adalah pihak yang mengaplikasikan prinsip dan pengetahuan dari kasus-kasus sebelumnya untuk memecahkan masalah atau membuat keputusan yang lebih baik dalam situasi yang sedang dihadapi.

Peran pengguna dalam penalaran berbasis kasus sangat penting dan mencakup sejumlah aspek yang signifikan:

1. Identifikasi Masalah: Pengguna harus mengidentifikasi masalah atau situasi yang memerlukan penalaran berbasis kasus. Hal ini memerlukan kemampuan untuk mengenali kesamaan antara masalah yang sedang dihadapi dan kasus-kasus yang telah ada sebelumnya.
2. Pencarian Kasus yang Relevan: Pengguna perlu mencari kasus-kasus yang relevan dalam basis data atau sumber pengetahuan yang ada. Mereka harus dapat menentukan kasus-kasus yang memiliki karakteristik serupa dengan masalah saat ini.
3. Analisis Kasus: Pengguna harus menganalisis kasus-kasus yang telah diidentifikasi. Dengan melibatkan pemahaman mendalam tentang bagaimana masalah dipecahkan atau keputusan apa yang diambil dalam kasus-kasus tersebut. Analisis ini membantu pengguna memahami konteks dan faktor-faktor kunci yang mempengaruhi hasil.
4. Penyesuaian Kasus: Setelah menganalisis kasus, pengguna perlu menyesuaikan solusi atau keputusan dari kasus-kasus tersebut agar sesuai dengan situasi yang sedang dihadapi. Hal ini mencakup penyesuaian strategi, parameter, atau langkah-langkah yang digunakan dalam kasus sebelumnya.
5. Pengambilan Keputusan: Pengguna berperan dalam pengambilan keputusan akhir berdasarkan pengetahuan yang mereka peroleh dari kasus-kasus sebelumnya. Mereka harus dapat menerapkan pengetahuan ini untuk merancang solusi atau keputusan yang paling sesuai dengan situasi mereka.
6. Evaluasi Hasil: Setelah tindakan diambil, pengguna harus terus-menerus memantau dan mengevaluasi hasilnya. Jika hasilnya tidak sesuai dengan yang diharapkan, pengguna perlu mengubah tindakan mereka berdasarkan pengalaman yang baru saja mereka dapatkan.

7. Penambahan Pengetahuan: Pengguna juga dapat berperan dalam menambahkan pengetahuan baru ke dalam basis data kasus. Jika mereka menemukan solusi yang efektif dalam situasi unik, mereka dapat menyimpan kasus tersebut sebagai referensi untuk penggunaan masa depan, yang akan bermanfaat bagi orang lain yang menghadapi masalah serupa.
8. Proses Pembelajaran: Penalaran berbasis kasus juga merupakan alat pembelajaran yang kuat. Pengguna dapat terus belajar dari pengalaman mereka sendiri dan dari pengalaman orang lain yang terdokumentasi dalam basis data kasus. Hal ini membantu meningkatkan pemahaman mereka tentang berbagai solusi yang mungkin untuk masalah tertentu.

Dengan demikian, peran pengguna dalam penalaran berbasis kasus melibatkan identifikasi, analisis, aplikasi, dan evaluasi pengetahuan dari kasus-kasus yang ada untuk memecahkan masalah atau membuat keputusan yang lebih baik. Mereka berfungsi sebagai pengambil keputusan yang bijak yang dapat memanfaatkan pengetahuan yang tersedia untuk merancang solusi yang efektif dalam berbagai konteks.

12.3 Algoritma Inference

Algoritma Inference adalah suatu proses komputasional atau metode yang digunakan dalam bidang kecerdasan buatan untuk mencapai kesimpulan atau mengambil keputusan berdasarkan informasi yang tersedia (Shiu and Pal, 2004). Dengan melibatkan penggunaan aturan logika, pengetahuan, dan data yang ada untuk melakukan penalaran atau inferensi guna mencapai solusi atau jawaban yang sesuai dalam suatu masalah. Algoritma Inference dapat digunakan dalam berbagai konteks, termasuk dalam sistem pakar, mesin pencarian, sistem rekomendasi, dan banyak lagi.

Berikut adalah beberapa konsep dasar yang terkait dengan Algoritma Inference:

1. Pengetahuan: Algoritma Inference memanfaatkan pengetahuan yang tersedia, baik dalam bentuk aturan logika, basis data, atau pengetahuan yang diambil dari pengalaman sebelumnya, untuk membuat penilaian atau mengambil keputusan. Pengetahuan ini dapat berupa fakta, aturan, atau informasi yang relevan dengan masalah yang sedang dihadapi.
2. Aturan Logika: Algoritma Inference sering kali menggunakan aturan logika, seperti aturan inferensi, modus ponens, dan modus tollens, untuk melakukan penalaran. Aturan-aturan ini membantu dalam mengekstrak kesimpulan berdasarkan premis atau fakta yang ada.
3. Representasi Pengetahuan: Informasi dan pengetahuan harus direpresentasikan dalam format yang dapat dipahami oleh algoritma. Bisa dalam bentuk struktur data seperti graf, pohon keputusan, atau bahkan dalam bentuk basis data.
4. Proses Penalaran: Algoritma Inference melakukan penalaran dengan menghubungkan premis atau fakta-fakta yang ada dengan aturan-aturan yang relevan. Melibatkan penyelidikan berbagai jalur atau hipotesis untuk mencapai kesimpulan yang paling sesuai.
5. Kepastian dan Ketidakpastian: Beberapa masalah dapat memiliki ketidakpastian terkait dengan data atau informasi yang tersedia. Algoritma Inference dapat mengatasi ketidakpastian ini dengan menggunakan teknik seperti logika fuzzy atau probabilistik untuk menghasilkan solusi yang lebih tepat.
6. Penilaian Kualitas Solusi: Setelah melakukan penalaran, algoritma akan mengevaluasi kesimpulan atau solusi yang

dihasilkan. Evaluasi ini dapat melibatkan kriteria-kriteria tertentu, seperti akurasi, efisiensi, atau relevansi.

7. Pengambilan Keputusan: Akhirnya, algoritma Inference akan mengambil keputusan atau menghasilkan output berdasarkan hasil penalaran dan evaluasi yang telah dilakukan. Output ini dapat berupa rekomendasi, tindakan konkret, atau jawaban atas masalah yang diberikan.

Algoritma Inference sangat bervariasi tergantung pada konteks aplikasinya. Misalnya, dalam sistem pakar, algoritma Inference akan menggunakan pengetahuan para ahli untuk membantu dalam diagnosis masalah. Dalam mesin pencarian, algoritma Inference akan menentukan relevansi dokumen terhadap kueri pengguna. Dalam semua kasus, tujuan utamanya adalah menggunakan informasi yang tersedia untuk mencapai solusi yang tepat atau keputusan yang bijak.

12.3.1 Algoritma – Algoritma dalam Penalaran Berbasis Kasus

Algoritma dalam penalaran berbasis kasus merupakan serangkaian langkah logis yang digunakan untuk memproses informasi dan pengetahuan yang terkait dengan kasus tertentu.

Berikut adalah beberapa algoritma populer yang digunakan dalam penalaran berbasis kasus:

1. CBR (Case-Based Reasoning):
   CBR adalah pendekatan utama dalam penalaran berbasis kasus. Melibatkan penentuan solusi untuk masalah saat ini dengan mencari kasus yang mirip yang telah dipecahkan sebelumnya dan menerapkan solusi dari kasus tersebut.
2. K-NN (K-Nearest Neighbors):
   K-NN adalah algoritma pembelajaran mesin yang sering digunakan dalam penalaran berbasis kasus. Ini mencari kasus terdekat dalam dataset berdasarkan ukuran jarak (misalnya, jarak Euclidean) dan mengambil solusi dari kasus terdekat tersebut.

3. RBR (Rule-Based Reasoning):
   Penalaran berbasis aturan melibatkan penggunaan aturan-aturan yang ditentukan sebelumnya untuk menghasilkan solusi berdasarkan situasi saat ini. Aturan ini menghubungkan masalah dengan tindakan yang harus diambil.
4. Genetic Algorithms (GA):
   Algoritma genetik dapat digunakan dalam penalaran berbasis kasus untuk mencari solusi optimal dengan evolusi populasi kasus melalui proses seleksi, crossover, dan mutasi.
5. *Fuzzy Logic*:
   Fuzzy logic digunakan untuk mengatasi ketidakpastian dalam penalaran berbasis kasus. Hal ini memungkinkan kasus memiliki tingkat keanggotaan dalam berbagai kategori, sehingga dapat memberikan solusi yang lebih fleksibel.
6. *Bayesian Networks*:
   Jaringan Bayesian adalah model probabilistik yang digunakan dalam penalaran berbasis kasus untuk menggambarkan hubungan probabilistik antara masalah dan solusi berdasarkan informasi statistik.
7. *Analogical Reasoning*:
   Analogical reasoning melibatkan pencarian analogi antara masalah saat ini dengan kasus-kasus yang telah dipecahkan sebelumnya. Digunakan untuk menemukan pola-pola analogi dan menerapkan solusi yang serupa.
8. *Reinforcement Learning*:
   Dalam konteks penalaran berbasis kasus, *reinforcement learning digunakan* untuk memutuskan tindakan yang akan diambil berdasarkan hasil-hasil dari kasus yang telah dipecahkan sebelumnya.
9. *Explanatory Reasoning*:
   Penalaran penjelasan (*explanatory reasoning*) digunakan untuk memahami mengapa suatu solusi dari kasus yang ada

cocok atau tidak cocok untuk masalah saat ini, membantu dalam pengambilan keputusan yang lebih baik.

Setiap algoritma diatas memiliki karakteristik dan kegunaan yang berbeda, untuk pemilihan tergantung pada jenis masalah, data yang tersedia, dan preferensi pengembang. Penalaran berbasis kasus adalah pendekatan yang kuat dalam menghadapi masalah yang kompleks dan berkaitan dengan pengalaman dan pengetahuan yang terakumulasi dari waktu ke waktu.

12.3.2 Algoritma Pencarian

Algoritma pencarian adalah konsep yang mendasar dalam ilmu komputer dan kecerdasan buatan. Metode atau prosedur yang digunakan untuk menemukan elemen tertentu dalam kumpulan data atau ruang informasi yang lebih besar. Algoritma pencarian memiliki berbagai aplikasi dalam berbagai bidang, termasuk pemrosesan teks, pengembangan perangkat lunak, pengoptimalan, dan bahkan kecerdasan buatan.

Algoritma pencarian memiliki tujuan dasar: menemukan elemen tertentu dalam kumpulan data atau ruang informasi. Mirip dengan mencari "jarum dalam tumpukan jerami." Beberapa elemen kunci dalam konsep dasar algoritma pencarian adalah (Russell, 2010):

1. Ruang Pencarian: Ruang di mana algoritma mencari eleen yang dicari. Bisa berupa kumpulan data terstruktur seperti array, daftar, atau grafik, atau bahkan ruang informasi yang lebih abstrak seperti internet.
2. Elemen yang Dicari: Elemen yang ingin ditemukan dalam ruang pencarian. Bisa berupa angka, teks, objek, atau bahkan aturan tertentu dalam konteks yang lebih kompleks.
3. Solusi: Solusi adalah hasil dari algoritma pencarian yang menunjukkan apakah elemen yang dicari ditemukan atau tidak. Jika ditemukan, solusi biasanya juga mencakup posisi atau informasi tambahan tentang elemen tersebut.

4. Kriteria Pencarian: Kriteria pencarian adalah aturan atau persyaratan yang harus dipenuhi oleh elemen yang dicari. Misalnya, mencari angka terkecil dalam daftar atau kata tertentu dalam dokumen teks.

Ada beberapa jenis algoritma pencarian yang digunakan dalam berbagai konteks. Beberapa yang paling umum meliputi:

1. Pencarian Linear: Metode pencarian sederhana dimana algoritma memeriksa setiap elemen dalam ruang pencarian satu per satu hingga elemen yang dicari ditemukan atau seluruh ruang pencarian dijelajahi. Sangat efektif untuk kumpulan data kecil, tetapi menjadi tidak efisien untuk kumpulan data besar.
2. Pencarian Biner: Pencarian biner digunakan terutama pada kumpulan data terurut. Algoritma ini membagi ruang pencarian menjadi dua bagian pada setiap langkah, menghilangkan setengahnya pada setiap iterasi. Hal ini menghasilkan pencarian yang jauh lebih efisien daripada pencarian linear.
3. Pencarian Graf: Dalam konteks ini, algoritma mencari jalur atau rute dalam grafik dari satu titik ke titik lain. Algoritma seperti algoritma Dijkstra atau A* sering digunakan dalam pencarian graf.
4. Pencarian Teks: Algoritma pencarian teks digunakan untuk mencari pola atau kata dalam teks. Algoritma KMP (Knuth-Morris-Pratt) dan algoritma Boyer-Moore adalah contoh algoritma pencarian teks yang efisien.

Algoritma pencarian memiliki banyak aplikasi di berbagai bidang. Beberapa contoh aplikasi termasuk:

1. Mesin Pencari Web: Mesin pencari seperti Google menggunakan algoritma pencarian kompleks untuk menemukan dan menampilkan hasil yang relevan dari miliaran halaman web.

2. Sistem Pemrosesan Bahasa Alami: Dalam pemrosesan teks dan bahasa alami, algoritma pencarian digunakan untuk menemukan kata kunci, entitas, atau struktur bahasa tertentu dalam teks.
3. Rekomendasi Produk: Dalam e-commerce, algoritma pencarian digunakan untuk merekomendasikan produk kepada pengguna berdasarkan preferensi dan riwayat pembelian mereka.
4. Permainan Komputer: Dalam permainan video, algoritma pencarian digunakan untuk mengatur perilaku entitas dalam permainan dan mencari jalan terbaik untuk mencapai tujuan.
5. Pemecahan Masalah Optimasi: Algoritma pencarian digunakan untuk mencari solusi optimal dalam berbagai masalah optimasi, seperti rancangan jaringan, perencanaan rute, dan banyak lagi.

Dalam dunia yang semakin terhubung secara digital, algoritma pencarian tetap menjadi bagian integral dari teknologi dan pengembangan perangkat lunak. Kemampuan untuk menavigasi labirin informasi dengan efisien adalah kunci dalam memproses, mengakses, dan memahami data yang semakin kompleks dalam era informasi saat ini.

12.3.3 Teknik Pencarian dalam Penalaran Berbasis Kasus

Salah satu komponen kunci dari penalaran berbasis kasus adalah teknik pencarian, yang memungkinkan sistem untuk menemukan kasus yang relevan dalam basis pengetahuan.

Konsep dasar teknik pencarian dalam penalaran berbasis kasus mencakup langkah-langkah berikut:

1. Pencarian Kasus: Tahap pertama adalah mencari kasus yang relevan dalam basis pengetahuan. Dengan melibatkan perbandingan karakteristik masalah saat ini dengan karakteristik kasus-kasus yang ada. Teknik pencarian harus

dapat mengidentifikasi kasus-kasus yang mirip secara signifikan dengan masalah saat ini.

2. Pemilihan Kasus: Setelah sejumlah kasus yang potensial ditemukan, sistem perlu memilih kasus yang paling relevan untuk dijadikan dasar penalaran. Melibatkan peringkat atau penilaian kasus-kasus berdasarkan kemiripan dan relevansinya dengan masalah saat ini.
3. Adaptasi: Setelah kasus yang sesuai dipilih, sistem harus mampu mengadaptasi solusi dari kasus tersebut untuk memecahkan masalah saat ini. Hal ini sering melibatkan modifikasi solusi berdasarkan konteks atau persyaratan yang berbeda.
4. Evaluasi: Solusi yang dihasilkan kemudian dievaluasi untuk memastikan bahwa mereka memenuhi tujuan dan persyaratan masalah saat ini. Jika solusi tersebut tidak memadai, sistem dapat kembali mencari atau memodifikasi kasus-kasus yang digunakan.

Ada beberapa teknik pencarian yang dapat digunakan dalam penalaran berbasis kasus, tergantung pada kompleksitas masalah dan struktur basis pengetahuan. Beberapa di antaranya adalah:

1. Pencarian Berbasis Indeks: Dalam teknik ini, basis pengetahuan diberi indeks berdasarkan karakteristik atau atribut kasus. Memungkinkan pencarian yang cepat berdasarkan kriteria tertentu.
2. Pencarian Berbasis Cluster: Pencarian ini mengelompokkan kasus-kasus yang mirip ke dalam cluster atau kelompok. Ketika ada masalah baru, pencarian dilakukan pada cluster yang paling relevan.
3. Pencarian Berbasis Propagasi: Dalam pendekatan ini, solusi dari kasus yang mirip disesuaikan dengan masalah saat ini dengan menggunakan teknik propagasi pengetahuan.

4. Pencarian Berbasis Seleksi: Sistem memilih sejumlah kasus yang mirip, dan kemudian solusi dari kasus-kasus tersebut digabungkan atau diubah sesuai dengan karakteristik masalah saat ini.

Penalaran berbasis kasus dengan teknik pencarian yang efisien memiliki beragam aplikasi dalam berbagai bidang, termasuk:

1. Diagnostik Medis: Dalam dunia kedokteran, penalaran berbasis kasus digunakan untuk membantu dokter dalam mendiagnosis penyakit berdasarkan gejala pasien dan catatan medis sebelumnya.
2. Pemecahan Masalah Teknis: Dalam perawatan peralatan teknis, teknisi dapat menggunakan penalaran berbasis kasus untuk menemukan solusi untuk masalah yang muncul dengan merujuk ke kasus-kasus serupa.
3. Rekomendasi Produk: Dalam e-commerce, penalaran berbasis kasus digunakan untuk merekomendasikan produk kepada pelanggan berdasarkan sejarah pembelian mereka dan preferensi lainnya.
4. Manajemen Pengetahuan: Penalaran berbasis kasus dapat digunakan untuk menyimpan dan memanfaatkan pengetahuan yang ada dalam organisasi, sehingga memungkinkan karyawan untuk mengambil keputusan yang lebih baik.
5. Pengaturan Perjalanan: Dalam aplikasi perjalanan, penalaran berbasis kasus membantu dalam merencanakan rute perjalanan dan penginapan berdasarkan preferensi pengguna dan pengalaman pengguna sebelumnya.

Teknik pencarian dalam penalaran berbasis kasus adalah salah satu komponen kunci yang memungkinkan sistem untuk memanfaatkan pengetahuan masa lalu untuk memecahkan masalah saat ini. Dengan penggunaan yang cermat, penalaran berbasis kasus dapat meningkatkan efisiensi dan efektivitas dalam berbagai aspek kehidupan dan bisnis.

12.4 Algoritma Inference

Dalam konteks ini, "kasus" mengacu pada rangkaian informasi yang berkaitan dengan situasi atau masalah tertentu. Strategi pemilihan kasus adalah langkah penting dalam proses penalaran berbasis kasus, di mana kasus yang paling relevan dan mirip harus dipilih untuk digunakan sebagai pedoman dalam menghadapi situasi saat ini. Berikut ini beberapa strategi penting dalam pemilihan kasus :

1. Similarity-Based Selection

Salah satu strategi utama dalam pemilihan kasus adalah berdasarkan tingkat kesamaan antara kasus yang ada dengan situasi yang sedang dihadapi. Kasus yang paling mirip dengan situasi saat ini cenderung memberikan panduan yang lebih relevan. Hal ini dapat dicapai dengan mengukur kesamaan antara atribut atau karakteristik kasus dengan situasi yang ada.

Contoh: Dalam sistem diagnosis medis berbasis kasus, jika pasien saat ini mengalami gejala yang mirip dengan kasus pasien sebelumnya yang sudah terdiagnosis, maka kasus yang mirip tersebut dapat menjadi pedoman untuk diagnosis saat ini.

2. Relevance-Based Selection

Penting untuk mempertimbangkan relevansi informasi dalam kasus saat memilihnya. Kasus yang relevan adalah yang mengandung informasi yang paling relevan dengan situasi saat ini. Faktor-faktor seperti konteks, waktu, dan kepentingan relatif dari informasi dalam kasus harus dipertimbangkan.

Contoh: Dalam sistem rekomendasi produk e-commerce, pemilihan kasus dapat didasarkan pada preferensi dan riwayat pembelian pengguna yang paling relevan dengan produk yang sedang dilihat oleh pengguna saat ini.

3. Weighted Selection

Dalam beberapa kasus, atribut atau karakteristik tertentu dalam kasus dapat dianggap lebih penting daripada yang lain. Dalam strategi pemilihan berbasis bobot, kasus diberi bobot berdasarkan pentingnya atribut atau karakteristik tersebut. Kasus dengan bobot tertinggi akan lebih mungkin dipilih.

Contoh: Dalam sistem peramalan cuaca, atribut seperti tekanan udara dan suhu mungkin memiliki bobot yang lebih tinggi daripada atribut lain seperti kecepatan angin, karena mereka memiliki dampak yang lebih besar pada prediksi cuaca.

4. Learning-Based Selection

Dalam beberapa sistem berbasis kasus, algoritma pembelajaran mesin dapat digunakan untuk mempelajari pola pemilihan kasus yang efektif berdasarkan sejarah penggunaan kasus sebelumnya. Algoritma ini dapat mengadaptasi pemilihan kasus seiring waktu berdasarkan pengalaman yang diperoleh.

Contoh: Dalam asisten virtual, seperti ChatGPT, algoritma pembelajaran mesin dapat mempelajari kasus yang paling relevan berdasarkan interaksi pengguna sebelumnya untuk memberikan jawaban yang lebih baik.

12.5 Perkembangan Penalaran Berbasis Kasus

Dalam perkembangannya, ada beberapa hal terbaru yang telah muncul dalam penalaran berbasis kasus yang patut diperhatikan.

1. Integrasi dengan Teknologi Pemrosesan Bahasa Alami (Natural Language Processing, NLP): Salah satu perkembangan signifikan dalam penalaran berbasis kasus adalah integrasinya dengan teknologi NLP. Sistem penalaran berbasis kasus menggunakan kemampuan NLP untuk mengambil informasi dari teks, dokumen, atau sumber daya berbasis bahasa, yang memungkinkan penggunaan kasus-kasus yang lebih kaya dalam pengambilan keputusan.

Contoh: Dalam sistem pelayanan pelanggan, CBR dapat mengintegrasikan NLP untuk menganalisis keluhan pelanggan dalam bentuk teks dan memberikan solusi yang lebih baik berdasarkan permasalahan yang diungkapkan oleh pelanggan.

2. Penerapan dalam Bidang Kesehatan dengan Data Medis yang Lebih Kompleks: Penalaran berbasis kasus terus berkembang dalam domain kesehatan. Dengan perkembangan teknologi medis, data medis menjadi lebih kompleks. Penalaran berbasis kasus digunakan untuk membantu dokter dalam mendiagnosis penyakit, merencanakan perawatan, dan mengambil keputusan klinis yang lebih baik.
   Contoh: Sistem penalaran berbasis kasus dapat menganalisis data radiologi seperti gambar MRI atau CT scan untuk membantu dalam mendiagnosis penyakit tertentu dengan tingkat akurasi yang tinggi (Bichindaritz and Marling, 2006).

3. Penggunaan penalaran berbasis kasus dalam Sistem Kendaraan Otonom: Penalaran berbasis kasus juga berkembang dalam konteks kendaraan otonom. Sistem ini dapat memanfaatkan data dari kendaraan otonom lainnya untuk membuat keputusan yang lebih aman dan efisien di jalan raya.
   Contoh: Jika kendaraan otonom menghadapi situasi yang belum pernah dialaminya sebelumnya, penalaran berbasis kasus dapat merujuk pada pengalaman kendaraan otonom lain yang telah mengatasi situasi serupa untuk membuat keputusan yang tepat.

4. Pengembangan Algoritma Pembelajaran Mesin dalam penalaran berbasis kasus: Perkembangan terbaru dalam algoritma pembelajaran mesin seperti *deep learning* telah

diintegrasikan dengan penalaran berbasis kasus. Memungkinkan sistem penalaran berbasis kasus untuk memahami dan mengambil manfaat dari data dengan lebih baik, meningkatkan kualitas rekomendasi dan keputusan yang dihasilkan.

Contoh: Sistem penalaran berbasis kasus dapat menggunakan *deep learning* untuk mengekstrak fitur-fitur penting dari kasus-kasus yang ada, memungkinkan identifikasi pola yang lebih kompleks dalam pengambilan keputusan.

5. Penerapan dalam Bisnis dan Pengambilan Keputusan: Penalaran berbasis kasus semakin banyak digunakan dalam konteks bisnis untuk mengoptimalkan pengambilan keputusan. Hal ini mencakup perencanaan sumber daya, manajemen rantai pasokan, dan analisis risiko bisnis.

   Contoh: Dalam bisnis *e-commerce*, sistem penalaran berbasis kasus dapat membantu dalam mengelola inventaris dan mengoptimalkan proses pengiriman berdasarkan pengalaman pengelolaan pesanan yang ada.

6. Penalaran Berbasis Kasus dalam Pendidikan: Penalaran berbasis kasus juga telah digunakan dalam konteks pendidikan. Sistem berbasis kasus dapat memberikan rekomendasi pembelajaran yang disesuaikan dengan kebutuhan individual siswa berdasarkan pemahaman mereka terhadap materi pelajaran.

   Contoh: Sistem penalaran berbasis kasus telah digunakan dalam platform pembelajaran online untuk memberikan saran dan materi tambahan kepada siswa berdasarkan kinerja mereka dalam ujian dan tugas sebelumnya (Mamcenko, Kurilovas and Krikun, 2019).

7. Penalaran Berbasis Kasus Berbasis Konteks: Penalaran berbasis kasus berkembang untuk memasukkan konteks yang lebih luas dalam pengambilan keputusan. Termasuk

mempertimbangkan faktor lingkungan, situasional, dan sosial yang dapat memengaruhi keputusan yang diambil oleh sistem.

Contoh: Dalam kendaraan otonom, sistem dapat menggunakan penalaran berbasis kasus untuk mengambil keputusan berdasarkan situasi lalu lintas, kondisi cuaca, perilaku pengemudi lain, dan informasi kontekstual lainnya (Kiran et al., 2021).

Perkembangan terbaru dalam penalaran berbasis kasus menggabungkan teknologi canggih seperti NLP dan deep learning dengan aplikasi yang semakin beragam. Hal ini membuat penalaran berbasis kasus menjadi pendekatan yang lebih kuat dalam pengambilan keputusan, tidak hanya dalam konteks kecerdasan buatan, tetapi juga dalam berbagai industri dan disiplin ilmu lainnya. Dengan kemampuannya untuk mengadaptasi pengalaman sebelumnya ke situasi baru, penalaran berbasis kasus tetap menjadi alat berharga dalam pemecahan masalah dan pengambilan keputusan.

DAFTAR PUSTAKA

Bichindaritz, I. and Marling, C. (2006) 'Case-based reasoning in the health sciences: What's next?', *Artificial intelligence in medicine*, 36(2), pp. 127–135.

Kiran, B.R. *et al.* (2021) 'Deep reinforcement learning for autonomous driving: A survey', *IEEE Transactions on Intelligent Transportation Systems*, 23(6), pp. 4909–4926.

López, B. (2022) *Case-based reasoning: a concise introduction*. Springer Nature.

Main, J., Dillon, T.S. and Shiu, S.C.K. (2001) 'A tutorial on case based reasoning', *Soft computing in case based reasoning*, pp. 1–28.

Mamcenko, J., Kurilovas, E. and Krikun, I. (2019) 'On application of case-based reasoning to personalise learning', *Informatics in Education*, 18(2), pp. 345–358.

Recio-García, J.A., González-Calero, P.A. and Díaz-Agudo, B. (2014) 'jcolibri2: A framework for building Case-based reasoning systems', *Science of Computer Programming*, 79, pp. 126–145.

Russell, S.J. (2010) *Artificial intelligence a modern approach*. Pearson Education, Inc.

Shiu, S.C.K. and Pal, S.K. (2004) 'Case-based reasoning: concepts, features and soft computing', *Applied Intelligence*, 21, pp. 233–238.

BAB 13
PENGOLAHAN BAHASA ALAMI

Oleh Yulius Denny Prabowo

13.1 Pendahuluan

Dalam beberapa tahun terakhir, cabang-cabang kecerdasan buatan (AI) telah menarik begitu banyak orang dan menciptakan banyak penerapan (konteks) dimana metode dan pendekatan kecerdasan buatan tersebut diterapkan, cabang-cabang tersebut antara lain adalah: ilmu data (*data science*), analisis data (*data analysis*), analisis prediktif (*predictive analysis*), pengolahan bahasa alami (*natural language processing*/NLP) dan sebagainya. Bab ini akan fokus pada pengolahan bahasa alami.

Bab ini akan mencoba menjawab pertanyaan-pertanyaan seputar pengolahan bahasa alami, apa itu bahasa alami? Apa itu pemrosesan bahasa alami? Apa saja cabang lain yang terlibat dalam pembangunan sistem pakar menggunakan berbagai konsep pemrosesan bahasa alami? dan bagaimana kita bisa membangun sistem cerdas menggunakan konsep NLP?

Untuk mempermudah pemahaman, kita bisa membayangkan kalau kita ingin membangun sebuah mesin yang dapat berinteraksi dengan manusia menggunakan bahasa alami. Sistem cerdas semacam ini memerlukan teknologi komputasi dan linguistik komputasi untuk membangunnya, agar sistem tersebut mampu memproses bahasa alami seperti manusia.

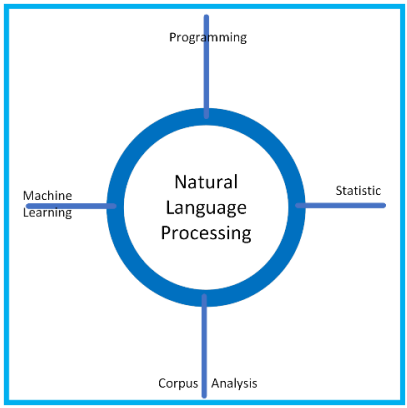
Kebutuhan pembangunan sistem yang dipaparkan di atas merupakan sebuah produk teknologi berupa perangkat lunak yang menggunakan Konsep NLP. Konsep NLP di atas bisa Anda

kaitkan dengan produk NLP yang saat ini sudah ada di pasaran, biasanya produk ini dikembangkan oleh perusahaan teknologi ternama dunia, seperti *Google Assistant* dari Google, *Siri Speech Assistance* dari Apple, dan lain sebagainya.

Dari contoh nyata di atas, sekarang kita dapat memahami pengertian NLP, yaitu sebagai berikut:

- Pemrosesan bahasa alami adalah kemampuan teknologi komputasi dan/atau linguistik komputasi untuk memproses bahasa alami manusia.
- Pemrosesan bahasa alami adalah bidang ilmu komputer, yang menggabungkan antara konsep kecerdasan buatan dan komputasi linguistik, yang berkaitan dengan interaksi antara komputer dan bahasa (alami) manusia.
- Pemrosesan bahasa alami dapat didefinisikan sebagai pemrosesan bahasa alami manusia secara otomatis (atau semi-otomatis).

Meskipun titik berat dari pengolahan bahasa alami adalah irisan dari kecerdasan buatan dan komputasi linguistik, dua hal ini bukanlah satu-satunya bidang yang mendasari pengolahan bahasa alami. Gambar berikut mengilustrasikan bidang-bidang yang berkontribusi dan mempengaruhi pengolahan bahasa alami.



Gambar 13.1 Bangunan Bidang NLP

NLP adalah cabang dari kecerdasan buatan. Pada awal perkembangannya konsep dari NLP digunakan dalam sistem pakar untuk melakukan fungsi seperti berikut: sistem pengenalan ucapan, sistem tanya jawab, penerjemahan dari satu bahasa tertentu ke bahasa tertentu lainnya, peringkasan teks, analisis sentimen, chatbot berbasis template, klasifikasi teks, dan segmentasi topik.

Pada tahap perkembangan selanjutnya NLP digunakan dalam beberapa perangkat lunak tingkat lanjut , seperti:

- Robot manusia yang memahami perintah bahasa alami dan berinteraksi dengan manusia dalam bahasa alami.
- Membangun sistem terjemahan mesin universal dengan bantuan pembelajaran mendalam (deep learning), saat ini kita dapat mengembangkan sistem terjemahan mesin universal dan Google baru-baru ini mengumumkan bahwa mereka hampir mencapai tujuan ini.
- Sistem NLP, yang menghasilkan judul logis untuk dokumen tertentu adalah juga salah satu contoh aplikasi tingkat lanjut. Dengan bantuan pembelajaran mendalam, Anda dapat membuat judul dokumen dan melakukan ringkasan dokumen tersebut.

- Sistem NLP, yang menghasilkan teks untuk topik tertentu atau untuk gambar juga dianggap sebagai aplikasi NLP tingkat lanjut.
- Chatbot tingkat lanjut, yang menghasilkan teks yang dipersonalisasi untuk manusia dan mengabaikan kesalahan dalam tulisan manusia.

Karena keterbatasan halaman, penulis akan membahas banyak informasi tentang pengolahan bahasa alami dengan sangat cepat dalam bab ini, jadi jika Anda melihat sesuatu yang tidak langsung masuk akal atau tidak anda pahami untuk saat ini, mohon jangan merasa tersesat dan bersabarlah.

Pembelajaran yang lebih mendalam dari sub-bab yang dijelaskan disini perlu dipelajari lebih lanjut, anda harus mengeksplorasi semua detail dan contoh yang disebutkan tadi secara mandiri, dengan demikian pembelajaran mandiri itu pasti akan membantu Anda menghubungkan titik-titik yang anda dapatkan saat ini.

13.2 Corpus dan Dataset

Aplikasi yang terkait dengan pemrosesan bahasa alami dibangun menggunakan data dalam jumlah besar. Dalam istilah awam, kumpulan data yang besar disebut basis data, basis data Bahasa disebut dengan istilah korpus. Jadi, secara lebih formal dan teknis, korpus dapat diartikan sebagai kumpulan materi bahasa alami tertulis atau lisan, disimpan di komputer, dan digunakan untuk mengetahui bagaimana bahasa digunakan.

Lebih tepatnya, korpus adalah kumpulan bahasa otentik yang terkomputerisasi dan sistematis yang digunakan untuk analisis linguistik serta analisis korpus. Jika mempunyai lebih dari satu korpus atau kumpulan dari korpus disebut korpora.

Untuk mengembangkan aplikasi NLP diperlukan korpus yang berupa materi bahasa alami tertulis atau lisan. Bahan atau data

tersebut akan kita gunakan sebagai data masukan dan mencoba mencari tahu fakta-fakta yang dapat membantu kita mengembangkan aplikasi NLP. Terkadang, aplikasi NLP menggunakan satu korpus sebagai masukan, dan di lain waktu, aplikasi tersebut menggunakan beberapa korpus sebagai masukan.

Ada banyak alasan penggunaan corpus untuk mengembangkan aplikasi NLP, beberapa di antaranya adalah sebagai berikut:

- Dengan bantuan korpus, kita dapat melakukan beberapa analisis statistik seperti distribusi frekuensi, kemunculan kata, dan sebagainya.
- Kita dapat mendefinisikan dan memvalidasi aturan linguistik untuk berbagai aplikasi NLP. Jika Anda sedang membangun sistem koreksi tata bahasa, Anda akan menggunakan korpus teks dan mencoba menemukan contoh tata bahasa yang salah, lalu Anda akan menentukan aturan tata bahasa yang membantu memperbaiki contoh tersebut.
- Kita dapat mendefinisikan beberapa aturan linguistik tertentu yang bergantung pada penggunaan bahasa tersebut. Dengan bantuan sistem berbasis aturan (rule based system), Anda dapat menentukan aturan linguistik dan memvalidasi aturan menggunakan korpus

Dalam sebuah korpus, kumpulan data yang besar dapat dalam format berikut: Pertama, data teks, yanb berarti berbentuk data/materi tertulis dan kedua, data ucapan, yang berarti berbentuk data/materi lisan.

13.2.1 Korpus Data Teks dan Data Ucapan

Data teks adalah kumpulan informasi tertulis. Ada beberapa sumber yang dapat digunakan untuk mendapatkan informasi tertulis seperti artikel berita, buku, perpustakaan digital, pesan email, halaman web, blog, dan sebagainya. Saat ini, kita semua hidup di dunia digital, sehingga jumlah informasi teks

berkembang pesat. Jadi, kita bisa menggunakan semua sumber daya yang ada di internet untuk mendapatkan data teks dan kemudian membuat korpus kita sendiri.

Mari kita ambil contoh: jika Anda ingin membangun sistem yang merangkum artikel berita, pertama-tama Anda mengumpulkan berbagai artikel berita yang ada di web dan menghasilkan kumpulan artikel baru sehingga koleksi tersebut menjadi korpus artikel berita Anda dan memiliki data teks. Anda dapat menggunakan *web-scrapper* untuk mendapatkan informasi dari halaman HTML.

Korpus data ucapan umumnya memiliki dua hal: satu adalah file audio, dan yang lainnya adalah transkripsi teksnya. Umumnya kita dapat memperoleh data ucapan dari rekaman audio. Rekaman audio ini mungkin berisi dialog atau percakapan orang. Contoh: saat Anda menelepon bank departemen layanan pelanggan, jika Anda memperhatikan, Anda akan mengetahui bahwa setiap panggilan direkam. Ini adalah cara Anda menghasilkan data ucapan atau korpus ucapan.

Korpus juga disebut sebagai kumpulan data dalam beberapa kasus. Ada tiga jenis korpus:

- Korpus monolingual: Korpus jenis ini memiliki satu bahasa
- Korpus bilingual: Korpus jenis ini memiliki dua bahasa
- Korpus multibahasa: Korpus jenis ini memiliki lebih dari satu bahasa

13.2.2 Analisis Korpus

Analisis korpus dapat didefinisikan sebagai metodologi untuk melakukan penyelidikan mendalam terhadap konsep linguistik yang didasarkan pada konteks situasi otentik dan komunikatif. Ketika kita membahas analisis korpus artinya, kita berbicara tentang korpora bahasa yang disimpan secara digital, yang

tersedia untuk diakses, diambil, dan dianalisis melalui komputer.

Analisis korpus untuk data ucapan memerlukan analisis pemahaman fonetik dari masing-masing contoh data. Selain analisis fonetik, kita juga perlu melakukan analisis percakapan, yang memberi kita gambaran tentang bagaimana interaksi sosial terjadi dalam kehidupan sehari-hari dalam konteks tertentu.

Misalkan dalam kehidupan nyata, jika Anda melakukan analisis percakapan untuk bahasa Indonesia yang digunakan dalam keseharian atau bahasa pergaulan, mungkin Anda akan menemukan kalimat seperti ”gimana kabar bro?” lebih sering digunakan dalam percakapan dibandingkan dengan ”Apa kabar tuan (atau nyonya)?”.

Analisis korpus untuk data teks terdiri dari penyelidikan statistik, manipulasi, dan generalisasi kumpulan data. Jadi untuk kumpulan data teks, biasanya dilakukan analisis terhadap berapa banyak kata berbeda yang ada dalam korpus dan berapa frekuensi kata tertentu dalam korpus. Jika korpus mengandung noise, kami mencoba menghilangkan noise tersebut. Hampir di setiap aplikasi NLP, kita perlu melakukan beberapa analisis korpus dasar agar kita dapat memahami korpus kita dengan baik. Secara praktik, library nltk memberi kita beberapa korpus bawaan. Jadi, kita dapat melakukan analisis korpus menggunakan korpus bawaan ini. Namun sebelum beralih ke bagian praktis, sangat penting untuk mengetahui jenis corpora apa yang ada di nltk.

Library nltk memiliki empat jenis corpora. Mari kita lihat masing-masing jenis tersebut:

Korpus terisolasi: Korpus jenis ini merupakan kumpulan teks atau bahasa alami. Contoh korpus semacam ini adalah gutenberg, webtext, dan sebagainya.

Korpus yang dikategorikan: Korpus jenis ini merupakan kumpulan teks yang dikelompokkan ke dalam berbagai jenis kategori. Contoh korpus jenis ini adalah korpus brown yang berisi data untuk berbagai kategori seperti berita, hobi, humor, dan sebagainya.

Korpus yang tumpang tindih: Korpus jenis ini merupakan kumpulan teks yang dikategorikan, namun kategorinya saling tumpang tindih. Contoh korpus jenis ini adalah korpus reuters, yang berisi data yang dikategorikan, namun kategori yang ditentukan saling tumpang tindih. Misalnya, jika Anda menganggap berbagai jenis kelapa sebagai satu kategori, Anda dapat melihat subkategori minyak kelapa, dan Anda juga memiliki minyak kapas. Jadi, di korpus reuters, berbagai kategori data saling tumpang tindih.

Korpus temporal: Korpus jenis ini merupakan kumpulan penggunaan bahasa alami selama periode waktu tertentu. Contoh korpus jenis ini adalah korpus inaugural address. Mengapa temporer? Misalkan Anda mengumpulkan penggunaan suatu bahasa di kota mana pun di Indonesia pada tahun 1950. Kemudian Anda mengulangi aktivitas yang sama untuk melihat penggunaan bahasa di kota tersebut pada tahun 1970 dan sekali lagi pada tahun 2017. Anda akan mencatat berbagai atribut data mengenai penggunaan bahasa, ada istilah baru dan ada juga istilah yang tidak lagi digunakan atau bahkan dikenal di periode waktu lainnya.

13.3 Mempersiapkan Dataset untuk aplikasi NLP

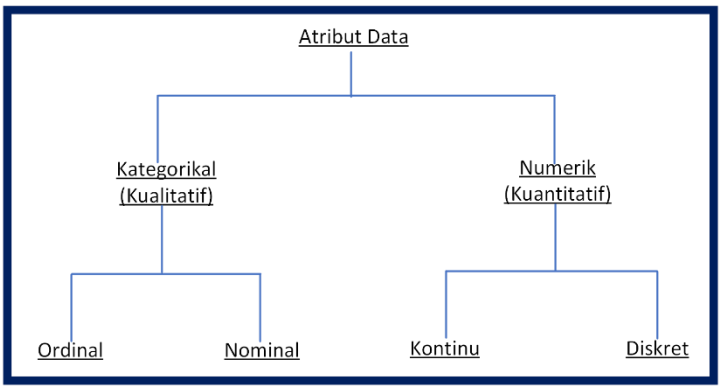
Semua korpora nltk tidak terlalu banyak noise nya. Meski demikian beberapa teknik *preprocessing* (pemrosesan awal) dasar diperlukan agar dapat menghasilkan fitur dari korpora di NLTK. Misalkan Anda memiliki korpus biokimia, maka Anda mungkin memiliki banyak persamaan dan nama kompleks

bahan kimia lainnya yang tidak dapat diuraikan secara akurat menggunakan parser yang ada. Untuk kondisi seperti inilah anda kemudian dapat, membuat keputusan apakah anda harus menghapusnya pada tahap preprocessing atau menyimpannya dan melakukan beberapa penyesuaian pada penguraian di tingkat *part-of-speech (POS) tagging*.

Dalam penerapannya di kehidupan nyata, korpus sangatlah kotor. Karena itu disediakan berberapa fungsi untuk memperbaiki data dalam NLTK. Misalnya dengan menggunakan FreqDist, Anda dapat melihat bagaimana kata-kata didistribusikan dan apa yang harus dan tidak boleh kita pertimbangkan. Pada umumnya langkah ini dilakukan saat *preprocessing*, bersamaan dengan metode lainnya. Misalnya anda perlu memeriksa banyak atribut kompleks seperti apakah hasil parsing, POS tagging, dan pemisahan kalimat sudah sesuai atau belum.

13.3.1 Memahami Atribut Data

Mari kita fokus pada atribut data seperti apa yang bisa muncul di korpus, berikut adalah gambar ilustrasi untuk atribut data.



Gambar 13.2 Atribut Data

Atribut data kategorikal atau kualitatif adalah sebagai berikut :

- Atribut data seperti ini lebih deskriptif

- Contohnya adalah catatan tertulis, corpora disediakan oleh nltk, corpus yang mencatat berbagai jenis ras anjing, seperti collie, Shepherd, dan terrier.

Ada dua sub-tipe atribut data kategorikal:

1. Data biasa (Ordinal):

   Atribut data jenis ini digunakan untuk mengukur konsep non-numerik seperti tingkat kepuasan, tingkat kebahagiaan, tingkat ketidaknyamanan, dan sebagainya. Pertimbangkan pertanyaan-pertanyaan berikut :

   Pertanyaan 1: Bagaimana perasaan Anda hari ini?

   Pilihan untuk Pertanyaan 1:

- Sangat buruk
- Buruk
- Bagus
- Senang
- Sangat senang

Sekarang Anda akan memilih salah satu opsi yang diberikan. Misalkan Anda memilih Bagus, tidak ada yang bisa mengubah perasaan Anda menjadi nilai numerik.

Semua pilihan sebelumnya adalah konsep non-numerik. Oleh karena itu, data tersebut termasuk dalam kategori data ordinal.

- Pertanyaan 2: Bagaimana Anda menilai layanan hotel kami?
- Pilihan untuk Pertanyaan 2:
- Buruk
- Rata-rata
- Diatas rata-rata
- Bagus
- Besar

Sekarang anggaplah Anda memilih salah satu opsi yang diberikan. Semua pilihan di atas akan mengukur tingkat

kepuasan Anda, dan sulit untuk mengubah jawaban Anda menjadi nilai numerik karena jawaban akan berbeda-beda dari orang ke orang.

Karena satu orang mengatakan Baik dan satu lagi mengatakan Di Atas Rata-Rata, maka ada kemungkinan keduanya merasakan hal yang sama terhadap pelayanan hotel namun memberikan respon yang berbeda. Dengan kata sederhana, Anda dapat mengatakan bahwa perbedaan antara satu opsi dan opsi lainnya tidak diketahui. Jadi, Anda tidak dapat menentukan secara tepat nilai numerik untuk jenis data ini.

2. Data nominal

Atribut data jenis ini digunakan untuk mencatat data yang tidak tumpang tindih. Contoh : Apa jenis kelamin anda ? Jawabannya laki-laki atau perempuan, dan jawabannya tidak tumpang tindih. Ambil contoh lain : Apa warna mata Anda ? Jawabannya bisa hitam, coklat, biru, atau abu-abu.

Dalam aplikasi terkait NLP, kita akan menangani atribut data kategorikal. Jadi, untuk mendapatkan titik data yang sesuai dari korpus yang memiliki atribut data kategorikal adalah bagian dari rekayasa fitur.

13.3.2. Mempersiapkan Dataset

Pada dasarnya ada tiga langkah untuk mempersiapkan kumpulan data Anda, sebagai berikut:

1. Memilih data (data selection)
2. Memproses data sebelumnya (preprocessing)
3. Mengubah data (data transforming)

Misalkan Anda bekerja dengan raksasa teknologi dunia seperti Google, Apple, Facebook, dan sebagainya. Maka Anda bisa dengan mudah mendapatkan data dalam jumlah besar, tetapi

jika Anda tidak bekerja dengan raksasa dan malah melakukan penelitian independen atau mempelajari beberapa konsep NLP, lalu bagaimana dan dari mana Anda bisa mendapatkan kumpulan data?

Pertama, tentukan jenis dataset yang Anda perlukan sesuai dengan aplikasi NLP yang ingin Anda kembangkan. Pertimbangkan juga hasil akhir dari aplikasi NLP yang Anda coba bangun. Jika Anda ingin membuat chatbot untuk domain layanan kesehatan, sebaiknya Anda tidak menggunakan dataset dialog layanan pelanggan perbankan. Jadi, pahami aplikasi atau pernyataan masalah Anda secara menyeluruh. Setelah pemilihan data dilakukan, selanjutnya dilakukan preprocessing data, berikut adalah Langkah umum melakukan preprocessing data:

Pemformatan, Pada langkah ini, dihasilkan format kumpulan data yang Anda rasa paling nyaman untuk digunakan. Jika Anda memiliki kumpulan data dalam format JSON dan Anda merasa paling nyaman bekerja dengan CSV, konversikan kumpulan data tersebut dari JSON ke CSV.

Cleaning, Pada langkah ini kita membersihkan data. Jika kumpulan data memiliki nilai yang hilang, hapus rekaman data tersebut atau ganti dengan nilai terdekat yang paling sesuai. Jika Anda menemukan atribut data yang tidak diperlukan, Anda juga dapat menghapusnya. Misalkan Anda membuat sistem koreksi tata bahasa, maka Anda dapat menghapus persamaan matematika dari dataset Anda karena aplikasi koreksi tata bahasa Anda tidak menggunakan persamaan.

Pengambilan Sampel, Pada tahap ini, Anda sebenarnya dapat mencoba mencari tahu atribut data mana yang tersedia yang dimiliki kumpulan data kita saat ini dan atribut data mana yang dapat kita peroleh. Kita juga mencoba mencari tahu atribut data apa yang paling penting sesuai aplikasi. Misalkan kita sedang membangun chatbot. Kita kemudian akan mencoba memecah kalimat menjadi kata-kata sehingga dapat mengidentifikasi kata

kunci dari kalimat tersebut. Jadi, informasi tingkat kata dapat diperoleh dari kalimat, dan informasi tingkat kata dan tingkat kalimat penting untuk aplikasi chatbot. Oleh karena itu, kami tidak menghapus kalimat, kecuali kalimat sampah. Dengan menggunakan pengambilan sampel, kami mencoba mengekstrak atribut data terbaik yang mewakili keseluruhan kumpulan data dengan sangat baik.

Transforming Data, Pada tahap ini, kita akan menerapkan beberapa teknik rekayasa fitur yang membantu kita mengubah data teks menjadi data numerik sehingga mesin dapat memahami dataset kita dan mencoba mencari tahu pola dalam dataset tersebut. Jadi tahap ini pada dasarnya adalah tahap manipulasi data. Pada domain NLP, untuk tahap transformasi kita dapat menggunakan beberapa teknik coding dan vektorisasi.

Semua tahapan sebelumnya adalah langkah dasar untuk mempersiapkan kumpulan data untuk aplikasi apa pun yang terkait dengan NLP atau ilmu data (data sains). Sekarang, mari kita lihat bagaimana Anda dapat menghasilkan data menggunakan web scraping.

13.3.3. Penerapan Algoritme

Berikut adalah penerapan algoritme untuk pelabelan dokumen. Perhatikan pseudo code di bawah ini:

1. Import library yang diperlukan
2. Definisikan fungsi untuk melakukan labeling teks:
    a. Menerima input teks yang akan dilabeli
    b. Lakukan preprocessing teks (misalnya, menghapus tanda baca, mengubah huruf menjadi lowercase)
    c. Lakukan tokenisasi teks menjadi kata-kata
    d. Lakukan proses labeling pada setiap kata
    e. Mengembalikan teks yang telah dilabeli

3. Definisikan kamus atau daftar kata yang akan digunakan untuk labeling
4. Menerima input teks dari pengguna
5. Panggil fungsi labeling teks dengan input teks dari pengguna sebagai argumen
6. Tampilkan teks yang telah dilabeli
7. Selesai

Pseudocode di atas memberikan gambaran umum tentang langkah-langkah yang perlu dilakukan untuk melakukan labeling teks menggunakan Python. Anda dapat mengimplementasikan pseudocode ini dalam bahasa pemrograman Python sesuai dengan kebutuhan dan preferensi Anda. Berikut adalah contoh implementasi pos tagging menggunakan library NLTK dan Python.

```
import nltk
# Download data yang diperlukan untuk pos tagging
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')

# Fungsi untuk melakukan pos tagging
def pos_tagging(text):
    # Tokenisasi teks menjadi kalimat-kalimat
    sentences = nltk.sent_tokenize(text)

    # Loop melalui setiap kalimat
    for sentence in sentences:
        # Tokenisasi kalimat menjadi kata-kata
        words = nltk.word_tokenize(sentence)

        # Melakukan pos tagging pada kata-kata
        tagged_words = nltk.pos_tag(words)

        # Menampilkan hasil pos tagging
```

```
    print(tagged_words)

# Contoh penggunaan
text = "I love to eat pizza."
pos_tagging(text)
```

Dalam contoh di atas, kita menggunakan library NLTK untuk melakukan pos tagging. Pertama, kita perlu mengunduh data yang diperlukan dengan menggunakan fungsi nltk.download(). Kemudian, kita mendefinisikan fungsi pos_tagging() yang menerima input teks.

Di dalam fungsi ini, teks akan dipecah menjadi kalimat-kalimat menggunakan nltk.sent_tokenize(), dan setiap kalimat akan dipecah menjadi kata-kata menggunakan nltk.word_tokenize(). Selanjutnya, kita menggunakan nltk.pos_tag() untuk melakukan pos tagging pada setiap kata. Hasil pos tagging akan ditampilkan menggunakan print().

Dalam contoh di atas, teks "I love to eat pizza." akan di-pos tagging dan hasilnya akan ditampilkan. Anda dapat mengganti teks dengan teks yang Anda inginkan untuk melakukan pos tagging.

Berikut adalah contoh implementasi labeling teks menggunakan library sckit-learn dan Python :

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import LabelEncoder

# Data teks yang akan dilabeli
texts = [
    "Ini adalah teks pertama.",
    "Ini adalah teks kedua.",
    "Ini adalah teks ketiga.",
    "Ini adalah teks keempat."
```

```
]

# Label untuk setiap teks
labels = [
    "label1",
    "label2",
    "label1",
    "label2"
]

# Inisialisasi objek CountVectorizer
vectorizer = CountVectorizer()

# Mengubah teks menjadi vektor fitur
X = vectorizer.fit_transform(texts)


# Inisialisasi objek LabelEncoder
label_encoder = LabelEncoder()

# Mengubah label menjadi angka
y = label_encoder.fit_transform(labels)

# Menampilkan hasil labeling
for i in range(len(texts)):
    print("Teks:", texts[i])
    print("Label:", label_encoder.inverse_transform([y[i]]))
    print()
```

Dalam contoh di atas, kita menggunakan library scikit-learn untuk melakukan labeling teks. Pertama, kita mendefinisikan data teks yang akan dilabeli dalam variabel texts dan label untuk setiap teks dalam variabel labels.

Selanjutnya, kita menggunakan objek CountVectorizer untuk mengubah teks menjadi vektor fitur. Kemudian, kita menggunakan objek LabelEncoder untuk mengubah label menjadi angka. Hasil labeling ditampilkan menggunakan loop for untuk setiap teks. Anda dapat mengganti data teks dan label sesuai dengan kebutuhan Anda. Berikut adalah contoh implementasi labeling korpus menggunakan library sckit-learn dan Python :

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import LabelEncoder

# Corpus teks yang akan dilabeli
corpus = [
    "Ini adalah teks pertama.",
    "Ini adalah teks kedua.",
    "Ini adalah teks ketiga.",
    "Ini adalah teks keempat."
]

# Label untuk setiap teks dalam korpus
labels = [
    "label1",
    "label2",
    "label1",
    "label2"
]

# Inisialisasi objek CountVectorizer
vectorizer = CountVectorizer()

# Mengubah korpus menjadi vektor fitur
X = vectorizer.fit_transform(corpus)
```

```
# Inisialisasi objek LabelEncoder
label_encoder = LabelEncoder()

# Mengubah label menjadi angka
y = label_encoder.fit_transform(labels)

# Menampilkan hasil labeling
for i in range(len(corpus)):
    print("Teks:", corpus[i])
    print("Label:", label_encoder.inverse_transform([y[i]]))
    print()
```

Dalam contoh di atas, kita menggunakan library scikit-learn untuk melakukan labeling pada sebuah korpus. Pertama, kita mendefinisikan korpus teks dalam variabel corpus dan label untuk setiap teks dalam variabel labels. Kita menggunakan objek CountVectorizer untuk mengubah korpus menjadi vektor fitur. Kemudian, kita menggunakan objek LabelEncoder untuk mengubah label menjadi angka. Hasil labeling ditampilkan menggunakan loop for untuk setiap teks dalam korpus.

Untuk membangun korpus menggunakan artikel berita, Anda dapat menggunakan library BeautifulSoup untuk melakukan web scraping dan mengambil teks dari artikel berita. Berikut adalah contoh kode Python untuk membangun korpus menggunakan artikel berita dari sebuah situs web:

```
import requests

from bs4 import BeautifulSoup


# URL dari situs web berita

url = "https://www.contohwebsiteberita.com"
```

```
# Mengirim permintaan HTTP ke situs web
response = requests.get(url)

# Mengecek apakah permintaan berhasil
if response.status_code == 200:
    # Mengambil konten HTML dari halaman web
    html_content = response.content

    # Membuat objek BeautifulSoup untuk melakukan parsing HTML
    soup = BeautifulSoup(html_content, "html.parser")

    # Mencari elemen HTML yang berisi artikel berita
    articles = soup.find_all("div", class_="article")

    # Membangun korpus teks dari artikel berita
    corpus = []
    for article in articles:
        # Mengambil teks dari artikel berita
        text = article.get_text()
        corpus.append(text)
```

```
    # Menampilkan korpus teks
    for text in corpus:
        print(text)
else:
    print("Gagal mengambil konten dari situs web.")
```

Dalam contoh di atas, kita menggunakan library requests untuk mengirim permintaan HTTP ke situs web berita dan mendapatkan konten HTML dari halaman web. Kemudian, kita menggunakan library BeautifulSoup untuk melakukan parsing HTML dan mencari elemen HTML yang berisi artikel berita. Setiap artikel berita diambil teksnya dan ditambahkan ke dalam korpus teks. Terakhir, kita menampilkan korpus teks dengan menggunakan loop for.

Pastikan untuk mengganti url dengan URL yang sesuai dengan situs web berita yang ingin Anda gunakan. Selain itu, Anda mungkin perlu menyesuaikan kode untuk menemukan elemen HTML yang berisi artikel berita pada situs web yang Anda pilih.

DAFTAR PUSTAKA

Ana Frankenberg Garcia, Lynne Flowerdew dan Guy Aston. 1988. *New Trend in Corpora and Language Learning.* New York: Continuum International Publishing Group.

James Pustejovsky dan Amber Stubbs. 2010. *Natural Language Annotation for Machine Learning*. Koln: Oreilly Publishing.

Krishna Bhavsar, Naresh Kumar dan Pratap Dangeti. 2017. *Natural Language with Python Cookbook*. Birmingham: Packt Publishing.Ltd.

Palash Goyal, Sumit Pandey dan Karan Jain. 2018*. Deep Learning for Natural Language Processing: Creating Neural Network with Python.* Bangalore: Apress.

Stefan TH Gries, Stefanie Wulff dan Mark Davies. 2017. *Corpus-Linguistic Application: current studies new directions*. Amsterdam: Rodopi Publishing.

Yuji Kawaguchi, Susumu Zaima dan Toshihiro Kawaguchi. 2011. *Spoken Language Corpus and Linguistic Informatics*. Jakarta Utara: John Benjamins Publishing Company.

BAB 14
SISTEM PAKAR

Oleh Budi Wijaya

Sistem pakar merupakan aplikasi berbasis kecerdasan buatan yang memiliki basis pengetahuan dari pakar. Peneliti kecerdasan buatan telah berhasil dalam 50 tahun terakhir dalam menyelesaikan masalah yang sangat rumit untuk dipecahkan menggunakan metode konvensional dengan memanfaatkan pengaplikasian kecerdasan buatan dan pengetahuan dari para pakar yang kini disebut sebagai sistem pakar.
Sistem pakar pertama kali diciptakan pada tahun 1970 dan mulai berkembang pada tahun 1980-an. Dapat dikatakan bahwa sistem pakar merupakan salah satu pengaplikasian kecerdasan buatan dalam bentuk perangkat lunak. Sistem pakar merupakan hasil pendekatan baru yang disebut sebagai sistem berbasis aturan. Untuk dapat mengembangkan sistem pakar, maka harus ada pengetahuan yang berasal dari ahlinya, contoh seorang dokter dapat memasukkan pengetahuan tentang penyakit beserta gejalanya ke dalam suatu sistem yang lalu sistem tersebut akan belajar bagaimana cara seorang dokter mengambil keputusan.

14.1 Definisi Sistem Pakar

Sistem pakar merupakan sebuah sistem komputer yang dapat mensimulasikan pengambilan keputusan layak seorang pakar(Stubblefield & Lugger, 1993). Dalam menciptakan sistem pakar, diperlukan pengetahuan dari pakar. Jika ingin membuat sebuah sistem pakar tentang penyakit umum, maka dibutuhkan

pengetahuan seorang dokter umum yang sudah berpengalaman di bidangnya lalu dimasukkan ke dalam sebuah sistem pakar. Lalu, apakah sistem pakar akan menggantikan profesi masa depan? Tentu tidak. Pada dasarnya sistem pakar ini dibuat untuk membantu pekerjaan seorang pakar di bidangnya. Seorang pakar juga manusia biasa yang tidak luput akan penyakit umur yaitu lupa. Nah, untuk mengatasi hal tersebut maka diperlukan sistem pakar ini agar pengetahuan seorang pakar dapat disimpan dan bahkan digunakan dalam membantu pengambilan keputusan.

Jika begitu, sistem pakar dapat digunakan dimana saja? Sistem pakar bisa digunakan hampir di semua sektor yang ada di negara kita. Penelitian yang paling banyak dilakukan untuk sistem pakar ini adalah penelitian di bidang kesehatan.

14.2 Fitur yang Harus Ada dalam Sistem Pakar

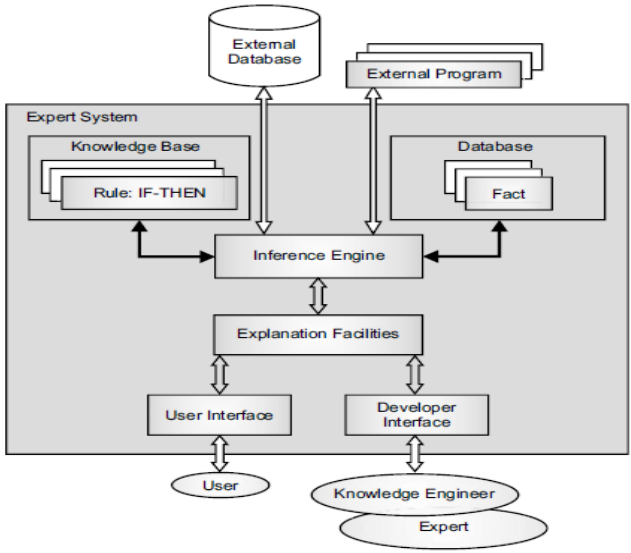
Dalam perancangan sistem pakar yang baik, maka harus mengikuti kaidah-kaidah berikut ini :

1. Bermanfaat : pengembangan sistem pakar harus memenuhi kebutuhan spesifik.
2. Mudah digunakan : sistem pakar yang dikembangkan dapat digunakan oleh orang awam.
3. Bersifat mendidik.
4. Dapat menjelaskan alasan dibalik keputusan yang diambil
5. Dapat mempelajari hal baru
6. Dapat menghasilkan keputusan yang berkualitas
7. Dapat menghasilkan keputusan dalam waktu cepat
8. Bersifat heuristik

14.3 Arsitektur dan Komponen Sistem Pakar

Arsitektur sistem pakar yang biasa digunakan adalah sistem berbasis aturan atau sistem produksi. Sistem ini menggunakan pengetahuan tertulis yang berada dalam bentuk aturan produksi

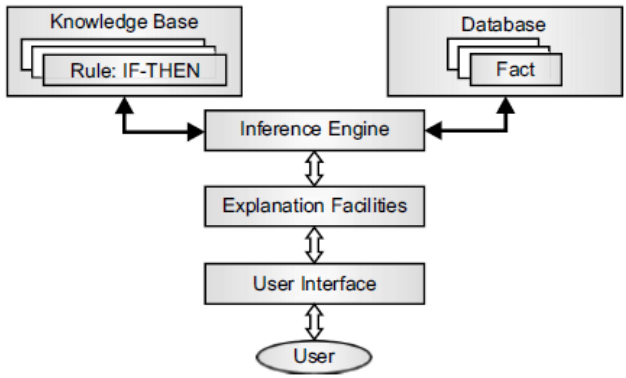
yaitu menggunakan fungsi Jika-Maka (IF-THEN) dalam bahasa pemrograman.



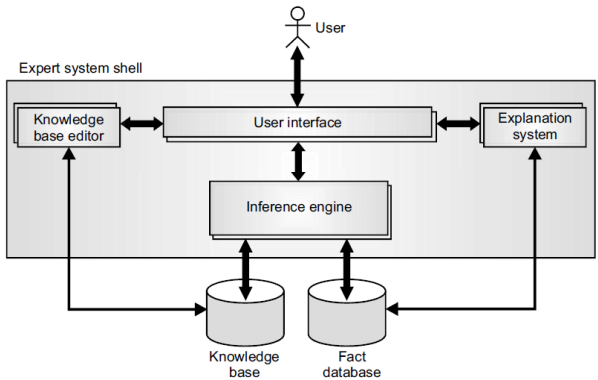
Gambar 14.1 Contoh Komponen dalam Sistem Pakar
(Sumber: Ithisa & Garima, 2020)

Contoh : Jika terjadi hujan, maka membawa payung

Setiap aturan akan merepresentasikan bagian terkecil dari pengetahuan pakar yang ada di dalam sebuah sistem. Ketika sebuah fakta cocok dengan kondisi 'jika' maka akan diberikan sebuah aksi/konklusi yang dapat berguna untuk pengguna.



Gambar 14.2 Komponen dalam Sistem Pakar
(Sumber: Ithisa & Garima, 2020)



Gambar 14.3 Arsitektur dalam Sistem Pakar
(Sumber: Artificial Intelligence and Expert System, 2020)

14.3.1 Antarmuka Pengguna (*User Interface*)

Dalam perancangan sistem pakar sangat bergantung dengan kualitas antarmuka pengguna. Proses komunikasi antara pengguna dan sistem akan melalui antarmuka pengguna, semakin ramah dan mudah digunakan maka semakin berhasil desain antarmuka pengguna ini.

Desain antarmuka pengguna dapat berupa berbasis teks dan berbasis grafik. Tergantung penggunaannya kepada siapa. Desain berbasis grafik umumnya lebih mudah digunakan oleh

orang awam. Pengguna dapat menjawab pertanyaan dari sistem, dan proses jawabnya pun harus didesain sebaik mungkin sehingga tidak menyulitkan pengguna dalam melakukan konsultasi kepada sistem.

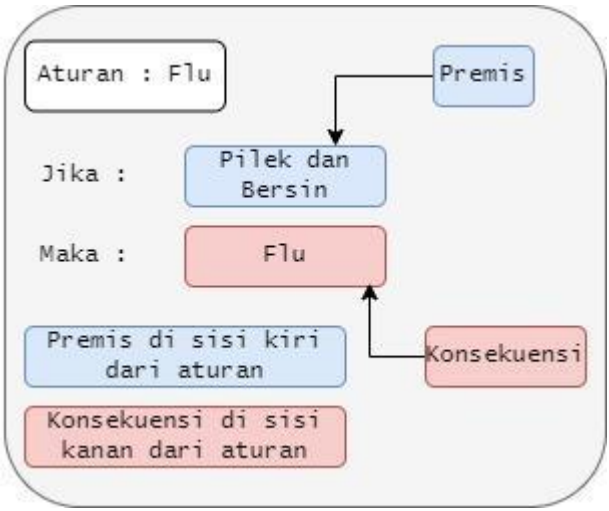
1. Basis Pengetahuan

Basis pengetahuan merupakan inti dari sebuah sistem pakar. Semua pengetahuan dari seorang pakar akan ditampung ke dalam sebuah basis pengetahuan dengan menggunakan teknik akuisisi. Contohnya, jika sebuah sistem pakar merupakan sistem yang berbasis pengetahuan kedokteran, maka ilmu dari seorang dokter yang telah berpengalaman di bidangnya dapat memasukkan pengetahuannya tersebut ke dalam sistem. Dokter tersebut lalu membuat peraturan-peraturan yang sesuai dengan bidang keilmuannya. Semua pengetahuan yang memuat keilmuan dari dokter sebut lalu disimpan dalam sebuah basis pengetahuan yang berbentuk aturan.

Aturan JIKA-MAKA memiliki dua bagian yaitu:

a. Bagian JIKA merupakan sebuah premis (bagian kondisi)
b. Bagian THEN merupakan sebuah kesimpulan atau konsekuensi atas premis sebuah premis (bagian aksi)

Lalu struktur dari sebuah aturan dapat berbentuk seperti berikut:



Gambar 14.4 Aturan Jika-Maka

Aturan JIKA-MAKA tidak serta-merta hanya dapat memiliki satu kondisi saja, tetapi dapat lebih dari satu. Dengan menambahkan aturan DAN(AND) maka akan menghasilkan sebuah aturan JIKA dimana kondisinya bisa lebih dari satu.
Sistem pakar bukan hanya dapat memberikan saran atas pengetahuan tentang gejala dan penyakit yang diderita berdasarkan gejala tersebut, tetapi juga bisa berupa hubungan, heuristik, atau sebuah rekomendasi.

a. Rekomendasi:

JIKA langit berawan,
DAN angin kencang,
MAKA disarankan untuk membawa payung

b. Hubungan:

JIKA bensin habis
MAKA mobil tidak dapat berjalan

c. Heuristik:

JIKA benda berbentuk cair
DAN PH benda < 6

DAN benda beraroma cuka
MAKA benda tersebut adalah asam asetat

Aturan produksi (JIKA-MAKA) adalah cara yang paling umum digunakan dalam merancang sebuah sistem pakar. Ketika bagian JIKA terpenuhi, konsekuensi yaitu MAKA akan terjadi. Sebuah sistem pakar yang termuat dalam sebuah basis pengetahuan berbentuk aturan produksi maka akan disebut sebagai sistem berbasis aturan.

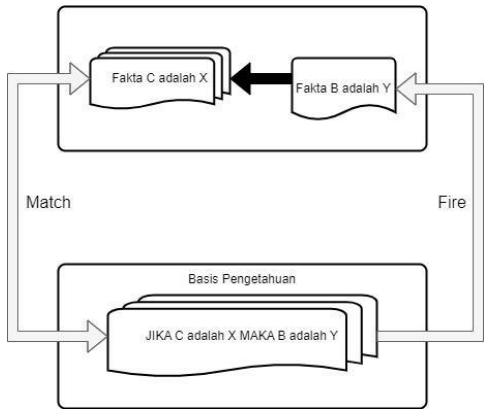
Sebuah basis pengetahuan terdiri dari fakta dan pertanyaan, serta aturan yang dimuat oleh pakar. Bisa dikatakan bahwa basis pengetahuan dari seorang pakar dari bidang spesifik merupakan gudang ilmu yang memiliki nilai yang sangat tinggi. Basis pengetahuan dari sebuah sistem pakar dapat berupa seperti berikut:

a. Pengetahuan Prosedural: sebuah pengetahuan dapat dikatakan sebagai prosedural jika memuat informasi langkah-langkah untuk menyelesaikan masalah.
b. Pengetahuan Faktual: Pengetahuan yang bersifat fakta dan dapat ditemukan dalam sebuah buku maupun jurnal ilmiah.
c. Pengetahuan Heuristik: Berbeda dengan pengetahuan faktual, yang disebarkan kepada seluruh orang. Pengetahuan heuristik bersifat individual dan lebih eksperimental. Pengetahuan heuristik ini dapat ditemukan dari sebuah pengalaman dalam praktik dan pengambilan keputusan yang baik.

Selain itu, lima proses yang sangat dibutuhkan dalam membentuk sebuah basis pengetahuan adalah akuisisi pengetahuan, analisis dan representasi pengetahuan, validasi, penyimpulan desain, lalu penjelasan dan justifikasi.

2. Mesin Inferensi

Mesin Inferensi merujuk kepada sebuah komponen sistem pakar yang dapat memikirkan keputusan apa yang akan diambil berdasarkan variabel-variabel yang telah dipilih. Mesin inferensi dapat dikatakan sebagai otak dari sebuah sistem pakar, karena komponen inilah yang menjadikan sistem pakar dapat bekerja selayak yang diinginkan oleh pembuatnya. Mesin inferensi ini akan mencocokkan fakta yang telah termuat di dalam basis pengetahuan dengan data yang dimasukkan oleh pengguna, selanjutnya keduanya lalu diolah dengan menggunakan aturan produksi. Ketika fakta dan data yang dimasukkan tersebut cocok dengan aturan JIKA (kondisi), aturan MAKA (konsekuensi) akan dieksekusi.



Gambar 14.5 Cara Kerja Mesin Inferensi Prosedur *Match-Fire*

Proses dalam mencocokkan fakta baru atau fakta yang telah ada terhadap aturan produksi disebut sebagai pencocokan pola. Sebuah mesin inferensi akan memprioritaskan aturan *FIRE*, dimana jika data dan fakta cocok, maka *FIRE*-nya adalah konsekuensi dari pencocokan kondisi. Mesin inferensi memiliki tiga metode dalam mengeksekusi sistem berbasis aturan, yaitu *forward chaining, backward chaining* dan *hybrid chaining.*

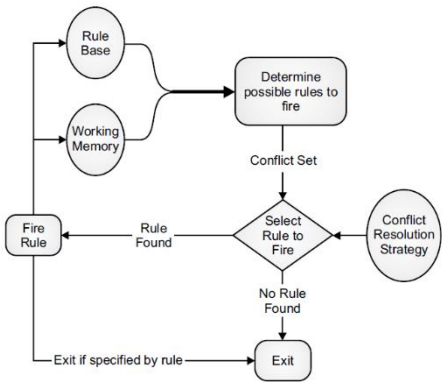
14.4 Algoritma

Seperti yang dijelaskan sebelumnya, dalam pembuatan sistem pakar dapat menggunakan sistem berbasis aturan dengan memanfaatkan mesin inferensi yang menjadi dasar atau otak dari sebuah sistem.

14.4.1 *Forward Chaining*

Forward chaining merupakan sebuah algoritma yang sangat umum digunakan dalam mesin inferensi. Algoritma ini akan mencari fakta ke depan hingga mencapai sebuah kesimpulan. *Forward chaining* juga disebut sebagai cara berpikir berdasarkan data (data-driven reasoning) karena mengambil kesimpulan berdasarkan data-data yang telah tersimpan di dalam basis pengetahuan.

Metode *forward chaining* akan mengecek masing-masing aturan, jika ada aturan yang terpenuhi. Maka akan terbentuk fakta baru dan yang memungkinkan dapat dipakai oleh aturan yang lain. Pengecekan aturan disebut sebagai interpretasi aturan yang berbasis pengetahuan dan dilakukan oleh mesin inferensi.



Gambar 14.6 *Forward Chaining*

Algoritma *forward chaining* juga sangat baik digunakan dalam membuat sebuah sistem rekomendasi. Contoh dibawah

merupakan aturan yang telah dimasukkan oleh seorang pakar tour guide:

a. Uang > 20 juta
b. Uang < 20 juta
c. Memiliki pemandangan yang indah
d. Negara yang bersih
e. Budaya yang menarik
f. Jepang
g. Singapura

Lalu seorang pengguna memasukkan referensinya sesuai aturan yang telah ada di sebuah sistem pakar.

Aturan:
R1: JIKA pengguna memiliki uang > 20 juta DAN memiliki pemandangan indah DAN budaya menarik MAKA jepang
R2: JIKA pengguna memiliki uang < 20 juta DAN Negara bersih MAKA Singapura
R3: JIKA pengguna memiliki uang > 20 juta DAN negara bersih MAKA Jepang
R4: JIKA uang < 20 juta maka Singapura
R5: JIKA uang > 20 juta maka Jepang

Disederhanakan menjadi:
R1: JIKA A DAN C DAN E MAKA F
R2: JIKA B DAN D MAKA G
R3: JIKA A DAN D MAKA F
R4: JIKA B MAKA G
R5: JIKA A MAKA F

Lalu fakta yang dimasukkan oleh pengguna adalah

- Uang < 20 juta (B)
- Negara yang Bersih (D)
- Budaya yang menarik (E)

Berdasarkan fakta dan aturan yang telah dimasukkan ke dalam sistem pakar, maka kesimpulan dari contoh diatas adalah G yaitu Singapura.

```
# Mendefinisikan aturan
rules = [
    (['uang', 'pemandangan_indah', 'budaya_menarik'], 'Jepang'),
    (['uang', 'negara_bersih'], 'Singapura'),
    (['uang', 'negara_bersih'], 'Jepang'),
    (['uang'], 'Singapura'),
    (['uang'], 'Jepang')
]
```

Gambar 14.7 Mendefinisikan Aturan (Google Colab)

Untuk implementasinya, tahap pertama kita bisa membangun kode aturan. Mari ubah aturan yang telah ditentukan di atas ke dalam sebuah kode bahasa pemrograman python. Dimana *rules* merupakan variabel yang menyimpan aturan yang telah kita tentukan.

Selanjutnya, mari sematkan fakta-fakta yang mungkin akan dimasukkan oleh pengguna. Potongan kode untuk variabel fakta dapat dilihat pada gambar 14.8 berikut ini.

```
# Definisikan Fakta
facts = {
    'uang': 25000000,  # Dalam bentuk rupiah
    'pemandangan_indah': True,
    'budaya_menarik': True,
    'negara_bersih': True,
    'Recommended_Destination': None
}
```

Gambar 14.8 Mendefinisikan Fakta (Google Colab)

Langkah selanjutnya yaitu membuat fungsi *forward chaining* dengan menggunakan *def*.

```
# Fungsi Forward Chaining
def forward_chaining(rules, facts):
    for conditions, destination in rules:
        if all(facts[cond] for cond in conditions):
            facts['Recommended_Destination'] = destination
            break  # Berhenti ketika mendapatkan hasil yang cocok
```

Gambar 14.9 Fungsi Forward Chaining (Google Colab)

Setelah fungsi telah didefinisikan, maka langkah terakhir yaitu memasukkan fakta dan aturan ke dalam *fungsi forward chaining.* Hasil dari sistem pakar sederhana *forward chaining* dapat dilihat pada gambar 14.10 berikut.

```
# Memasukkan aturan dan fakta ke fungsi forward chaining
forward_chaining(rules, facts)

# Mengubah rekomendasi yang awalnya kosong menjadi rekomendasi yang diperoleh sistem
recommended_destination = facts['Recommended_Destination']

# Mengeluarkan output rekomendasi
if recommended_destination:
    print(f"Berdasarkan kriteria yang dimasukkan, kamu bisa pergi ke {recommended_destination}.")
else:
    print("Maaf berdasarkan kriteria yang dimasukkan, sistem tidak dapat memberi")
```

```
Berdasarkan kriteria yang dimasukkan, kamu bisa pergi ke Jepang.
```

Gambar 14.10 Hasil *Forward Chaining* (Google Colab)

Hasilnya menunjukkan bahwa berdasarkan fakta dan aturan yang telah ditetapkan, rekomendasi dari sistem pakar dengan algoritma *forward chaining* adalah pengguna disarankan untuk pergi ke jepang.

14.4.2 *Backward Chaining*

Backward chaining atau rantai penelusuran ke belakang merupakan kebalikan dari *forward chaining*. Jika *forward chaining* merupakan metode yang merunut berdasarkan fakta dan dimulai dari premis JIKA, maka *backward chaining* akan menelusuri dari belakang yaitu dari konsekuensi/kesimpulan.

Dengan begitu, penelusuran pun dilakukan dari kanan ke kiri, yang berarti dimulai dari MAKA (dari bagian JIKA-MAKA), lalu

pencarian yang dilakukan akan menelusuri fakta-fakta yang cocok dengan premis JIKA. Ketika mendapatkan sebuah premis yang cocok dengan fakta yang dimasukkan oleh pengguna, maka aturan tersebut akan dieksekusi, selanjutnya akan dilihat kecocokan antara hasil eksekusi dan konsekuensi yang sedang dipilih oleh sistem. Jika konsekuensi dan hasil eksekusi berbeda, maka sistem akan mengganti *cursor* kesimpulan lain dan menjalankan aturan yang cocok dengan premis JIKA. Hal tersebut dilakukan hingga mendapatkan sebuah hasil yang cocok dengan kesimpulan. Mari mengambil contoh kasus diatas, lalu kita masukkan fakta baru seperti berikut.

Fakta baru yang dimasukkan pengguna:

- Uang < 20 juta
- Pemandangan yang indah

Lalu dengan menggunakan algoritma *backward chaining* akan dilakukan penelusuran seperti berikut.

- Mencari konsekuensi yang tersedia pada semua aturan yang ada, R1-R5.
- Dua konsekuensi MAKA ada dua, yaitu Jepang dan Singapura (F dan G).
- Sistem pakar akan menelusuri dari konsekuensi Jepang dan sistem akan mencocokkan fakta dan aturan yang memiliki konsekuensi Jepang di dalamnya.
- Didapatkan bahwa tidak ada aturan yang memiliki premis dan fakta yang cocok dengan konsekuensi Jepang, maka sistem akan mengganti cursor ke konsekuensi berikutnya, Singapura.
- Premis pun didapatkan, sebuah aturan pun dieksekusi dan hasilnya cocok dengan cursor konsekuensi saat ini.

Dengan begitu, kesimpulan dari fakta yang dimasukkan oleh pengguna adalah Singapura. Untuk implementasinya ke dalam sebuah bahasa pemrograman dapat dilihat pada gambar berikut.

```
# Define rules
rules = {
    'Jepang': [
        (lambda facts: facts['uang'] > 20000000 and facts['pemandangan_indah'] and facts['budaya_menarik'],)
    ],
    'Singapura': [
        (lambda facts: facts['uang'] < 20000000 and facts['negara_bersih'],),
    ]
}
```

Gambar 14.11 Aturan dalam backward chaining (Google Colab)

Pada gambar 16.11, sebuah variabel *rules* akan menyimpan data berupa aturan yang akan diimplementasikan di sebuah sistem pakar.

```
# Function for backward chaining
def backward_chaining(rules, goals, facts):
    results = {}

    for goal in goals:
        if goal in facts:
            # The goal is already known
            results[goal] = facts[goal]
        elif goal in rules:
            for conditions_tuple in rules[goal]:
                conditions_met = all(condition(facts) for condition in conditions_tuple)
                if conditions_met:
                    facts[goal] = True
                    results[goal] = True
                    break
            else:
                results[goal] = False

    return results
```

Gambar 14.12 Mendefinisikan fungsi *backward chaining* (Google Colab)

Dalam gambar 16.12 terlihat pembuatan fungsi *backward_chaining* yang mana akan mencocokkan tujuan yang disebut sebagai *goal*, dengan fakta-fakta yang akan dimasukan oleh pengguna ke dalam sistem pakar. Untuk faktanya bisa dilihat pada gambar berikut.

```
# Define the initial facts based on user input
facts = {
    'uang': 2500000,  # User's budget (in local currency),
    'pemandangan_indah': True,
    'budaya_menarik': True,
    'negara_bersih': True
}
```

Gambar 14.13 Mendefinisikan Variabel Fakta (Google Colab)

Lalu hasilnya dapat dilihat pada gambar 16.14 berikut ini.

```
# Specify the goals (recommend Jepang and Singapura)
goals = ['Jepang', 'Singapura']

# Perform backward chaining to find the recommended destinations
recommended_destinations = backward_chaining(rules, goals, facts)

# Print the recommended destinations
for goal, recommendation in recommended_destinations.items():
    if recommendation:
        print(f"Based on your criteria, we recommend traveling to: {goal}.")
    else:
        print(f"We couldn't make a recommendation for {goal} based on the provided criteria.")
```

```
We couldn't make a recommendation for Jepang based on the provided criteria.
Based on your criteria, we recommend traveling to: Singapura.
```

Gambar 14.14 Hasil Dari *Backward Chaining* (Google Colab)

Dari fakta dan aturan yang dicocokkan berdasarkan hasil implementasi ke dalam sebuah program sederhana diatas, menunjukan bahwa rekomendasi perjalanan pengguna dengan uang dibawah 20 juta, dengan pemandangan indah, budaya menarik dan negara bersih adalah perjalanan ke Singapura.

DAFTAR PUSTAKA

Azar, A. T., & Koubaa, A. (n.d.). *Studies in Computational Intelligence 1093*.

Ithisa, G., & Garima, N. 2020. *Artificial intelligence and expert System* (Vol. 1). MERCURY LEARNING AND INFORMATION.

Kim, J., Poong, S. L., & Seong, H. (n.d.). *Lecture Notes in Energy 94 Autonomous Nuclear Power Plants with Artificial Intelligence*.

Lucas, P. J. F., & Van Der Gaag, L. C. (n.d.). *Principles of Expert Systems*.

Russel, S., & Norvig, P. 2010. *Artificial Intelligence A Modern Approach* (Third Edition). MERCURY LEARNING AND INFORMATION. www.PlentyofeBooks.net

Sharma, A. (n.d.). *INTRODUCTION TO ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS DCAP310*.

BAB 15
Tren Terkini dalam Kecerdasan Buatan

Oleh Angga Aditya Permana

15.1 Pendahuluan

Gambar 15. 1 Tren *Artificial Intelligence*
(Sumber : https://ids.ac.id/)

Seiring pergantian waktu, kita telah menyaksikan perkembangan yang luar biasa dalam bidang Kecerdasan Buatan (Artificial Intelligence atau AI). Mari kita melangkah sejenak untuk menjelajahi perjalanan tren AI dari awal hingga hari ini. (Harvey dan Harvey 1998; Chowdhary 2020)

Pada awalnya, AI adalah impian ilmuan komputer. Ide untuk menciptakan mesin yang dapat berpikir dan belajar seperti

manusia telah menjadi fokus perhatian sejak awal abad ke-20. Namun, baru setelah beberapa dekade, pada tahun 1956, konsep AI mendapatkan pengakuan formal dalam konferensi Dartmouth College, di mana istilah "kecerdasan buatan" pertama kali digunakan.

Tahun-tahun pertama AI penuh tantangan. Meskipun memiliki potensi besar, komputer pada masa itu terbatas dalam kemampuan mereka untuk mengeksekusi tugas-tugas yang memerlukan pemahaman manusia. Algoritma-algoritma awal seperti "pohon pencarian" dan "logika predikat" digunakan untuk memodelkan pemikiran manusia, tetapi kemajuan nyata terbukti sulit dicapai.

Kemudian, pada tahun 1980-an dan 1990-an, AI mengalami periode ketidakjelasan yang dikenal sebagai "musim dingin AI." Minat dan pendanaan terhadap AI menurun karena kegagalan sistem AI untuk memenuhi harapan yang tinggi.

Namun, semuanya berubah pada awal abad ke-21. Kemajuan dalam perangkat keras, peningkatan besar dalam kapasitas komputasi, dan perkembangan algoritma seperti "pemelajaran mesin" dan "jaringan saraf tiruan" memulihkan AI dari tidur panjangnya. Kita menyaksikan lahirnya asisten virtual seperti Siri dan Alexa, yang mampu merespons pertanyaan kita dan mengikuti perintah.

Dalam beberapa tahun terakhir, AI telah mencapai puncak ketenaran. Teknologi ini menggemparkan dunia dengan kemampuannya dalam pengenalan wajah, penerjemahan bahasa otomatis, mobil otonom, dan bahkan merancang obat-obatan baru. Keberhasilan ini tidak hanya ditemukan dalam laboratorium riset, tetapi juga menginvasi banyak aspek kehidupan sehari-hari kita.

Saat ini, AI menjadi pendorong utama inovasi di berbagai sektor, termasuk kesehatan, transportasi, finansial, dan industri

manufaktur. Ia bahkan menciptakan peluang bisnis baru dan mengubah cara kita berinteraksi dengan teknologi.

Namun, seperti yang sering terjadi dalam sejarah teknologi, kemajuan AI juga membawa tantangan baru. Pertimbangan etika, privasi, dan keamanan menjadi semakin penting. Peraturan dan pedoman etis menjadi subjek diskusi yang mendalam.

Kita tidak bisa memprediksi dengan pasti apa yang akan terjadi selanjutnya dalam dunia AI, tetapi satu hal yang pasti adalah bahwa inovasi ini akan terus berkembang dan merubah dunia kita. AI tidak lagi hanya menjadi mimpi, ia telah menjadi kenyataan yang mengubah cara kita hidup, bekerja, dan berinteraksi satu sama lain. Dan sebagai kita melangkah maju, kita harus tetap berpikir kritis dan bertanggung jawab dalam mengelola teknologi yang semakin cerdas ini.

Beberapa tren utama yang paling meningkat dalam AI termasuk:

1. **Peningkatan Penggunaan Pemelajaran Mesin**: Pemelajaran mesin adalah salah satu subbidang utama dalam AI yang mengalami pertumbuhan pesat. Banyak industri, termasuk perbankan, e-commerce, kesehatan, dan lainnya, telah mengadopsi teknologi pemelajaran mesin untuk meningkatkan analisis data dan pengambilan keputusan.
2. **Pengolahan Bahasa Alami (NLP)**: Teknologi pemrosesan bahasa alami (NLP) terus berkembang. GPT-3, yang diluncurkan oleh OpenAI, adalah salah satu contoh terbaru yang memunculkan kemampuan generasi teks yang sangat maju.
3. **AI dalam Kesehatan**: AI digunakan secara luas dalam bidang kesehatan, termasuk dalam diagnois penyakit, pengembangan obat, dan manajemen data medis.

Pemrosesan citra medis, misalnya, telah menjadi lebih canggih.

4. **Kendaraan Otonom**: Perkembangan mobil otonom terus berlanjut dengan banyak perusahaan besar seperti Tesla, Waymo, dan General Motors yang mengembangkan teknologi ini. Namun, statistik terbaru tentang kecelakaan dan adopsi kendaraan otonom perlu diperbarui.
5. **AI di Industri Manufaktur**: Di sektor manufaktur, AI digunakan untuk mengoptimalkan rantai pasokan, perencanaan produksi, dan pemeliharaan prediktif.
6. **AI di Pemerintahan dan Layanan Publik**: Banyak pemerintah dan lembaga layanan publik mulai menggunakan AI untuk meningkatkan efisiensi dalam pengelolaan data, pelayanan pelanggan, dan pengambilan keputusan.
7. **AI Etika dan Regulasi**: Meningkatnya perhatian terhadap etika AI dan perlindungan privasi telah mengarah pada perkembangan regulasi dan pedoman yang lebih ketat di berbagai negara.
8. **AI dalam Bisnis**: Bisnis semakin menggunakan AI untuk meningkatkan proses bisnis, seperti analisis data, personalisasi pelanggan, dan otomatisasi tugas-tugas rutin.
9. **AI dalam E-commerce**: AI digunakan untuk memberikan rekomendasi produk yang lebih akurat kepada pelanggan, meningkatkan pengalaman berbelanja online, dan mengelola rantai pasokan.
10. **AI di Pendidikan**: AI digunakan untuk meningkatkan pengalaman pembelajaran, dengan sistem pengajaran adaptif dan tutor AI yang membantu siswa dalam belajar.

Untuk data terbaru tentang tren AI, Anda perlu merujuk ke sumber-sumber berita teknologi terkini, laporan riset industri, atau sumber informasi lain yang terkini. Tren AI dapat

bervariasi dari tahun ke tahun, dan perkembangannya sangat bergantung pada inovasi teknologi dan kebutuhan pasar.

15.2 Tren AI dalam Bidang Pendidikan

Gambar 15.2 Tren AI Bidang Pendidikan
(Sumber : https://www.refoindonesia.com/)

Tren Kecerdasan Buatan (Artificial Intelligence atau AI) dalam bidang pendidikan telah mengubah cara kita belajar, mengajar, dan mengelola pendidikan. Ini adalah cerita tentang bagaimana AI telah memasuki dunia pendidikan dan membuka peluang baru bagi siswa, guru, dan lembaga pendidikan. (Cardona et al. 2023)

Pada awalnya, pendidikan adalah pengalaman yang sangat personal. Siswa duduk dalam kelas, mendengarkan guru, dan memeriksa buku pelajaran mereka. Namun, dengan munculnya AI, pendidikan menjadi lebih adaptif dan individualis. Sistem pembelajaran adaptif yang didukung AI dapat menyesuaikan kurikulum dan tingkat kesulitan dengan kemampuan setiap

siswa. Ini berarti bahwa siswa yang lebih cepat dalam satu mata pelajaran dapat diberikan materi yang lebih menantang, sementara siswa yang memerlukan bantuan tambahan dapat menerima dukungan lebih lanjut.

Sistem pembelajaran adaptif juga membantu mengidentifikasi potensi masalah dalam pembelajaran siswa. AI dapat mengumpulkan data tentang kemajuan belajar siswa, mengidentifikasi tren, dan memberi tahu guru jika ada siswa yang memerlukan perhatian khusus. Hal ini memungkinkan intervensi dini yang dapat meningkatkan hasil belajar.

Selain itu, AI juga telah mengubah cara kita mengakses informasi. Dengan adanya asisten virtual yang didukung AI, seperti chatbot, siswa dapat dengan mudah mencari informasi, menjawab pertanyaan, dan mendapatkan bantuan dalam waktu nyata. Guru juga dapat menggunakan AI untuk mengelola tugas-tugas administratif mereka, membebaskan lebih banyak waktu untuk fokus pada pengajaran.

Di era pembelajaran jarak jauh, AI telah menjadi kunci untuk memungkinkan pembelajaran online yang lebih interaktif. Platform pembelajaran online menggunakan AI untuk memberikan konten yang sesuai dengan kebutuhan individu, membuat rekomendasi kursus, dan bahkan mengukur tingkat keterlibatan siswa selama sesi pembelajaran virtual.

Namun, dengan segala inovasi dan potensi yang dimiliki, ada juga tantangan yang perlu dihadapi. Salah satunya adalah masalah privasi data, di mana perlindungan data pribadi siswa menjadi penting. Selain itu, penting juga untuk memastikan bahwa pembelajaran dengan dukungan AI tidak menggantikan peran guru, tetapi lebih sebagai alat bantu yang dapat meningkatkan pengalaman belajar.

Tren AI dalam pendidikan adalah cerita tentang bagaimana teknologi dapat memajukan dunia pendidikan dengan cara yang

belum pernah terbayangkan sebelumnya. Ini adalah perjalanan yang terus berlanjut, di mana AI akan terus mengubah cara kita belajar dan mengajar, membawa pendidikan lebih dekat ke arah masa depan yang lebih adaptif, inklusif, dan efektif.

Mari kita lihat contoh perbandingan antara masa sebelum adanya Kecerdasan Buatan (Artificial Intelligence atau AI) dan masa setelah adanya AI dalam dunia pendidikan:

Sebelum AI dalam Pendidikan:

1. **Kelas Tradisional**: Sebelum adanya AI, kelas-kelas di sekolah dan perguruan tinggi cenderung berlangsung dalam format yang lebih tradisional. Guru menyampaikan materi dengan cara yang seragam kepada semua siswa dalam kelas.
2. **Kurikulum Standar**: Kurikulum pendidikan didasarkan pada standar nasional atau regional, yang mungkin tidak selalu mempertimbangkan perbedaan individu dalam tingkat pemahaman dan minat.
3. **Penilaian Manual**: Penilaian dan ujian dilakukan secara manual oleh guru, yang memerlukan waktu dan dapat menjadi subjektif.
4. **Keterbatasan Sumber Daya**: Sumber daya pendidikan terbatas, dan siswa mungkin tidak selalu memiliki akses ke materi pendidikan yang berkualitas.

Setelah AI dalam Pendidikan:

1. **Pembelajaran Adaptif**: AI digunakan untuk membuat pengalaman pembelajaran lebih adaptif. Sistem pembelajaran adaptif dapat menyesuaikan materi dan tingkat kesulitan dengan kebutuhan individu, memungkinkan siswa belajar pada kecepatan mereka sendiri.

2. **Pemantauan dan Analisis**: AI dapat digunakan untuk memantau kemajuan belajar siswa secara real-time. Data ini membantu guru mengidentifikasi masalah dan memberikan bantuan tambahan jika diperlukan.
3. **Personalisasi Kurikulum**: Dengan AI, lembaga pendidikan dapat merancang kurikulum yang lebih personal untuk setiap siswa. Ini memungkinkan siswa untuk mengejar minat mereka sendiri.
4. **Asisten Virtual**: AI seperti chatbot dapat membantu siswa menjawab pertanyaan mereka kapan saja, sehingga mendukung pembelajaran mandiri.
5. **Evaluasi Otomatis**: AI dapat digunakan untuk mengevaluasi tugas dan ujian secara otomatis, mengurangi beban kerja guru dan meningkatkan konsistensi penilaian.
6. **Pembelajaran Jarak Jauh**: AI mendukung pembelajaran jarak jauh dengan platform pembelajaran online yang menggunakan teknologi pemelajaran mesin untuk memberikan rekomendasi kursus dan konten yang sesuai.
7. **Kemampuan Analisis Data**: Lembaga pendidikan dapat menggunakan analisis data yang didukung AI untuk membuat keputusan berbasis bukti dan mengidentifikasi tren dalam hasil pembelajaran.
8. **Efisiensi Administratif**: AI digunakan untuk mengotomatisasi tugas-tugas administratif di lembaga pendidikan, memungkinkan staf administrasi untuk fokus pada pekerjaan yang lebih strategis.

Pengenalan AI dalam pendidikan telah membawa perubahan signifikan dalam cara pendidikan disampaikan dan diakses. Ini membantu mempersonalisasi pembelajaran, meningkatkan efisiensi, dan memberikan pengalaman yang lebih inklusif bagi siswa dengan berbagai tingkat kemampuan dan minat. Namun, juga penting untuk mengelola tantangan terkait privasi data dan

memastikan bahwa AI adalah alat bantu yang mendukung, bukan pengganti peran guru.

15.3 Tren AI dalam Bidang Kesehatan

Gambar 15.3 Tren AI dalam Bidang Kesehatan
(Sumber : https://jagad.id/)

Mari kita memasuki dunia tren Kecerdasan Buatan (Artificial Intelligence atau AI) dalam bidang kesehatan, di mana teknologi ini telah memberikan perubahan mendalam dalam cara kita merawat, mendiagnosis, dan mengelola kesehatan kita. (Nallamothu dan Cuthrell 2023)

Pada masa lalu, praktik medis dan perawatan kesehatan sering kali didasarkan pada pengetahuan dan pengalaman manusia yang terbatas. Pengambilan keputusan medis sering kali merupakan proses yang lamban dan tergantung pada diagnosa dan pengobatan yang bervariasi. Namun, dengan munculnya AI, lanskap kesehatan telah berubah secara dramatis.

Salah satu aspek paling mencolok dari tren AI dalam kesehatan adalah dalam diagnosa medis. Sistem AI yang dilatih dengan ribuan data medis dapat mengenali pola-pola dan tanda-tanda penyakit dengan tingkat akurasi yang tinggi. Ini telah memungkinkan deteksi dini penyakit seperti kanker, diabetes,

dan penyakit jantung dengan lebih cepat dan lebih tepat. Selain itu, robot bedah yang didukung AI telah mengubah cara operasi dilakukan, dengan presisi yang lebih besar dan risiko yang lebih rendah.

Penggunaan AI juga meluas ke perawatan pasien. Pemantauan pasien secara real-time dengan menggunakan perangkat medis terkoneksi AI memungkinkan tim medis untuk mengidentifikasi perubahan yang signifikan dalam kondisi pasien lebih cepat. Ini dapat mengarah pada tindakan yang lebih cepat dan peningkatan dalam merawat pasien secara keseluruhan.

AI juga digunakan dalam pengembangan obat. Proses penemuan dan pengembangan obat yang awalnya memakan waktu dan biaya besar telah ditingkatkan melalui simulasi komputer dan pemodelan molekuler yang didukung AI. Ini dapat menghasilkan penemuan obat-obatan baru dengan lebih cepat dan efisien.

Selain itu, AI berperan besar dalam manajemen data medis. Pengolahan besar-besaran data medis dapat mengidentifikasi tren dalam epidemiologi, membantu dalam perencanaan sumber daya kesehatan, dan mendukung penelitian medis lanjutan.

Namun, seperti semua teknologi, AI dalam kesehatan juga menghadapi tantangan. Perlindungan data pasien menjadi sangat penting, dan perlu adanya peraturan yang ketat untuk memastikan bahwa data medis aman dan privasi pasien terjaga. Selain itu, penting juga untuk memastikan bahwa AI digunakan secara etis dan tidak menggantikan peran profesional medis, tetapi mendukung mereka dalam pengambilan keputusan.

Tren AI dalam bidang kesehatan adalah kisah tentang bagaimana teknologi dapat menyelamatkan nyawa, meningkatkan akurasi diagnosa, dan mengubah cara kita merawat dan memahami kesehatan. Ini adalah perjalanan yang masih terus berkembang, dengan potensi untuk membawa

inovasi besar dan perbaikan dalam sektor kesehatan yang sangat penting ini.

Berikut adalah contoh perbandingan antara masa sebelum adanya Kecerdasan Buatan (Artificial Intelligence atau AI) dan masa setelah adanya AI dalam bidang kesehatan:

Sebelum AI dalam Kesehatan:

1. **Diagnosa Manual**: Diagnosa medis dan interpretasi hasil uji sering kali dilakukan secara manual oleh tenaga medis. Ini bisa memakan waktu, dan tingkat akurasi dapat bervariasi tergantung pada pengalaman individu.
2. **Pemantauan Pasien Terbatas**: Pemantauan pasien yang konstan dalam rumah sakit sering kali terbatas oleh jumlah perawat dan staf medis yang terbatas. Pasien mungkin hanya dimonitor secara berkala.
3. **Pengolahan Data Lambat**: Pengolahan data medis dan rekam medis sering kali memerlukan waktu yang lama dan seringkali dilakukan secara manual.
4. **Penemuan Obat Lambat**: Proses penemuan dan pengembangan obat adalah proses yang mahal dan memakan waktu. Pencarian obat baru seringkali terbatas oleh pendekatan percobaan dan kesalahan.

Setelah AI dalam Kesehatan:

1. **Diagnosa Cepat dan Akurat**: AI digunakan dalam interpretasi gambar medis, seperti pemindaian CT atau MRI, untuk membantu dokter dalam mendiagnosis penyakit seperti kanker atau penyakit jantung dengan akurasi yang lebih tinggi dan lebih cepat.
2. **Pemantauan Pasien Real-time**: AI digunakan dalam perangkat medis yang terkoneksi untuk pemantauan pasien secara real-time. Ini memungkinkan pengawasan pasien

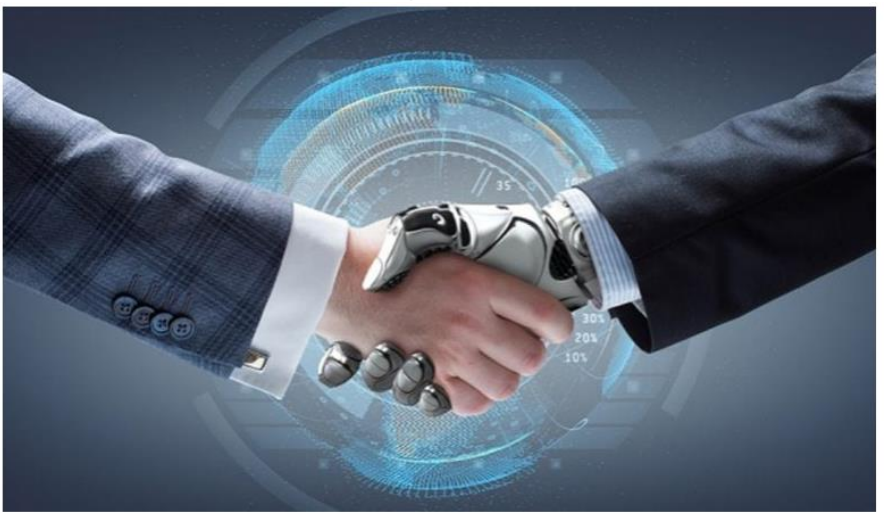
yang lebih ketat dan deteksi dini perubahan yang signifikan dalam kondisi pasien.

3. **Pengolahan Data Cepat**: AI digunakan untuk memproses dan menganalisis data medis dengan cepat. Ini dapat mengidentifikasi pola dan tren dalam data yang mungkin tidak terdeteksi oleh manusia.
4. **Penemuan Obat yang Cepat dan Efisien**: Dengan pemodelan molekuler dan simulasi komputer yang didukung AI, peneliti obat dapat mengidentifikasi kandidat obat potensial dengan lebih cepat dan efisien. Ini mempercepat pengembangan obat baru.
5. **Asisten Virtual Kesehatan**: Chatbot dan asisten virtual kesehatan yang didukung AI dapat memberikan informasi kesehatan yang akurat dan menjawab pertanyaan pasien 24/7.
6. **Manajemen Data Medis**: AI digunakan untuk mengelola dan menganalisis data medis besar-besaran untuk identifikasi tren epidemiologi dan perencanaan sumber daya kesehatan.
7. **Perawatan Kesehatan Jarak Jauh**: AI digunakan dalam telemedicine untuk memberikan perawatan kesehatan jarak jauh, yang lebih terjangkau dan dapat diakses oleh individu yang berada di daerah terpencil.
8. **Pengembangan Terapi Personalisasi**: AI dapat digunakan untuk merancang terapi dan perawatan yang lebih personal untuk pasien berdasarkan data genetik dan medis mereka.

Perbandingan ini menunjukkan bagaimana AI telah mengubah lanskap kesehatan dengan menyediakan alat yang lebih kuat dan cepat untuk mendiagnosis, merawat, dan mengelola data medis. Ini juga menciptakan peluang untuk perawatan yang lebih personal dan efisien. Namun, penting untuk memastikan bahwa

AI digunakan dengan etika dan privasi data pasien yang dijaga dengan baik.

15.4 Tren AI dalam Bidang Bisnis



Gambar 15.4 Tren AI dalam Bidang Bisnis
(Sumber : https://supportmymac.ca/)

Tren Kecerdasan Buatan (Artificial Intelligence atau AI) dalam dunia bisnis adalah kisah tentang bagaimana teknologi ini telah membuka pintu untuk inovasi, efisiensi, dan perubahan yang mendalam dalam cara perusahaan beroperasi dan berkomunikasi dengan pelanggan. Mari kita telusuri bagaimana AI telah membentuk lanskap bisnis saat ini. (Bharadiya *et al.* 2023)

Sebelum AI memasuki dunia bisnis secara luas, proses bisnis sering kali tergantung pada pengambilan keputusan manusia, yang mungkin rentan terhadap bias, kesalahan, dan terbatas dalam skala. Namun, saat AI mulai digunakan dalam bisnis, segalanya berubah.

Salah satu tren utama adalah penggunaan AI dalam analisis data. Perusahaan kini memiliki akses ke alat analisis data yang

mampu mengurai besar-besaran data dalam waktu yang singkat dan mengidentifikasi pola, tren, dan wawasan yang berharga. Ini membantu perusahaan membuat keputusan berbasis bukti yang lebih baik dalam hal pengelolaan stok, pemasaran, dan perencanaan strategis.

Selain itu, AI telah mengubah cara perusahaan berkomunikasi dengan pelanggan. Chatbot AI dan asisten virtual digunakan untuk memberikan dukungan pelanggan dalam waktu nyata, 24/7, mengatasi pertanyaan umum, dan mengarahkan pelanggan ke solusi yang sesuai. Ini memberikan pengalaman pelanggan yang lebih baik dan mengurangi beban pekerjaan agen layanan pelanggan manusia.

Dalam pengelolaan rantai pasokan, AI memainkan peran penting dalam perencanaan, pemantauan, dan pengoptimalan. Prediksi permintaan yang akurat, penggunaan algoritma routing cerdas untuk pengiriman, dan manajemen stok otomatis semuanya dapat meminimalkan biaya dan meningkatkan efisiensi operasional.

AI juga telah memasuki dunia pemasaran dengan kuat. Mesin pemelajaran yang dikendalikan AI dapat menganalisis perilaku konsumen, menganalisis kampanye pemasaran, dan membuat rekomendasi produk yang sesuai. Ini memungkinkan perusahaan untuk menciptakan kampanye yang lebih efektif dan meningkatkan konversi.

Namun, seiring dengan semua inovasi ini, perusahaan juga menghadapi tantangan baru, termasuk masalah privasi data, etika AI, dan persaingan dalam sumber daya manusia yang semakin canggih. Pengembangan dan penerapan AI yang sukses membutuhkan visi yang jelas, kebijakan yang bijaksana, dan pemahaman yang mendalam tentang teknologi ini.

Tren AI dalam bisnis adalah kisah tentang bagaimana teknologi mengubah lanskap bisnis dengan menyediakan alat-alat yang

kuat untuk analisis, pengambilan keputusan, dan pelayanan pelanggan. Ini adalah era di mana perusahaan yang dapat memanfaatkan AI secara efektif akan mendapatkan keunggulan kompetitif yang signifikan di pasar yang semakin kompetitif.

Tren Kecerdasan Buatan (*Artificial Intelligence* atau AI) telah membawa perubahan signifikan dalam dunia bisnis. Di bawah ini adalah beberapa contoh perbandingan antara masa sebelum adanya AI dan masa setelah adanya AI dalam bidang bisnis:

Sebelum AI dalam Bisnis:

1. **Pengambilan Keputusan Berdasarkan Intuisi**: Keputusan bisnis sering kali didasarkan pada pengalaman dan intuisi manajer. Analisis data terbatas, dan seringkali keputusan dibuat berdasarkan "perasaan."
2. **Pemasaran Manual**: Upaya pemasaran cenderung manual, dengan perencanaan kampanye yang dilakukan secara tradisional dan pemilihan target pasar yang kurang tepat.
3. **Manajemen Rantai Pasokan Konvensional**: Manajemen rantai pasokan sering kali tergantung pada perkiraan manual, yang dapat mengakibatkan kekurangan atau kelebihan persediaan.
4. **Layanan Pelanggan Terbatas**: Layanan pelanggan umumnya terbatas pada jam kerja normal, dan interaksi pelanggan seringkali melibatkan lamanya waktu tunggu dan berbagai pertanyaan yang harus dijawab oleh agen manusia.

Setelah AI dalam Bisnis:

1. **Pengambilan Keputusan Berbasis Data**: AI memungkinkan perusahaan untuk membuat keputusan berdasarkan data dan analisis yang akurat. Algoritma AI dapat memproses dan menganalisis data dalam skala besar untuk memberikan wawasan yang berharga.

2. **Pemasaran yang Digerakkan Data**: AI digunakan dalam pemasaran untuk menganalisis perilaku konsumen, mempersonalisasi rekomendasi, dan mengotomatisasi kampanye yang lebih efektif.
3. **Manajemen Rantai Pasokan yang Efisien**: AI digunakan dalam manajemen rantai pasokan untuk meramalkan permintaan, memantau persediaan secara real-time, dan mengoptimalkan proses pengiriman.
4. **Layanan Pelanggan 24/7**: Asisten virtual dan chatbot AI memberikan dukungan pelanggan yang tersedia 24/7, menjawab pertanyaan dengan cepat, dan mengarahkan pelanggan ke solusi yang tepat.
5. **Automatisasi Tugas Rutin**: AI digunakan untuk mengotomatisasi tugas-tugas rutin dalam bisnis, seperti pemrosesan data, sehingga staf dapat fokus pada pekerjaan yang lebih strategis.
6. **Analisis Sentimen dan Umpan Balik Konsumen**: AI dapat menganalisis sentimen konsumen dari data sosial media dan ulasan online untuk mendapatkan wawasan tentang preferensi pelanggan dan umpan balik produk.
7. **Prediksi Kerentanan Keamanan**: AI dapat digunakan untuk menganalisis data keamanan dan memprediksi potensi ancaman serta mengambil tindakan pencegahan yang diperlukan.
8. **Optimasi Harga dan Persediaan**: AI dapat membantu dalam mengoptimalkan harga produk dan mengelola persediaan dengan lebih baik, meningkatkan profitabilitas perusahaan.
9. **Pengenalan Pola dalam Data Keuangan**: AI digunakan dalam analisis keuangan untuk mengidentifikasi pola anomali atau peluang investasi.

Perbandingan ini mencerminkan bagaimana AI telah mengubah berbagai aspek bisnis, dari pengambilan keputusan hingga pemasaran, manajemen rantai pasokan, dan layanan pelanggan. AI telah membuka peluang baru untuk efisiensi, inovasi, dan pertumbuhan bisnis yang lebih baik, sambil menghadirkan tantangan baru dalam hal keamanan data dan etika penggunaan teknologi.

15.5 Tren AI dalam Bidang Otomotif

Gambar 15.5 Tren AI dalam Bidang Otomotif
(Sumber : https://tirto.id/)

Tren Kecerdasan Buatan (Artificial Intelligence atau AI) dalam bidang otomotif adalah kisah yang sangat menarik tentang bagaimana teknologi ini telah mengubah industri kendaraan bermotor secara fundamental. Ini adalah cerita tentang bagaimana AI telah membawa otomotif ke era baru yang penuh inovasi dan mobilitas cerdas.. (Vinayak Pujari 2022)

Sebelum AI merambah otomotif, mobil dianggap sebagai kendaraan yang berfungsi untuk mengantarkan manusia dari satu tempat ke tempat lain. Namun, dengan munculnya AI, mobil

telah menjadi lebih dari sekadar alat transportasi. Mereka telah menjadi platform bergerak yang cerdas, yang dapat mengambil keputusan, berkomunikasi dengan pengemudi dan penumpang, dan bahkan menghindari kecelakaan.

Salah satu tren utama dalam otomotif AI adalah perkembangan kendaraan otonom. Mobil otonom menggunakan AI dan sensor-sensor canggih untuk mengemudi tanpa pengemudi manusia. Ini telah membuka peluang besar dalam hal keselamatan, efisiensi, dan mobilitas. Mobil otonom dapat membantu mengurangi kecelakaan lalu lintas karena kemampuannya untuk mendeteksi dan merespons situasi berpotensi berbahaya lebih cepat daripada manusia.

AI juga digunakan dalam sistem bantuan pengemudi yang cerdas, seperti pengereman otomatis, peringatan tabrakan, dan asisten parkir. Sistem-sistem ini membantu pengemudi dalam mengemudi lebih aman dan nyaman.

Selain itu, AI juga telah memasuki interior kendaraan. Sistem infotainment AI yang canggih memungkinkan pengemudi dan penumpang untuk berinteraksi dengan mobil melalui suara atau sentuhan, mengakses informasi, hiburan, dan layanan online dalam perjalanan.

Dalam hal perawatan dan pemeliharaan, AI digunakan untuk pemantauan kondisi kendaraan secara real-time. Ini membantu pemilik mobil untuk melakukan perawatan preventif yang tepat waktu, mengidentifikasi masalah potensial, dan menghindari kerusakan yang mahal.

Penggunaan AI dalam otomotif juga berdampak pada produksi. Pabrik-pabrik otomotif menggunakan robot yang dikendalikan AI untuk memproduksi mobil dengan presisi tinggi dan efisiensi yang lebih besar. Ini dapat mengurangi biaya produksi dan meningkatkan kualitas produk.

Tren AI dalam otomotif adalah kisah tentang bagaimana teknologi ini telah mengubah cara kita berkendara, memperbaiki keselamatan jalan raya, meningkatkan pengalaman pengemudi dan penumpang, serta membuka peluang baru dalam industri otomotif. Saat teknologi ini terus berkembang, kita dapat mengharapkan lebih banyak inovasi yang akan mengubah cara kita berinteraksi dengan kendaraan dan lingkungan transportasi kita.

Berikut adalah beberapa contoh perbandingan antara masa sebelum adanya Kecerdasan Buatan (Artificial Intelligence atau AI) dan masa setelah adanya AI dalam bidang otomotif:

Sebelum AI dalam Otomotif:

1. **Kendaraan Manual**: Semua kendaraan bergantung pada pengemudi manusia untuk mengemudi, mengambil keputusan, dan menghindari kecelakaan.
2. **Sistem Keamanan Dasar**: Sistem keamanan dalam mobil cenderung terbatas pada rem, airbag, dan sabuk pengaman, yang berfungsi sebagai respons pasif terhadap kecelakaan.
3. **Navigasi Terbatas**: Navigasi dalam mobil bergantung pada peta fisik atau GPS dasar yang memberikan arahan dasar.
4. **Kendaraan Tidak Terhubung**: Mobil tidak memiliki kemampuan untuk terhubung dengan internet atau jaringan lainnya.

Setelah AI dalam Otomotif:

1. **Kendaraan Otonom**: AI digunakan untuk mengendalikan kendaraan otonom yang dapat mengemudi tanpa intervensi manusia. Ini membuka peluang untuk mobilitas yang lebih aman dan efisien.
2. **Sistem Keamanan Canggih**: AI digunakan dalam sistem keamanan yang canggih seperti pengereman otomatis,

peringatan tabrakan, dan deteksi jalan raya yang memungkinkan kendaraan untuk merespons situasi berbahaya secara otomatis.

3. **Navigasi Pintar**: Sistem navigasi AI menggunakan data real-time dan pemetaan digital yang canggih untuk memberikan petunjuk dan rekomendasi lalu lintas yang tepat waktu. Mereka bahkan dapat menyarankan rute alternatif untuk menghindari kemacetan.
4. **Kendaraan Terhubung**: Kendaraan terhubung ke internet dan sistem lainnya, memungkinkan layanan seperti streaming media, pembaruan perangkat lunak otomatis, dan layanan telematika yang memantau kondisi kendaraan.
5. **Asisten Pribadi**: Sistem asisten pribadi dalam kendaraan menggunakan suara dan pemahaman bahasa alami untuk membantu pengemudi dan penumpang dalam berbagai tugas, termasuk menjawab pertanyaan, mengendalikan fungsi mobil, dan memberikan informasi lalu lintas.
6. **Perawatan yang Terprediksi**: AI digunakan untuk memantau kondisi kendaraan secara real-time dan memberikan peringatan dini tentang perawatan yang diperlukan, mengurangi risiko kerusakan besar.
7. **Manajemen Lalu Lintas yang Efisien**: AI digunakan untuk mengelola lalu lintas di jalan raya dengan lebih baik, termasuk penggunaan lampu lalu lintas cerdas dan sistem koordinasi kendaraan otonom.
8. **Perkiraan Bahan Bakar yang Akurat**: AI digunakan untuk menghitung perkiraan konsumsi bahan bakar berdasarkan gaya mengemudi, rute, dan kondisi jalan, membantu pengemudi mengoptimalkan penggunaan bahan bakar.

Perbandingan ini mencerminkan bagaimana AI telah mengubah dunia otomotif dengan meningkatkan keselamatan, efisiensi, dan

pengalaman pengemudi. Dengan perkembangan teknologi AI yang terus berlanjut, kita dapat mengharapkan transformasi lebih lanjut dalam industri otomotif di masa depan.

15.6 Tren AI dalam Bidang Entertainment

Gambar 15.6 Tren AI dalam Bidang *Entertainment*
(Sumber : https://www.openpr.com/)

Tren Kecerdasan Buatan (*Artificial Intelligence* atau AI) dalam bidang hiburan adalah kisah tentang bagaimana teknologi ini telah mengubah cara kita mengkonsumsi hiburan, mulai dari film dan musik hingga permainan video dan konten digital. Ini adalah cerita tentang bagaimana AI telah meningkatkan pengalaman hiburan kita dengan cara yang belum pernah terbayangkan sebelumnya. (Lachman dan Joffe 2020)

Pada awalnya, hiburan seringkali merupakan pengalaman pasif di mana kita adalah penonton yang hanya menerima konten yang telah dibuat oleh seniman dan kreator. Namun, dengan munculnya AI, hiburan telah menjadi lebih interaktif, personal, dan disesuaikan.

Di Dunia Film dan Televisi:

- AI digunakan dalam produksi film untuk menghasilkan efek visual yang luar biasa dan grafis yang kompleks, serta untuk mengotomatisasi tugas-tugas pascaproduksi seperti pengeditan dan pemrosesan audio.
- Algoritma pemilihan konten AI digunakan dalam platform streaming untuk merekomendasikan film dan acara TV yang sesuai dengan preferensi pengguna, meningkatkan pengalaman menonton.
- Dalam animasi, AI memungkinkan pembuatan karakter digital yang lebih realistis dan ekspresif.
- Dalam penulisan naskah, AI dapat membantu seniman dalam merencanakan plot, menghasilkan dialog, atau bahkan menciptakan skenario alternatif.

Dalam Musik:

- AI digunakan untuk menciptakan musik yang baru dan orisinal, serta untuk mereplikasi gaya musisi terkenal.
- Dalam streaming musik, algoritma AI digunakan untuk membuat playlist yang disesuaikan dengan selera pendengar, membantu pengguna menemukan musik yang mereka sukai.
- AI juga dapat digunakan untuk menghasilkan suara yang realistis dalam permainan video dan aplikasi hiburan.

Dalam Permainan Video:

- AI digunakan dalam permainan untuk menghasilkan karakter non-pemain (NPC) yang lebih cerdas dan reaktif, sehingga pengalaman bermain lebih dinamis.
- Algoritma pembelajaran mesin digunakan dalam permainan untuk menyesuaikan tingkat kesulitan dengan kemampuan pemain, sehingga setiap pengalaman bermain menjadi lebih menantang dan adil.

- AI juga digunakan dalam permainan untuk memprediksi perilaku pemain dan memberikan rekomendasi bermain yang sesuai.

Dalam Konten Digital Lainnya:

- Dalam platform media sosial, AI digunakan untuk analisis sentimen, mengidentifikasi konten yang melanggar pedoman, dan memberikan rekomendasi berdasarkan preferensi pengguna.
- Dalam seni digital, AI digunakan untuk menciptakan karya seni yang unik, bahkan dalam bentuk lukisan dan patung.
- Di dunia virtual reality (VR) dan augmented reality (AR), AI digunakan untuk menciptakan pengalaman yang lebih imersif dan realistis.

Namun, bersama dengan semua inovasi ini, AI juga menghadirkan pertanyaan etika dan privasi, terutama dalam hal penggunaan data pribadi dan pembuatan konten palsu. Tetapi secara keseluruhan, tren AI dalam hiburan adalah cerita tentang bagaimana teknologi ini telah menghadirkan kemungkinan baru dalam dunia hiburan, membuat pengalaman kita lebih mendalam, lebih interaktif, dan lebih dinamis. Perjalanan ini terus berlanjut, dan kita dapat mengharapkan lebih banyak inovasi menarik di masa depan.

Berikut adalah beberapa contoh perbandingan antara masa sebelum adanya Kecerdasan Buatan (Artificial Intelligence atau AI) dan masa setelah adanya AI dalam bidang hiburan:

Sebelum AI dalam Hiburan:

1. **Produksi Konten Konvensional**: Film, musik, dan permainan video diproduksi dengan metode konvensional yang memakan waktu dan biaya besar. Efek

khusus dan grafik canggih memerlukan tim yang besar dan anggaran yang tinggi.

2. **Pemilihan Konten Manual**: Platform hiburan mengandalkan pemilihan konten manual oleh editor dan kritik untuk menyarankan kepada pemirsa. Rekomendasi berdasarkan preferensi pribadi minim.
3. **Interaksi Pasif**: Pemirsa adalah penonton pasif yang hanya dapat mengonsumsi konten yang telah dibuat. Tidak ada tingkat interaksi yang signifikan.
4. **Produksi Musik Manual**: Komposisi musik, penciptaan lirik, dan rekaman vokal adalah proses kreatif yang sepenuhnya manusia.

Setelah AI dalam Hiburan:

1. **Produksi Konten Lebih Efisien**: AI digunakan dalam produksi film, musik, dan permainan video untuk mempercepat proses dan menghasilkan efek khusus yang canggih dengan biaya yang lebih rendah.
2. **Pemilihan Konten yang Disesuaikan**: Algoritma AI digunakan dalam platform streaming untuk merekomendasikan konten yang sesuai dengan preferensi pengguna, memungkinkan pengalaman menonton yang lebih personal.
3. **Interaksi yang Meningkat**: AI digunakan dalam permainan video untuk membuat karakter yang lebih cerdas dan reaktif, sehingga pengalaman bermain menjadi lebih dinamis. Pengguna juga dapat berinteraksi dengan asisten virtual dalam aplikasi hiburan.
4. **Penciptaan Musik Otomatis**: AI dapat menciptakan musik orisinal dan lirik berdasarkan parameter tertentu, memungkinkan eksperimen kreatif yang lebih besar.
5. **Permainan Video yang Disesuaikan**: Algoritma pembelajaran mesin digunakan dalam permainan video untuk menyesuaikan tingkat kesulitan dengan

kemampuan pemain, sehingga setiap pengalaman bermain menjadi lebih menantang.

6. **Karya Seni Digital**: AI digunakan untuk menciptakan karya seni digital yang unik, bahkan dalam bentuk lukisan dan patung.
7. **Konten Interaktif**: Di dunia virtual reality (VR) dan augmented reality (AR), AI digunakan untuk menciptakan pengalaman yang lebih interaktif dan realistis.
8. **Analisis Sentimen**: AI digunakan dalam media sosial dan platform berbagi video untuk menganalisis sentimen dan mengidentifikasi konten yang melanggar pedoman.

Perbandingan ini mencerminkan bagaimana AI telah mengubah hiburan dari proses produksi hingga pengalaman konsumsi. Dengan kemampuannya untuk memproses data besar, AI membawa hiburan yang lebih personal, efisien, dan interaktif. Meskipun ada banyak manfaat, kita juga perlu mempertimbangkan implikasi etika AI dalam pembuatan konten dan privasi data.

DAFTAR PUSTAKA

Bharadiya JP, Thomas RK, Ahmed F. 2023. Rise of Artificial Intelligence in Business and Industry. *J. Eng. Res. Reports*. 25(3):85–103.doi:10.9734/jerr/2023/v25i3893.

Cardona M, Rodríguez R, Ishmael K. 2023. U.S. Department of Education, Office of Educational Technology, Artificial Intelligence and Future of Teaching and Learning: Insights and Recommendations, Washington, DC, 2023. *Artif. Intell. Futur. Teach. Learn.*(May).

Chowdhary KR. 2020. *Fundamentals of Artificial Intelligence*. New Delhi: Springer India.

Harvey S, Harvey R. 1998. An introduction to artificial intelligence. *Appita J.* 51(1).doi:10.2514/6.1994-294.

Lachman R, Joffe M. 2020. Applications of Artificial Intelligence in Media and Entertainment. *Anal. Futur. Appl. AI, Sensors, Robot. Soc.*:201–220.doi:10.4018/978-1-7998-3499-1.ch012.

Nallamothu PT, Cuthrell KM. 2023. Artificial Intelligence in Health Sector: Current Status and Future Perspectives. *Asian J. Res. Comput. Sci.* 15(4):1–14.doi:10.9734/ajrcos/2023/v15i4325.

Vinayak Pujari. 2022. Artificial Intelligence in Automobiles: an Overview. *Int. J. Eng. Appl. Sci. Technol.* 6(12):194–199.doi:10.33564/ijeast.2022.v06i12.032.

BIODATA PENULIS

Rahmadi Asri, S.Inf., M.Kom.
Dosen Program Studi Teknik Informatika
Fakultas Teknik Universitas Gajah Putih

Penulis lahir di Pante Raya, Kab. Bener Meriah tanggal 28 Agustus 1990. Penulis adalah dosen tetap pada Program Studi Teknik Informatika Fakultas Teknik, Universitas Gajah Putih. Menyelesaikan pendidikan S1 pada Jurusan Teknik Informatika dan melanjutkan S2 pada Jurusan Teknik Informatika Universitas Sumatera Utara. Konsentrasi penulis di bidang Technical Support Komputer, Web Technology, Cloud Computing, dan Data Science. Penulis juga aktif menulis sesuai dengan kepakarnya tersebut. Penulis juga aktif melaksanakan Tridarma Perguruan Tinggi yaitu Pendidikan, Penelitian dan Pengambdian Kepada Masyarakat.

BIODATA PENULIS

Wirawan Istiono, S.Kom., M.Kom.
Dosen Informatika
Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara

Penulis lahir di Jakarta tanggal 13 April 1983. Penulis adalah dosen pada Program Studi Informatika Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 pada Jurusan Teknik Informatika dan melanjutkan S2 pada bidang jurusan yang Teknik Informatika. Pada saat ini, penulis menekuni bidang Penelitian terkait komputer sains, sistem komputer dan teknologi informasi

BIODATA PENULIS

Yusuf Wahyu Setiya Putra, S.Kom., M.Kom.
Dosen Program Studi Sistem Informasi
STMIK Bina Patria Magelang

Penulis lahir di Kota Magelang pada tanggal 2 Mei. Ia Lulus jenjang pendidikan Sarjana pada tahun 2015 dengan masa studi 3 tahun di Program Studi Sistem Informasi. Dan Lulus jenjang Magister pada tahun 2019 di Universitas Amikom Yogyakarta. Berkarir sebagai dosen sejak tahun 2019, ia memulai karirnya di salah satu Perguruan Tinggi Swasta di Kabupaten Kendal hingga tahun 2022. Dan tahun 2022 hingga saat ini ia sudah kembali ke Kota asalnya dan tercatat sebagai dosen tetap pada Program Studi Sistem Informasi di Sekolah Tinggi Manajemen Informatika dan Komputer Bina Patria Magelang. Selain mengajar ia aktif dalam kegiatan tridarma lainnya diantaranya ialah penelitian dan pengabdian.

BIODATA PENULIS

Hermila A. S.SI., M.Pd.
Dosen Program Studi Pendidikan Teknologi Informasi
Fakultas Teknik Universitas Negeri Gorontalo

Penulis lahir di Polewali Mandar tanggal 25 Juli 1993. Penulis adalah dosen tetap pada Program Studi Pendidikan Teknologi Informasi Fakultas Teknik, Universitas Negeri Gorontalo. Menyelesaikan pendidikan S1 pada Jurusan Sistem Informasi STMIK Profesional Makassar dan melanjutkan S2 pada Jurusan Pendidikan Teknologi Kejuruan Kekhususan Pendidikan Teknik Informatika dan Komputer Universita Negeri Makassar. Penulis sekarang mulai menekuni bidang Menulis.

BIODATA PENULIS

Suwito Pomalingo, S.Kom., M.Kom
Dosen Program Studi Informatika
Universitas Multimedia Nusantara

Penulis adalah dosen tetap pada Program Studi Informatika Fakultas Teknik dan Informatika Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 pada Jurusan Teknik Informatika dan melanjutkan S2 pada Program Magister Informatika Universitas Islam Indonesia.

Saat ini sementara menyelesaikan studi Doktoral dengan topik penelitian di bidang Security Science dengan pendekatan Deep Learning. Penulis menekuni bidang keilmuan meliputi Cyber Security, Malware Analytics, Data Science, Machine Learning, dan Deep Learning, selain itu, penulis juga memiliki beberapa serifikasi internasional di bidang Cyber Security, Digital Forensics, Data Science dan Machine Learning.

BIODATA PENULIS

Rudi Sutomo, S.Kom., M.Si., M.Kom.
Dosen Program Studi Sistem Informasi
Fakultas Teknik dan Informatika
Universitas Multimedia Nusantara

Rudi Sutomo memulai karirnya di bidang IT sebagai *IT Lecturer* di tahun 2002 dan mendalami perkembangan Software Engineering di Kampus dan IT Development di bidang Hospitality (Perhotelan), terlibat dalam beberapa Project Pembangunan Aplikasi untuk menunjang kemajuan dan kelancaran sistem IT di Hotel Novotel di Kota Palembang sebagai Asisten IT Manager. Aktif dan giat sebagai Web Developer yang berpengalaman di pembangunan di bidang Aplikasi Perkantoran dan bidang Pengembangan Website terutama Project Web Desain dan Website Development dengan terlibat dalam beberapa pembangunan project aplikasi seperti Aplikasi Visual Basic "*Work Order Multi User*" digunakan untuk seluruh Divisi dalam Hotel Novotel Palembang, Sistem Informasi Akademik (SIA) Kampus Tanri Abeng University dan Project berhubungan dengan PUSMENJAR (Pusat Asesmen dan Pembelajaran) Kemendikbud yaitu Project PUSPENDIK AKSI dan beberapa Project Minor lainnya.

Beliau juga aktif berpatisipasi dalam workshop mengenai Pembuatan website baik di sekolah maupun di masyarakat dan beberapa kali dipercaya sebagai Narasumber salah satunya di

“Pelatihan Pengenalan dan Pembangunan Website” di Sekolah SMKN 43 Jakarta Tahun 2019. Pada Event Penghargaan Website Tingkat Nasional diberi kepercayaan sebagai Juri pada IWA (Indonesia Website Awards) Penilaian periode Tahun 2018 dan 2019.

BIODATA PENULIS

Rahmat Taufik R.L Bau, M.Kom.
Dosen Program Studi Sistem Informasi
Fakultas Teknik Universitas Negeri Gorontalo

Penulis lahir di Gorontalo tanggal 27 Juli 1989. Penulis adalah seorang dosen pada Program Studi Sistem Informasi Fakultas Teknik, Universitas Negeri Gorontalo. Menyelesaikan pendidikan S1 dan S2 pada Jurusan Teknologi Informasi. Penulis mengajar dan meneliti di bidang keahlian analisis dan perancangan. Sejak awal tahun 2023 bergabung di komunitas menulis bab chapter oleh Get Press Indonesia.

BIODATA PENULIS

Jayanti Yusmah Sari, S.T., M.Kom.
Dosen Program Studi Ilmu Komputer
Fakultas Teknologi Informasi Universitas Sembilanbelas
November Kolaka

Penulis lahir di Kendari, Sulawesi Tenggara. Penulis menempuh pendidikan pada SDN 8 Baruga, SMPN 4 Kendari, dan SMAN 4 Kendari. Pada tahun 2008, penulis melanjutkan studi ke jenjang S1 pada Jurusan Teknik Informatika, Fakultas Teknik, Universitas Halu Oleo Kendari. Pada Juli 2013, penulis berhasil diterima di Magister Teknik Informatika Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya, melalui jalur BPPDN (Beasiswa Pendidikan Pascasarjana Dalam Negeri) sebagai calon dosen. Sejak tahun 2019 hingga saat ini, penulis mengabdikan diri sebagai seorang dosen di Prodi Ilmu Komputer Universitas Sembilanbelas November Kolaka. Dan di penghujung tahun 2023 ini, penulis kembali melanjutkan studi doktoral di Prodi Teknologi Pembelajaran, Universitas Negeri Malang dengan Beasiswa Pendidikan Indonesia. Penulis memiliki ketertarikan dalam

bidang digital image processing, computer vision, artificial intelligence dan biometrika. Penulis dapat dihubungi melalui email: jayanti.yusmah.sari@gmail.com.

BIODATA PENULIS

Fahmy Rinanda Saputri, S.T., M.Eng.
Dosen Program Studi Teknik Fisika
Fakultas Teknik dan Informatika Universitas Multimedia Nusantara

Penulis lahir pada bulan Agustus 1993 di Cilacap. Penulis adalah dosen tetap pada Program Studi Teknik Fisika Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 pada Jurusan Teknik Fisika Universitas Gadjah Mada dan melanjutkan studi pada Program Magister Teknik Fisika Universitas Gadjah Mada. Penulis menekuni bidang instrumentasi, fisika bangunan, dan energi baru terbarukan.

BIODATA PENULIS

Melissa Indah Fianty, S.Kom., MMSI
Dosen Program Studi Sistem Informasi Fakultas Teknik dan Informatika Universitas Multimedia Nusantara

Penulis adalah dosen full-time pada Program Studi Sistem Informasi Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 pada Teknik Informatika dan melanjutkan S2 pada Manajemen Sistem Informasi di Bina Nusantara University.

BIODATA PENULIS

Dr. Erick Fernando, S.Kom., M.S.I.
Dosen Program Studi Sistem Informasi
Fakultas Teknik dan Informatik

Penulis saat ini merupakan dosen tetap pada Program Studi Sistem Informasi Fakultas Teknik dan Informatik, Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 pada Jurusan Teknik Informatika dan melanjutkan S2 pada Jurusan Magister Sistem Informasi. Serta Menyelesaikan S3 pada Doktor of Computer Science. Penulis menekuni bidang Basis data, Teknologi Blockchain, Adopsi Teknologi, Big Data, Sistem Informasi Geografis, Emmbeded System Dlll.

BIODATA PENULIS

Dr. Irmawati, S.Kom., MMSI.
Dosen Program Studi Sistem Informasi
Fakultas Teknik dan Informatika
Universitas Multimedia Nusantara

Penulis adalah dosen tetap pada Program Studi Sistem Informasi Fakultas Teknik dan Informatika Universitas Multimedia Nusantara. Menyelesaikan pendidikan S1 dan S2 pada Program Studi Sistem Informasi Universitas Gunadarma dan S3 pada departemen Teknik Elektro Universitas Indonesia. Penulis menekuni bidang keilmuan meliputi *Artificial Intelligence*, *Image Processing, Machine Learning*, dan *Deep Learning*.

BIODATA PENULIS

Yulius Denny Prabowo
Dosen Program Studi Ilmu Komputer

Penulis lahir di Kutoarjo, Kab.Purworejo, Jawa Tengah pada tahun 1982, Saat ini penulis bekerja sebagai Dosen pada Universitas Bina Nusantara. Menyelesaikan pendidikan S1 pada Jurusan Teknik Industri di Universitas Atma Jaya Yogyakarta dan melanjutkan S2 pada Jurusan Teknik Informatika di Universitas Indonesia. Penulis kemudian menyelesaikan studi doktoral pada Program Studi Ilmu Komputer, Universitas Bina Nusantara.

BIODATA PENULIS

Budi Wijaya Rauf, S.Kom., M.Kom.
Dosen Program Studi Sistem dan Teknologi Informasi
Universitas Muhammadiyah Kendari

Penulis lahir di Kendari, Sulawesi tenggara pada tanggal 28 November 1996. Penulis merupakan dosen tetap di Universitas Muhammadiyah Kendari program studi Sistem dan Teknologi Informasi pada jenjang S-1. Penulis menekuni pada bidang kecerdasan buatan dengan fokus kepada pengembangan *machine learning*, *deep learning* dan sekarang sedang mendalami tentang *Generative AI*.

BIODATA PENULIS

Angga Aditya Permana
Dosen Program Studi Informatika Fakultas Teknik dan Informatika Universitas Multimedia Nusantara

Angga Aditya Permana (lahir pada Desember 1989 di Jakarta) seorang dosen full time di bidang *Computer Science* pada Universitas Multimedia Nusantara sejak tahun 2021, fokus penelitian pada kriptografi, steganografi serta keamanan computer, namun pada tahun 2021 sedang mendalami topik bioinformatika dan network science. Angga juga memiliki hobi yaitu touring dan juga bermain bulutangkis, saat ini sedang menjadi mahasiswa program doctoral pada IPB University. Memulai karir pertama kali sebagai Network Enginner tahun 2011 dan memulai profesinya sebagai dosen pada 2013 di kampus swasta yang ada di Jakarta, pernah juga mengajar di kampus negri yang berada di Jakarta dan Tangerang, juga pernah di undang menjadi dosen tamu untuk mengenalkan Bioinformatika di kampus negeri di Jakarta. Terima kasih.