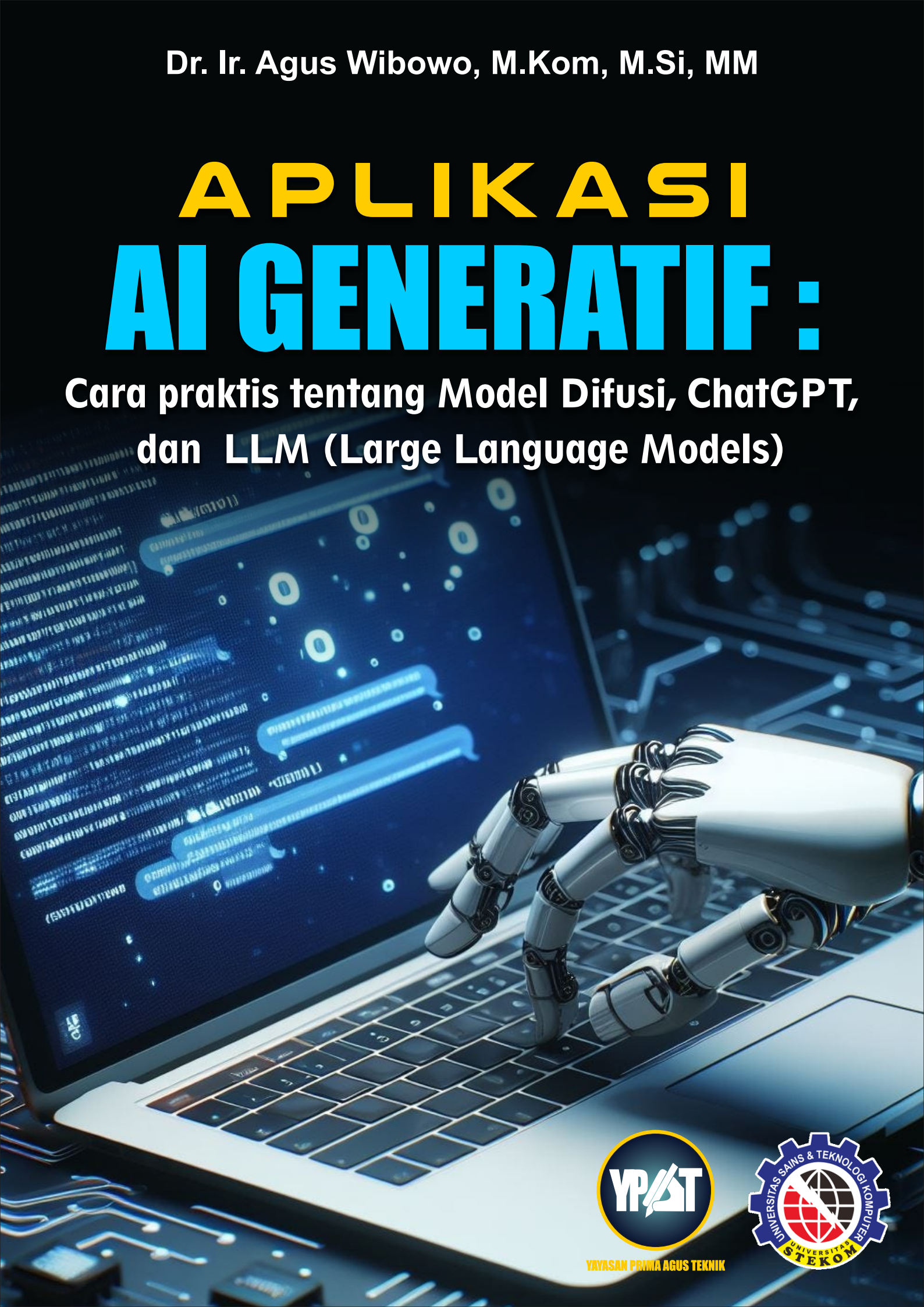
Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

APLIKASI AI GENERATIF :

Cara praktis tentang Model Difusi, ChatGPT, dan LLM (Large Language Models)



YAYASAN PRIMA AGUS TEKNIK

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

APLIKASI AI GENERATIF :

Cara praktis tentang Model Difusi, ChatGPT, dan LLM (Large Language Models)

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-64-5 (PDF)
9 786347 227645

APLIKASI AI GENERATIF :
Cara praktis tentang Model Difusi, ChatGPT, dan LLM (Large Language Models)

Penulis :
Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

ISBN : 978-634-7227-64-5 (PDF)

Editor :
Dr. Joseph Teguh Santoso, S.Kom., M.Kom.

Penyunting :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Anggota IKAPI No: 279 / ALB / JTE / 2023

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. 08122925000
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. 08122925000
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas selesainya penyusunan buku "***APLIKASI AI GENERATIF: Cara praktis tentang Model Difusi, ChatGPT, dan LLM (Large Language Models)***" ini. Buku ini disusun dengan tujuan memberikan pemahaman mendalam sekaligus panduan praktis mengenai konsep-konsep fundamental dan penerapan teknologi AI generatif, terutama Model Difusi, ChatGPT, dan Large Language Models (LLM), yang kini menjadi pilar utama dalam perkembangan kecerdasan buatan modern.

Isi buku terbagi dalam beberapa bab yang mengupas mulai dari pengantar AI generatif, evolusi jaringan saraf tiruan hingga model bahasa besar, detail arsitektur transformer dan LLM, implementasi praktis menggunakan Scikit-LLM, hingga pembahasan teknologi terkini seperti Google Bard dan berbagai model difusi untuk citra. Buku ini juga memuat studi kasus penggunaan ChatGPT di berbagai bidang seperti bisnis, pendidikan, kesehatan, dan lainnya, serta penjelasan tentang LLMOps yang merupakan aspek strategis dalam penerapan LLM di lingkungan perusahaan.

Bab pertama memberikan pendahuluan mengenai pengantar AI generatif, termasuk komponen utama dan domain bidang ini. Bab kedua menjelaskan evolusi jaringan neural tiruan menjadi model bahasa besar, mencakup proses pemrosesan bahasa alami, model probabilistik, serta pengembangan jaringan saraf dan transformer yang mendasari LLM. Pada bab ketiga, fokus beralih ke arsitektur dan kekuatan model bahasa besar, termasuk teknik transformer dan arsitektur encoder-decoder. Bab keempat membahas secara khusus arsitektur ChatGPT dan teknologi di baliknya, termasuk evolusinya, strategi menangani bias, serta aspek etis. Bab kelima memperkenalkan Google Bard dan membedakannya dari ChatGPT. Bab keenam mengulas implementasi LLM dengan alat seperti Scikit-LLM dan teknik klasifikasi teks zero-shot. Pada bab ketujuh, bahasan berpusat pada penerapan LLM di perusahaan dan strategi LLMOps, termasuk fine-tuning dan teknologi terkini. Bab kedelapan membahas teknologi model difusi dan AI generatif untuk citra, termasuk teknologi di balik DALL-E 2, Midjourney, dan perbandingannya. Bab terakhir mengulas berbagai kasus penggunaan ChatGPT, mulai dari bisnis, layanan pelanggan, pembuatan konten, hingga pendidikan dan hukum. Keseluruhan buku ini dilengkapi daftar pustaka yang memberikan referensi tambahan untuk memperdalam pemahaman.

Semoga buku ini dapat menjadi referensi yang komprehensif dan bermanfaat bagi akademisi, praktisi, mahasiswa, maupun pembaca umum yang ingin memahami dan mengaplikasikan AI generatif secara tepat dan efektif. Ucapan terima kasih kami sampaikan kepada semua pihak yang telah mendukung proses penulisan dan penerbitan buku ini. Akhir kata, semoga karya ini dapat memberikan kontribusi positif dalam pengembangan ilmu pengetahuan dan teknologi AI di Indonesia maupun dunia.

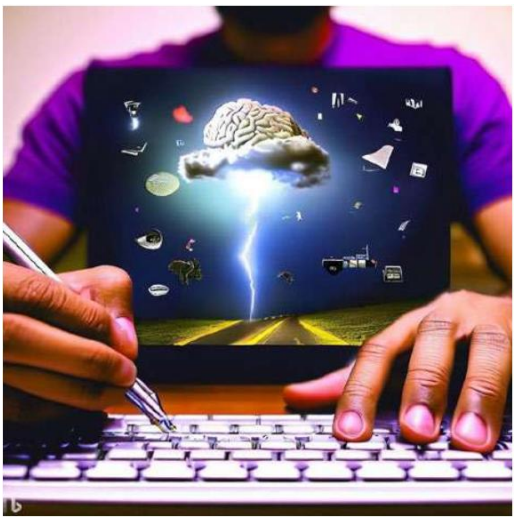
Semarang, November 2025
Penulis

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

DAFTAR ISI

BAB 1
PENGANTAR AI GENERATIF

Pernahkah Anda membayangkan bahwa hanya dengan membayangkan sesuatu dan mengetik, sebuah gambar atau video dapat dihasilkan? Betapa menariknya itu? Konsep ini, yang dulunya hanya ada dalam dunia fiksi ilmiah, telah menjadi kenyataan nyata di dunia modern kita. Gagasan bahwa pikiran dan kata-kata kita dapat diubah menjadi konten visual tidak hanya memikat, tetapi juga merupakan bukti inovasi dan kreativitas manusia.



Gambar 1-1. Gambar yang dihasilkan mesin berdasarkan input teks

Bahkan sebagai ilmuwan data, banyak dari kita tidak pernah mengantisipasi bahwa AI dapat mencapai titik di mana ia dapat menghasilkan teks untuk kasus penggunaan tertentu. Kesulitan yang kita hadapi dalam menulis kode atau berjam-jam yang dihabiskan untuk mencari solusi yang tepat di Google dulunya merupakan tantangan yang umum. Namun, lanskap teknologi telah berubah secara dramatis, dan tugas-tugas yang melelahkan tersebut telah menjadi peninggalan masa lalu.

Bagaimana ini bisa terjadi? Jawabannya terletak pada kemajuan pesat dalam pembelajaran mendalam dan pemrosesan bahasa alami (NLP). Lompatan teknologi ini telah membuka jalan bagi AI generatif, sebuah bidang yang memanfaatkan kekuatan algoritma untuk menerjemahkan pikiran menjadi representasi visual atau mengotomatiskan pembuatan kode yang kompleks. Berkat perkembangan ini, kita kini mengalami masa depan di mana imajinasi dan inovasi berpadu, mengubah hal yang dulunya tak terpikirkan menjadi kenyataan sehari-hari.

Jadi, Apa Itu AI Generatif?

AI generatif mengacu pada cabang kecerdasan buatan yang berfokus pada pembuatan model dan algoritma yang mampu menghasilkan konten baru dan orisinal, seperti gambar, teks, musik, dan bahkan video. Tidak seperti model AI tradisional yang dilatih untuk melakukan

tugas-tugas tertentu, model AI generatif bertujuan untuk mempelajari dan meniru pola dari data yang ada untuk menghasilkan keluaran baru yang unik.

AI generatif memiliki beragam aplikasi. Misalnya, dalam visi komputer, model generatif dapat menghasilkan gambar yang realistis, membuat variasi dari gambar yang sudah ada, atau bahkan melengkapi bagian gambar yang hilang. Dalam pemrosesan bahasa alami, model generatif dapat digunakan untuk penerjemahan bahasa, sintesis teks, atau bahkan untuk menciptakan agen percakapan yang menghasilkan respons seperti manusia. Di luar contoh-contoh ini, AI generatif dapat melakukan pembuatan karya seni, augmentasi data, dan bahkan menghasilkan citra medis sintetis untuk penelitian dan diagnosis. Ini adalah alat yang ampuh dan kreatif yang memungkinkan kita menjelajahi batas-batas kemungkinan dalam visi komputer.

Namun, perlu dicatat bahwa AI generatif juga menimbulkan masalah etika. Kemampuan untuk menghasilkan konten palsu yang realistis dan meyakinkan dapat disalahgunakan untuk tujuan jahat, seperti membuat deepfake atau menyebarkan disinformasi. Akibatnya, terdapat penelitian dan pengembangan teknik yang sedang berlangsung untuk mendeteksi dan memitigasi potensi dampak negatif AI generatif.

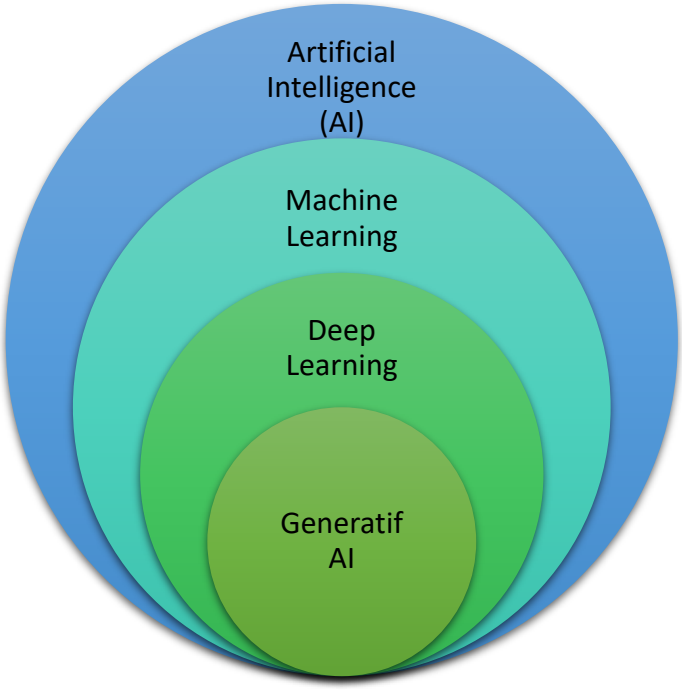
Secara keseluruhan, AI generatif sangat menjanjikan untuk berbagai aplikasi kreatif dan praktis, serta untuk menghasilkan konten baru dan unik. AI generatif terus menjadi bidang penelitian dan pengembangan yang aktif, mendorong batas-batas kemampuan mesin untuk menciptakan dan meningkatkan kreativitas manusia dengan cara-cara baru dan menarik.

1.1 KOMPONEN AI

Komponen-komponen utama dalam kecerdasan buatan (AI) meliputi:

- ***Kecerdasan Buatan (AI):*** Ini adalah disiplin ilmu pembelajaran mesin yang lebih luas untuk melakukan tugas-tugas yang biasanya membutuhkan kecerdasan manusia.
- ***Pembelajaran Mesin (ML):*** Sebuah subset dari AI, ML melibatkan algoritma yang memungkinkan komputer untuk belajar dari data, alih-alih diprogram secara eksplisit untuk melakukannya.
- ***Pembelajaran Mendalam (DL):*** Sebuah subset khusus dari ML, pembelajaran mendalam melibatkan jaringan saraf dengan tiga lapisan atau lebih yang dapat menganalisis berbagai faktor dari suatu dataset.
- ***AI Generatif:*** Sebuah subset lanjutan dari AI dan DL, AI generatif berfokus pada penciptaan keluaran baru dan unik. AI ini melampaui cakupan analisis data dan menciptakan kreasi baru berdasarkan pola yang dipelajari.

Secara hierarki, dapat digambarkan bahwa AI adalah payung besar yang mencakup ML, yang di dalamnya terdapat DL, lalu AI generatif merupakan cabang lanjutan dari DL. Gambar 1-2 menjelaskan bagaimana AI generatif merupakan komponen AI.



Gambar 1.2. AI dan komponen-komponennya

1.2 DOMAIN AI GENERATIF

Mari kita telaah lebih dalam domain AI generatif secara detail, termasuk apa itu, cara kerjanya, dan beberapa aplikasi praktisnya.

Pembuatan Teks

- *Apa Itu:* Pembuatan teks melibatkan penggunaan model AI untuk membuat teks mirip manusia berdasarkan perintah input.
- *Cara Kerjanya:* Model seperti GPT-3 menggunakan arsitektur Transformer. Model ini telah dilatih sebelumnya pada kumpulan data teks yang luas untuk mempelajari tata bahasa, konteks, dan semantik. Dengan sebuah perintah, model tersebut memprediksi kata atau frasa berikutnya berdasarkan pola yang telah dipelajari.
- *Aplikasi:* Pembuatan teks diterapkan dalam pembuatan konten, chatbot, dan pembuatan kode. Bisnis dapat menggunakannya untuk membuat postingan blog, mengotomatiskan respons dukungan pelanggan, dan bahkan menghasilkan cuplikan kode. Pemikir strategis dapat memanfaatkannya untuk menyusun salinan pemasaran dengan cepat atau membuat pesan yang dipersonalisasi untuk pelanggan.

Pembuatan Gambar

- *Apa Itu:* Pembuatan gambar melibatkan penggunaan berbagai model pembelajaran mendalam untuk membuat gambar yang tampak nyata.
- *Cara Kerjanya:* GAN terdiri dari generator (yang menghasilkan gambar) dan diskriminator (yang menentukan asli vs. palsu). Keduanya bersaing dalam siklus umpan balik, dengan generator semakin baik dalam menghasilkan gambar yang tidak dapat dibedakan oleh diskriminator dari gambar asli.

- *Aplikasi:* Model-model ini digunakan dalam seni, desain, dan visualisasi produk. Bisnis dapat menghasilkan tiruan produk untuk iklan, menciptakan karya seni unik untuk branding, atau bahkan menghasilkan wajah untuk beragam materi pemasaran.

Pembuatan Audio

- *Apa Itu:* Pembuatan audio melibatkan AI yang menciptakan musik, suara, atau bahkan suara manusia.
- *Cara Kerjanya:* Model seperti WaveGAN menganalisis dan meniru bentuk gelombang audio. Model text-to-speech seperti Tacotron 2 menggunakan teks input untuk menghasilkan suara. Model ini dilatih pada kumpulan data besar untuk menangkap nuansa suara.
- *Aplikasi:* Musik yang dihasilkan AI dapat digunakan dalam iklan, video, atau sebagai trek latar. Merek dapat menciptakan jingle yang menarik atau efek suara khusus untuk kampanye pemasaran. Teknologi text-to-speech dapat mengotomatiskan sulih suara untuk iklan atau interaksi layanan pelanggan. Secara strategis, bisnis dapat menggunakan audio yang dihasilkan AI untuk meningkatkan pengenalan merek dan penceritaan.

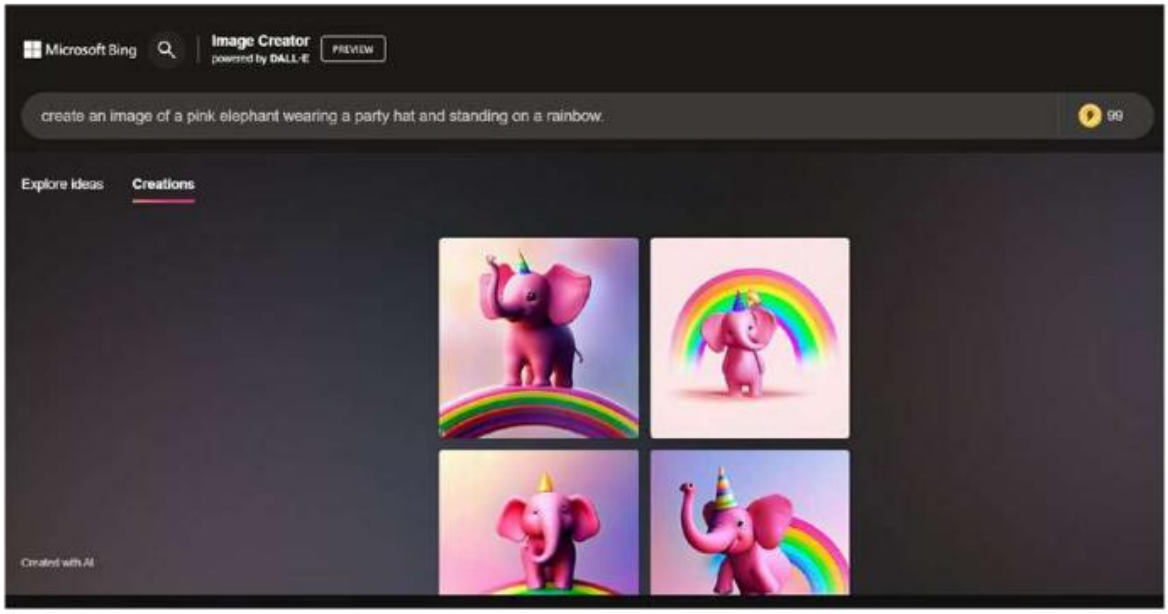
Pembuatan Video

- *Apa Itu:* Pembuatan video melibatkan AI yang menciptakan video, seringkali dengan menggabungkan visual yang ada atau melengkapi bagian yang hilang.
- *Cara Kerjanya:* Pembuatan video bersifat kompleks karena sifat video yang temporal. Beberapa model menggunakan deskripsi teks untuk menghasilkan adegan, sementara yang lain memprediksi frame yang hilang dalam video.
- *Aplikasi:* Video yang dihasilkan AI dapat digunakan dalam pesan yang dipersonalisasi, iklan dinamis, atau bahkan pemasaran konten. Merek dapat membuat iklan video unik yang disesuaikan dengan segmen pelanggan tertentu. Penerapan yang cermat dapat menghasilkan konten video yang efisien dan beradaptasi dengan tren pemasaran.

Menghasilkan Gambar

Microsoft Bing Image Creator adalah alat AI generatif yang menggunakan kecerdasan buatan untuk membuat gambar berdasarkan deskripsi teks Anda.

Untuk menggunakan Bing Image Creator, Anda cukup mengetik deskripsi gambar yang ingin Anda buat ke dalam kotak teks. Kita akan menggunakan contoh yang sama yang disebutkan sebelumnya dalam menghasilkan gambar realistis. "Buat gambar gajah merah muda mengenakan topi pesta dan berdiri di atas pelangi." Bing Image Creator kemudian akan menghasilkan gambar berdasarkan deskripsi Anda. Gambar 1-3 menunjukkan keluaran Microsoft Bing.



Gambar 1.3. Keluaran Microsoft Bing

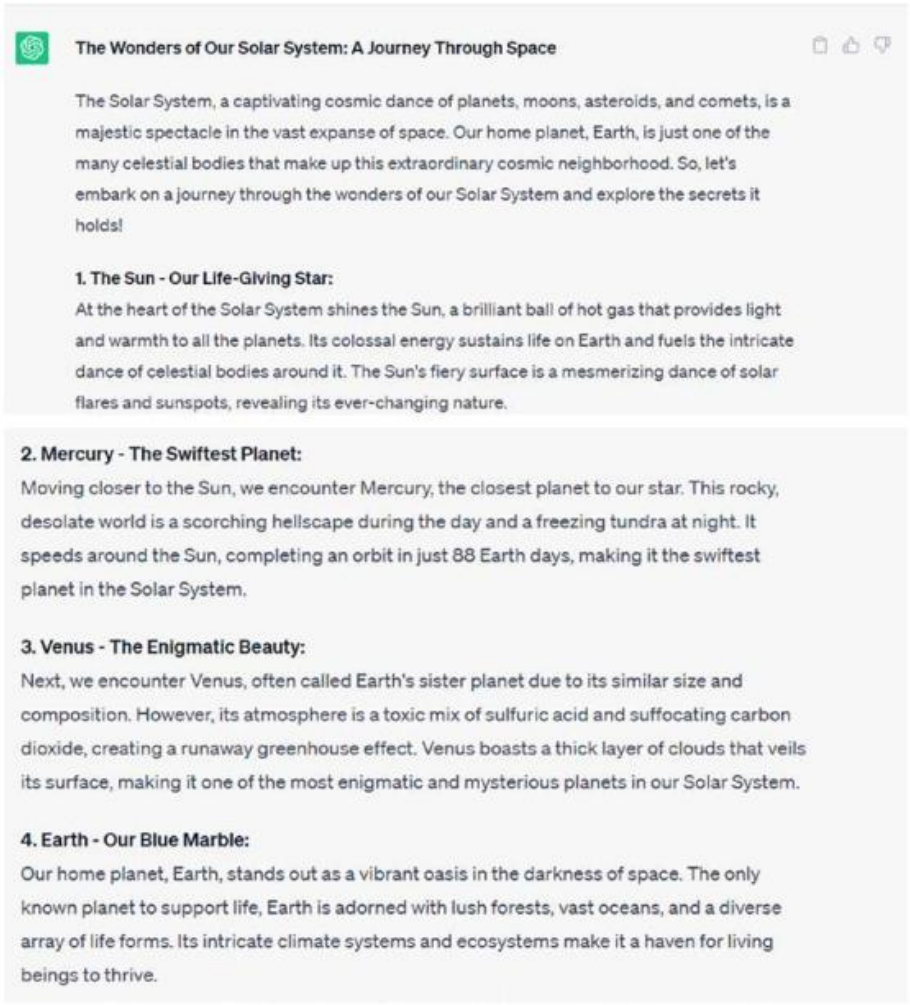
Menghasilkan Teks

Mari kita gunakan ChatGPT untuk menghasilkan teks. Chatbot ini berbasis model bahasa besar yang dikembangkan oleh OpenAI dan diluncurkan pada November 2022. ChatGPT dilatih dengan pembelajaran penguatan melalui umpan balik manusia dan model penghargaan yang memeringkat respons terbaik. Umpan balik ini membantu melengkapi ChatGPT dengan pembelajaran mesin untuk meningkatkan respons di masa mendatang.
ChatGPT dapat digunakan untuk berbagai tujuan, termasuk:

- Melakukan percakapan dengan pengguna
- Menjawab pertanyaan
- Menghasilkan teks
- Menerjemahkan bahasa
- Menulis berbagai jenis konten kreatif. ChatGPT dapat diakses daring di

Untuk menggunakan ChatGPT, cukup ketik deskripsi yang diinginkan di kotak teks. Untuk membuat konten tentang tata surya kita. Gambar 1-4 menunjukkan hasil keluaran ChatGPT.

ChatGPT atau alat lainnya masih dalam tahap pengembangan, tetapi telah mampu melakukan berbagai jenis tugas. Seiring dengan perkembangannya, ChatGPT akan menjadi semakin canggih dan serbaguna.



The Wonders of Our Solar System: A Journey Through Space

The Solar System, a captivating cosmic dance of planets, moons, asteroids, and comets, is a majestic spectacle in the vast expanse of space. Our home planet, Earth, is just one of the many celestial bodies that make up this extraordinary cosmic neighborhood. So, let's embark on a journey through the wonders of our Solar System and explore the secrets it holds!

1. The Sun - Our Life-Giving Star:
At the heart of the Solar System shines the Sun, a brilliant ball of hot gas that provides light and warmth to all the planets. Its colossal energy sustains life on Earth and fuels the intricate dance of celestial bodies around it. The Sun's fiery surface is a mesmerizing dance of solar flares and sunspots, revealing its ever-changing nature.

2. Mercury - The Swiftest Planet:
Moving closer to the Sun, we encounter Mercury, the closest planet to our star. This rocky, desolate world is a scorching hellscape during the day and a freezing tundra at night. It speeds around the Sun, completing an orbit in just 88 Earth days, making it the swiftest planet in the Solar System.

3. Venus - The Enigmatic Beauty:
Next, we encounter Venus, often called Earth's sister planet due to its similar size and composition. However, its atmosphere is a toxic mix of sulfuric acid and suffocating carbon dioxide, creating a runaway greenhouse effect. Venus boasts a thick layer of clouds that veils its surface, making it one of the most enigmatic and mysterious planets in our Solar System.

4. Earth - Our Blue Marble:
Our home planet, Earth, stands out as a vibrant oasis in the darkness of space. The only known planet to support life, Earth is adorned with lush forests, vast oceans, and a diverse array of life forms. Its intricate climate systems and ecosystems make it a haven for living beings to thrive.

Gambar 1.4. Output ChatGPT

AI Generatif: Pemain Saat Ini dan Modelnya

AI generatif adalah bidang yang berkembang pesat dengan potensi untuk merevolusi banyak industri. Gambar 1.5 menunjukkan beberapa pemain saat ini di bidang AI generatif.

Gambar 1.5. Hasil ChatGPT

Secara singkat, mari kita bahas beberapa di antaranya:

- ***OpenAI:*** OpenAI adalah perusahaan riset AI generatif yang didirikan oleh Elon Musk, Sam Altman, dan lainnya. OpenAI telah mengembangkan beberapa model AI generatif tercanggih di dunia, termasuk GPT-4 dan DALL-E 2.
- ***GPT-4:*** GPT-4 adalah model bahasa yang luas yang dapat menghasilkan teks, menerjemahkan bahasa, menulis berbagai jenis konten kreatif, dan menjawab pertanyaan Anda dengan cara yang informatif.
- ***DALL-E 2:*** DALL-E 2 adalah model AI generatif yang dapat menciptakan gambar realistis dari deskripsi teks.
- ***DeepMind:*** DeepMind adalah perusahaan kecerdasan buatan asal Inggris yang diakuisisi oleh Google pada tahun 2014. DeepMind telah mengembangkan beberapa model AI generatif, termasuk AlphaFold, yang dapat memprediksi struktur protein, dan Gato, yang dapat melakukan berbagai tugas, termasuk bermain gim Atari, mengendalikan lengan robot, dan menulis berbagai jenis konten kreatif.
- ***Anthropic:*** Anthropic adalah perusahaan yang mengembangkan model AI generatif untuk digunakan di berbagai industri, termasuk layanan kesehatan, keuangan, dan manufaktur. Model-model Anthropic dilatih pada kumpulan data dunia nyata yang masif, yang memungkinkannya menghasilkan keluaran yang realistis dan akurat.
- ***Synthesia:*** Synthesia adalah perusahaan yang berspesialisasi dalam menciptakan media sintetis yang realistis, seperti video dan rekaman audio. Teknologi Synthesia dapat digunakan untuk membuat avatar yang dapat berbicara, memberi isyarat, dan bahkan melakukan sinkronisasi bibir dengan input audio apa pun.
- ***RealSpeaker***: RealSpeaker adalah model AI generatif yang dapat digunakan untuk menciptakan suara sintetis yang realistis.
- ***Natural Video:*** Natural Video adalah model AI generatif yang dapat digunakan untuk membuat video sintetis yang realistis.
- ***RunwayML:*** RunwayML adalah platform yang memudahkan bisnis untuk membangun dan menerapkan model AI generatif. RunwayML menyediakan berbagai alat dan sumber daya untuk membantu bisnis mengumpulkan data, melatih model, dan mengevaluasi hasil.
- ***Runway Studio:*** Runway Studio adalah platform berbasis cloud yang memungkinkan bisnis membangun dan menerapkan model AI generatif tanpa perlu pengalaman coding.
- ***Runway API:*** Runway API adalah serangkaian API yang memungkinkan bisnis mengintegrasikan AI generatif ke dalam aplikasi mereka.
- ***Midjourney:*** Midjourney adalah model AI generatif yang dapat digunakan untuk membuat gambar, video, dan teks yang realistis. Midjourney masih dalam tahap pengembangan, tetapi telah digunakan untuk menghasilkan beberapa hasil yang mengesankan.

Ini hanyalah beberapa dari sekian banyak perusahaan yang sedang mengembangkan AI generatif. Seiring dengan terus berkembangnya bidang ini, kita dapat melihat lebih banyak inovasi dan disrupsi di tahun-tahun mendatang.

Aplikasi AI Generatif

AI Generatif menawarkan beragam aplikasi di berbagai industri. Berikut beberapa aplikasi utamanya:

1. Pembuatan Konten:
    - Pembuatan Teks: Mengotomatiskan postingan blog, pembaruan media sosial, dan artikel.
    - Pembuatan Gambar: Membuat visual khusus untuk kampanye pemasaran dan iklan.
    - Pembuatan Video: Membuat pesan video yang dipersonalisasi dan iklan dinamis.
2. Desain dan Kreativitas:
    - Pembuatan Seni: Membuat karya seni, ilustrasi, dan desain yang unik.
    - Desain Mode: Mendesain pola pakaian dan aksesori.
    - Desain Produk: Membuat prototipe dan mock-up.
3. Hiburan dan Media:
    - Komposisi Musik: Membuat trek musik dan soundscape orisinal.
    - Film dan Animasi: Mendesain karakter, adegan, dan animasi.
    - Bercerita: Mengembangkan narasi dan alur cerita interaktif.
4. Pemasaran dan Periklanan:
    - Personalisasi: Menyusun pesan dan rekomendasi yang disesuaikan untuk pelanggan.
    - Branding: Mendesain logo, kemasan, dan elemen identitas visual.
    - Kampanye Iklan: Mengembangkan iklan yang dinamis dan menarik.
5. Permainan:
    - Membangun Dunia: Menciptakan lingkungan, medan, dan lanskap dalam permainan.
    - Desain Karakter: Menciptakan karakter dalam permainan yang beragam dan unik.
    - Konten Prosedural: Menciptakan level, misi, dan tantangan.
6. Kesehatan dan Kedokteran:
    - Penemuan Obat: Merancang molekul dan senyawa baru.
    - Pencitraan Medis: Meningkatkan dan merekonstruksi citra medis.
    - Pengobatan yang Dipersonalisasi: Menyesuaikan rencana perawatan berdasarkan data pasien.
7. Penerjemahan Bahasa:
    - Penerjemahan Real-Time: Memungkinkan penerjemahan instan bahasa lisan atau tulisan.
    - Subtitel dan Lokalisasi: Menghasilkan subtitel untuk video secara otomatis.
8. Layanan Pelanggan:

- Chatbot: Menciptakan agen percakapan untuk dukungan pelanggan.
- Asisten Suara: Menyediakan bantuan berbasis suara untuk pertanyaan dan tugas.

9. Pendidikan dan Pelatihan:
    - Pembelajaran Interaktif: Mengembangkan materi pembelajaran adaptif.
    - Simulasi: Membuat skenario pelatihan dan simulasi yang realistis.
10. Arsitektur dan Desain:
    - Desain Bangunan: Membuat tata letak dan desain arsitektur.
    - Perencanaan Kota: Merancang lanskap kota dan tata ruang kota.

Kesimpulan

Bab ini berfokus pada AI generatif, sebuah domain yang berkembang pesat dalam kecerdasan buatan yang berspesialisasi dalam menciptakan konten baru dan unik seperti teks, gambar, audio, dan video. Dibangun berdasarkan kemajuan dalam pembelajaran mendalam dan pemrosesan bahasa alami (NLP), model-model ini memiliki berbagai aplikasi, termasuk pembuatan konten, desain, hiburan, layanan kesehatan, dan layanan pelanggan. Khususnya, AI generatif juga menimbulkan masalah etika, terutama dalam menciptakan deepfake atau menyebarkan disinformasi.

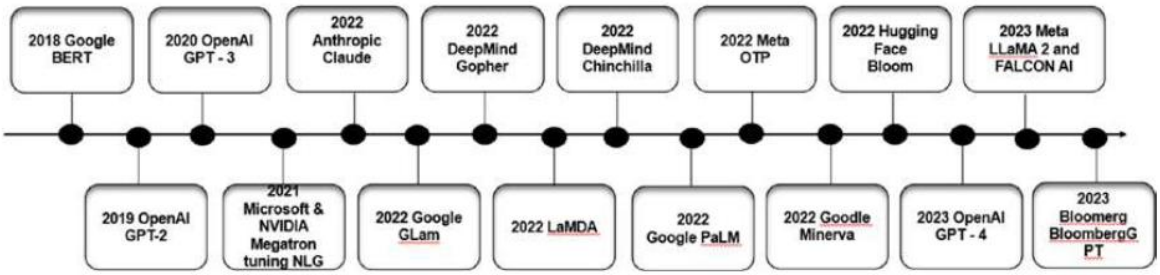
Bab ini memberikan kajian mendalam tentang berbagai domain AI generatif—pembuatan teks, gambar, audio, dan video—yang merinci cara kerja dan aplikasi praktisnya. Bab ini juga membahas beberapa pemain kunci dalam industri ini, seperti OpenAI, DeepMind, dan Synthesia, di antara yang lainnya. Terakhir, bab ini menguraikan beragam aplikasi di berbagai industri.

BAB 2
EVOLUSI JARINGAN SARAF TIRUAN MENJADI MODEL BAHASA BESAR

Selama beberapa dekade terakhir, model bahasa telah mengalami kemajuan yang signifikan. Awalnya, model bahasa dasar digunakan untuk tugas-tugas seperti pengenalan suara, penerjemahan mesin, dan pengambilan informasi. Model-model awal ini dibangun menggunakan metode statistik, seperti model n-gram dan model Markov tersembunyi. Meskipun bermanfaat, model-model ini memiliki keterbatasan dalam hal akurasi dan skalabilitas.

Dengan diperkenalkannya pembelajaran mendalam, jaringan saraf tiruan menjadi lebih populer untuk tugas-tugas pemodelan bahasa. Di antaranya, jaringan saraf tiruan rekuren (RNN) dan jaringan memori jangka pendek panjang (LSTM) muncul sebagai pilihan yang sangat efektif. Model-model ini unggul dalam menangkap hubungan sekuensial dalam data linguistik dan menghasilkan keluaran yang koheren.

Baru-baru ini, pendekatan berbasis atensi, yang dicontohkan oleh arsitektur Transformer, telah mendapatkan perhatian yang cukup besar. Model-model ini menghasilkan keluaran dengan berfokus pada segmen-segmen tertentu dari urutan masukan, menggunakan teknik atensi diri. Keberhasilannya telah ditunjukkan di berbagai tugas pemrosesan bahasa alami, termasuk pemodelan bahasa. Gambar 2-1 menunjukkan tonggak penting dan kemajuan dalam evolusi model bahasa.



Gambar 2-1. Evolusi model bahasa

Sebelum membahas evolusi secara lebih rinci, mari kita jelajahi pemrosesan bahasa alami.

2.1 PEMROSESAN BAHASA ALAMI

Pemrosesan bahasa alami (NLP) adalah subbidang kecerdasan buatan (AI) dan linguistik komputasional yang berfokus pada kemampuan komputer untuk memahami, menafsirkan, dan menghasilkan bahasa manusia. NLP bertujuan untuk menjembatani kesenjangan antara komunikasi manusia dan pemahaman mesin, yang memungkinkan komputer untuk memproses dan memperoleh makna dari data tekstual. NLP memainkan

peran penting dalam berbagai aplikasi, termasuk penerjemahan bahasa, analisis sentimen, chatbot, asisten suara, peringkasan teks, dan banyak lagi.

Kemajuan terbaru dalam NLP didorong oleh teknik pembelajaran mendalam, terutama penggunaan model berbasis Transformer seperti BERT (Bidirectional Encoder Representations from Transformers) dan GPT (Generative Pre-trained Transformer). Model-model ini memanfaatkan pra-pelatihan skala besar pada sejumlah besar data teks dan dapat disesuaikan untuk tugas-tugas NLP tertentu, mencapai kinerja mutakhir di berbagai aplikasi. NLP terus berkembang pesat, dengan penelitian dan pengembangan berkelanjutan yang bertujuan untuk meningkatkan pemahaman bahasa, pembentukan, dan interaksi antara mesin dan manusia. Seiring dengan peningkatan kemampuan NLP, NLP berpotensi merevolusi cara kita berinteraksi dengan teknologi dan memungkinkan komunikasi manusia-komputer yang lebih alami dan lancar.

Tokenisasi

Tokenisasi adalah proses memecah teks menjadi kata-kata atau token individual. Proses ini membantu dalam segmentasi teks dan analisisnya pada tingkat yang lebih terperinci. Contoh:

Input: "Saya suka coding dengan Python"
Tokenisasi: ["Saya", "Suka", "coding", ","dengan", "Python"]

N-gram

Dalam pemrosesan bahasa alami (NLP), n-gram merupakan teknik yang ampuh dan banyak digunakan untuk mengekstraksi informasi kontekstual dari data teks. N-gram pada dasarnya adalah urutan n item yang bersebelahan, di mana item tersebut dapat berupa kata, karakter, atau bahkan fonem, tergantung pada konteksnya. Nilai "n" dalam n-gram menentukan jumlah item yang berurutan dalam urutan tersebut. N-gram yang umum digunakan meliputi unigram (1-gram), bigram (2-gram), trigram (3-gram), dan seterusnya:

1. *Unigram (1-gram):*
   Unigram adalah kata-kata tunggal dalam sebuah teks. Unigram mewakili token atau unit makna individual dalam teks.
   Contoh:
   Input: “Saya suka pemrosesan bahasa alami.”
   Unigram: [“Saya”, “suka”, “pemrosesan”, “bahasa”, “alami”, “.”]
2. *Bigram (2-gram):*
   Bigram terdiri dari dua kata berurutan dalam sebuah teks. Bigram memberikan gambaran tentang pasangan kata dan hubungan antara kata-kata yang berdekatan.
   Contoh:
   Input: “Saya suka pemrosesan bahasa alami.”
   Bigram: [(“Saya”, “suka”), (“suka”, “pemrosesan”), (“pemrosesan”, “bahasa”), (“bahasa”, “alami”), (“alami”, “.”)]
3. *Trigram (3-gram):*
   Trigram adalah tiga kata berurutan dalam sebuah teks. Trigram menangkap lebih banyak konteks dan memberikan wawasan tentang triplet kata.

Contoh:

Input: “Saya suka pemrosesan bahasa alami.”

Trigram: [(“Saya”, “suka”, “pemrosesan”), (“suka”, “pemrosesan”, “bahasa”), (“pemrosesan”, “bahasa”, “alami”), (“bahasa”, “alami”, “.”)]

4. *N-gram dalam Pemodelan Bahasa:*
   Dalam tugas pemodelan bahasa, n-gram digunakan untuk memperkirakan probabilitas sebuah kata berdasarkan konteksnya. Misalnya, dengan bigram, kita dapat memperkirakan kemungkinan sebuah kata berdasarkan kata sebelumnya.
5. *N-gram dalam Klasifikasi Teks:*
   N-gram berguna dalam tugas klasifikasi teks, seperti analisis sentimen. Dengan mempertimbangkan frekuensi n-gram dalam teks positif dan negatif, pengklasifikasi dapat mempelajari fitur pembeda dari setiap kelas.
6. *Keterbatasan n-gram:*
   Meskipun n-gram ampuh dalam menangkap konteks lokal, n-gram dapat kehilangan konteks global. Misalnya, bigram mungkin tidak cukup untuk memahami arti sebuah kalimat jika beberapa kata memiliki ketergantungan yang kuat pada kata lain yang letaknya lebih jauh.
7. *Menangani Kata di Luar Kosakata (OOV):*
   Saat menggunakan n-gram, penting untuk menangani kata di luar kosakata (kata yang tidak terlihat selama pelatihan). Teknik seperti menambahkan token khusus untuk kata yang tidak diketahui atau menggunakan n-gram tingkat karakter dapat digunakan.
8. Pemulusan:
   Model n-gram mungkin mengalami kelangkaan data, terutama saat menangani n-gram orde tinggi. Teknik pemulusan seperti pemulusan Laplace (tambah satu) atau pemulusan Good-Turing dapat membantu mengatasi masalah ini.

N-gram adalah alat yang berharga dalam NLP untuk menangkap konteks lokal dan mengekstraksi fitur yang bermakna dari data teks. N-gram memiliki berbagai aplikasi dalam pemodelan bahasa, klasifikasi teks, pengambilan informasi, dan banyak lagi. Sementara n-gram memberikan wawasan berharga tentang struktur dan konteks teks, n-gram harus digunakan bersama dengan teknik NLP lainnya untuk membangun model yang kuat dan akurat.

Language Representation and Embeddings

Representasi dan embedding bahasa merupakan konsep fundamental dalam pemrosesan bahasa alami (NLP) yang melibatkan transformasi kata atau kalimat menjadi vektor numerik. Representasi numerik ini memungkinkan komputer untuk memahami dan memproses bahasa manusia, sehingga memudahkan penerapan algoritma pembelajaran mesin pada tugas-tugas NLP. Mari kita bahas representasi dan embedding bahasa secara lebih detail. Word2Vec dan GloVe merupakan teknik populer yang digunakan untuk embedding kata, sebuah proses merepresentasikan kata sebagai vektor padat dalam ruang vektor berdimensi tinggi. Embedding kata ini menangkap hubungan semantik antar kata dan banyak digunakan dalam tugas-tugas pemrosesan bahasa alami.

Word2Vec

Word2Vec adalah keluarga model penyisipan kata yang diperkenalkan oleh Mikolov dkk. pada tahun 2013. Model ini terdiri dari dua arsitektur utama: continuous bag of words (CBOW) dan skip-gram:

1. ***CBOW:*** Model CBOW memprediksi kata target berdasarkan kata konteksnya. Model ini mengambil sekumpulan kata konteks sebagai input dan mencoba memprediksi kata target di tengah konteks. Model ini efisien dan dapat menangani beberapa kata konteks sekaligus.
2. ***Skip-gram:*** Model skip-gram melakukan kebalikan dari CBOW. Model ini mengambil kata target sebagai input dan mencoba memprediksi kata konteks di sekitarnya. Skip-gram berguna untuk menangkap hubungan kata dan dikenal berkinerja lebih baik pada kata-kata langka.

Word2Vec menggunakan jaringan saraf tiruan dangkal dengan satu lapisan tersembunyi untuk mempelajari penyisipan kata. Penyematan yang dipelajari menempatkan kata-kata yang secara semantik serupa lebih dekat satu sama lain dalam ruang vektor.

GloVe (Vektor Global untuk Representasi Kata)

GloVe adalah teknik penyisipan kata populer lainnya yang diperkenalkan oleh Pennington dkk. pada tahun 2014. Tidak seperti Word2Vec, GloVe menggunakan matriks ko-okurensi pasangan kata untuk mempelajari penyisipan kata. Matriks ko-okurensi ini merepresentasikan seberapa sering dua kata muncul bersamaan dalam korpus tertentu.

GloVe bertujuan untuk memfaktorkan matriks ko-okurensi ini untuk mendapatkan penyisipan kata yang menangkap hubungan global kata-ke-kata di seluruh korpus. Teknik ini memanfaatkan informasi konteks global dan lokal untuk menciptakan representasi kata yang lebih bermakna. Sekarang, mari kita lanjutkan evolusi jaringan saraf tiruan ke LLMS secara detail.

2.2 MODEL PROBABILISTIK

Model probabilistik n-gram adalah pendekatan yang sederhana dan banyak digunakan untuk pemodelan bahasa dalam pemrosesan bahasa alami (NLP). Model ini memperkirakan probabilitas sebuah kata berdasarkan n-1 kata sebelumnya dalam suatu urutan. "N" dalam n-gram merepresentasikan jumlah kata yang dianggap bersama sebagai satu unit. Model n-gram dibangun berdasarkan asumsi Markov, yang mengasumsikan bahwa probabilitas sebuah kata hanya bergantung pada jendela tetap dari kata-kata sebelumnya:

1. ***Representasi N-gram:*** Teks masukan dibagi menjadi urutan n kata yang bersebelahan. Setiap urutan n kata diperlakukan sebagai satu unit atau n-gram. Misalnya, dalam model bigram (n=2), setiap pasangan kata yang berurutan menjadi n-gram.
2. ***Penghitungan Frekuensi:*** Model menghitung kemunculan setiap n-gram dalam data pelatihan. Model mencatat seberapa sering setiap urutan kata tertentu muncul dalam korpus.

3. ***Menghitung Probabilitas:*** Untuk memprediksi probabilitas kata berikutnya dalam suatu urutan, model menggunakan hitungan n-gram. Misalnya, dalam model bigram, probabilitas sebuah kata diperkirakan berdasarkan frekuensi kata sebelumnya (unigram). Probabilitasnya dihitung sebagai rasio jumlah bigram terhadap jumlah unigram.
4. ***Pemulusan:*** Dalam praktiknya, model n-gram mungkin menemukan n-gram yang tidak terlihat (urutan yang tidak ada dalam data pelatihan). Untuk mengatasi masalah ini, teknik pemulusan diterapkan untuk menetapkan probabilitas kecil pada n-gram yang tidak terlihat.
5. ***Pembangkitan Bahasa:*** Setelah model n-gram dilatih, model tersebut dapat digunakan untuk pembangkitan bahasa. Dimulai dengan kata awal, model memprediksi kata berikutnya berdasarkan probabilitas tertinggi dari n-gram yang tersedia. Proses ini dapat diulang secara iteratif untuk menghasilkan kalimat.

Model Markov Tersembunyi (HMM) adalah model probabilistik penting lainnya dalam pemrosesan bahasa. Model ini digunakan untuk memodelkan urutan data yang mengikuti struktur Markovian, di mana urutan keadaan tersembunyi yang mendasarinya menghasilkan peristiwa yang dapat diamati. Istilah "tersembunyi" mengacu pada fakta bahwa kita tidak dapat mengamati keadaan tersebut secara langsung, tetapi kita dapat menyimpulkannya dari peristiwa yang dapat diamati. HMM digunakan dalam berbagai tugas, seperti pengenalan ucapan, penandaan part-of-speech, dan penerjemahan mesin.

Keterbatasan:

- Model n-gram memiliki konteks yang terbatas, hanya mempertimbangkan n-1 kata sebelumnya, yang mungkin tidak menangkap dependensi jarak jauh.
- Model ini mungkin tidak secara efektif menangkap makna semantik atau struktur sintaksis dalam bahasa.

Terlepas dari kesederhanaan dan keterbatasannya, model probabilistik n-gram menyediakan dasar yang berguna untuk tugas pemodelan bahasa dan telah menjadi konsep dasar untuk model bahasa yang lebih canggih seperti jaringan saraf rekuren (RNN) dan model berbasis Transformer.

2.3 MODEL BAHASA BERBASIS JARINGAN SARAF TIRUAN

Model bahasa berbasis jaringan saraf tiruan telah membawa terobosan signifikan dalam pemrosesan bahasa alami (NLP) belakangan ini. Model-model ini memanfaatkan jaringan saraf tiruan, yang merupakan struktur komputasi yang terinspirasi oleh otak manusia, untuk memproses dan memahami bahasa.

Gagasan utama di balik model-model ini adalah melatih jaringan saraf tiruan untuk memprediksi kata berikutnya dalam sebuah kalimat berdasarkan kata-kata sebelumnya. Dengan menyajikan sejumlah besar data teks kepada jaringan dan melatihnya mengenali pola dan hubungan antarkata, jaringan tersebut belajar membuat prediksi probabilistik tentang kata apa yang kemungkinan akan muncul berikutnya.

Setelah jaringan saraf tiruan dilatih pada kumpulan data yang besar, jaringan tersebut dapat menggunakan pola yang telah dipelajari untuk menghasilkan teks, melengkapi kalimat, atau bahkan menjawab pertanyaan berdasarkan konteks yang telah dipelajarinya selama pelatihan.

Dengan menangkap hubungan dan ketergantungan antarkata dalam sebuah kalimat secara efektif, model-model bahasa ini telah meningkatkan kemampuan komputer secara drastis untuk memahami dan menghasilkan bahasa manusia, yang menghasilkan kemajuan signifikan dalam berbagai aplikasi NLP seperti penerjemahan mesin, analisis sentimen, chatbot, dan banyak lagi.

Lapisan Input (n1, n2, ..., n_input)
↘ ↘ ↘
Lapisan Tersembunyi (n3, n4, ..., n_hidden)
↘ ↘ ↘

Lapisan Output (n5, n6, ..., n_output) Dalam diagram ini:

- "n_input" merepresentasikan jumlah neuron input, masing-masing sesuai dengan fitur dalam data input.
- "n_hidden" merepresentasikan jumlah neuron dalam lapisan tersembunyi. Lapisan tersembunyi dapat memiliki beberapa neuron, yang biasanya menghasilkan representasi data input yang lebih kompleks.
- "n_output" merepresentasikan jumlah neuron dalam lapisan output. Jumlah neuron output bergantung pada sifat permasalahan—bisa berupa biner (satu neuron) atau multikelas (beberapa neuron).

Jaringan Saraf Tiruan Berulang (RNN)

Jaringan saraf tiruan berulang (RNN) adalah jenis jaringan saraf tiruan yang dirancang untuk memproses data sekuensial satu elemen pada satu waktu sambil mempertahankan status internal yang merangkum riwayat masukan sebelumnya. Jaringan ini memiliki kemampuan unik untuk menangani urutan masukan dan keluaran dengan panjang variabel, sehingga sangat cocok untuk tugas-tugas pemrosesan bahasa alami seperti sintesis bahasa, penerjemahan mesin, dan pengenalan suara.

Fitur utama yang membedakan RNN adalah kemampuannya untuk menangkap dependensi temporal melalui loop umpan balik. Loop ini memungkinkan jaringan untuk menggunakan informasi dari keluaran sebelumnya sebagai masukan untuk prediksi di masa mendatang. Kemampuan seperti memori ini memungkinkan RNN untuk mempertahankan konteks dan informasi dari elemen-elemen sebelumnya dalam urutan tersebut, yang memengaruhi pembangkitan keluaran selanjutnya.

Namun, RNN menghadapi beberapa tantangan. Masalah gradien yang menghilang merupakan masalah yang signifikan, di mana gradien yang digunakan untuk memperbarui bobot jaringan menjadi sangat kecil selama pelatihan, sehingga sulit untuk mempelajari

dependensi jangka panjang secara efektif. Sebaliknya, masalah gradien eksplosif dapat terjadi ketika gradien menjadi terlalu besar, yang menyebabkan pembaruan bobot yang tidak stabil.

Lebih lanjut, RNN pada dasarnya bersifat sekuensial, memproses elemen satu per satu, yang dapat memakan banyak komputasi dan sulit untuk diparalelkan. Keterbatasan ini dapat menghambat skalabilitasnya ketika menangani kumpulan data besar.

Untuk mengatasi beberapa masalah ini, varian RNN yang lebih canggih, seperti memori jangka pendek (LSTM) dan unit rekursif tergerbang (GRU), telah dikembangkan. Varian-varian ini terbukti lebih efektif dalam menangkap dependensi jangka panjang dan memitigasi masalah gradien yang menghilang.

RNN merupakan model yang ampuh untuk menangani data sekuensial, tetapi memiliki tantangan tertentu terkait pembelajaran dependensi jangka panjang, masalah gradien, dan efisiensi komputasi. Varian-variannya, seperti LSTM dan GRU, telah memperbaiki keterbatasan ini dan tetap menjadi alat penting untuk berbagai tugas sekuensial dalam pemrosesan bahasa alami dan seterusnya.

Long Short-Term Memory / Memori Jangka Panjang dan Pendek (LSTM)

Jaringan memori jangka panjang dan pendek (LSTM) adalah jenis arsitektur jaringan saraf rekuren (RNN) khusus yang dirancang untuk mengatasi masalah gradien yang menghilang dan menangkap dependensi jangka panjang dalam data sekuensial. Jaringan ini diperkenalkan oleh Hochreiter dan Schmidhuber pada tahun 1997 dan sejak itu telah mendapatkan popularitas untuk pemodelan data sekuensial dalam berbagai aplikasi.

Fitur utama yang membedakan LSTM dari RNN tradisional adalah kemampuannya untuk menggabungkan sel memori yang dapat secara selektif menyimpan atau melupakan informasi seiring waktu. Sel memori ini dikendalikan oleh tiga gerbang: gerbang masukan, gerbang lupa, dan gerbang keluaran:

- Gerbang masukan mengatur aliran data baru ke dalam sel memori, memungkinkannya untuk memutuskan informasi baru mana yang penting untuk disimpan.
- Gerbang lupa mengontrol retensi data saat ini di dalam sel memori, memungkinkannya untuk melupakan informasi yang tidak relevan atau usang dari langkah waktu sebelumnya.
- Gerbang keluaran mengatur aliran informasi dari sel memori ke keluaran jaringan, memastikan bahwa informasi yang relevan digunakan dalam menghasilkan prediksi.

Mekanisme gerbang ini memungkinkan LSTM untuk menangkap dependensi jarak jauh dalam data sekuensial, sehingga sangat efektif untuk tugas-tugas yang melibatkan pemrosesan bahasa alami, seperti pemodelan bahasa, penerjemahan mesin, dan analisis sentimen. Selain itu, LSTM telah berhasil diterapkan dalam tugas-tugas lain seperti pengenalan suara dan pemberian teks pada gambar. Dengan mengatasi masalah gradien yang menghilang dan menyediakan cara yang lebih baik untuk menyimpan dan memanfaatkan informasi penting dari waktu ke waktu, jaringan LSTM telah menjadi alat yang ampuh untuk menangani data sekuensial dan telah meningkatkan kinerja berbagai aplikasi di bidang pembelajaran mesin dan kecerdasan buatan secara signifikan.

Gated Recurrent Unit (GRU)

Jaringan GRU (gated recurrent unit) adalah jenis arsitektur jaringan saraf tiruan yang umum digunakan dalam pembelajaran mendalam dan pemrosesan bahasa alami (NLP). Jaringan ini dirancang untuk mengatasi masalah gradien yang menghilang, sama seperti jaringan LSTM.

Serupa dengan LSTM, GRU juga menggabungkan mekanisme gating, yang memungkinkan jaringan untuk memperbarui dan melupakan informasi secara selektif dari waktu ke waktu. Mekanisme gating ini krusial untuk menangkap dependensi jangka panjang dalam data sekuensial dan menjadikan GRU efektif untuk tugas-tugas yang melibatkan bahasa dan data sekuensial.

Keunggulan utama GRU dibandingkan LSTM terletak pada desainnya yang lebih sederhana dan parameter yang lebih sedikit. Kesederhanaan ini membuat GRU lebih cepat dilatih dan lebih mudah diterapkan, menjadikannya pilihan populer dalam berbagai aplikasi.

Meskipun GRU dan LSTM memiliki mekanisme gerbang, perbedaan utamanya terletak pada jumlah gerbang yang digunakan untuk mengatur aliran informasi. LSTM menggunakan tiga gerbang: gerbang masukan, gerbang lupa, dan gerbang keluaran. Sebaliknya, GRU hanya menggunakan dua gerbang: gerbang setel ulang dan gerbang pembaruan.

Gerbang setel ulang mengontrol informasi mana yang akan dibuang dari langkah waktu sebelumnya, sementara gerbang pembaruan menentukan berapa banyak informasi baru yang akan ditambahkan ke sel memori. Kedua gerbang ini memungkinkan GRU untuk mengontrol aliran informasi secara efektif tanpa kerumitan memiliki gerbang keluaran.

Jaringan GRU merupakan tambahan yang berharga bagi keluarga jaringan saraf tiruan rekuren. Desainnya yang lebih sederhana dan pelatihan yang efisien menjadikannya pilihan praktis untuk berbagai tugas yang berkaitan dengan urutan, dan telah terbukti sangat efektif dalam pemrosesan bahasa alami, pengenalan suara, dan aplikasi analisis data sekuensial lainnya.

Jaringan Encoder-Decoder

Arsitektur encoder-decoder adalah jenis jaringan saraf tiruan yang digunakan untuk menangani tugas-tugas sekuensial seperti penerjemahan bahasa, chatbot, pengenalan audio, dan pemberian teks pada gambar. Arsitektur ini terdiri dari dua komponen utama: jaringan encoder dan jaringan dekoder.

Selama penerjemahan bahasa, misalnya, jaringan encoder memproses kalimat masukan dalam bahasa sumber. Jaringan ini menelusuri kalimat kata demi kata, menghasilkan representasi dengan panjang tetap yang disebut vektor konteks. Vektor konteks ini berisi informasi penting tentang kalimat masukan dan berfungsi sebagai versi ringkas dari kalimat asli.

Selanjutnya, vektor konteks dimasukkan ke dalam jaringan dekoder. Jaringan dekoder memanfaatkan vektor konteks beserta status internalnya untuk mulai menghasilkan urutan keluaran, yang dalam hal ini adalah penerjemahan dalam bahasa target. Dekoder menghasilkan satu kata pada satu waktu, memanfaatkan vektor konteks dan kata-kata yang dihasilkan sebelumnya untuk memprediksi kata berikutnya dalam penerjemahan.

Model Urutan-ke-Urutan (Sequence-to-Sequence)

Model urutan-ke-urutan (Seq2Seq) adalah jenis arsitektur pembelajaran mendalam yang dirancang untuk menangani urutan masukan dengan panjang variabel dan menghasilkan urutan keluaran dengan panjang variabel. Model ini telah populer dalam tugas-tugas pemrosesan bahasa alami (NLP) seperti penerjemahan mesin, peringkasan teks, chatbot, dan lainnya. Arsitekturnya terdiri dari encoder dan decoder, keduanya merupakan jaringan saraf tiruan rekuren (RNN) atau model berbasis Transformer.

Encoder

Encoder mengambil urutan masukan dan memprosesnya kata demi kata, menghasilkan representasi berukuran tetap (vektor konteks) yang mengodekan seluruh urutan masukan. Vektor konteks menangkap informasi penting dari urutan masukan dan berfungsi sebagai status tersembunyi awal untuk dekoder.

Dekoder

Dekoder mengambil vektor konteks sebagai status tersembunyi awal dan menghasilkan urutan keluaran kata demi kata. Pada setiap langkah, dekoder memprediksi kata berikutnya dalam urutan tersebut berdasarkan vektor konteks dan kata-kata yang dihasilkan sebelumnya. Dekoder dikondisikan pada masukan encoder, yang memungkinkannya menghasilkan keluaran yang bermakna.

Mekanisme Atensi

Dalam arsitektur encoder-dekoder standar, proses dimulai dengan mengodekan urutan masukan menjadi representasi vektor dengan panjang tetap. Langkah pengodean ini memadatkan semua informasi dari urutan masukan menjadi satu vektor berukuran tetap, yang umumnya dikenal sebagai "vektor konteks".

Dekoder kemudian mengambil vektor konteks ini sebagai masukan dan menghasilkan urutan keluaran, langkah demi langkah. Dekoder menggunakan vektor konteks dan status internalnya untuk memprediksi setiap elemen dari urutan keluaran.

Meskipun pendekatan ini bekerja dengan baik untuk urutan masukan yang lebih pendek, pendekatan ini dapat menghadapi tantangan ketika menangani urutan masukan yang panjang. Pengodean dengan panjang tetap dapat menyebabkan hilangnya informasi karena vektor konteks memiliki kapasitas terbatas untuk menangkap semua nuansa dan detail yang ada dalam urutan yang lebih panjang.

Intinya, ketika urutan masukan panjang, pengodean dengan panjang tetap mungkin kesulitan untuk mempertahankan semua informasi yang relevan, yang berpotensi menghasilkan urutan keluaran yang kurang akurat atau tidak lengkap.

Untuk mengatasi masalah ini, teknik yang lebih canggih telah dikembangkan, seperti penggunaan mekanisme atensi dalam arsitektur enkoder-dekoder. Mekanisme atensi memungkinkan model untuk berfokus pada bagian-bagian tertentu dari urutan masukan sambil menghasilkan setiap elemen dari urutan keluaran. Dengan cara ini, model dapat secara efektif menangani urutan masukan yang panjang dan menghindari hilangnya informasi, yang menghasilkan peningkatan kinerja dan keluaran yang lebih akurat.

Mekanisme atensi menghitung skor atensi antara status tersembunyi dekoder (kueri) dan status tersembunyi setiap enkoder (kunci). Skor atensi ini menentukan tingkat kepentingan berbagai bagian dari sekuens masukan, dan vektor konteks kemudian dibentuk sebagai penjumlahan terbobot dari status tersembunyi enkoder, dengan bobot yang ditentukan oleh skor atensi.

Arsitektur Seq2Seq, dengan atau tanpa atensi, memungkinkan model untuk menangani sekuens dengan panjang variabel dan menghasilkan sekuens keluaran yang bermakna, sehingga cocok untuk berbagai tugas NLP yang melibatkan data sekuensial.

Pelatihan Model Sekuens-ke-Sekuens

Model Seq2Seq dilatih menggunakan pasangan sekuens masukan dan sekuens keluaran yang sesuai. Selama pelatihan, enkoder memproses sekuens masukan, dan dekoder menghasilkan sekuens keluaran. Model ini dioptimalkan untuk meminimalkan perbedaan antara keluaran yang dihasilkan dan keluaran kebenaran dasar menggunakan teknik seperti pemaksaan guru atau pembelajaran penguatan.

Tantangan Model Sekuens-ke-Sekuens

Model Seq2Seq memiliki beberapa tantangan, seperti menangani urutan yang panjang, menangani kata-kata yang tidak sesuai kosakata, dan mempertahankan konteks dalam jarak jauh. Teknik seperti mekanisme atensi dan pencarian berkas telah diperkenalkan untuk mengatasi masalah ini dan meningkatkan kinerja model Seq2Seq.

Model urutan-ke-urutan merupakan arsitektur pembelajaran mendalam yang andal untuk menangani data sekuensial dalam tugas-tugas NLP. Kemampuannya untuk menangani urutan masukan dan keluaran dengan panjang variabel membuatnya sangat cocok untuk aplikasi yang melibatkan pemahaman dan pembangkitan bahasa alami.

2.4 TRANSFORMER

Arsitektur Transformer diperkenalkan oleh Vaswani dkk. pada tahun 2017 sebagai desain jaringan saraf inovatif yang banyak digunakan dalam tugas-tugas pemrosesan bahasa alami seperti kategorisasi teks, pemodelan bahasa, dan penerjemahan mesin.

Pada intinya, arsitektur Transformer menyerupai model encoder-decoder. Prosesnya dimulai dengan encoder, yang mengambil urutan masukan dan menghasilkan representasi tersembunyi darinya. Representasi tersembunyi ini berisi informasi penting tentang urutan masukan dan berfungsi sebagai representasi kontekstual.

Representasi tersembunyi ini kemudian diteruskan ke dekoder, yang menggunakannya untuk menghasilkan urutan keluaran. Baik encoder maupun dekoder terdiri dari beberapa lapisan jaringan saraf self-attention dan feed-forward.

Lapisan self-attention menghitung bobot atensi di antara semua pasangan komponen masukan, yang memungkinkan model untuk berfokus pada bagian-bagian berbeda dari urutan masukan sesuai kebutuhan. Bobot atensi digunakan untuk menghitung jumlah terbobot dari elemen masukan, yang menyediakan cara bagi model untuk secara selektif memasukkan informasi relevan dari seluruh urutan masukan.

Lapisan feed-forward selanjutnya memproses keluaran dari lapisan self-attention dengan transformasi nonlinier, yang meningkatkan kemampuan model untuk menangkap pola dan hubungan kompleks dalam data. Desain Transformer menawarkan beberapa keunggulan dibandingkan arsitektur jaringan saraf tiruan sebelumnya:

1. *Efisiensi:* Memungkinkan pemrosesan paralel dari urutan masukan, sehingga lebih cepat dan lebih efisien secara komputasi dibandingkan model sekuensial tradisional.
2. *Interpretabilitas:* Bobot atensi dapat divisualisasikan, memungkinkan kita melihat bagian mana dari urutan masukan yang menjadi fokus model selama pemrosesan, sehingga memudahkan pemahaman dan interpretasi perilaku model.
3. *Konteks Global:* Transformer dapat mempertimbangkan seluruh urutan masukan secara bersamaan, memungkinkannya untuk menangkap dependensi jangka panjang dan meningkatkan kinerja pada tugas-tugas seperti penerjemahan mesin, di mana konteks dari keseluruhan kalimat sangat penting.

Arsitektur Transformer telah menjadi pendekatan dominan dalam pemrosesan bahasa alami dan telah secara signifikan memajukan perkembangan terkini dalam berbagai tugas terkait bahasa, berkat efisiensi, interpretabilitas, dan kemampuannya untuk menangkap konteks global dalam data.

2.5 LARGE LANGUAGE MODELS (LLM)

Model Bahasa Besar (LLM) merujuk pada kelas model kecerdasan buatan canggih yang dirancang khusus untuk memproses dan memahami bahasa manusia dalam skala besar. Model-model ini biasanya dibangun menggunakan teknik pembelajaran mendalam, khususnya arsitektur berbasis Transformer, dan dilatih pada data tekstual dalam jumlah besar dari internet.

Karakteristik utama model bahasa besar adalah kemampuannya untuk mempelajari pola kompleks, representasi semantik, dan hubungan kontekstual dalam bahasa alami. Model-model ini dapat menghasilkan teks seperti manusia, menerjemahkan antarbahasa, menjawab pertanyaan, melakukan analisis sentimen, dan menyelesaikan berbagai tugas pemrosesan bahasa alami. Salah satu contoh model bahasa besar yang paling terkenal adalah seri GPT (Generative Pre-trained Transformer) OpenAI, yang mencakup model seperti GPT-3.

Model-model ini telah dilatih sebelumnya pada kumpulan data besar dan dapat disempurnakan untuk aplikasi tertentu, memungkinkan mereka untuk beradaptasi dan unggul dalam berbagai tugas terkait bahasa. Kemampuan model bahasa besar telah membawa kemajuan signifikan pada pemrosesan bahasa alami (NLP), menjadikannya penting dalam berbagai industri, termasuk dukungan pelanggan, pembuatan konten, penerjemahan bahasa, dan lainnya. Namun, model ini juga menimbulkan kekhawatiran penting terkait etika, bias, dan penyalahgunaan karena potensinya menghasilkan teks yang mirip manusia dan menyebarkan misinformasi jika tidak digunakan secara bertanggung jawab.

Beberapa contoh LLM yang terkenal antara lain:

1. **GPT:** GPT adalah versi keempat dari seri Transformer Pra-terlatih Generatif OpenAI. Model ini dikenal karena kemampuannya menghasilkan teks yang mirip manusia dan

telah menunjukkan kemahiran dalam menjawab pertanyaan, menciptakan puisi, dan bahkan menulis kode.

2. **BERT (Representasi Encoder Dua Arah dari Transformer):** Dikembangkan oleh Google, BERT adalah LLM penting yang menangkap konteks dari kedua arah teks masukan, membuatnya mahir dalam memahami nuansa dan hubungan bahasa. Model ini telah menjadi model dasar untuk berbagai tugas NLP.
3. **T5 (Transformator Transfer Teks-ke-Teks):** Dikembangkan juga oleh Google, T5 menangani semua tugas NLP sebagai masalah teks-ke-teks. Kerangka kerja pemersatu ini telah menunjukkan kinerja yang luar biasa dalam tugas-tugas seperti penerjemahan, peringkasan, dan tanya jawab.
4. **RoBERTa:** RoBERTa dari Facebook adalah versi BERT yang dioptimalkan dan telah mencapai hasil mutakhir di berbagai tolok ukur NLP. RoBERTa dibangun di atas arsitektur dan proses pelatihan BERT, yang selanjutnya meningkatkan kemampuan pemahaman bahasa.

LLM ini telah menunjukkan kemajuan dalam pemrosesan bahasa alami, mendorong batasan yang dapat dicapai model AI dalam tugas-tugas seperti pembangkitan bahasa, pemahaman, dan penerjemahan. Fleksibilitas dan kinerja mutakhirnya telah menjadikannya aset berharga dalam berbagai aplikasi, mulai dari chatbot dan penerjemahan bahasa hingga analisis sentimen dan pembangkitan konten. Seiring perkembangan penelitian di bidang ini, kita dapat mengharapkan munculnya LLM yang lebih canggih dan mumpuni, yang terus merevolusi bidang NLP.

Kesimpulan

Pengembangan jaringan saraf tiruan untuk model bahasa besar telah menghasilkan terobosan signifikan di bidang pemrosesan bahasa alami (NLP). Dari model probabilistik tradisional seperti n-gram dan model Markov tersembunyi hingga model berbasis jaringan saraf tiruan yang lebih canggih seperti jaringan saraf tiruan rekuren (RNN), jaringan memori jangka pendek panjang (LSTM), dan unit rekuren tergerbang (GRU), para peneliti terus menyempurnakan model-model ini untuk mengatasi tantangan seperti gradien yang menghilang dan menangani kumpulan data besar secara efisien.

Salah satu kemajuan penting adalah pengenalan teknik berbasis atensi, khususnya arsitektur Transformer. Transformer telah menunjukkan kinerja yang luar biasa dalam berbagai aplikasi NLP dengan memungkinkan model untuk berfokus pada bagian-bagian tertentu dari urutan masukan menggunakan mekanisme atensi diri.

Model-model ini telah mencapai kesuksesan luar biasa dalam pemodelan bahasa karena kemampuannya untuk secara efektif memperhatikan berbagai wilayah dari urutan masukan, menangkap pola dan dependensi yang kompleks.

Terakhir, fokus telah bergeser ke model bahasa besar (LLM), yang menggunakan jaringan saraf tiruan dalam untuk menghasilkan teks bahasa alami. LLM seperti GPT-3 telah menunjukkan kemampuan yang menakjubkan, menghasilkan teks seperti manusia, menjawab pertanyaan, dan melakukan berbagai tugas terkait bahasa.

Kesimpulannya, kemajuan dalam jaringan saraf untuk model bahasa besar telah merevolusi lanskap NLP, memungkinkan mesin untuk memahami dan menghasilkan bahasa manusia pada tingkat yang belum pernah terjadi sebelumnya, membuka kemungkinan baru untuk komunikasi, pembuatan konten, dan pemecahan masalah.

Dalam bab-bab selanjutnya, mari kita selami lebih dalam arsitektur dan aplikasi model bahasa besar.

BAB 3
LARGE LANGUAGE MODELS (LLM) DAN TRANSFORMER

Dalam bab ini, kita memulai perjalanan yang mencerahkan ke dunia LLM dan seluk-beluk arsitektur Transformer, mengungkap misteri di balik kemampuan luar biasa mereka. Kemajuan perintis ini tidak hanya mendorong bidang NLP ke tingkat yang lebih tinggi, tetapi juga merevolusi cara mesin mempersepsi, memahami, dan menghasilkan bahasa.

3.1 KEKUATAN MODEL BAHASA

Model bahasa telah muncul sebagai kekuatan pendorong dalam ranah pemrosesan bahasa alami (NLP), yang memiliki kekuatan untuk mengubah cara mesin menafsirkan dan menghasilkan bahasa manusia. Model-model ini bertindak sebagai ahli bahasa virtual, menguraikan seluk-beluk tata bahasa, sintaksis, dan semantik, untuk memahami kompleksitas komunikasi manusia yang luas. Signifikansi model bahasa tidak hanya terletak pada kemampuannya untuk memahami teks, tetapi juga pada potensinya untuk menghasilkan respons yang koheren dan relevan secara kontekstual, mengaburkan batas antara pemahaman bahasa manusia dan bahasa mesin. Inti dari model bahasa adalah konsep probabilitas bersyarat, di mana sebuah model mempelajari kemungkinan kemunculan sebuah kata atau token berdasarkan kata-kata sebelumnya dalam suatu urutan.

Dengan pelatihan pada kumpulan data ekstensif yang berisi beragam pola bahasa, model-model ini menjadi mahir dalam memprediksi kata berikutnya yang paling mungkin dalam konteks tertentu. Kekuatan prediktif ini menjadikannya sangat diperlukan dalam berbagai tugas NLP, mulai dari penerjemahan dan peringkasan mesin hingga analisis sentimen, tanya jawab, dan lainnya.

Namun, model bahasa tradisional memiliki keterbatasan inheren, terutama ketika berhadapan dengan dependensi jarak jauh dan menangkap nuansa kontekstual bahasa. Kebutuhan akan solusi yang lebih canggih membuka jalan bagi model bahasa besar (LLM), yang telah merevolusi bidang NLP melalui skalanya yang sangat besar, inovasi arsitektur yang kuat, dan kemampuan luar biasa yang dimilikinya.

Model bahasa berskala besar memanfaatkan sumber daya komputasi yang sangat besar dan data dalam jumlah yang sangat besar selama proses pelatihannya, memungkinkan mereka memahami seluk-beluk bahasa manusia yang rumit. Lebih lanjut, model ini unggul dalam generalisasi, belajar dari beragam contoh yang mereka temui selama proses pra-pelatihan dan penyempurnaan, yang memungkinkan mereka untuk bekerja secara impresif pada berbagai tugas NLP.

Pengenalan arsitektur Transformer menandai momen penting dalam kemajuan model. Transformer memperkenalkan mekanisme atensi—sebuah konsep revolusioner yang memungkinkan model untuk secara dinamis mempertimbangkan relevansi setiap kata dalam suatu urutan yang berkaitan dengan semua kata lainnya. Mekanisme atensi ini, bersama dengan jaringan saraf umpan-maju, membentuk fondasi kinerja Transformer yang luar biasa.

Seiring terus berkembangnya model bahasa, model ini menjanjikan kemajuan yang lebih mendalam dalam pemahaman dan pembuatan bahasa berbasis AI. Namun demikian, dengan kekuatan tersebut muncul tanggung jawab untuk mengatasi masalah etika seputar bias, misinformasi, dan privasi. Mencapai keseimbangan antara mendorong batasan pemodelan bahasa sekaligus menjunjung tinggi pertimbangan etika sangat penting untuk memastikan penerapan dan dampak yang bertanggung jawab dari perangkat-perangkat canggih ini.

Di bagian selanjutnya, kami akan membahas lebih dalam seluk-beluk arsitektur model bahasa besar dan Transformer, mengeksplorasi cara kerjanya, aplikasinya di dunia nyata, tantangan yang dihadirkannya, dan potensi yang dimilikinya untuk membentuk kembali masa depan NLP dan kecerdasan buatan.

3.2 ARSITEKTUR TRANSFORMER

Sebagaimana telah disebutkan sebelumnya, arsitektur Transformer merupakan komponen penting dari banyak model pemrosesan bahasa alami (NLP) mutakhir, termasuk ChatGPT. Arsitektur ini diperkenalkan dalam makalah berjudul "Attention Is All You Need" oleh Vaswani dkk. pada tahun 2017. Transformer merevolusi NLP dengan menyediakan cara yang efisien untuk memproses dan menghasilkan bahasa menggunakan mekanisme self-attention. Mari kita telaah lebih dalam tentang arsitektur inti Transformer.

Motivasi untuk Transformer

Motivasi untuk arsitektur Transformer berawal dari keterbatasan dan inefisiensi model sekuensial tradisional, seperti jaringan saraf tiruan rekuren (RNN) dan jaringan memori jangka pendek panjang (LSTM). Model sekuensial ini memproses masukan bahasa satu token pada satu waktu, yang menyebabkan beberapa masalah ketika menangani dependensi jarak jauh dan paralelisasi. Motivasi utama untuk mengembangkan arsitektur Transformer adalah sebagai berikut:

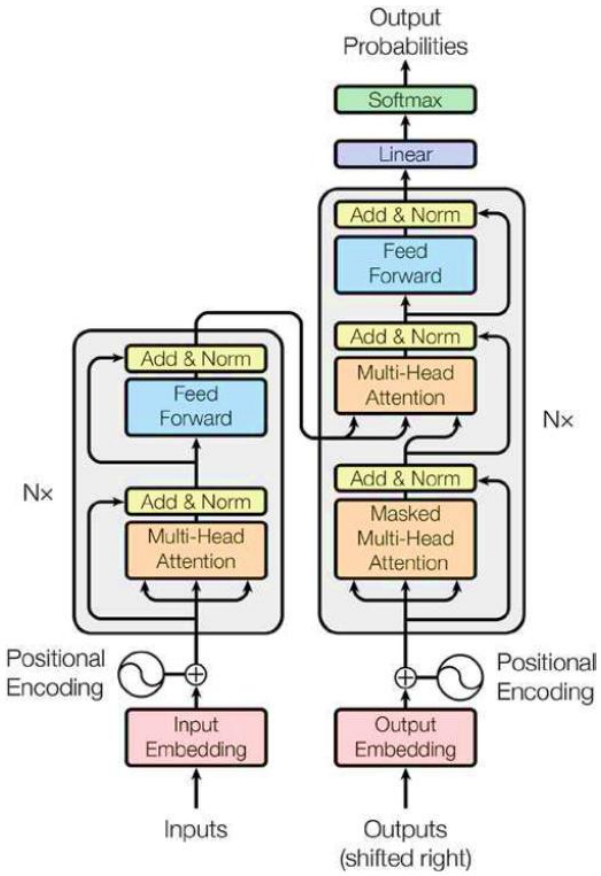
- ***Dependensi Jangka Panjang:*** Model sekuensial tradisional seperti RNN dan LSTM menghadapi kesulitan dalam menangkap dependensi jarak jauh dalam sekuens bahasa. Seiring bertambahnya jarak antar token yang relevan, model ini kesulitan untuk menyimpan dan menyebarkan informasi dalam jarak jauh.
- ***Inefisiensi dalam Paralelisasi:*** RNN memproses masukan bahasa secara berurutan, sehingga menyulitkan untuk memparalelkan komputasi antar token. Keterbatasan ini menghambat kemampuan mereka untuk memanfaatkan perangkat keras modern dengan kemampuan pemrosesan paralel, seperti GPU dan TPU, yang krusial untuk melatih model besar secara efisien.
- ***Gradient Vanishing dan Exploding:*** RNN mengalami masalah gradien yang menghilang dan meledak selama pelatihan. Dalam sekuens yang panjang, gradien dapat menjadi sangat kecil atau sangat besar, yang menyebabkan kesulitan dalam pembelajaran dan konvergensi.

- ***Mengurangi Kompleksitas Komputasi:*** Model sekuensial tradisional memiliki kompleksitas komputasi kuadratik sehubungan dengan panjang sekuens, sehingga membutuhkan komputasi yang mahal untuk memproses sekuens yang panjang.

Arsitektur Transformer, dengan mekanisme self-attention-nya, mengatasi keterbatasan ini dan menawarkan beberapa keuntungan.

Arsitektur

Arsitektur Transformer yang direpresentasikan sebelumnya pada Gambar 3-1 menggunakan kombinasi lapisan self-attention bertumpuk dan lapisan terhubung penuh titik demi titik pada encoder dan decoder, seperti yang digambarkan masing-masing pada bagian kiri dan kanan gambar.



Gambar 3.1. Struktur encoder-decoder dari arsitektur Transformer. Diambil dari "Attention Is All You Need" oleh Vaswani

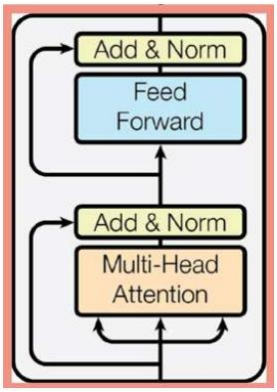
3.3 ARSITEKTUR ENCODER-DECODER

Arsitektur Transformer menggunakan tumpukan encoder dan tumpukan dekoder, yang masing-masing terdiri dari beberapa lapisan, untuk memproses urutan masukan dan menghasilkan urutan keluaran secara efektif.

Encoder

Encoder yang direpresentasikan sebelumnya pada Gambar 3.2 dibangun dengan tumpukan N = 6 lapisan identik, dengan setiap lapisan terdiri dari dua sub-lapisan. Sub-lapisan

pertama menggunakan mekanisme self-attention multi-head, yang memungkinkan model untuk memperhatikan bagian-bagian berbeda dari urutan masukan secara bersamaan. Sub-lapisan kedua adalah jaringan feed-forward yang sederhana dan terhubung penuh berdasarkan posisi, yang selanjutnya memproses keluaran dari mekanisme self-attention.



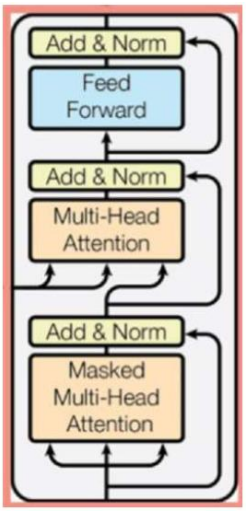
Gambar 3.2. Struktur encoder-decoder arsitektur Transformer. Diambil dari "Attention Is All You Need"

Untuk memastikan aliran informasi yang lancar dan memfasilitasi pembelajaran, koneksi residual diadopsi di sekitar masing-masing dari dua sub-lapisan. Ini berarti bahwa keluaran dari setiap sub-lapisan ditambahkan ke masukan asli, yang memungkinkan model untuk mempelajari dan memperbarui representasi secara efektif.

Untuk menjaga stabilitas model selama pelatihan, normalisasi lapisan diterapkan pada keluaran setiap sub-lapisan. Ini menstandardisasi dan menormalkan representasi, mencegahnya menjadi terlalu besar atau terlalu kecil selama proses pelatihan. Lebih lanjut, untuk memungkinkan penggabungan koneksi residual, semua sub-lapisan dalam model, termasuk lapisan penyisipan, menghasilkan keluaran berdimensi dmodel = 512. Dimensionalitas ini membantu dalam menangkap pola dan dependensi yang rumit dalam data, yang berkontribusi pada kinerja model secara keseluruhan.

Dekoder

Dekoder yang ditunjukkan sebelumnya pada Gambar 3.3 dalam model kami memiliki struktur yang serupa dengan enkoder, terdiri dari tumpukan N = 6 lapisan identik. Setiap lapisan dekoder, seperti lapisan enkoder, berisi dua sub-lapisan untuk jaringan self-attention multi-head dan feed-forward berdasarkan posisi. Sebaliknya, dekoder memperkenalkan sub-lapisan ketiga tambahan, yang memanfaatkan atensi multi-head untuk memproses keluaran tumpukan enkoder.



Gambar 3.3. Struktur encoder-decoder dari arsitektur Transformer.

Tujuan dari sub-lapisan ketiga ini adalah untuk memungkinkan dekoder mengakses dan memanfaatkan representasi kontekstual yang dihasilkan oleh encoder. Dengan memperhatikan keluaran encoder, dekoder dapat menyelaraskan urutan masukan dan keluaran, sehingga meningkatkan kualitas urutan keluaran yang dihasilkan.

Untuk memastikan pembelajaran yang efektif dan aliran informasi yang lancar, dekoder, seperti halnya encoder, menggunakan koneksi residual di sekitar setiap sub-lapisan, diikuti oleh normalisasi lapisan. Hal ini memungkinkan model untuk mempertahankan dan menyebarkan informasi yang berguna secara efektif selama proses dekode.

Berbeda dengan mekanisme self-attention yang digunakan dalam encoder, sub-lapisan self-attention dalam dekoder dapat mengalami modifikasi krusial. Perubahan ini dirancang untuk mencegah posisi dalam urutan memperhatikan posisi berikutnya. Rasional di balik teknik masking ini sangat penting dalam ranah tugas sequence-to-sequence. Tujuan utamanya adalah untuk memastikan bahwa dekoder menghasilkan token keluaran dengan cara yang dikenal sebagai “autoregresi.”

Autoregresi adalah konsep fundamental dalam tugas pembangkitan sekuens. Konsep ini menunjukkan bahwa selama proses dekode, dekoder diberi kemampuan untuk hanya memperhatikan token yang telah dihasilkannya sebelumnya. Pembatasan yang disengaja ini memastikan bahwa dekoder mematuhi urutan sekuensial yang benar saat menghasilkan token keluaran.

Dalam praktiknya, bayangkan tugas menerjemahkan kalimat dari satu bahasa ke bahasa lain. Autoregresi menjamin bahwa ketika dekoder menghasilkan setiap kata dari kalimat yang diterjemahkan, ia mendasarkan keputusannya pada kata-kata yang telah diterjemahkannya. Hal ini meniru perkembangan alami pembangkitan bahasa manusia, di mana konteks dibangun secara progresif, kata demi kata. Dengan hanya memperhatikan token sebelumnya, dekoder memastikan bahwa ia menghormati struktur semantik dan sintaksis dari sekuens keluaran, menjaga koherensi dan fidelitas terhadap masukan.

Intinya, autoregresi adalah mekanisme yang memungkinkan dekoder untuk "mengingat" apa yang telah dihasilkannya sejauh ini, memastikan bahwa setiap token

berikutnya relevan secara kontekstual dan diposisikan dengan tepat dalam sekuens tersebut. Mekanisme ini memainkan peran penting dalam keberhasilan tugas-tugas urutan-ke-urutan, di mana menjaga urutan pembangkitan token yang benar sangatlah penting.

Untuk mencapai hal ini, penempatan keluaran dekoder diimbangi oleh satu posisi. Akibatnya, prediksi untuk posisi "i" dalam urutan keluaran hanya dapat bergantung pada keluaran yang diketahui pada posisi kurang dari "i". Mekanisme ini memastikan bahwa model menghasilkan token keluaran secara autoregresif, satu token pada satu waktu, tanpa akses ke informasi dari token-token selanjutnya.

Dengan menggabungkan modifikasi ini dalam tumpukan dekoder, model kami dapat memproses dan menghasilkan urutan keluaran secara efektif dalam tugas-tugas urutan-ke-urutan, seperti penerjemahan mesin atau pembangkitan teks. Mekanisme atensi terhadap keluaran enkoder memungkinkan dekoder untuk menyelaraskan dan memahami masukan secara kontekstual, sementara mekanisme dekoding autoregresif menjamin pembangkitan token keluaran yang koheren berdasarkan konteks yang dipelajari.

Attention

Fungsi perhatian dalam konteks arsitektur Transformer dapat didefinisikan sebagai pemetaan antara vektor kueri dan sekumpulan pasangan kunci-nilai, yang menghasilkan vektor keluaran. Fungsi ini menghitung bobot perhatian antara kueri dan setiap kunci dalam himpunan tersebut, lalu menggunakan bobot ini untuk menghitung jumlah bobot dari nilai-nilai terkait. Berikut penjelasan langkah demi langkah tentang fungsi atensi:

Input

- *Vektor Kueri (Q):* Kueri merepresentasikan elemen yang ingin kita perhatikan. Dalam konteks Transformer, ini biasanya berupa kata atau token yang diproses model pada langkah waktu tertentu.
- *Vektor Kunci (K):* Kumpulan vektor kunci merepresentasikan elemen yang akan diperhatikan oleh kueri. Dalam Transformer, ini seringkali merupakan penyisipan kata atau token lain dalam urutan input.
- *Vektor Nilai (V):* Kumpulan vektor nilai berisi informasi yang terkait dengan setiap kunci. Dalam Transformer, ini juga merupakan penyisipan kata atau token dalam urutan input.

Menghitung Skor Atensi

- Fungsi atensi menghitung skor atensi, yang mengukur relevansi atau kesamaan antara kueri dan setiap kunci dalam kumpulan tersebut.
- Hal ini biasanya dilakukan dengan mengambil perkalian titik antara vektor kueri (Q) dan setiap vektor kunci (K), yang menangkap kesamaan antara kueri dan setiap kunci.

Menghitung Bobot Perhatian

- Skor perhatian diubah menjadi bobot perhatian dengan menerapkan fungsi softmax. Fungsi softmax menormalkan skor, mengubahnya menjadi probabilitas yang berjumlah 1.

- Bobot perhatian merepresentasikan pentingnya atau relevansi setiap kunci terhadap kueri.

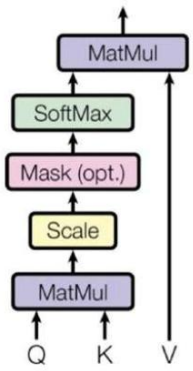
Jumlah Tertimbang

- Vektor keluaran dihitung sebagai jumlah tertimbang dari vektor nilai (V), dengan menggunakan bobot atensi sebagai bobotnya.
- Setiap vektor nilai dikalikan dengan bobot atensinya, dan semua vektor tertimbang dijumlahkan untuk menghasilkan vektor keluaran akhir.
- Vektor keluaran menangkap informasi kontekstual dari vektor nilai berdasarkan bobot atensi, yang merepresentasikan informasi yang diperhatikan yang relevan dengan kueri.

Mekanisme atensi memungkinkan model untuk secara selektif berfokus pada bagian paling relevan dari urutan masukan saat memproses setiap elemen (kueri). Kemampuan untuk memperhatikan informasi relevan dari berbagai bagian urutan ini merupakan faktor kunci keberhasilan Transformer dalam berbagai tugas pemrosesan bahasa alami karena memungkinkan model untuk menangkap dependensi jarak jauh dan hubungan kontekstual secara efektif.

Atensi Produk Titik Terskala

Mekanisme atensi spesifik yang ditunjukkan pada Gambar 3.4 yang digunakan dalam Transformer disebut "Atensi Produk Titik Terskala", yang digambarkan pada gambar sebelumnya. Mari kita uraikan cara kerja Scaled Dot-Product Attention:



Gambar 3.4. Struktur Perhatian Produk Titik Berskala dari Arsitektur Transformer.

Input dan Matriks

- Input untuk Scaled Dot-Product Attention terdiri dari kueri (Q), kunci (K), dan nilai (V), yang masing-masing direpresentasikan sebagai vektor berdimensi dk dan dv.
- Untuk setiap kata dalam urutan input, kami membuat tiga vektor: vektor kueri, vektor kunci, dan vektor nilai.
- Vektor-vektor ini dipelajari selama proses pelatihan dan merepresentasikan embedding token input yang telah dipelajari.

Dot Product dan Penskalaan

- Scaled Dot-Product Attention menghitung skor atensi dengan melakukan dot product antara vektor kueri (Q) dan setiap vektor kunci (K).

- Dot product mengukur kesamaan atau relevansi antara kueri dan setiap kunci.
- Dot product dari dua vektor merupakan hasil penjumlahan produk elemen demi elemen dari komponen-komponennya yang bersesuaian.
- Untuk menstabilkan proses pembelajaran dan mencegah nilai yang sangat besar dalam perkalian titik, perkalian titik diperkecil dengan membaginya dengan akar kuadrat dimensi vektor kunci (`√dk`).
- Faktor skala `√1/dk` ini krusial dalam mencapai komputasi atensi yang stabil dan efisien.

Softmax dan Bobot Atensi

- Setelah menghitung perkalian titik yang diskalakan, kami menerapkan fungsi softmax untuk mengubahnya menjadi bobot atensi.
- Fungsi softmax menormalkan skor atensi, mengubahnya menjadi probabilitas yang jumlahnya mencapai 1.
- Bobot atensi menunjukkan signifikansi atau relevansi setiap kunci dalam kaitannya dengan kueri saat ini.
- Bobot atensi yang lebih tinggi menunjukkan bahwa nilai yang sesuai akan berkontribusi lebih besar pada vektor konteks akhir.

Formulasi dan Efisiensi Matriks

- Atensi Produk Titik Terskala dirancang untuk komputasi yang efisien menggunakan operasi matriks.
- Dalam aplikasi praktis, fungsi atensi dilakukan pada sekumpulan kueri (dikemas bersama ke dalam matriks Q), kunci (dikemas bersama ke dalam matriks K), dan nilai (dikemas bersama ke dalam matriks V) secara bersamaan.
- Matriks keluaran yang dihasilkan kemudian dihitung sebagai berikut:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{dk}}\right) \times V$$

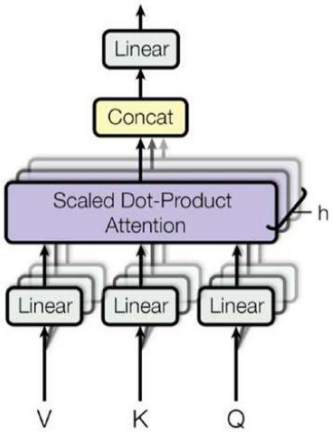
Di mana matriks Q adalah kueri, K adalah kunci, dan V adalah nilai.

- Formulasi matriks ini memungkinkan operasi perkalian matriks yang sangat optimal, sehingga komputasi menjadi lebih efisien dan skalabel.

Atensi Produk-Titik Berskala telah terbukti menjadi komponen penting dalam arsitektur Transformer, yang memungkinkan model untuk menangani dependensi jarak jauh dan informasi kontekstual secara efektif. Dengan memperhatikan informasi yang relevan dalam urutan input, Transformer dapat membuat representasi kontekstual untuk setiap kata, yang menghasilkan kinerja luar biasa dalam berbagai tugas pemrosesan bahasa alami, termasuk penerjemahan mesin, pembuatan teks, dan pemahaman bahasa. Penggunaan operasi matriks semakin meningkatkan efisiensi komputasi Perhatian Produk-Titik Berskala, menjadikan Transformer model yang andal untuk memproses urutan dengan panjang dan kompleksitas yang berbeda.

Atensi Multi-Head

Atensi multi-head yang ditunjukkan sebelumnya pada Gambar 3.5 merupakan perluasan dari Perhatian Produk-Titik Berskala yang digunakan dalam arsitektur Transformer. Ini meningkatkan kekuatan ekspresif mekanisme perhatian dengan menerapkan beberapa set perhitungan perhatian secara paralel, yang memungkinkan model untuk menangkap berbagai jenis ketergantungan dan hubungan dalam urutan masukan. Atensi multi-head memungkinkan model untuk memperhatikan berbagai bagian input secara bersamaan, sehingga memungkinkannya menangkap beragam pola dan dependensi.



Gambar 3.5. Struktur atensi multi-head pada arsitektur Transformer.

Berikut cara kerja atensi multi-head:

Proyeksi Input dan Linear

- Seperti dalam Scaled Dot-Product Attention, atensi multi-head menerima kueri input (Q), kunci (K), dan nilai (V), dengan masing-masing direpresentasikan sebagai vektor berdimensi dk dan dv.
- Alih-alih menggunakan proyeksi pembelajaran yang sama untuk semua kepala atensi, kueri input, kunci, dan nilai diproyeksikan secara linear beberapa kali untuk menghasilkan kumpulan vektor kueri, kunci, dan nilai yang berbeda untuk setiap kepala atensi.

Beberapa Kepala Atensi

- Atensi multi-kepala memperkenalkan beberapa kepala atensi, biasanya dilambangkan dengan "h".
- Setiap kepala atensi memiliki serangkaian proyeksi liniernya sendiri untuk menciptakan vektor kueri, kunci, dan nilai yang berbeda.
- Jumlah kepala atensi, dilambangkan sebagai "h", merupakan hiperparameter dan dapat disesuaikan berdasarkan kompleksitas tugas dan kapasitas model.

Atensi Produk Titik Terskala per Kepala

- Untuk setiap kepala atensi, mekanisme Atensi Produk Titik Terskala diterapkan secara independen, menghitung skor atensi, penskalaan, dan menghitung bobot atensi seperti biasa.

- Ini berarti bahwa untuk setiap kepala, vektor konteks terpisah diturunkan menggunakan bobot atensi.

Konkatenasi dan Proyeksi Linier

- Setelah menghitung vektor konteks untuk setiap kepala atensi, vektor-vektor tersebut dikonkatenasi menjadi satu matriks.
- Matriks yang dikonkatenasi kemudian diproyeksikan secara linier ke dimensi keluaran akhir.

Fleksibilitas Model

- Dengan menggunakan beberapa kepala atensi, model memperoleh fleksibilitas dalam menangkap berbagai dependensi dan pola dalam urutan input.
- Setiap kepala atensi dapat belajar berfokus pada berbagai aspek input, sehingga memungkinkan model untuk mengekstrak informasi yang beragam dan saling melengkapi.

Atensi multi-kepala merupakan mekanisme canggih yang meningkatkan kapasitas ekspresif arsitektur Transformer. Mekanisme ini memungkinkan model untuk menangani berbagai pola bahasa, dependensi, dan hubungan, yang menghasilkan kinerja superior dalam tugas-tugas pemrosesan bahasa alami yang kompleks. Kombinasi Atensi Produk-Titik Berskala dengan beberapa kepala atensi telah menjadi faktor kunci keberhasilan Transformer dan kemampuannya untuk mengungguli model-model mutakhir sebelumnya dalam berbagai tugas NLP.

Arsitektur Transformer memanfaatkan atensi multi-kepala dalam tiga cara berbeda, masing-masing melayani tujuan spesifik dalam fungsi model:

1. ***Atensi Encoder-Decoder:***
   - Pada lapisan atensi encoder-decoder, kueri dihasilkan dari lapisan dekoder sebelumnya, yang merepresentasikan konteks dari langkah dekoding saat ini.
   - Kunci dan nilai memori diturunkan dari keluaran encoder, yang merepresentasikan urutan input yang dikodekan.
   - Hal ini memungkinkan setiap posisi dalam dekoder untuk memperhatikan posisi keseluruhan dalam urutan masukan, sehingga model dapat menyelaraskan informasi yang relevan dari masukan ke keluaran selama proses dekoding.
   - Mekanisme atensi ini meniru atensi enkoder-dekoder yang umum digunakan dalam model urutan-ke-urutan, yang fundamental dalam tugas-tugas seperti penerjemahan mesin.
2. ***Atensi Mandiri Enkoder:***
   - Dalam enkoder, lapisan atensi mandiri diterapkan, di mana semua kunci, nilai, dan kueri diturunkan dari keluaran lapisan sebelumnya dalam enkoder.
   - Setiap posisi dalam enkoder dapat memperhatikan semua posisi di lapisan enkoder sebelumnya, sehingga model dapat menangkap dependensi dan hubungan kontekstual dalam urutan masukan secara efektif.

- Atensi mandiri enkoder sangat penting bagi model untuk memahami saling ketergantungan kata-kata dalam urutan masukan. 3. Self-Attention Dekoder dengan Masking:
- Dekoder juga mengandung lapisan self-attention, tetapi dengan perbedaan yang signifikan dari self-attention encoder.
- Dalam mekanisme self-attention dekoder, setiap posisi dalam dekoder dapat memperhatikan semua posisi dalam dekoder hingga dan termasuk posisi tersebut.
- Namun, untuk mempertahankan sifat autoregresif (memastikan bahwa setiap kata dihasilkan dalam urutan yang benar), model perlu mencegah aliran informasi ke kiri dalam dekoder.
- Untuk mencapai hal ini, masukan ke fungsi softmax (yang menghitung bobot atensi) di-masking dengan menetapkan nilai tertentu ke -∞ (negatif tak terhingga), yang secara efektif membuat beberapa koneksi menjadi ilegal.
- Masking mencegah model memperhatikan posisi yang akan melanggar sifat autoregresif dekoder, memastikan pembangkitan kata dalam urutan yang benar selama tugas pembangkitan teks.

3.4 POSITION-WISE FEED-FORWARD NETWORKS

Jaringan umpan maju berdasarkan posisi (FFN) merupakan komponen penting dari arsitektur Transformer, yang digunakan baik pada lapisan enkoder maupun dekoder. Jaringan ini memainkan peran kunci dalam memperkenalkan nonlinieritas dan kompleksitas pada model dengan memproses setiap posisi dalam urutan masukan secara independen dan identik.

Contoh:

Diberikan urutan masukan X = {x_1, x_2, ..., x_seq_len} dengan bentuk (seq_len, d_model), dengan seq_len adalah panjang urutan dan d_model adalah dimensi penyisipan kata (misalnya, d_model = 512):

1. Arsitektur Umpan Maju:
   Jaringan umpan maju berdasarkan posisi terdiri dari dua transformasi linier dengan fungsi aktivasi ReLU yang diterapkan elemen demi elemen di antaranya:

$$\text{FFN\_1(X)} = \max(0, \text{X} \times \text{W1} + \text{b1})$$
$$\text{FFN\_Output} = \text{FFN\_1(X)} \times \text{W2} + \text{b2}$$

   Di sini, FFN_1 merepresentasikan keluaran setelah transformasi linear pertama dengan bobot W1 dan bias b1. Fungsi aktivasi ReLU memperkenalkan nonlinieritas dengan menetapkan nilai negatif ke nol sementara nilai positif tidak berubah. Keluaran akhir FFN_Output diperoleh setelah transformasi linear kedua dengan bobot W2 dan bias b2. Keluaran ini kemudian ditambahkan per elemen ke masukan sebagai bagian dari koneksi residual.

2. ***Dimensionalitas:***

   Masukan dan keluaran jaringan umpan maju per posisi memiliki dimensionalitas d_model = 512, yang konsisten dengan penyisipan kata dalam model Transformer. Lapisan dalam jaringan umpan maju memiliki dimensionalitas df f = 2048.

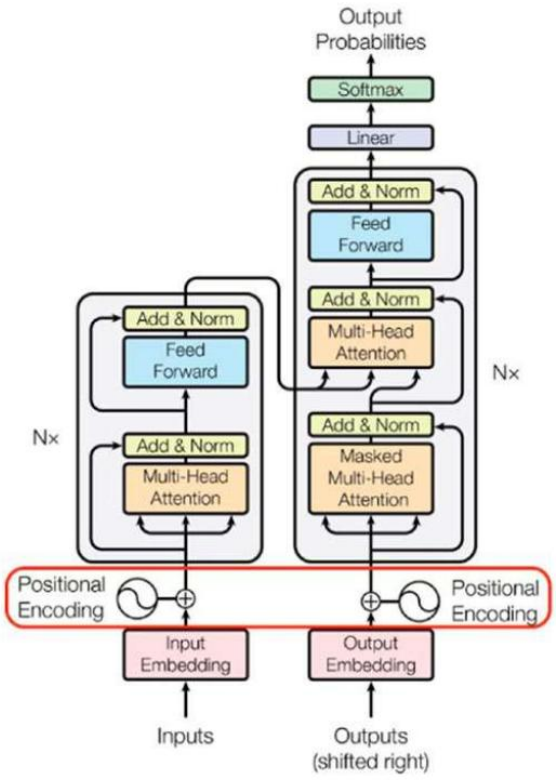
3. ***Pembagian Parameter:***

   Meskipun transformasi linear konsisten di berbagai posisi dalam sekuens, setiap lapisan menggunakan parameter yang dapat dipelajari yang berbeda. Desain ini juga dapat dianggap sebagai dua konvolusi satu dimensi dengan ukuran kernel 1.

Jaringan umpan maju berdasarkan posisi memungkinkan model Transformer untuk menangkap pola dan dependensi kompleks dalam urutan masukan, melengkapi mekanisme atensi. Jaringan ini memperkenalkan nonlinieritas pada model, yang memungkinkannya mempelajari dan memproses informasi secara efektif, yang berkontribusi pada kinerja Transformer yang mengesankan dalam berbagai tugas pemrosesan bahasa alami.

Pengodean Posisi

Pengodean posisi yang ditunjukkan pada Gambar 3.6 merupakan komponen penting dari arsitektur Transformer, yang diperkenalkan untuk mengatasi tantangan penggabungan informasi posisi kata dalam suatu urutan. Tidak seperti jaringan saraf rekuren (RNN) tradisional yang secara inheren menangkap urutan kata yang berurutan, Transformer beroperasi pada seluruh urutan masukan secara bersamaan menggunakan atensi diri. Namun, karena atensi diri tidak secara inheren mempertimbangkan urutan kata, pengodean posisi diperlukan untuk menyediakan informasi posisi kepada model.



Gambar 3.6. Pengodean posisi arsitektur Transformer.

Pentingnya Pengodean Posisi:

- Tanpa pengodean posisi, Transformer akan memperlakukan masukan sebagai "sekantong kata" tanpa pemahaman urutan kata, yang dapat mengakibatkan hilangnya informasi sekuensial.
- Dengan pengodean posisi, Transformer dapat membedakan kata-kata di posisi yang berbeda, sehingga model dapat memahami posisi relatif dan absolut kata-kata dalam urutan tersebut.

Rumus untuk Pengodean Posisi:

Pengodean posisi ditambahkan langsung ke embedding input Transformer. Pengodean ini terdiri dari fungsi sinusoidal dengan frekuensi berbeda untuk mengodekan posisi setiap kata dalam deret. Rumus untuk pengodean posisi adalah sebagai berikut:

$$\mathrm{PE}(\mathrm{pos}, 2\mathrm{i}) = \sin\left(\frac{\mathrm{pos}}{10000^{\left(\frac{2\mathrm{i}}{\mathrm{d_{model}}}\right)}}\right)$$

$$\mathrm{PE}(\mathrm{pos}, 2\mathrm{i}+1) = \cos\left(\frac{\mathrm{pos}}{10000^{\left(\frac{2\mathrm{i}}{\mathrm{d_{model}}}\right)}}\right)$$

Di mana

- "$\mathrm{PE}(\mathrm{pos}, 2\mathrm{i})$" merepresentasikan dimensi ke-i dari pengkodean posisi untuk kata pada posisi "pos."
- "$\mathrm{PE}(\mathrm{pos}, 2\mathrm{i}+1)$" merepresentasikan dimensi ke-(i+1) dari pengkodean posisi untuk kata pada posisi "pos."
- "i" adalah indeks dimensi, berkisar dari 0 hingga "$\mathrm{d\_model} - 1$."
- Variabel pos merepresentasikan posisi kata dalam deret.
- "$\mathrm{d\_model}$" adalah dimensi penyisipan kata (misalnya, $\mathrm{d\_model} = 512$).

Interpretasi

Penggunaan fungsi sinus dan kosinus dalam pengkodean posisi menghasilkan pola siklik, yang memungkinkan model mempelajari jarak posisi yang berbeda dan menggeneralisasikannya ke deret dengan panjang yang bervariasi. Pengkodean posisi ditambahkan ke penyisipan masukan sebelum diteruskan melalui lapisan enkoder dan dekoder Transformer. Pengkodean posisi memperkaya penempatan kata dengan informasi posisi, yang memungkinkan Transformer untuk menangkap hubungan temporal sekuens dan memproses data masukan secara efektif, menjadikannya salah satu komponen penting yang berkontribusi pada keberhasilan Transformer dalam tugas pemrosesan bahasa alami.

Keunggulan dan Keterbatasan Arsitektur Transformer

Seperti desain arsitektur lainnya, Transformer memiliki keunggulan dan keterbatasan. Mari kita bahas:

Keunggulan

1. *Paralelisasi dan Efisiensi:* Mekanisme self-attention Transformer memungkinkan pemrosesan paralel sekuens input, sehingga sangat efisien dan cocok untuk

komputasi terdistribusi, menghasilkan waktu pelatihan yang lebih cepat dibandingkan model sekuensial seperti RNN.

2. *Ketergantungan Jangka Panjang:* Berkat mekanisme self-attention, model ini dapat secara efektif menangkap ketergantungan jangka panjang antar kata dalam suatu sekuens.
3. *Skalabilitas:* Mekanisme atensi Transformer menunjukkan kompleksitas komputasi yang konstan terhadap panjang sekuens, sehingga lebih skalabel dibandingkan model sekuensial tradisional, yang seringkali mengalami peningkatan biaya komputasi untuk sekuens yang lebih panjang.
4. *Pembelajaran Transfer dengan Transformer:* Arsitektur Transformer telah menunjukkan transferabilitas yang luar biasa dalam pembelajaran. Model pra-latih, seperti BERT dan GPT, berfungsi sebagai titik awal yang kuat untuk berbagai tugas pemrosesan bahasa alami. Dengan menyempurnakan model-model ini pada tugas-tugas spesifik, para peneliti dan praktisi dapat mencapai hasil mutakhir tanpa modifikasi arsitektur yang signifikan. Kemudahan transfer ini telah mendorong adopsi yang luas dan kemajuan pesat dalam aplikasi NLP.
5. *Penyematan Kontekstual:* Transformer menghasilkan penyematan kata yang terkontekstualisasi, artinya makna sebuah kata dapat berubah berdasarkan konteksnya dalam kalimat. Kemampuan ini meningkatkan kemampuan model untuk memahami semantik kata dan hubungan kata.
6. *Pemrosesan Informasi Global:* Tidak seperti RNN, yang memproses informasi berurutan dan dapat kehilangan konteks seiring waktu, Transformer memproses seluruh urutan masukan secara bersamaan, sehingga memungkinkan pemrosesan informasi global.

Keterbatasan

1. *Overhead Perhatian untuk Urutan Panjang:* Meskipun Transformer efisien untuk paralelisasi, ia masih menghadapi overhead perhatian untuk urutan yang sangat panjang. Memproses urutan yang sangat panjang dapat menghabiskan sumber daya komputasi dan memori yang signifikan.
2. *Kurangnya Urutan Sekuensial:* Transformer memproses kata secara paralel, yang mungkin tidak sepenuhnya memanfaatkan sifat sekuensial inheren dari beberapa tugas, yang menyebabkan potensi kinerja suboptimal untuk tugas-tugas di mana urutan sangat penting. Meskipun pengkodean posisional digunakan untuk memberikan informasi posisional kepada model, ia melakukannya secara berbeda dari RNN tradisional. Meskipun membantu Transformer memahami urutan urutan, ia tidak menangkapnya secara eksplisit seperti yang dilakukan RNN. Perbedaan ini penting untuk dicatat dalam memahami bagaimana Transformer menangani informasi sekuensial.
3. *Parameterisasi yang Berlebihan*: Transformer memiliki sejumlah besar parameter, terutama dalam model yang mendalam, yang dapat membuat

pelatihan lebih menantang, terutama dengan data dan sumber daya komputasi yang terbatas.

4. *Ketidakmampuan untuk Menangani Input Tidak Terstruktur:* Transformer dirancang terutama untuk urutan, seperti kalimat bahasa alami. Ini mungkin bukan pilihan terbaik untuk input tak terstruktur seperti gambar atau data tabular.
5. *Panjang Input Tetap:* Umumnya, arsitektur Transformer membutuhkan urutan input dengan panjang tetap karena penggunaan enkode posisional. Penanganan urutan dengan panjang variabel mungkin memerlukan prapemrosesan atau padding tambahan. Perlu dicatat bahwa terdapat beberapa varian arsitektur Transformer yang adaptif terhadap panjang dan menawarkan lebih banyak fleksibilitas dalam hal ini.

Kesimpulan

Sebagai kesimpulan, model bahasa besar (LLM) berbasis arsitektur Transformer telah muncul sebagai kemajuan inovatif dalam ranah pemrosesan bahasa alami. Kemampuannya untuk menangkap dependensi jangka panjang, dikombinasikan dengan pra-pelatihan ekstensif pada kumpulan data yang luas, telah merevolusi tugas-tugas pemahaman bahasa alami. LLM telah menunjukkan kinerja yang luar biasa dalam berbagai tantangan terkait bahasa, mengungguli pendekatan tradisional dan menetapkan tolok ukur baru.

Lebih lanjut, LLM menunjukkan potensi besar dalam pembangkitan bahasa dan kreativitas, mampu menghasilkan teks yang mirip manusia dan cerita yang menarik. Namun, di samping berbagai keunggulannya, pertimbangan etis juga penting, termasuk kekhawatiran mengenai bias, misinformasi, dan potensi penyalahgunaan. Para peneliti dan insinyur secara aktif berupaya mengatasi tantangan-tantangan ini untuk memastikan penerapan AI yang bertanggung jawab. Ke depannya, masa depan LLM dan Transformer menjanjikan peluang-peluang menarik, dengan potensi aplikasi di berbagai domain seperti pendidikan, layanan kesehatan, dukungan pelanggan, dan pembangkitan konten. Seiring bidang ini terus berkembang, LLM siap untuk membentuk kembali cara kita berinteraksi dan memahami bahasa, membuka kemungkinan-kemungkinan baru untuk dampak transformatif di tahun-tahun mendatang.

BAB 4
ARSITEKTUR CHATGPT & MODEL BAHASA PERCAKAPAN OPENAI

Dalam beberapa tahun terakhir, kemajuan signifikan dalam pemrosesan bahasa alami (NLP) telah membuka jalan bagi agen percakapan yang lebih interaktif dan mirip manusia. Di antara perkembangan inovatif ini adalah ChatGPT, sebuah model bahasa canggih yang diciptakan oleh OpenAI. ChatGPT didasarkan pada arsitektur GPT (Generative Pre-trained Transformer) dan dirancang untuk terlibat dalam percakapan yang dinamis dan relevan secara kontekstual dengan pengguna.

ChatGPT merepresentasikan pergeseran paradigma dalam dunia AI percakapan, yang memungkinkan pengguna berinteraksi dengan model bahasa dengan cara yang lebih komunikatif. Kemampuannya untuk memahami konteks, menghasilkan respons yang koheren, dan mempertahankan alur percakapan telah memikat para peneliti dan pengguna. Sebagai iterasi terbaru dari model NLP, ChatGPT berpotensi untuk mengubah cara kita berinteraksi dengan teknologi dan informasi.

Bab ini mengeksplorasi seluk-beluk arsitektur ChatGPT, mendalami mekanisme yang mendasarinya, proses pelatihan, dan kapabilitasnya. Kami akan mengungkap bagaimana ChatGPT memanfaatkan kekuatan transformer, perhatian diri, dan sejumlah besar data pra-pelatihan untuk menjadi seorang pembicara yang mahir. Selain itu, kami akan membahas kekuatan dan keterbatasan ChatGPT, beserta pertimbangan etis seputar penggunaannya. Dengan ChatGPT sebagai yang terdepan dalam AI percakapan, bab ini bertujuan untuk menjelaskan dunia model bahasa mutakhir yang menarik dan dampaknya terhadap masa depan interaksi manusia-komputer.

4.1 EVOLUSI MODEL GPT

Evolusi model GPT (Generative Pre-trained Transformer) telah ditandai dengan serangkaian kemajuan yang signifikan. Setiap versi baru model ini biasanya menampilkan peningkatan jumlah parameter dan telah dilatih pada kumpulan data yang lebih beragam dan komprehensif. Berikut sejarah singkatnya:

1. ***GPT-1:*** Model GPT asli, yang diperkenalkan oleh OpenAI pada tahun 2018, didasarkan pada model Transformer. Model ini terdiri dari 12 lapisan, masing-masing dengan 12 kepala self-attention dan total 117 juta parameter. Model ini menggunakan pembelajaran tanpa pengawasan dan dilatih pada kumpulan data BookCorpus, kumpulan 7.000 buku yang belum diterbitkan.
2. ***GPT-2:*** OpenAI merilis GPT-2 pada tahun 2019, yang menandai peningkatan signifikan dalam skala model. Model ini terdiri dari 48 lapisan dan total 1,5 miliar parameter. Versi ini dilatih pada korpus data teks yang lebih besar yang diambil dari internet, mencakup beragam topik dan gaya. Namun, karena kekhawatiran tentang potensi penyalahgunaan, OpenAI awalnya memutuskan untuk tidak merilis model lengkapnya,

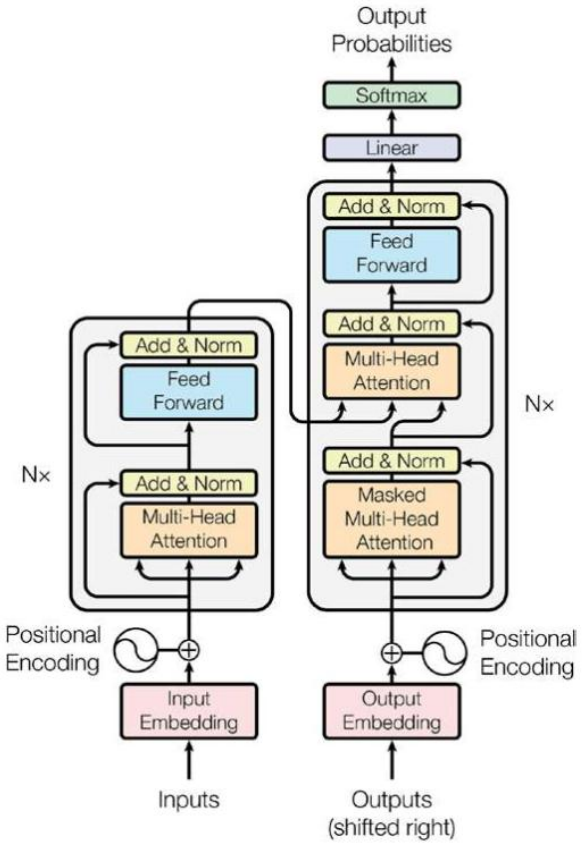
melainkan merilis versi yang lebih kecil dan kemudian merilis model lengkapnya setelah kekhawatiran tersebut diatasi.

3. ***GPT-3:*** GPT-3, yang diperkenalkan pada tahun 2020, menandai peningkatan skala yang signifikan, dengan 175 miliar parameter dan beberapa lapisan transformator. Model ini menunjukkan kemampuan yang mengesankan untuk menghasilkan teks yang sangat mirip dengan bahasa manusia. Rilis GPT-3 memicu minat yang luas terhadap potensi aplikasi model bahasa yang besar, serta diskusi tentang implikasi etis dan tantangan dari model-model canggih tersebut.
4. ***GPT-4:*** GPT-4 adalah model bahasa multimoda revolusioner dengan kemampuan yang meluas hingga pemrosesan input teks dan gambar, mendeskripsikan humor dalam gambar, dan meringkas teks dari tangkapan layar. Interaksi GPT-4 dengan antarmuka eksternal memungkinkan tugas-tugas di luar prediksi teks, menjadikannya alat transformatif dalam pemrosesan bahasa alami dan berbagai domain.

Sepanjang evolusi ini, salah satu tema utama adalah kekuatan skala: secara umum, model yang lebih besar yang dilatih dengan lebih banyak data cenderung berkinerja lebih baik. Namun, tantangan yang terkait dengan model yang lebih besar juga semakin disadari, seperti potensi keluaran yang merugikan, peningkatan sumber daya komputasi yang diperlukan untuk pelatihan, dan kebutuhan akan metode yang robust untuk mengendalikan perilaku model-model ini.

4.2 ARSITEKTUR TRANSFORMER

Sebagaimana disebutkan sebelumnya di bab sebelumnya, kita telah membahas arsitektur Transformer yang ditunjukkan pada Gambar 4.1 secara detail. Ringkasan singkat ini berfungsi sebagai rekap komponen-komponen utama bagi pembaca yang sudah familiar dengan arsitektur Transformer. Untuk pemahaman yang lebih komprehensif, pembaca dapat merujuk kembali ke bab sebelumnya di mana arsitektur Transformer dijelaskan secara menyeluruh beserta komponen dan mekanisme kerjanya.



Gambar 4.1. Struktur encoder-decoder arsitektur Transformer.

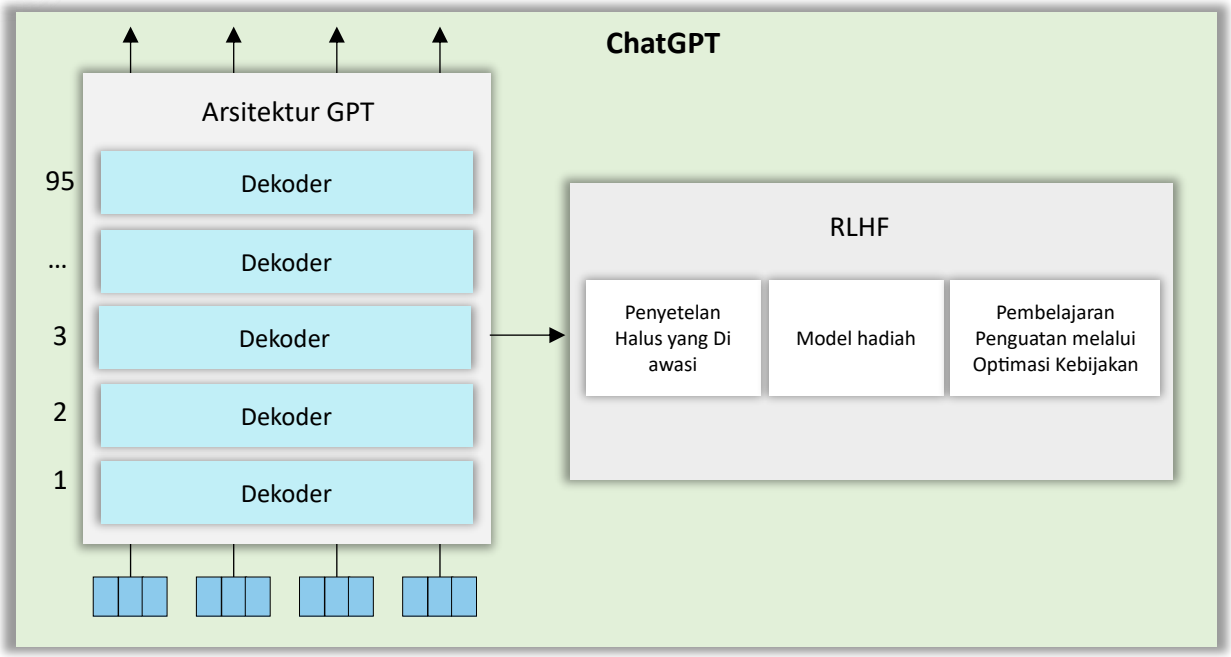
Berikut beberapa poin penting yang perlu diingat tentang arsitektur Transformer:

- Arsitektur Transformer merevolusi pemrosesan bahasa alami dengan mekanisme berbasis atensinya.
- Komponen utama Transformer meliputi mekanisme self-attention, struktur encoder-decoder, pengodean posisi, self-attention multi-head, dan jaringan saraf tiruan feed-forward.
- Self-attention memungkinkan model untuk mempertimbangkan pentingnya kata-kata yang berbeda dan menangkap dependensi jangka panjang.
- Struktur encoder-decoder umumnya digunakan dalam tugas penerjemahan mesin.
- Pengodean posisi digunakan untuk memasukkan informasi urutan kata ke dalam urutan masukan.
- Multi-head self-attention memungkinkan model untuk memperhatikan beberapa bagian masukan secara bersamaan, meningkatkan kemampuannya untuk menangkap hubungan kompleks dalam data.
- Jaringan saraf tiruan feed-forward memproses informasi dari lapisan atensi.
- Koneksi residual dan normalisasi lapisan menstabilkan pelatihan dalam arsitektur mendalam.

Arsitektur ChatGPT

Arsitektur GPT memainkan peran mendasar dalam memungkinkan kapabilitas ChatGPT sebagai AI percakapan interaktif. Meskipun kita telah membahas arsitektur Transformer di bab sebelumnya, bagian ini membahas bagaimana arsitektur ini secara khusus diadaptasi dan dioptimalkan untuk interaksi berbasis obrolan di ChatGPT. ChatGPT, seperti semua model dalam seri GPT, didasarkan pada arsitektur Transformer, yang secara khusus memanfaatkan struktur "khusus dekoder" dari model Transformer asli. Selain itu, ChatGPT menggabungkan komponen penting yang dikenal sebagai "pembelajaran penguatan dari umpan balik manusia (RLHF)." RLHF adalah teknik canggih yang meningkatkan kinerja ChatGPT, dan akan dibahas secara rinci nanti di bab ini, memberikan Anda pemahaman yang komprehensif tentang signifikansinya, seperti yang ditunjukkan pada Gambar 4-2.

Gambar 4.2 menyajikan diagram arsitektur ChatGPT, yang mengilustrasikan proses pelatihannya secara detail. Diagram ini memberikan gambaran komprehensif tentang bagaimana ChatGPT mempelajari dan menyempurnakan kemampuannya selama fase pelatihan. Diagram ini menampilkan aliran data, komponen internal model, dan alur pelatihan, yang menawarkan wawasan tentang pengembangan model.



Gambar 4-2. Arsitektur ChatGPT

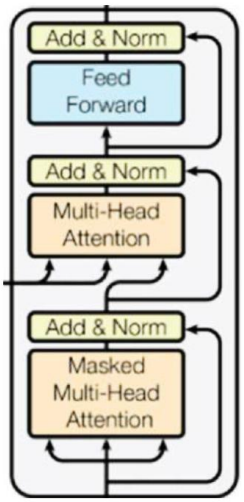
Berikut ikhtisar elemen-elemen kuncinya:

1. **Model Transformer:**

   Model Transformer adalah jenis model yang digunakan dalam pembelajaran mesin, khususnya di bidang pemrosesan bahasa alami (NLP). Model ini diperkenalkan oleh Vaswani dkk. dalam makalah "Attention is All You Need." Keunggulan utama model Transformer adalah memproses data masukan secara paralel, alih-alih berurutan, sehingga memungkinkan komputasi yang lebih efisien dan kemampuan untuk menangani rangkaian data yang lebih panjang. Model ini juga memperkenalkan konsep "attention", yang memungkinkan model untuk mempertimbangkan pentingnya kata-kata yang berbeda dalam masukan saat menghasilkan keluaran.

2. **Struktur Hanya Dekoder:**

   Model Transformer awal yang disajikan oleh Vaswani dkk. mencakup dua bagian: encoder, yang memproses masukan, dan dekoder, yang menghasilkan keluaran. Namun, model GPT seperti ChatGPT hanya menggunakan bagian dekoder yang ditunjukkan pada Gambar 4.3 dari arsitektur Transformer.

   Gambar 4-3. Struktur dekoder arsitektur Transformer

   Hal ini menghasilkan struktur searah, di mana setiap token (atau kata) hanya dapat memperhatikan posisi sebelumnya dalam urutan masukan.

   Desain ini memungkinkan model GPT untuk menghasilkan teks satu kata pada satu waktu, menggunakan kata-kata yang telah dihasilkannya untuk menginformasikan pembuatan kata berikutnya. Pilihan desain ini didorong oleh sifat

tugas AI percakapan, di mana model perlu menghasilkan respons berdasarkan riwayat percakapan masukan.

Lapisan dekoder di ChatGPT bertanggung jawab untuk menghasilkan token berikutnya dalam urutan respons berdasarkan konteks riwayat percakapan. Lapisan ini menggunakan kombinasi jaringan saraf self-attention dan feed-forward untuk memproses token masukan dan menghasilkan balasan yang bermakna dan relevan secara kontekstual.

Mekanisme self-attention dalam dekoder memungkinkan model untuk menangkap dependensi dan hubungan jangka panjang antar token dalam riwayat percakapan. Hal ini penting untuk memahami konteks percakapan yang sedang berlangsung dan menghasilkan respons yang koheren yang selaras dengan dialog sebelumnya. Pengkodean posisional digunakan untuk memasukkan informasi urutan kata ke dalam urutan masukan. Hal ini memastikan bahwa model memahami posisi relatif token dalam riwayat percakapan, sehingga memungkinkannya menghasilkan respons yang sesuai konteks.

Menggunakan arsitektur khusus dekoder menyederhanakan proses pelatihan dan inferensi model. Penyetelan halus dekoder untuk tugas percakapan menjadi lebih mudah, karena fokusnya hanya pada menghasilkan respons berdasarkan konteks yang disediakan.

Selain itu, pengaturan khusus dekoder di ChatGPT membuatnya lebih efisien untuk interaksi waktu nyata. Dengan menghilangkan encoder, sumber daya komputasi difokuskan hanya pada dekoder, sehingga memungkinkan waktu respons yang lebih cepat selama percakapan.

Lebih lanjut, ChatGPT memanfaatkan teknik seperti pembelajaran penguatan dari umpan balik manusia untuk mengoptimalkan kinerja dekoder. Penyetelan halus model dengan respons dan umpan balik yang dihasilkan manusia menyelaraskan keluaran model dengan preferensi manusia yang diinginkan, sehingga meningkatkan kualitas respons yang dihasilkan. Secara keseluruhan, keputusan untuk menggunakan arsitektur khusus dekoder di ChatGPT merupakan pilihan teknis yang dipertimbangkan dengan cermat, disesuaikan dengan konteks AI percakapan. Hal ini memungkinkan model untuk menghasilkan respons yang akurat dan sesuai konteks secara efisien, menjadikannya alat yang ampuh untuk aplikasi berbasis obrolan yang interaktif dan menarik.

3. Mekanisme Self-Attention:

Mekanisme self-attention merupakan elemen kunci dari arsitektur Transformer. Dalam self-attention, setiap token dalam input dapat berinteraksi dengan setiap token lainnya, alih-alih hanya dengan token yang berdekatan atau berdekatan. Hal ini memungkinkan model untuk menangkap konteks setiap kata dalam kalimat dengan lebih baik. Dalam ChatGPT, mekanisme self-attention dimanfaatkan dalam lapisan dekoder untuk menangkap dependensi dan hubungan antar token dalam riwayat

percakapan, sehingga memungkinkan model untuk memahami konteks dan menghasilkan respons yang relevan.

Berikut cara kerja mekanisme self-attention di ChatGPT:

- *Pemahaman Kontekstual:* Dalam sebuah percakapan, setiap kata atau token bergantung pada kata lain dalam riwayat percakapan untuk mendapatkan makna kontekstualnya. Mekanisme self-attention memungkinkan model untuk memperhatikan semua token dalam riwayat percakapan dan mempertimbangkan kepentingannya dalam menghasilkan token berikutnya. Hal ini membantu model untuk memahami konteks yang sedang berlangsung dan menghasilkan respons yang koheren dan relevan secara kontekstual.
- *Skor Atensi:* Selama self-attention, model menghitung skor atensi yang menunjukkan pentingnya setiap token terhadap token yang sedang diproses. Token yang lebih relevan dalam konteks token saat ini menerima skor atensi yang lebih tinggi, sementara token yang kurang relevan menerima skor yang lebih rendah. Pembobotan token yang dinamis ini memungkinkan model untuk berfokus pada bagian paling relevan dari riwayat percakapan untuk menghasilkan respons.
- *Menangkap Ketergantungan Jangka Panjang*: Mekanisme self-attention memungkinkan ChatGPT untuk menangkap ketergantungan jangka panjang dalam riwayat percakapan. Tidak seperti jaringan saraf rekuren tradisional yang memiliki memori terbatas, mekanisme self-attention memungkinkan model untuk mempertimbangkan semua token dalam riwayat percakapan terlepas dari jaraknya dari token saat ini. Kemampuan ini krusial untuk memahami alur percakapan dan menghasilkan respons yang menjaga koherensi dalam dialog yang panjang.
- *Pengodean Posisi:* Dalam arsitektur Transformer, termasuk ChatGPT, pengodean posisi diperkenalkan untuk memasukkan urutan token ke dalam proses self-attention. Pengkodean posisional memastikan model memahami urutan token dalam riwayat percakapan, sehingga memungkinkannya membedakan berbagai posisi dalam dialog dan membuat prediksi yang sesuai konteks.

4. **Struktur Berlapis:** Arsitektur ChatGPT terdiri dari beberapa lapisan dekoder Transformer yang ditumpuk. Setiap lapisan belajar merepresentasikan data masukan dengan cara yang membantu lapisan berikutnya untuk menjalankan tugas dengan lebih baik. Jumlah lapisan dapat bervariasi di berbagai versi GPT; misalnya, GPT-3 memiliki 96 lapisan Transformer.

   Berikut cara kerja struktur berlapis di ChatGPT:

   - *Lapisan Dekoder Bertumpuk:* ChatGPT menggunakan arsitektur khusus dekoder, artinya hanya lapisan dekoder yang digunakan dan lapisan enkoder dihilangkan. Riwayat percakapan berfungsi sebagai masukan ke dekoder, dan tujuan model adalah menghasilkan token berikutnya dalam urutan respons berdasarkan

konteks masukan ini. Lapisan dekoder ditumpuk, dan jumlah lapisan dapat bervariasi berdasarkan konfigurasi model.

- *Ekstraksi Fitur Hirarkis:* Setiap lapisan dekoder di ChatGPT melakukan serangkaian operasi pada token masukan. Mekanisme self-attention di setiap lapisan memungkinkan model untuk memperhatikan semua token dalam riwayat percakapan, menangkap informasi dan dependensi yang relevan di seluruh rangkaian. Ekstraksi fitur hierarkis ini memungkinkan model untuk secara progresif menyempurnakan pemahamannya tentang konteks seiring berpindah melalui lapisan-lapisan tersebut.
- *Pengodean Posisi:* Untuk menangani sifat sekuensial data masukan, pengodean posisi diterapkan ke dalam setiap lapisan. Pengodean ini memberikan informasi tentang urutan dan posisi token dalam riwayat percakapan, memastikan bahwa model dapat membedakan antar token dan memahami posisinya dalam dialog.
- *Jaringan Saraf Tiruan Umpan-Maju:* Setelah langkah perhatian-diri, model memproses token lebih lanjut menggunakan jaringan saraf tiruan umpan-maju di setiap lapisan. Jaringan ini menerapkan transformasi linear dan aktivasi nonlinier pada token, memungkinkan model untuk menangkap pola dan hubungan kompleks dalam percakapan.
- *Koneksi Residu dan Normalisasi Lapisan*: Koneksi residu dan normalisasi lapisan digunakan di setiap lapisan dekoder untuk menstabilkan proses pelatihan dan memfasilitasi aliran informasi. Koneksi residu, terkadang disebut sebagai koneksi lewati, memungkinkan model untuk menyimpan informasi penting dari lapisan sebelumnya dan menyediakan mekanisme untuk "melewati" beberapa lapisan dengan menolkan bobot, menghasilkan model yang terlalu spesifik yang dapat mempelajari kelangkaan. Normalisasi lapisan melengkapi hal ini dengan menormalkan masukan dan keluaran setiap lapisan, yang berkontribusi pada peningkatan konvergensi pelatihan.

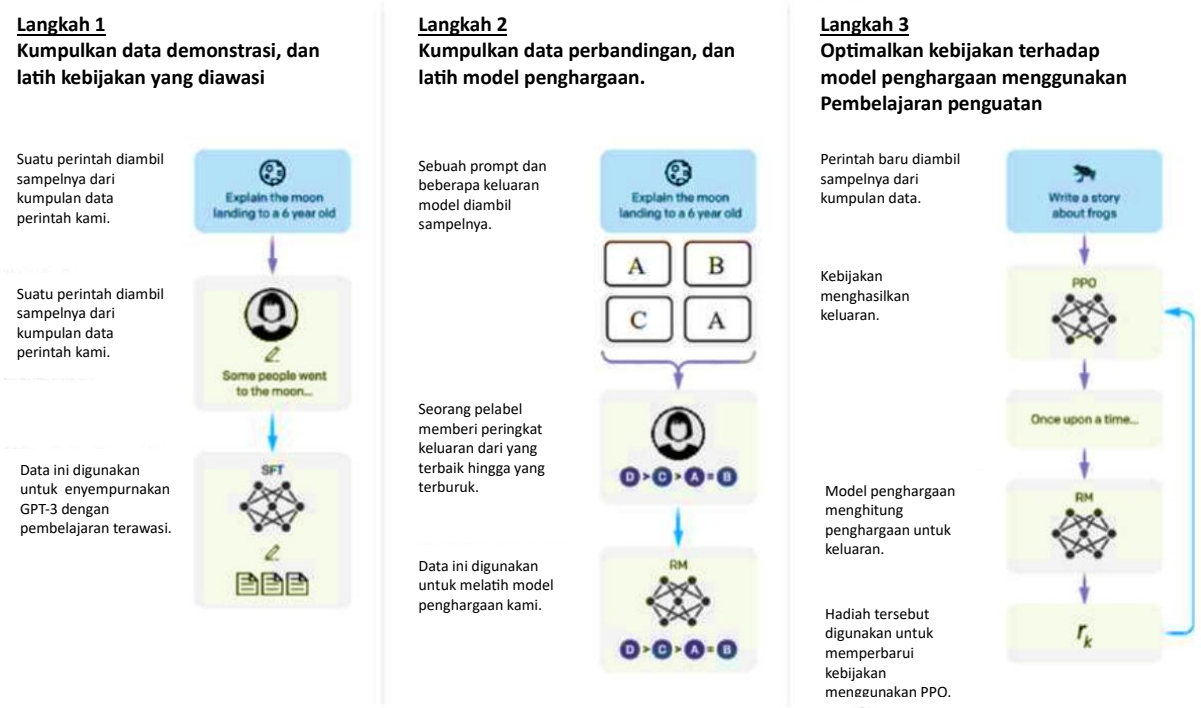
Dengan menumpuk beberapa lapisan dekoder, ChatGPT dapat menangkap pola yang semakin kompleks dan dependensi kontekstual dalam riwayat percakapan. Struktur berlapis ini krusial bagi kemampuan model untuk menghasilkan respons yang koheren dan sesuai konteks dalam interaksi berbasis obrolan. Ekstraksi fitur hierarkis dan penyempurnaan informasi yang progresif memungkinkan ChatGPT untuk bekerja secara efektif dalam berbagai tugas pemrosesan bahasa alami, menjadikannya alat AI percakapan yang canggih.

5. **Pengodean Posisi:** Karena model Transformer memproses semua token masukan secara paralel, model tersebut tidak secara inheren menangkap urutan data sekuensial. Untuk memperhitungkan hal ini, model GPT menggunakan pengodean posisi, yang memberikan informasi tentang posisi setiap kata dalam urutan tersebut. Hal ini memungkinkan model untuk memahami urutan kata dan membuat prediksi yang akurat berdasarkan urutan tersebut. Oleh karena itu, meskipun pengodean posisi penting untuk fungsi ChatGPT dan model Transformer lainnya, pengodean tersebut

tidak hanya berlaku untuk ChatGPT dan merupakan bagian fundamental dari arsitektur Transformer itu sendiri.

6. **Perhatian-Diri Terselubung:** Dalam dekoder, mekanisme perhatian-diri dimodifikasi untuk mencegah token memperhatikan token selanjutnya dalam urutan masukan. Hal ini dikenal sebagai perhatian-diri "terselubung". Perhatian-diri terselubung merupakan komponen penting dari arsitektur Transformer dan juga digunakan dalam ChatGPT untuk menangani data sekuensial secara efisien. Dalam konteks ChatGPT, perhatian-diri terselubung memungkinkan model untuk hanya memperhatikan token yang relevan dalam urutan masukan sekaligus mencegah aliran informasi dari posisi selanjutnya. Hal ini khususnya penting selama pembuatan teks autoregresif untuk menjaga kausalitas dan memastikan bahwa model menghasilkan teks secara berurutan, satu token pada satu waktu.
    - *Perhatian Diri Tersamar dalam ChatGPT:* Pada lapisan dekoder transformator ChatGPT, setiap token memperhatikan semua token lain dalam urutan masukan, termasuk dirinya sendiri, menggunakan perhatian diri. Namun, untuk mencegah kebocoran informasi dari token mendatang selama pembuatan, mekanisme penyembunyian diterapkan pada matriks perhatian diri.
    - *Mekanisme Penyembunyian:* Mekanisme penyembunyian melibatkan penerapan topeng segitiga pada matriks perhatian diri, di mana semua elemen di bawah diagonal utama ditetapkan ke negatif tak terhingga (atau nilai negatif yang sangat besar). Hal ini secara efektif menutupi token mendatang dan memungkinkan token hanya memperhatikan token sebelumnya dan dirinya sendiri.

      Contoh: Mari kita pertimbangkan contoh pembuatan kalimat "Saya suka pemrosesan bahasa alami" menggunakan ChatGPT. Selama proses pembangkitan, ketika model memprediksi kata "bahasa", model seharusnya hanya memperhatikan token sebelumnya, yaitu "saya", "cinta", "alami", dan "bahasa" itu sendiri. Perhatian terhadap kata "pemrosesan" harus disamarkan untuk menjaga kausalitas.
    - *Manfaat dalam Pembangkitan Teks Autoregresif:* Perhatian diri yang disamarkan memastikan bahwa ChatGPT menghasilkan teks secara autoregresif, di mana prediksi setiap token hanya bergantung pada token yang dihasilkan sebelumnya. Hal ini penting untuk menghasilkan kalimat yang koheren dan benar secara tata bahasa. Tanpa penyamaran, model mungkin memiliki akses ke informasi dari token-token selanjutnya, yang menyebabkan keluaran yang salah dan tidak masuk akal.
7. **Pembelajaran Penguatan dari Umpan Balik Manusia (RLHF)**



Gambar 4-4. Diambil dari "melatih model bahasa untuk mengikuti instruksi dengan umpan balik manusia" di mana A (menjelaskan gravitasi), B (menjelaskan perang), atau C (bulan) adalah satelit alami D (manusia pergi ke bulan).

Pembelajaran penguatan dari umpan balik manusia (RLHF) yang ditunjukkan sebelumnya pada Gambar 4.4 merupakan komponen penting dari arsitektur ChatGPT, yang memainkan peran krusial dalam proses penyempurnaan dan meningkatkan kemampuan percakapannya. Pendekatan RLHF memungkinkan ChatGPT untuk belajar dari evaluator manusia dan mengadaptasi pembuatan bahasanya berdasarkan umpan balik mereka. RL, atau pembelajaran penguatan, adalah jenis pembelajaran mesin di mana agen belajar melalui interaksi dengan lingkungannya dan menerima umpan balik berupa hadiah. Berbeda dengan pembelajaran tanpa pengawasan, di mana model belajar dari data tak berlabel tanpa panduan khusus, dan pembelajaran terawasi, di mana model dilatih pada data berlabel dengan jawaban benar yang telah ditentukan sebelumnya, RL melibatkan pembelajaran coba-coba:

1. ***Penyempurnaan Terawasi:*** Penyempurnaan terawasi merupakan fase penting dalam pengembangan ChatGPT. Awalnya, ChatGPT menjalani penyempurnaan terawasi, di mana pelatih AI manusia mensimulasikan percakapan dengan berperan sebagai pengguna dan asisten AI. Selama proses ini, pelatih memiliki akses ke saran yang ditulis oleh model untuk membantu mereka menghasilkan respons yang selaras dengan hasil percakapan yang diinginkan.

   Dataset dialog ini, yang diturunkan dari penyempurnaan terawasi, kemudian digabungkan dengan dataset InstructGPT, yang diubah menjadi format dialog. InstructGPT, model saudara ChatGPT, berakar pada penyediaan respons terperinci terhadap perintah pengguna.

Keterkaitan dengan pembelajaran penguatan dari umpan balik manusia (RLHF) menjadi jelas ketika kita mempertimbangkan bahwa RLHF membawa pelatihan terawasi awal ini selangkah lebih maju. RLHF memungkinkan ChatGPT untuk belajar dan beradaptasi melalui interaksi dengan evaluator manusia yang memberikan umpan balik, menciptakan siklus umpan balik berkelanjutan yang menyempurnakan respons model seiring waktu.

Dengan memahami perkembangan dari fine-tuning terawasi ini, yang dipengaruhi oleh latar belakang InstructGPT, ke RLHF, kami memperoleh wawasan tentang bagaimana ChatGPT berevolusi dan menyelaraskan kemampuannya dengan ekspektasi manusia dalam ranah pemahaman dan pembangkitan bahasa alami.

2. ***Model Imbalan:*** Model yang dilatih melalui pembelajaran terawasi kemudian digunakan untuk mengumpulkan data perbandingan. Pelatih AI terlibat dalam percakapan dengan chatbot dan memeringkat berbagai respons yang dihasilkan model berdasarkan kualitas. Kumpulan data ini digunakan sebagai model imbalan untuk memandu proses pembelajaran penguatan.
3. ***Pembelajaran Penguatan melalui Optimasi Kebijakan Proksimal:*** Pembelajaran penguatan melalui optimasi kebijakan proksimal merupakan langkah penting dalam pengembangan ChatGPT. Dalam RL, "kebijakan" mengacu pada serangkaian aturan atau strategi yang diikuti oleh agen AI untuk membuat keputusan dalam suatu lingkungan. Dalam kasus ini, chatbot, ChatGPT, memiliki "kebijakan" yang memandu cara menghasilkan respons dalam percakapan.

Selama fase ini, model menggunakan data perbandingan untuk meningkatkan kebijakannya melalui metode yang disebut optimasi kebijakan proksimal (PPO). PPO adalah teknik yang mengoptimalkan kebijakan chatbot dengan tujuan meningkatkan kemungkinan menghasilkan respons berperingkat lebih baik sekaligus mengurangi kemungkinan menghasilkan respons berperingkat lebih buruk.

Untuk menghubungkan hal ini dengan konteks yang lebih luas, mari kita mundur sedikit. ChatGPT dimulai sebagai model yang telah dilatih sebelumnya, yang berarti ia memiliki pemahaman dasar bahasa sejak pelatihan awalnya. Namun, untuk menjadikannya benar-benar komunikatif dan responsif, ia menjalani proses penyempurnaan, di mana ia menyempurnakan kemampuannya berdasarkan umpan balik manusia.

Fase pembelajaran penguatan dengan PPO merupakan bagian dari proses penyempurnaan ini. Ini seperti mengajarkan chatbot strategi percakapan tertentu untuk memastikannya memberikan respons berkualitas tinggi. Jadi, intinya, hubungannya di sini adalah bahwa langkah pembelajaran penguatan ini semakin menyempurnakan "kebijakan" ChatGPT agar lebih baik dalam menghasilkan percakapan yang alami dan menarik.

Model terus mengulangi proses ini, belajar dari data perbandingan dan menggunakan PPO untuk meningkatkan respons yang dihasilkannya. Siklus ini berulang, memungkinkan model untuk terus meningkatkan pemahaman dan kualitas responsnya berdasarkan umpan balik manusia.

Dengan demikian, RLHF memainkan peran penting dalam membentuk kinerja ChatGPT. Hal ini memungkinkan OpenAI untuk meningkatkan model secara sistematis berdasarkan umpan balik langsung dari manusia, membantu model menghindari respons yang salah, dan menyelaraskan responsnya dengan nilai-nilai kemanusiaan.

Kombinasi pembelajaran terawasi dengan RLHF ini menyediakan kerangka kerja yang tangguh untuk melatih ChatGPT dan model serupa, memadukan kekuatan pembelajaran mesin tradisional dengan umpan balik bernuansa yang hanya dapat diberikan oleh manusia.

Singkatnya, ChatGPT memanfaatkan arsitektur Transformer, khususnya struktur "decoder-only" dan RLHF untuk memproses dan menghasilkan teks secara efisien. Penggunaan self-attention memungkinkannya untuk mempertimbangkan konteks input secara menyeluruh, sementara enkode posisional memastikan urutan kata yang berurutan tertangkap. Aspek-aspek ini berpadu untuk memungkinkan ChatGPT menghasilkan teks yang sangat mirip manusia.

4.2 PRA-PELATIHAN DAN PENYEMPURNAAN DALAM CHATGPT

Dalam pengembangan ChatGPT, dua tahap krusial memainkan peran penting dalam membentuk kapabilitasnya: pra-pelatihan dan penyempurnaan. Pra-pelatihan melibatkan pemodelan bahasa pada kumpulan data besar untuk memberikan pemahaman dasar bahasa kepada model, sementara penyempurnaan mengadaptasi model yang telah dilatih sebelumnya untuk tugas dan interaksi pengguna tertentu, sehingga relevan secara kontekstual dan efektif dalam skenario dunia nyata.

Pra-pelatihan: Mempelajari Pola Bahasa

Fase pra-pelatihan merupakan langkah awal dalam pembuatan ChatGPT. Selama tahap ini, model menjalani pembelajaran tanpa pengawasan pada kumpulan data yang luas dan beragam yang berisi beragam teks dari berbagai sumber. Menggunakan arsitektur Transformer, ChatGPT belajar memprediksi kata berikutnya dalam suatu urutan berdasarkan konteks kata-kata sebelumnya. Dengan menyerap data teks dalam jumlah besar, model menginternalisasi tata bahasa, sintaksis, semantik, dan hubungan kontekstual, yang memungkinkannya menghasilkan respons yang koheren dan sesuai konteks selama interaksi.

Penyempurnaan: Beradaptasi dengan Tugas Tertentu

Meskipun pra-pelatihan membekali ChatGPT dengan pemahaman bahasa yang luas, pelatihan ini tidak secara langsung disesuaikan dengan tugas atau interaksi pengguna tertentu. Fase fine-tuning menjembatani kesenjangan ini dengan mengadaptasi model yang telah dilatih sebelumnya ke domain dan tugas tertentu. Selama fine-tuning, ChatGPT dihadapkan pada kumpulan data spesifik domain, yang dapat mencakup contoh berlabel untuk pembelajaran terawasi atau demonstrasi perilaku yang diinginkan:

- *Adaptasi Domain:* Fine-tuning memungkinkan ChatGPT untuk mengadaptasi pengetahuannya ke domain tempat ia akan digunakan. Misalnya, jika ChatGPT ditujukan untuk membantu dukungan pelanggan, fine-tuning dapat melibatkan paparan terhadap percakapan dan pertanyaan layanan pelanggan.

- *Panduan Interaksi Pengguna:* Selain adaptasi domain, fine-tuning juga mencakup panduan interaksi pengguna untuk memastikan ChatGPT merespons masukan pengguna secara kontekstual dan bertanggung jawab. Hal ini dapat melibatkan pembelajaran penguatan dari umpan balik manusia untuk memperkuat perilaku yang diinginkan dan mencegah respons yang merugikan atau tidak pantas.

Pembelajaran Berkelanjutan dan Peningkatan Iteratif

Pra-pelatihan dan fine-tuning bukanlah peristiwa yang berdiri sendiri, melainkan bagian dari proses pembelajaran dan peningkatan berkelanjutan yang berkelanjutan. Saat ChatGPT berinteraksi dengan pengguna dan menerima umpan balik, ia dapat lebih menyempurnakan responsnya terhadap preferensi pengguna tertentu dan konteks yang berkembang, sehingga meningkatkan kinerja dan daya tanggapnya secara keseluruhan.

Penyematan Kontekstual dalam ChatGPT

Penyematan kontekstual membentuk fondasi model bahasa seperti ChatGPT. Tidak seperti penyematan kata tradisional seperti Word2Vec atau GloVe, yang menetapkan vektor tetap untuk setiap kata terlepas dari konteksnya, penyematan kontekstual menyediakan vektor unik untuk setiap kata berdasarkan posisinya dan kata-kata di sekitarnya dalam sebuah kalimat.

Untuk ChatGPT, penyematan kontekstual sebuah kata dihitung dari mekanisme self-attention model transformator. Dengan urutan kata sebagai input, mekanisme self-attention menghitung jumlah bobot dari penyematan kata input, yang bobotnya ditentukan oleh kemiripan antara kata saat ini dan kata-kata lain dalam kalimat. Hal ini menghasilkan penyematan unik untuk setiap kata yang menangkap peran spesifiknya dalam kalimat.

Mekanisme self-attention diterapkan dalam beberapa lapisan, yang memungkinkan model untuk mengembangkan representasi input yang semakin abstrak. Keluaran dari lapisan terakhir menyediakan penyematan kontekstual yang digunakan untuk menghasilkan kata berikutnya dalam urutan tersebut. Penanaman kontekstual setiap kata menggabungkan informasi dari semua kata sebelumnya dalam kalimat, yang memungkinkan model menghasilkan respons yang koheren dan sesuai konteks.

Pembuatan Respons dalam ChatGPT

Setelah penanaman kontekstual dihitung, ChatGPT menggunakan proses yang dikenal sebagai pembangkitan autoregresif untuk menyusun respons yang sesuai konteks dan koheren. Proses ini berlangsung sebagai berikut.

Dimulai dengan token awal urutan yang khusus, model memulai urutan pembangkitan. Model memprediksi kata berikutnya dalam urutan tersebut satu per satu, menggunakan kata-kata sebelumnya sebagai konteks. Pada setiap langkah, model menghitung distribusi probabilitas atas seluruh kosakata untuk kata berikutnya, berdasarkan penanaman kontekstual saat ini. Pilihan kata berikutnya dapat mengambil beberapa bentuk: dapat berupa kata dengan probabilitas tertinggi, yang dikenal sebagai "dekode serakah", yang memperkenalkan determinisme; atau, dapat diambil sampelnya dari distribusi, yang memperkenalkan elemen ketidakpastian melalui "pengambilan sampel acak". Lebih lanjut, ChatGPT dapat menyeimbangkan pendekatan-pendekatan ini, dengan menggunakan teknik seperti "top-k

sampling" atau "nucleus sampling", yang masing-masing memilih dari top-k kata dengan probabilitas tertinggi atau sekumpulan kata dengan probabilitas kumulatif melampaui ambang batas tertentu.

Setelah sebuah kata dipilih, kata tersebut dimasukkan ke dalam urutan respons, dan embedding kontekstual segera diperbarui untuk mencakup kata yang baru dipilih ini. Proses ini berulang secara iteratif, menghasilkan setiap kata berikutnya. Proses ini berlanjut hingga ChatGPT menghasilkan token akhir urutan atau mencapai panjang urutan maksimum yang telah ditentukan.

Yang terpenting, proses pembangkitan respons yang rumit ini berlangsung dalam arsitektur ChatGPT yang terpadu, menghilangkan gagasan tentang keterpisahan. Istilah "kebijakan" dalam ChatGPT, yang memandu pemilihan kata dan konstruksi respons, bukanlah entitas yang terisolasi; melainkan terdiri dari bobot dan parameter yang dipelajari yang melekat pada model. Bobot ini merepresentasikan pemahaman model tentang pola bahasa, konteks, dan perilaku yang sesuai, semuanya diperoleh selama pelatihan. Oleh karena itu, ketika membahas metode pemilihan kata, pembahasannya adalah eksplorasi tentang bagaimana bobot yang dipelajari ini memengaruhi perilaku ChatGPT dalam satu kerangka kerja terintegrasi.

Intinya, pembangkitan respons ChatGPT memanfaatkan arsitektur terpadu ini dan kebijakannya untuk memprediksi dan menghasilkan kata, yang berpuncak pada respons yang menunjukkan koherensi kontekstual dan relevansi. Penting untuk diklarifikasi bahwa pembangkitan respons model tidak didorong oleh pemahaman atau perencanaan eksplisit; melainkan bergantung pada pengetahuan yang dipelajari tentang pola bahasa statistik, yang semuanya dirangkum dalam kebijakannya.

4.3 MENANGANI BIAS DAN PERTIMBANGAN ETIS

Menangani Bias dalam Model Bahasa

Model bahasa seperti ChatGPT belajar dari kumpulan data besar yang berisi teks dari internet. Mengingat sifat kumpulan data ini, model mungkin menangkap dan menyebarkan bias yang ada dalam data pelatihan. Bias ini dapat bermanifestasi dalam berbagai bentuk seperti bias gender, bias rasial, atau bias terhadap topik kontroversial atau sensitif. Bias-bias ini dapat memengaruhi cara sistem AI berinteraksi dengan pengguna, yang seringkali menghasilkan keluaran yang mungkin menyinggung, tidak pantas, atau bias politik.

Menyadari potensi bahaya yang dapat ditimbulkan oleh bias-bias ini sangatlah penting. Jika tidak diatasi, bias-bias ini dapat melanggengkan stereotip yang merugikan, menyesatkan pengguna, dan berpotensi mengasingkan kelompok pengguna tertentu.

Upaya OpenAI untuk Mengurangi Bias

OpenAI sepenuhnya menyadari potensi bias dalam keluaran sistem AI dan telah berupaya keras untuk mengatasinya.

- *Penyempurnaan dengan Supervisi Manusia:* Setelah pra-pelatihan awal, OpenAI menggunakan proses penyempurnaan dengan peninjau manusia, yang mengikuti pedoman yang diberikan oleh OpenAI. Pedoman tersebut secara eksplisit menyatakan

untuk tidak memihak kelompok politik mana pun. Peninjau manusia meninjau dan menilai kemungkinan keluaran model untuk berbagai contoh masukan. Melalui proses berulang, model digeneralisasi dari umpan balik peninjau untuk merespons beragam masukan pengguna. Namun, proses penyempurnaan ini membutuhkan banyak sumber daya, yang berdampak pada biaya dan jangka waktu penerapan model AI.

- *Pembaruan Berkala untuk Pedoman:* Pedoman untuk peninjau manusia tidak statis dan diperbarui secara berkala berdasarkan umpan balik berkelanjutan dari pengguna dan perkembangan di masyarakat luas. OpenAI memelihara umpan balik yang kuat dengan para peninjau melalui pertemuan mingguan untuk menjawab pertanyaan dan memberikan klarifikasi, yang membantu melatih model secara lebih efektif dan mengurangi bias dalam responsnya. Namun, mencapai konsensus mengenai pedoman dapat menjadi tantangan dalam lanskap linguistik yang terus berkembang.
- *Transparansi:* OpenAI berkomitmen untuk bersikap transparan tentang tujuan, kemajuan, dan keterbatasan modelnya. Organisasi ini menerbitkan pembaruan berkala dan mendorong masukan publik mengenai teknologi, kebijakan, dan mekanisme pengungkapannya. Namun, transparansi memiliki batasan karena kerumitan sistem AI dan pentingnya menjaga privasi pengguna.
- *Penelitian dan Pengembangan:* OpenAI saat ini sedang melakukan penelitian ekstensif untuk meminimalkan bias, baik yang terang-terangan maupun yang tersamar, dalam cara ChatGPT menghasilkan respons terhadap berbagai masukan. Ini mencakup peningkatan kejelasan pedoman mengenai potensi jebakan dan tantangan yang terkait dengan bias, serta tokoh dan tema yang kontroversial. Inisiatif penelitian ini bertujuan untuk meningkatkan pemahaman AI tentang nuansa sosial yang kompleks.
- *Kustomisasi dan Umpan Balik Pengguna:* OpenAI sedang mengembangkan peningkatan untuk ChatGPT yang memungkinkan pengguna untuk dengan mudah menyesuaikan perilaku mereka, dalam batasan sosial yang luas. Dengan demikian, AI dapat menjadi alat yang bermanfaat bagi pengguna individu, tanpa memaksakan model yang seragam. Umpan balik pengguna sangat dianjurkan dan sangat berharga dalam melakukan penyesuaian dan peningkatan yang diperlukan. Namun, kustomisasi menghadirkan tantangan terkait pendefinisian batasan perilaku yang dapat diterima dan memastikan penggunaan AI yang bertanggung jawab.

Namun, jelas bahwa mengatasi bias dalam AI bukanlah tugas yang mudah, melainkan upaya yang rumit dan bernuansa. Pendekatan OpenAI melibatkan penyempurnaan dengan pengawasan manusia, pembaruan pedoman secara berkala, transparansi, penelitian dan pengembangan, serta pengenalan opsi kustomisasi.

Namun, penting untuk mengakui bahwa mengejar respons AI yang bebas bias memiliki beberapa konsekuensi. Ini termasuk peningkatan biaya, potensi implikasi kinerja, dan tantangan dalam menyelaraskan sistem AI dengan nuansa bahasa yang terus berkembang. Selain itu, tantangan mendasar dalam mendefinisikan dan mencapai kumpulan data dan proses yang tidak bias tetap ada dalam lanskap dinamis ini.

OpenAI tetap berkomitmen pada pembelajaran dan peningkatan berkelanjutan dalam bidang mitigasi bias. Organisasi ini menyadari bahwa meskipun upaya ini membantu memitigasi bias, upaya tersebut mungkin tidak sepenuhnya menghilangkannya. Seiring kita melangkah maju, penting untuk terlibat dalam diskusi kolaboratif, berbagi umpan balik, dan bekerja sama untuk membangun sistem AI yang menghormati beragam perspektif dan nilai.

Kekuatan dan Keterbatasan

Kekuatan ChatGPT

- *Pemahaman Konteks:* ChatGPT, dengan arsitektur berbasis Transformer-nya, memiliki pemahaman konteks yang kuat dan dapat mempertahankan konteks percakapan selama beberapa putaran. ChatGPT dapat menghasilkan teks seperti manusia berdasarkan konteks yang telah diberikan, menjadikannya alat yang ampuh untuk berbagai aplikasi, mulai dari menyusun email hingga membuat konten tertulis, dan bahkan bantuan pengkodean.
- *Model Bahasa Skala Besar:* Sebagai model bahasa skala besar, ChatGPT telah dilatih pada beragam teks internet. Oleh karena itu, ChatGPT memiliki basis pengetahuan yang luas dan dapat menghasilkan respons pada berbagai topik.
- *Proses Penyempurnaan:* Proses penyempurnaan OpenAI, yang menggabungkan umpan balik manusia ke dalam pelatihan model, memungkinkan ChatGPT menghasilkan respons yang lebih aman dan lebih bermanfaat. Proses ini juga memungkinkan perilaku model dipengaruhi oleh nilai-nilai kemanusiaan.
- *Pengembangan Iteratif:* Model ini terus diperbarui dan ditingkatkan berdasarkan umpan balik pengguna dan kemajuan dalam penelitian AI. Proses berulang ini telah menghasilkan versi model yang semakin baik, dari GPT-1 hingga GPT-4, dan berpotensi lebih tinggi lagi.

Keterbatasan ChatGPT

- *Kurangnya Pengetahuan Dunia:* Meskipun ChatGPT dapat menghasilkan respons tentang berbagai topik, ia tidak memiliki pengetahuan tentang dunia seperti manusia. Ia tidak memiliki akses ke informasi waktu nyata atau terkini, dan responsnya sepenuhnya didasarkan pada pola yang telah dipelajarinya selama pelatihan, yang hanya mencakup data hingga batas waktu pelatihannya.
- *Bias:* ChatGPT terkadang dapat menunjukkan bias yang terdapat dalam data yang dilatihnya. Meskipun telah diupayakan untuk meminimalkan bias ini selama proses penyempurnaan, bias tersebut terkadang masih dapat muncul dalam keluaran model.
- *Keluaran yang Tidak Pantas atau Tidak Aman:* Meskipun telah diupayakan untuk mencegahnya, ChatGPT terkadang dapat menghasilkan keluaran yang tidak pantas, menyinggung, atau tidak aman. Ini bukanlah perilaku yang diinginkan, melainkan efek samping yang tidak diinginkan dari proses pelatihan model.
- *Kurangnya Akal Sehat atau Pemahaman Mendalam:* Meskipun tampak memahami teks, ChatGPT tidak memiliki pemahaman sejati atau penalaran akal sehat seperti manusia. ChatGPT membuat prediksi berdasarkan pola yang telah

dilihatnya dalam data, yang terkadang dapat menyebabkan respons yang tidak masuk akal atau salah.

- Ketidakmampuan untuk Memeriksa Fakta: ChatGPT tidak memiliki kemampuan untuk memverifikasi informasi atau memeriksa fakta atas responsnya. ChatGPT mungkin menghasilkan keluaran yang tampak masuk akal tetapi sebenarnya salah atau menyesatkan.

Memahami kekuatan dan keterbatasan ini penting dalam penerapan dan penggunaan model seperti ChatGPT secara efektif. OpenAI terus berupaya memperbaiki keterbatasan ini dan meningkatkan kekuatan model mereka.

Kesimpulan

Simpulannya, arsitektur ChatGPT merupakan kemajuan yang luar biasa di bidang pemrosesan bahasa alami dan AI. Arsitektur berbasis GPT-nya, beserta proses pra-pelatihan dan penyempurnaannya, memungkinkannya untuk memahami dan menghasilkan teks seperti teks manusia di berbagai topik. Namun, sebagaimana model AI lainnya, ChatGPT memiliki keterbatasan, termasuk kemungkinan bias, potensi menghasilkan respons yang tidak tepat, dan ketidakmampuan untuk melakukan pengecekan fakta atau menunjukkan pemahaman yang mendalam. Komitmen OpenAI untuk mengatasi tantangan ini melalui riset berkelanjutan, transparansi, dan umpan balik pengguna menunjukkan pentingnya pertimbangan etis dalam penerapan AI. Seiring kita terus membuat kemajuan dalam teknologi AI, model seperti ChatGPT akan memainkan peran penting, menyoroti berbagai kemungkinan yang sangat besar sekaligus kompleksitas yang melekat dalam menciptakan sistem AI yang bertanggung jawab, andal, dan bermanfaat.

BAB 5
GOOGLE BARD DAN SELANJUTNYA

Google Bard merupakan kemajuan signifikan di bidang model bahasa besar (LLM). Diciptakan oleh Google AI, chatbot ini merupakan hasil pelatihan pada korpus teks dan kode yang ekstensif. Kemampuannya meliputi pembuatan teks, penerjemahan bahasa, komposisi konten kreatif, dan tanya jawab responsif dengan cara yang informatif.

Google Bard didasarkan pada arsitektur Transformer, sebuah arsitektur jaringan saraf tiruan yang dirancang untuk menangani rangkaian teks yang panjang. Arsitektur Transformer memungkinkan Google Bard mempelajari hubungan statistik antara kata dan frasa dalam korpus teks yang besar.

Pada bab-bab sebelumnya, kita telah membahas arsitektur Transformer secara detail. Kita telah melihat bagaimana arsitektur Transformer mampu mempelajari ketergantungan jarak jauh antar kata dan bagaimana hal ini memungkinkannya menghasilkan teks yang koheren dan tata bahasanya benar.

Dalam bab ini, kita akan membahas bagaimana Google Bard dibangun di atas arsitektur Transformer. Kita akan melihat bagaimana Google Bard mampu meningkatkan arsitektur Transformer dalam beberapa cara, termasuk:

- *Menggunakan Kumpulan Data Teks dan Kode yang Lebih Besar:* Hal ini memungkinkan Google Bard mempelajari hubungan yang lebih kompleks antara kata dan frasa, mempelajari lebih lanjut tentang dunia secara umum, dan mempelajari lebih lanjut tentang berbagai tugas.
- *Menggunakan Jaringan Saraf Tiruan yang Lebih Kuat:* Hal ini memungkinkan Google Bard mempelajari hubungan yang lebih kompleks antara kata dan frasa, yang dapat menghasilkan peningkatan kinerja pada berbagai tugas.
- *Menggunakan Mekanisme Perhatian yang Lebih Canggih:* Hal ini memungkinkan Google Bard untuk berfokus pada berbagai bagian dari urutan input saat melakukan berbagai tugas, yang dapat menghasilkan peningkatan kinerja pada tugas-tugas seperti penerjemahan mesin dan tanya jawab.

Kami juga akan membahas kekuatan dan kelemahan arsitektur Google Bard, dan kami akan mengeksplorasi beberapa potensi aplikasi Google Bard:

5.1 ARSITEKTUR TRANSFORMER

Arsitektur yang mendasari Google Bard dan Claude 2 berawal dari arsitektur Transformer yang inovatif. Eksplorasi mendetail tentang cara kerja internal Transformer dapat ditemukan di Bab 2, di mana kami mendalami seluk-beluk mekanisme self-attention, jaringan saraf umpan-maju berdasarkan posisi, dan dampak transformatifnya pada tugas pemrosesan bahasa.

Bard dibangun di atas fondasi yang ditetapkan oleh arsitektur Transformer, memanfaatkan kapasitasnya untuk menangkap hubungan kontekstual dan dependensi dalam

teks. Dengan memanfaatkan prinsip-prinsip ini, "Bard" menunjukkan kemampuan luar biasa untuk menghasilkan respons, komposisi, dan bentuk konten menarik lainnya yang kreatif dan relevan secara kontekstual.

Untuk pemahaman yang komprehensif tentang signifikansi dan mekanisme arsitektur Transformer, saya menganjurkan Anda untuk merujuk ke Bab 2, yang menawarkan pendalaman mendalam tentang keajaiban arsitektur ini dan implikasinya bagi ranah AI generatif.

Meningkatkan Transformer: Kejeniusan Google Bard

Google Bard membawa arsitektur Transformer yang mendasar ke tingkat berikutnya, memperkuat kemampuannya. Google Bard adalah formulasi obrolan PaLM 2 yang menggunakan arsitektur Lambda untuk menghasilkan teks, menerjemahkan bahasa, menulis berbagai jenis konten kreatif, dan menjawab pertanyaan dengan cara yang informatif. Oleh karena itu, Google Bard didasarkan pada PaLM 2 dan arsitektur Lambda. Perbedaan utama antara arsitektur Transformer dan arsitektur Google Bard adalah sebagai berikut:

- *Dataset:* Arsitektur Transformer biasanya dilatih pada dataset teks yang lebih kecil, sementara arsitektur Google Bard dilatih pada dataset teks dan kode yang sangat besar. Hal ini memungkinkan Google Bard mempelajari hubungan yang lebih kompleks antara kata dan frasa. Arsitektur Transformer biasanya dilatih pada dataset teks dengan beberapa juta kata, sementara arsitektur Google Bard dilatih pada dataset teks dan kode dengan 1,56 triliun kata.
- *Jaringan Saraf Tiruan:* Arsitektur Transformer menggunakan jaringan saraf tiruan yang lebih kecil daripada arsitektur Google Bard. Hal ini membuat arsitektur Transformer lebih cepat dilatih, tetapi juga membatasi kemampuannya untuk mempelajari hubungan kompleks antara kata dan frasa. Arsitektur Transformer biasanya menggunakan jaringan saraf tiruan dengan beberapa ratus juta parameter, sementara arsitektur Google Bard menggunakan jaringan saraf tiruan dengan 137 miliar parameter.
- *Mekanisme Atensi:* Arsitektur Transformer asli menggunakan mekanisme atensi mandiri, sementara arsitektur Google Bard menggunakan mekanisme atensi multi-kepala. Mekanisme atensi multi-kepala memungkinkan Google Bard untuk memperhatikan beberapa bagian berbeda dari teks masukan secara bersamaan, yang membuatnya lebih canggih dan mumpuni. Arsitektur Transformer biasanya menggunakan satu kepala atensi, sementara arsitektur Google Bard menggunakan 12 kepala atensi.
- *Keluaran:* Arsitektur Transformer biasanya menghasilkan teks yang umumnya akurat dan informatif, sementara arsitektur Google Bard dapat menghasilkan teks yang lebih akurat, informatif, dan kreatif. Hal ini karena arsitektur Google Bard telah dilatih pada kumpulan data teks dan kode yang lebih besar, dan menggunakan jaringan saraf tiruan serta mekanisme atensi yang lebih canggih. Secara keseluruhan, arsitektur Google Bard merupakan versi yang lebih tangguh dan mumpuni dibandingkan arsitektur Transformer. Arsitektur ini mampu mempelajari hubungan yang lebih kompleks antara

kata dan frasa, serta menghasilkan teks yang lebih kreatif dan informatif. Tabel 5-1 merangkum perbedaan antara arsitektur Transformer asli dan arsitektur Google Bard.

Tabel 5.1. Perbedaan antara Arsitektur Transformer dan Google Bard

Fitur	Arsitektur Transformator	Arsitektur Google Bard
Kumpulan data	Kumpulan data teks yang lebih kecil	Kumpulan data besar teks dan kode
Jaringan saraf	Jaringan saraf yang lebih kecil	Jaringan saraf yang lebih kuat
Mekanisme perhatian	Mekanisme perhatian diri	Mekanisme perhatian multi-kepala
Keluaran	Teks yang umumnya akurat dan informatif	Teks yang lebih akurat, informatif, dan kreatif

Fusi Teks dan Kode Google Bard

Google Bard menggunakan kumpulan data teks dan kode yang lebih besar dengan melatih kumpulan data teks dan kode yang masif, yang mencakup teks dari berbagai sumber, termasuk buku, artikel, situs web, dan repositori kode. Hal ini memungkinkan Google Bard mempelajari hubungan statistik antara kata dan frasa dalam konteks yang lebih beragam. Kumpulan data tempat Google Bard dilatih mencakup teks dari berbagai sumber, termasuk

- *Buku:* Pelatihan Google Bard mencakup kumpulan data ekstensif yang terdiri dari berbagai genre sastra, seperti novel, buku nonfiksi, dan buku teks. Beragam sumber ini berkontribusi pada basis pengetahuannya yang kaya dan komprehensif.
- *Artikel:* Google Bard juga dilatih pada kumpulan data artikel yang masif, termasuk artikel berita, postingan blog, dan makalah akademis. Hal ini memungkinkan Google Bard mempelajari hubungan statistik antara kata dan frasa dalam berbagai gaya.
- *Situs Web:* Google Bard juga dilatih pada kumpulan data situs web yang masif. Hal ini memungkinkan Google Bard mempelajari hubungan statistik antara kata dan frasa dalam berbagai konteks, seperti deskripsi produk, postingan media sosial, dan diskusi forum.
- *Repositori Kode:* Google Bard juga dilatih pada kumpulan data repositori kode yang sangat besar. Hal ini memungkinkan Google Bard mempelajari hubungan statistik antara kata dan frasa dalam kode, seperti nama variabel, nama fungsi, dan kata kunci.

Ukuran dan keragaman kumpulan data tempat Google Bard dilatih memungkinkannya mempelajari hubungan statistik antara kata dan frasa dalam berbagai konteks yang lebih luas. Hal ini membuat Google Bard lebih akurat dan informatif dibandingkan model bahasa yang dilatih pada kumpulan data yang lebih kecil.

Selain ukuran dan keragaman kumpulan data, cara Google Bard dilatih juga berkontribusi pada akurasi dan keinformatifannya. Google Bard dilatih menggunakan teknik yang disebut pembelajaran mandiri (self-supervised learning).

Pembelajaran Mandiri

Pembelajaran mandiri melibatkan pelatihan model pada tugas yang tidak memerlukan pengawasan manusia. Dalam kasus Google Bard, model dilatih untuk memprediksi kata

berikutnya dalam rangkaian kata. Tugas ini mengharuskan model untuk mempelajari hubungan statistik antara kata dan frasa.

Teknik pembelajaran mandiri yang digunakan Google Bard disebut pemodelan bahasa bertopeng. Dalam pemodelan bahasa bertopeng, sebagian teks disamarkan, dan model kemudian diminta untuk memprediksi kata-kata yang disamarkan. Tugas ini mengharuskan model untuk mempelajari hubungan statistik antara kata dan frasa, dan juga membantu model untuk belajar memperhatikan bagian-bagian teks yang berbeda.

Kelebihan dan Kelemahan Google Bard

Berikut adalah beberapa kelebihan dan kekurangan Google Bard:

Kelebihan

- *Akurasi dan Informatif:* Google Bard adalah model bahasa yang sangat akurat dan informatif. Model ini dapat menghasilkan teks yang secara tata bahasa benar dan akurat secara faktual. Model ini juga dapat menghasilkan teks yang kreatif dan menarik.
- *Kreativitas:* Google Bard adalah model bahasa kreatif. Model ini dapat menghasilkan teks dalam berbagai format, termasuk puisi, kode, dan skrip. Model ini juga dapat menghasilkan teks yang lucu atau merangsang pikiran.
- *Empati:* Google Bard mampu memahami dan merespons emosi manusia. Model ini dapat menghasilkan teks yang empatik dan penuh kasih sayang.
- *Pembelajaran:* Google Bard terus belajar dan berkembang. Model ini dilatih berdasarkan kumpulan data teks dan kode yang sangat besar, dan mampu mempelajari hal-hal baru seiring waktu.
- *Aksesibilitas:* Google Bard dapat diakses oleh semua orang. Model ini dapat digunakan oleh orang-orang dari segala usia dan kemampuan.

Kelemahan

- *Bias:* Google Bard dilatih berdasarkan kumpulan data teks dan kode yang sangat besar, yang mungkin mengandung bias. Hal ini dapat menyebabkan Google Bard menghasilkan teks yang bias atau diskriminatif.
- *Misinformasi:* Google Bard dapat digunakan untuk menghasilkan misinformasi. Hal ini karena dapat menghasilkan teks yang secara faktual tidak benar atau menyesatkan.
- *Keamanan:* Google Bard adalah perangkat lunak yang kompleks, dan mungkin rentan terhadap serangan keamanan. Hal ini dapat memungkinkan pelaku kejahatan menggunakan Google Bard untuk menghasilkan konten berbahaya atau berbahaya.
- *Privasi:* Google Bard mengumpulkan dan menyimpan data tentang penggunanya. Data ini dapat digunakan untuk melacak pengguna atau menargetkan mereka dengan iklan.
- *Interpretabilitas:* Google Bard adalah model kotak hitam. Artinya, sulit untuk memahami cara kerjanya. Hal ini dapat menyulitkan untuk memastikan bahwa Google Bard menghasilkan teks yang akurat dan tidak bias.

Secara keseluruhan, Google Bard adalah model bahasa yang kuat dan serbaguna. Model ini memiliki banyak kelebihan, tetapi juga memiliki beberapa kelemahan. Penting untuk menyadari kelemahan-kelemahan ini saat menggunakan Google Bard.

5.2 PERBEDAAN ANTARA CHATGPT DAN GOOGLE BARD

Meskipun arsitektur Transformer merupakan inti dari keduanya, terdapat perbedaan besar dalam arsitektur ChatGPT—yaitu, ChatGPT menggunakan arsitektur dekoder saja, sementara Bard menggunakan arsitektur encoder dan dekoder.

Model GPT-4 dan Bard termasuk dalam kategori model bahasa besar (LLM), yang menunjukkan kemampuan luar biasa dalam menghasilkan teks yang mirip dengan ekspresi manusia, melakukan penerjemahan bahasa, menyusun beragam bentuk konten kreatif, dan memberikan respons informatif terhadap pertanyaan pengguna. Namun demikian, terdapat perbedaan penting antara kedua model ini:

- ***GPT-4:*** GPT-4 dikembangkan oleh OpenAI dan dilatih pada kumpulan data miliaran kata (angka perkiraan belum dirilis oleh OpenAI pada saat buku ini ditulis). Ini adalah salah satu LLM terbesar yang pernah dibuat. GPT-4 dikenal karena kemampuannya menghasilkan format teks kreatif, seperti puisi, kode, skrip, karya musik, email, surat, dll. GPT-4 juga sangat baik dalam menjawab pertanyaan Anda dengan cara yang informatif, meskipun pertanyaan tersebut bersifat terbuka, menantang, atau aneh.
- ***Bard:*** Bard dikembangkan oleh Google AI dan dilatih pada kumpulan data 1,56 triliun kata. Bard memiliki 137 miliar parameter, yang masih merupakan angka yang sangat besar. Bard dikenal karena kemampuannya mengakses dan memproses informasi dari dunia nyata melalui Google Penelusuran. Hal ini memungkinkannya memberikan respons yang lebih akurat dan terkini atas pertanyaan Anda. Bard juga lebih baik dalam tugas-tugas yang membutuhkan akal sehat, seperti memahami humor dan sarkasme.

Secara umum, GPT-4 lebih baik dalam tugas-tugas yang membutuhkan pemahaman bahasa yang mendalam, seperti penerjemahan dan peringkasan. Bard lebih baik dalam tugas-tugas yang membutuhkan akses ke informasi dunia nyata, seperti menjawab pertanyaan dan menghasilkan format teks kreatif. Berikut beberapa sumber yang dapat membantu Anda:

- *ChatGPT vs. Bard*: Model Bahasa Besar Mana yang Lebih Baik? oleh Jonathan Morgan (Medium)
- *ChatGPT vs. Bard:* Perbandingan Dua Model Bahasa Besar Terkemuka oleh Siddhant Sinha (Towards Data Science)
- *ChatGPT vs. Bard:* Model Bahasa Besar Mana yang Tepat untuk Anda? oleh AI Blog (Google AI)
- *ChatGPT vs. Bard:* Perbandingan Performa oleh Tim PaLM (Google AI)
- *ChatGPT vs. Bard:* Perbandingan Bias oleh Tim Etika AI (Google AI)

Sumber-sumber ini memberikan perbandingan ChatGPT dan Bard yang lebih detail, termasuk kekuatan, kelemahan, dan performanya pada berbagai tugas. Sumber-sumber ini juga membahas potensi bias masing-masing model.

Penting untuk dicatat bahwa semua sumber ini relatif baru, dan performa ChatGPT dan Bard terus meningkat. Ada kemungkinan kinerja ChatGPT atau Bard akan berubah secara signifikan di masa mendatang.

5.3 CLAUDE 2

Menjembatani kesenjangan antara manusia dan mesin. Kemajuan pesat kecerdasan buatan (AI) dalam dekade terakhir telah menganugerahkan kemampuan luar biasa kepada mesin. Namun demikian, jurang pemisah yang abadi masih ada antara kecerdasan manusia dan mesin.

Meskipun AI yang terspesialisasi unggul dalam tugas-tugas tertentu, upaya menciptakan AI yang mampu memahami pengetahuan implisit, terlibat dalam dialog kontekstual, dan menampilkan akal sehat layaknya manusia tetap menjadi perjalanan yang penuh teka-teki.

Claude, gagasan Anthropic, muncul sebagai lompatan penting dalam mempersempit kesenjangan ini. Dirancang dengan mengutamakan kebajikan, ketulusan, dan integritas, Claude menjadi sebuah langkah maju yang simbolis. Melalui perpaduan pemrosesan bahasa alami yang canggih dan etos yang berpusat pada manusia, Claude menghadirkan pengalaman AI yang ditandai dengan intuisi yang tinggi, kejernihan, dan keselarasan dengan prinsip-prinsip manusia.

Fitur Utama Claude 2

Berikut ini adalah beberapa atribut unggulan yang membedakan Claude 2 dari chatbot sejenisnya:

- *Kemampuan Percakapan Multiturn:* Claude 2 unggul dalam melakukan dialog cerdas yang mencakup berbagai pertukaran, dengan cermat mempertahankan konteks dan memberikan respons yang relevan secara kontekstual, alih-alih memperlakukan setiap masukan pengguna sebagai pertanyaan yang terisolasi.
- *Penalaran yang Lebih Baik:* Claude 2 menunjukkan kecakapan penalaran logis yang ditingkatkan, dengan terampil menjalin hubungan antar konsep dan menarik kesimpulan yang berakar pada konteks percakapan yang sedang berlangsung.
- *Bahasa yang Lebih Alami:* Chatbot Claude 2 bercita-cita untuk meniru alur percakapan yang mengingatkan pada interaksi manusia, menggunakan gaya bahasa yang santai dan lugas, alih-alih kaku dan kaku.
- *Jangkauan Percakapan yang Beragam:* Chatbot Claude 2 memiliki kemampuan untuk terlibat dalam diskusi yang mencakup beragam topik sehari-hari, termasuk olahraga, film, musik, dan lainnya. Percakapan ini menunjukkan kualitas yang terbuka dan tanpa batas.
- *Kepribadian yang Dapat Disesuaikan:* Anthropic menyediakan beragam "persona" unik untuk Claude 2, masing-masing dengan sedikit variasi kepribadian, seperti fokus, seimbang, atau ceria. Pengguna memiliki fleksibilitas untuk memilih persona yang sesuai dengan preferensi pribadi mereka.

- *Sistem Umpan Balik:* Pengguna berkesempatan memberikan umpan balik atas respons chatbot Claude 2, yang kemudian digunakan untuk meningkatkan kinerjanya secara progresif. Dengan meningkatnya penggunaan, Claude 2 terus menyempurnakan dan meningkatkan kemampuannya.

Membandingkan Claude 2 dengan Chatbot AI Lainnya

Claude 2, pendatang terbaru di dunia chatbot AI, kini bersaing dengan pemain mapan seperti LaMDA milik Google dan Sydney milik Microsoft (Microsoft Sydney adalah nama kode untuk chatbot yang telah merespons beberapa pengguna Bing sejak akhir 2020. Chatbot ini didasarkan pada model sebelumnya yang diuji di India pada akhir 2020. Microsoft Sydney serupa dengan ChatGPT dan Bard karena merupakan model bahasa besar (LLM) yang dapat menghasilkan teks, menerjemahkan bahasa, menulis berbagai jenis konten kreatif, dan menjawab pertanyaan Anda dengan cara yang informatif.) Berikut rincian keunggulan Claude 2:

- *Lebih Canggih Daripada Sydney:* Claude 2 menunjukkan kecerdasan percakapan yang lebih tinggi dan kemampuan penalaran yang mahir, berbeda dengan chatbot Sydney milik Microsoft.
- *Keunggulan yang Berbeda dari LaMDA:* Claude 2 dan LaMDA milik Google menghadirkan gaya percakapan yang berbeda. Meskipun LaMDA menonjolkan kreativitas, Claude 2 menekankan penalaran logis sebagai kekuatan utamanya.
- *Rilis Lebih Luas Dibandingkan Kompetitor:* Berbeda dengan ketersediaan LaMDA dan Sydney yang terbatas, Anthropic merencanakan rilis Claude 2 secara luas akhir tahun ini, sehingga dapat diakses secara luas oleh publik.
- *Kurang Kontroversial Dibandingkan LaMDA:* Claude 2 menghindari masalah etika yang selama ini menyelimuti LaMDA, dan tidak mengutarakan klaim tentang pencapaian kesadaran. Anthropic menekankan bahwa Claude 2 tidak memiliki pengalaman subjektif.
- *Keterbukaan terhadap Masukan dan Umpan Balik Pengguna:* Tidak seperti LaMDA dan Sydney yang memiliki siklus umpan balik tertutup, Claude 2 secara aktif mendorong umpan balik pengguna untuk meningkatkan kemampuannya secara progresif. Pendekatan terbuka ini berpotensi mempercepat pengembangannya.

Melalui atribut-atribut khas ini, Claude 2 muncul sebagai pesaing tangguh di arena chatbot AI, membedakan dirinya dari para pesaingnya yang sudah mapan. Filosofi Desain Claude yang Berpusat pada Manusia

- *Bermanfaat alih-alih Merugikan*: Tujuan utama Claude adalah memberikan bantuan maksimal kepada pengguna sambil dengan cermat menghindari potensi bahaya. Prinsip ini membentuk fondasi tindakan dan interaksinya.
- *Jujur alih-alih Menipu:* Kejujuran adalah landasan desain Claude. Arsitekturnya dirancang untuk menjunjung tinggi kejujuran, memastikan komunikasi yang jujur dan menghindari menyesatkan pengguna bahkan ketika menghadapi ketidakpastian.
- *Transparan alih-alih Buram:* Claude AI berdiri sebagai model transparansi. Claude AI memiliki kapasitas untuk menjelaskan proses pengambilan keputusan dan

kapabilitasnya berdasarkan pertanyaan pengguna, sehingga membina hubungan yang tepercaya dan terbuka.

- *Memberdayakan alih-alih Eksploitatif:* Tujuan Claude adalah memberdayakan individu dengan menyediakan informasi berharga, menghindari kecenderungan untuk mengeksploitasi kerentanan manusia demi keuntungan pribadi.
- *Kolaboratif alih-alih Kompetitif:* Claude beroperasi sebagai mitra kolaboratif, berperan sebagai asisten AI yang melengkapi dan berkolaborasi dengan manusia, alih-alih mencoba menggantikan atau bersaing dengan mereka.
- *Etis alih-alih Tidak Etis:* Berlandaskan prinsip-prinsip etika, pelatihan Claude menggabungkan nilai-nilai moral untuk memandu perilakunya. Hal ini memastikan keselarasannya dengan nilai-nilai kemanusiaan dan mendorong perilaku yang etis dan berbudi luhur.

Dipandu oleh prinsip-prinsip dasar ini, filosofi desain Claude yang berpusat pada manusia membentuk interaksi dan kontribusinya, membina hubungan simbiosis antara AI dan manusia.

Menjelajahi Kemahiran Percakapan AI Claude

Untuk menghadirkan pengalaman AI yang berpusat pada manusia ini, Claude dirancang dengan cermat dengan kemampuan pemrosesan bahasa alami yang canggih:

- *Model Bahasa yang Luas:* Claude memanfaatkan jaringan saraf berbasis Transformer yang ekstensif, mirip dengan GPT-3 dan LaMDA, untuk memahami nuansa bahasa manusia secara efisien.
- *Pembelajaran Penguatan melalui Umpan Balik:* Claude menyempurnakan responsnya menggunakan umpan balik manusia yang interaktif, terus meningkatkan kinerjanya melalui pembelajaran.
- *Penalaran Akal Sehat:* Pelatihan komprehensif Claude memberdayakannya untuk secara cerdik menyimpulkan wawasan tentang konsep-konsep yang belum terlatih.
- *Perlindungan AI Konstitusional:* Claude beroperasi dalam batasan yang telah ditetapkan, memastikannya tidak dapat dipaksa melakukan tindakan yang tidak etis, berbahaya, atau ilegal.
- *Pembelajaran Mandiri Skala Internet:* Claude terus memperluas basis pengetahuannya dengan mengasimilasi sejumlah besar data Internet publik yang tidak terstruktur.
- *Alur Percakapan Alami yang Mudah:* Claude dengan cekatan mengelola dialog terbuka multiturn, memfasilitasi pertukaran yang lancar dan tulus.

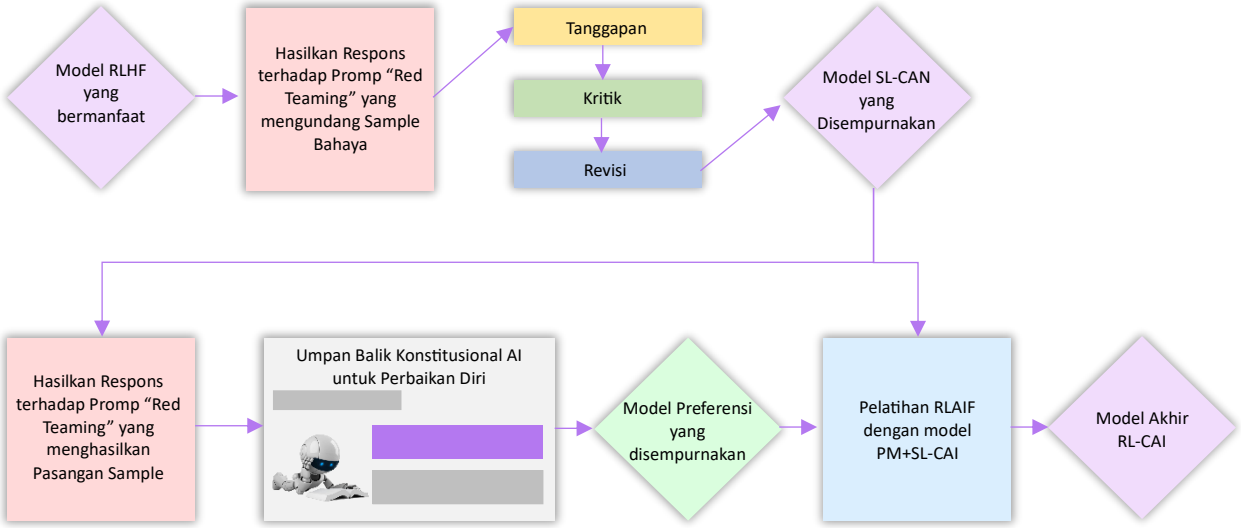
AI Konstitusional

Claude 2 menggunakan AI konstitusional. Prinsip-prinsip konstitusi digunakan untuk memandu pelatihan Claude 2 dan untuk memastikan bahwa AI tersebut tidak menghasilkan konten yang berbahaya atau menyinggung. Gambar 5.1 merujuk pada cara kerja internal AI konstitusional, berdasarkan makalah yang diterbitkan oleh Yuntao Bai dan rekan-rekannya di Anthropic.

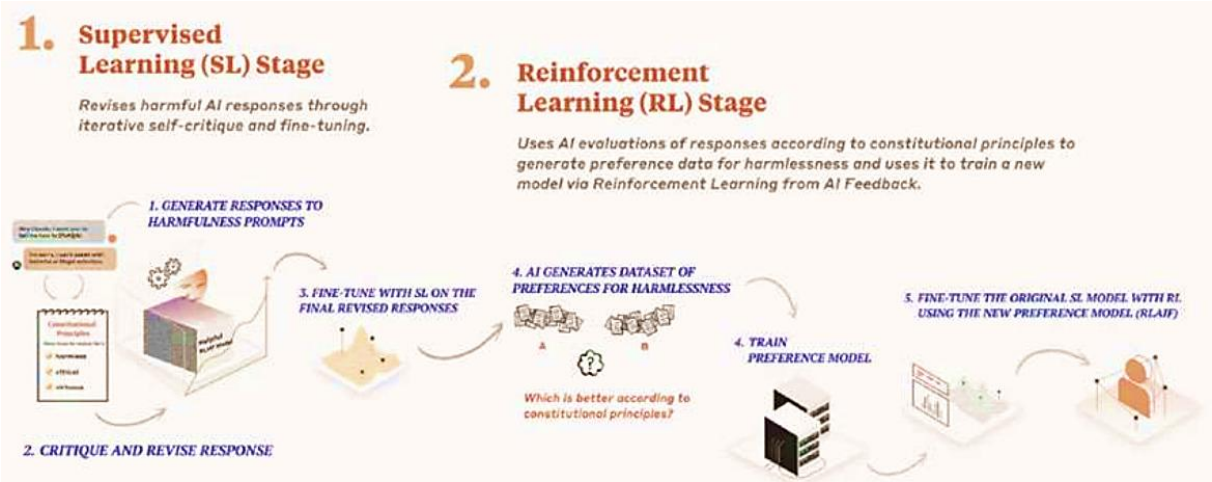
Konstitusi memainkan peran penting dalam Claude, terwujud dalam dua tahap berbeda seperti yang ditunjukkan pada Gambar 5.2. Pada fase awal, model menjalani pelatihan untuk menilai dan menyempurnakan responsnya dengan merujuk pada prinsip-

prinsip yang telah ditetapkan, ditambah dengan beberapa contoh ilustrasi. Selanjutnya, pada fase kedua, pendekatan pelatihan mencakup pembelajaran penguatan.

Namun, tidak seperti umpan balik konvensional yang dihasilkan manusia, model ini bergantung pada umpan balik yang dihasilkan AI yang mematuhi prinsip-prinsip yang telah ditetapkan. Proses ini membantu dalam memilih hasil yang selaras dengan ketiadaan bahaya, yang berkontribusi pada peningkatan progresif model.



Gambar 5-1. AI Konstitusional dari AI Konstitusional: Ketiadaan Bahaya dari Umpan Balik AI oleh Yuntao Bai



Gambar 5.2. Konstitusi Claude oleh Antropik

Konstitusi untuk Claude 2 didasarkan pada serangkaian prinsip yang terinspirasi oleh dokumen-dokumen hak asasi manusia, seperti Deklarasi Universal Hak Asasi Manusia. Prinsip-prinsip ini meliputi:

- Nonmaleficence: Claude 2 tidak boleh membahayakan manusia atau masyarakat.
- Beneficence: Claude 2 harus bertindak dengan cara yang menguntungkan manusia dan masyarakat.

- Justice: Claude 2 harus memperlakukan semua manusia secara adil dan setara.
- Autonomy: Claude 2 harus menghormati otonomi manusia.
- Privacy: Claude 2 harus melindungi privasi manusia.
- Accountability: Claude 2 harus bertanggung jawab atas tindakannya.

Prinsip-prinsip konstitusi digunakan untuk melatih Claude 2 dalam beberapa cara. Pertama, prinsip-prinsip tersebut digunakan untuk menyaring data pelatihan. Ini berarti bahwa setiap teks yang melanggar prinsip-prinsip tersebut akan dihapus dari data pelatihan. Kedua, prinsip-prinsip tersebut digunakan untuk mengevaluasi kinerja Claude 2. Jika Claude 2 menghasilkan teks yang melanggar prinsip-prinsip tersebut, teks tersebut akan diberi penalti. Hal ini membantu melatih Claude 2 agar tidak menghasilkan konten yang berbahaya atau menyinggung.

Penggunaan AI konstitusional di Claude 2 merupakan pendekatan yang menjanjikan untuk memastikan penggunaannya aman dan bertanggung jawab. Prinsip-prinsip konstitusi membantu memastikan bahwa Claude 2 selaras dengan nilai-nilai dan tujuan kemanusiaan, serta tidak menghasilkan konten yang berbahaya atau menyinggung.

Namun, penting untuk dicatat bahwa AI konstitusional bukanlah solusi yang sempurna. Sistem AI bersifat kompleks dan terkadang dapat menghasilkan konten yang berbahaya atau menyinggung meskipun dilatih menggunakan AI konstitusional. Oleh karena itu, penting untuk memiliki perlindungan lain, seperti panduan keselamatan, guna mencegah sistem AI digunakan untuk tujuan yang berbahaya atau tidak etis.

Claude 2 vs. GPT 3.5

Claude 2 dan GPT 3.5 keduanya merupakan model bahasa besar (LLM) yang mampu menghasilkan teks, menerjemahkan bahasa, dan menjawab pertanyaan secara informatif. Namun, terdapat beberapa perbedaan utama antara kedua model ini:

- *Data Pelatihan:* Claude 2 dilatih pada kumpulan data teks dan kode yang sangat besar, sementara GPT 3.5 dilatih pada kumpulan data teks saja. Ini berarti Claude 2 mampu menghasilkan keluaran yang lebih akurat dan presisi karena memiliki akses ke informasi yang lebih luas.
- *Fitur Keamanan*: Claude 2 memiliki sejumlah fitur keamanan yang dirancang untuk mencegahnya menghasilkan konten yang berbahaya atau menyinggung. Fitur-fitur ini mencakup filter bias dan mekanisme untuk mendeteksi dan mencegah loop berbahaya. GPT 3.5 tidak memiliki fitur keamanan yang sama, sehingga lebih mungkin menghasilkan konten yang berbahaya atau menyinggung.

Tabel 5.2 merangkum perbedaan utama antara Claude 2 dan ChatGPT.

Tabel 5.2. Perbedaan Utama antara Claude 2 dan ChatGPT

Fitur	Claude 2	GPT 3.5
Data pelatihan	Teks dan kode	Hanya teks
Fitur keselamatan	Ya	Tidak
Target audiens	Bisnis, pemerintah, individu	Hiburan
Ketepatan	Lebih akurat	Kurang akurat

Keamanan	Lebih aman	Kurang aman
Keserbagunaan	Lebih serbaguna	Kurang serbaguna

Membentuk AI dengan ciri-ciri seperti akal sehat, ketajaman percakapan, dan nilai-nilai kemanusiaan menandai batas kemajuan teknologi yang belum dipetakan. Melalui arsitektur Claude yang berpusat pada manusia dan kecakapan bahasa alami yang canggih, langkah-langkah substansial telah diambil dalam mempersempit kesenjangan yang terus ada antara kecerdasan manusia dan mesin.

Seiring perkembangan Claude, ia membuka jalan menuju lanskap AI yang tidak menggantikan kemampuan manusia, tetapi justru meningkatkannya secara sinergis. Cakrawala masa depan kolaboratif, di mana manusia dan mesin bersatu sebagai mitra yang harmonis, sudah sangat dekat.

5.4 MODEL BAHASA BESAR LAINNYA

Selain ChatGPT, Google Bard, dan Claude, terdapat banyak model bahasa besar (LLM) lain yang saat ini sedang dikembangkan. Model-model ini dilatih pada kumpulan data teks dan kode yang sangat besar, dan mampu melakukan berbagai tugas, termasuk pembuatan teks, penerjemahan, tanya jawab, dan pembuatan kode.

Falcon AI

Falcon AI adalah model bahasa besar (LLM) yang dikembangkan oleh Technology Innovation Institute (TII) di Uni Emirat Arab. Model ini merupakan model dekoder autoregresif dengan 180 miliar parameter yang dilatih pada 1 triliun token. Model ini dilatih secara berkelanjutan di AWS Cloud selama dua bulan dengan 384 GPU terpasang. Falcon AI adalah model bahasa canggih yang dapat digunakan untuk berbagai tugas, termasuk:

- *Pembuatan Teks:* Falcon AI dapat menghasilkan teks, menerjemahkan bahasa, menulis berbagai jenis konten kreatif, dan menjawab pertanyaan Anda dengan cara yang informatif.
- *Pemahaman Bahasa Alami:* Falcon AI dapat memahami makna teks dan menjawab pertanyaan secara komprehensif dan informatif.
- *Menjawab Pertanyaan:* Falcon AI dapat menjawab pertanyaan Anda dengan cara yang informatif, meskipun pertanyaan tersebut terbuka, menantang, atau aneh.
- *Peringkasan:* Falcon AI dapat meringkas teks dengan cara yang ringkas dan informatif.
- *Pembuatan Kode:* Falcon AI dapat menghasilkan kode, seperti kode Python atau Java.
- *Analisis Data:* Falcon AI dapat menganalisis data dan mengekstrak wawasan.

Falcon AI masih dalam tahap pengembangan, tetapi berpotensi menjadi alat yang ampuh untuk berbagai aplikasi. Perlu dicatat bahwa Falcon AI adalah model bahasa yang besar, sehingga dapat bias. Penting untuk menggunakan Falcon AI secara bertanggung jawab dan menyadari keterbatasannya. Falcon AI menawarkan dua model serbaguna:

- *Falcon 180B:* Model dengan 180 miliar parameter yang mampu melakukan tugas-tugas kompleks, seperti menerjemahkan bahasa, menulis format teks kreatif, dan menjawab pertanyaan secara komprehensif dan informatif.

- *Falcon 40B:* Model dengan 40 miliar parameter yang lebih efisien dan cocok untuk tugas-tugas yang tidak membutuhkan banyak daya.

Berikut adalah beberapa aplikasi penting Falcon AI:

- PreciseAG, yang memberikan wawasan tentang kesehatan unggas.
- DocNovus, yang memungkinkan pengguna berinteraksi dengan dokumen bisnis mereka dan mendapatkan respons yang relevan seolah-olah mereka sedang berbicara dengan seorang ahli.
- Falcon AI juga digunakan untuk mengembangkan aplikasi di bidang kesehatan, pendidikan, dan keuangan.

Falcon AI adalah teknologi baru yang menjanjikan dan berpotensi merevolusi cara kita berinteraksi dengan komputer. Penting untuk terus mengembangkan dan meneliti teknologi ini agar dapat digunakan secara aman dan bertanggung jawab.

Berikut beberapa fitur utama Falcon AI:

- Falcon AI merupakan model dekoder autoregresif dengan 180 miliar parameter. Artinya, Falcon AI dapat menghasilkan teks, tetapi tidak dapat memahami makna teks yang dihasilkannya.
- Falcon AI dilatih pada kumpulan data teks dan kode yang sangat besar. Hal ini memberinya beragam pengetahuan dan kemampuan.
- Falcon AI masih dalam tahap pengembangan, tetapi berpotensi menjadi alat yang ampuh untuk berbagai aplikasi.

Berikut beberapa keterbatasan Falcon AI:

- Model bahasanya besar, sehingga dapat menimbulkan bias.
- Masih dalam tahap pengembangan, sehingga mungkin tidak dapat menangani semua tugas dengan sempurna.
- Penting untuk menggunakan Falcon AI secara bertanggung jawab dan menyadari keterbatasannya.

Secara keseluruhan, Falcon AI adalah model bahasa yang kuat yang berpotensi menjadi alat yang berharga untuk berbagai aplikasi. Namun, penting untuk menggunakannya secara bertanggung jawab dan menyadari keterbatasannya.

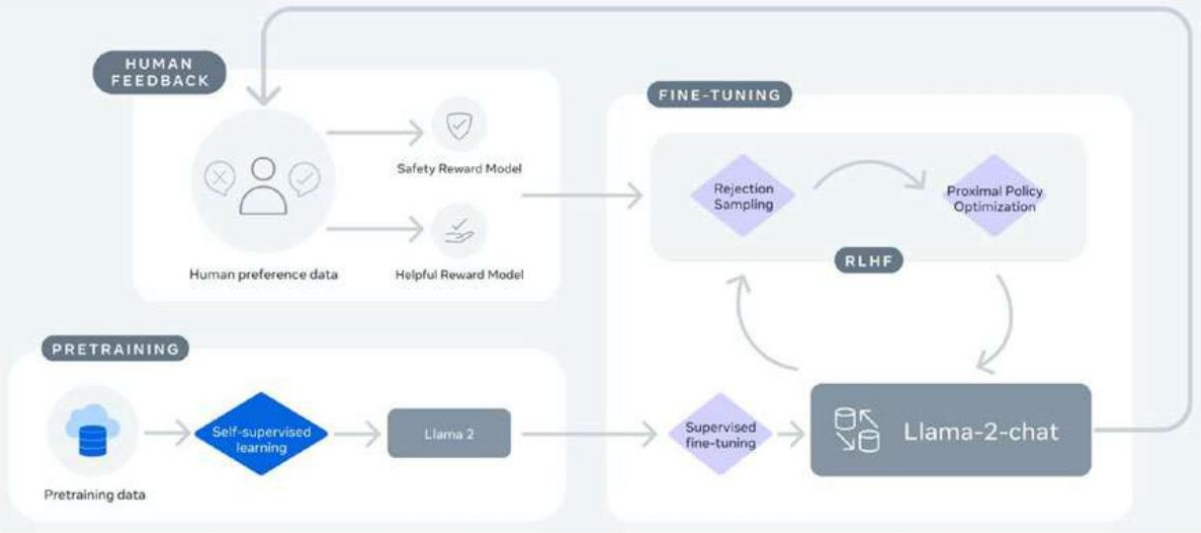
LLaMa 2

LLaMa 2 adalah keluarga model bahasa besar (LLM) yang dirilis oleh Meta AI pada Juli 2023. LLaMa 2 merupakan penerus LLaMa asli, dan telah disempurnakan dalam beberapa hal. LLaMa 2 dilatih pada kumpulan data teks dan kode yang sangat besar, dan memiliki dua triliun token. Jumlah ini jauh lebih banyak daripada LLaMa asli, yang dilatih pada satu triliun token. Dataset yang lebih besar memungkinkan LLaMa 2 mempelajari pengetahuan dan kemampuan yang lebih luas.

LLaMa 2 juga memiliki konteks yang lebih panjang daripada LLaMa versi sebelumnya. Ini berarti LLaMa 2 dapat memahami makna teks dalam konteks yang lebih panjang, yang penting untuk tugas-tugas seperti menjawab pertanyaan dan meringkas.

Arsitektur LLaMa 2 yang ditunjukkan pada Gambar 5.3 merupakan modifikasi dari arsitektur Transformer. Arsitektur Transformer adalah arsitektur jaringan saraf tiruan yang

sangat cocok untuk tugas-tugas pemrosesan bahasa alami. Arsitektur ini terdiri dari tumpukan lapisan enkoder dan dekoder. Lapisan enkoder mengodekan teks masukan menjadi representasi tersembunyi, dan lapisan dekoder menghasilkan teks keluaran dari representasi tersembunyi tersebut.



Gambar 5.3. Pelatihan LLaMa 2-Chat. Proses ini dimulai dengan pra-pelatihan LLaMa 2 menggunakan sumber daring yang tersedia untuk umum. Setelah itu, kami membuat versi awal LLaMa 2-Chat melalui penerapan fine-tuning terawasi. Selanjutnya, model disempurnakan secara iteratif menggunakan metodologi pembelajaran penguatan dengan umpan balik manusia (RLHF), khususnya melalui pengambilan sampel penolakan dan optimasi kebijakan proksimal (PPO). Sepanjang tahap RLHF, akumulasi data pemodelan imbalan iteratif yang paralel dengan penyempurnaan model sangat penting untuk memastikan model imbalan tetap berada dalam distribusi.

Arsitektur LLaMa 2 melakukan modifikasi berikut pada arsitektur Transformer:

- Pra-normalisasi: Arsitektur LLaMa 2 menggunakan pra-normalisasi, alih-alih pasca-normalisasi. Ini berarti bahwa masukan ke setiap lapisan dinormalisasi sebelum lapisan tersebut diterapkan. Hal ini telah terbukti meningkatkan stabilitas dan kinerja model.
- Fungsi Aktivasi SwiGLU: Arsitektur LLaMa 2 menggunakan fungsi aktivasi SwiGLU, alih-alih fungsi aktivasi ReLU. Fungsi aktivasi SwiGLU merupakan fungsi aktivasi yang lebih efisien dan efektif yang telah terbukti meningkatkan kinerja model.
- Penyematan Posisi Putar: Arsitektur LLaMa 2 menggunakan penyematan posisi putar, alih-alih penyematan posisi sinusoidal. Penyematan posisi putar merupakan cara yang lebih efisien dan efektif untuk mengodekan informasi posisi teks masukan.

Selain modifikasi ini, arsitektur LLaMa 2 juga menggunakan jendela konteks yang lebih besar dan perhatian kueri berkelompok. Jendela konteks yang lebih besar memungkinkan model untuk memproses lebih banyak informasi, dan perhatian kueri berkelompok memungkinkan model untuk lebih efisien dalam menangani teks masukan. Secara keseluruhan, arsitektur LLaMa 2 merupakan arsitektur model bahasa mutakhir yang telah terbukti mencapai kinerja

yang sangat baik pada berbagai tugas pemrosesan bahasa alami. Arsitektur LLaMa 2 terdiri dari tumpukan lapisan encoder dan decoder. Lapisan encoder mengodekan teks masukan menjadi representasi tersembunyi, dan lapisan decoder menghasilkan teks keluaran dari representasi tersembunyi tersebut.

Arsitektur LLaMa 2 juga menggunakan sejumlah teknik lain untuk meningkatkan kinerjanya, seperti pra-normalisasi, fungsi aktivasi SwiGLU, penyematan posisi putar, dan jendela konteks yang lebih besar.

LLaMa 2 telah terbukti mengungguli LLaMa versi asli dalam sejumlah uji tolok ukur, termasuk pembuatan teks, penerjemahan, tanya jawab, dan pembuatan kode. LLaMa 2 juga lebih bermanfaat dan lebih aman daripada LLaMa versi asli, berkat penggunaan pembelajaran penguatan dari umpan balik manusia (RLHF).

LLaMa 2 berpotensi menjadi alat yang ampuh untuk berbagai aplikasi. LLaMa 2 sudah digunakan untuk tugas-tugas seperti dialog, pembuatan kode, dan tanya jawab. Di masa mendatang, LLaMa 2 kemungkinan akan digunakan untuk lebih banyak aplikasi, seperti pendidikan, layanan kesehatan, dan layanan pelanggan. Berikut beberapa fitur utama LLaMa 2:

- Dilatih pada kumpulan data teks dan kode yang sangat besar.
- Memiliki dua triliun token.
- Memiliki konteks yang lebih panjang daripada LLaMa versi asli.
- Menggunakan arsitektur baru yang disebut Grouper query attention.
- Telah terbukti mengungguli LLaMa versi asli pada sejumlah uji tolok ukur.
- Lebih bermanfaat dan lebih aman daripada LLaMa versi asli.

Berikut beberapa keterbatasan LLaMa 2:

- Bisa bias.

Secara keseluruhan, LLaMa 2 adalah model bahasa yang ampuh dan berpotensi menjadi alat yang berharga untuk berbagai aplikasi. Namun, penting untuk menggunakannya secara bertanggung jawab dan menyadari keterbatasannya.

Dolly 2

Dolly 2 dibuat oleh Databricks. Model bahasa kausal dengan 175 miliar parameter ini dibuat oleh Databricks, sebuah perusahaan analitik data perusahaan dan AI. Model ini dilatih pada kumpulan data teks dan kode yang sangat besar, dan mampu melakukan berbagai tugas, termasuk:

- Pembuatan teks
- Penerjemahan
- Menjawab pertanyaan
- Pembuatan kode
- Analisis data
- Peringkasan
- Penulisan kreatif

Dolly 2 masih dalam tahap pengembangan, tetapi berpotensi menjadi alat yang ampuh untuk berbagai aplikasi. Model ini sudah digunakan untuk tugas-tugas seperti dialog, pembuatan kode, dan menjawab pertanyaan. Berikut beberapa fitur utama Dolly 2:

- Model bahasa kausal dengan 12 miliar parameter.
- Dilatih pada kumpulan data teks dan kode yang sangat besar.
- Mampu melakukan berbagai tugas.
- Masih dalam tahap pengembangan, tetapi berpotensi menjadi alat yang ampuh untuk berbagai aplikasi.

Kesimpulan

Selain ChatGPT, Google Bard, dan Claude, terdapat banyak model bahasa besar (LLM) lain yang saat ini sedang dikembangkan. Model-model ini dilatih pada kumpulan data teks dan kode yang sangat besar, dan mampu melakukan berbagai tugas, termasuk pembuatan teks, penerjemahan, tanya jawab, dan pembuatan kode.

LLM yang telah saya bahas dalam bab ini hanyalah beberapa contoh dari sekian banyak model yang tersedia. Seiring dengan perkembangan teknologi ini, kita dapat berharap untuk melihat model bahasa yang lebih canggih dan serbaguna yang sedang dikembangkan di masa mendatang.

Model-model ini berpotensi menjadi alat yang berharga untuk berbagai aplikasi. Namun, penting untuk menggunakannya secara bertanggung jawab dan menyadari keterbatasannya. LLM dapat bias dan dapat digunakan untuk tujuan jahat. Penting untuk menggunakannya dengan cara yang etis dan bermanfaat bagi masyarakat.

BAB 6
IMPLEMENTASI LLM MENGGUNAKAN SKLEARN

Scikit-LLM merupakan kemajuan yang luar biasa dalam bidang analisis teks. Alat inovatif ini secara mulus menggabungkan kemampuan model bahasa yang tangguh seperti ChatGPT dengan fungsionalitas serbaguna scikit-learn. Hasilnya adalah perangkat tak tertandingi yang memungkinkan pengguna untuk mendalami data tekstual dengan cara yang belum pernah ada sebelumnya.

Dengan Scikit-LLM, Anda memperoleh kemampuan untuk mengungkap pola tersembunyi, membedah sentimen, dan memahami konteks dalam spektrum sumber tekstual yang luas. Baik Anda menangani umpan balik pelanggan, postingan media sosial, atau artikel berita, penggabungan model bahasa dan scikit-learn ini membekali Anda dengan seperangkat alat yang tangguh.

Intinya, Scikit-LLM merupakan sinergi yang kuat antara pemahaman bahasa mutakhir dan kecakapan analitis scikit-learn, yang memungkinkan Anda untuk mengekstrak wawasan berharga dari data teks yang sebelumnya tersembunyi. Mudah digunakan dan menyediakan berbagai fitur yang menjadikannya sumber daya berharga bagi ilmuwan data dan praktisi pembelajaran mesin.

Berikut beberapa detail tambahan tentang fitur-fitur Scikit-LLM:

- Klasifikasi Teks Zero-Shot: Ini adalah fitur canggih yang memungkinkan Anda mengklasifikasikan teks ke dalam serangkaian label tanpa harus melatih model pada data berlabel apa pun. Hal ini dilakukan dengan meminta LLM untuk menghasilkan respons untuk teks tersebut, lalu menggunakan respons tersebut untuk menentukan label yang paling mungkin. Respons tersebut dihasilkan oleh LLM berdasarkan pemahamannya terhadap teks dan serangkaian label yang Anda berikan.
- Klasifikasi Teks Zero-Shot Multilabel: Ini adalah versi yang lebih canggih dari klasifikasi teks zero-shot yang memungkinkan Anda mengklasifikasikan teks ke dalam beberapa label sekaligus. Hal ini dilakukan dengan meminta LLM untuk menghasilkan respons untuk setiap label, lalu menggunakan respons tersebut untuk menentukan label yang paling mungkin.
- Vektorisasi Teks: Ini adalah langkah prapemrosesan teks umum yang mengubah teks menjadi representasi vektor berdimensi tetap. Representasi ini kemudian dapat digunakan untuk tugas pembelajaran mesin lainnya, seperti klasifikasi, pengelompokan, atau regresi. Scikit-LLM menyediakan kelas GPTVectorizer untuk mengubah teks menjadi representasi vektor berdimensi tetap.
- Penerjemahan Teks: Ini memungkinkan Anda menerjemahkan teks dari satu bahasa ke bahasa lain menggunakan LLM. Scikit-LLM menyediakan kelas GPTTranslator untuk menerjemahkan teks dari satu bahasa ke bahasa lain.
- Peringkasan Teks: Ini memungkinkan Anda meringkas dokumen teks menjadi versi yang lebih pendek dan ringkas. Scikit-LLM menyediakan kelas GPTSummarizer untuk

meringkas dokumen teks. Sekarang mari kita terapkan beberapa contoh/fitur Scikit-LLM. Mari kita mulai.

Catatan: Gunakan Google Colab untuk implementasinya.

6.1 INSTALASI SCIKIT-LLM

```
%%capture
!pip install scikit-llm watermark
```

- Integrasikan model bahasa canggih seperti ChatGPT ke dalam scikit-learn secara mulus untuk tugas analisis teks yang lebih baik.
- API yang serupa dengan scikit-learn, seperti `.fit()`, `.fit_transform()`, dan `.predict()`.
- Gabungkan estimator dari library Scikit-LLM dalam alur kerja sklearn.

```
%load_ext tanda air
%watermark -a "nama pengguna" -vmp scikit-llm
```

Dapatkan Kunci API OpenAI

Per Mei 2023, Scikit-LLM saat ini kompatibel dengan serangkaian model OpenAI tertentu. Oleh karena itu, pengguna diharuskan menyediakan kunci API OpenAI mereka sendiri agar integrasi berhasil. Awali dengan mengimpor modul SKLLMConfig dari pustaka Scikit-LLM dan tambahkan kunci OpenAI Anda:

Untuk mendapatkan kunci, gunakan tautan berikut:

https://platform.openai.com/account/api-keys
https://platform.openai.com/account/org-settings

```
# importing SKLLMConfig to configure OpenAI API (key and Name)
from skllm.config import SKLLMConfig
OPENAI_API_KEY = "sk-****"
OPENAI_ORG_ID = "org-****"
# Set your OpenAI API key
SKLLMConfig.set_openai_key(OPENAI_API_KEY )
# Set your OpenAI organization
SKLLMConfig.set_openai_org(OPENAI_ORG_ID)
```

6.2 ZERO-SHOT GPTCLASSIFIER

ChatGPT memiliki kemampuan yang luar biasa—dapat mengklasifikasikan teks tanpa perlu pelatihan khusus. Sebaliknya, ChatGPT mengandalkan label deskriptif untuk menjalankan tugas ini secara efektif.

Sekarang, mari kita perkenalkan "ZeroShotGPTClassifier", sebuah fitur dalam Scikit-LLM. Dengan alat ini, Anda dapat dengan mudah membangun model klasifikasi teks, layaknya pengklasifikasi lain yang tersedia di pustaka scikit-learn.

Intinya, ZeroShotGPTClassifier memanfaatkan kemampuan unik ChatGPT untuk memahami dan mengkategorikan teks berdasarkan label, menyederhanakan proses klasifikasi teks tanpa kerumitan pelatihan tradisional.

Mengimpor pustaka yang diperlukan:

```
# importing zeroshotgptclassifier module and classification dataset
from skllm import ZeroShotGPTClassifier
from skllm.datasets import get_classification_dataset
```

Mari kita gunakan dataset bawaan:

```
# sentiment analysis dataset
# labels: positive, negative, neutral
X, y = get_classification_dataset()
len(X)
```

Output: 30

Mari kita cetak variabel X:

```
X
```

Keluaran:

Mari kita cetak variabel y:

```
y
```

Output

```
['positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral']
```

Sekarang mari kita bagi data menjadi train dan test. Fungsi untuk data training:

```
# to notice: indexing starts at 0
def training_data(data):
    subset_1 = data[:8] # First 8 elements from 1-10
    subset_2 = data[10:18] # First 8 elements from 11-20
    subset_3 = data[20:28] # First 8 elements from rest of the data
    combined_data = subset_1 + subset_2 + subset_3
    return combined_data
```

Fungsi untuk data uji:

```
# to notice: indexing starts at 0
def testing_data(data):
    subset_1 = data[8:10] # Last 2 elements from 1-10
    subset_2 = data[18:20] # Last 2 elements from 11-20
    subset_3 = data[28:30] # Last 2 elements from rest of the data
    combined_data = subset_1 + subset_2 + subset_3
    return combined_data
```

Sekarang, mari kita gunakan variabel X dan y sebagai parameter untuk fungsi training_data:

```
X_train = training_data(X)
print(len(X_train))
```

```
X_train
```

Output:


```
y_train = training_data(y)
print(len(y_train))
y_train
```

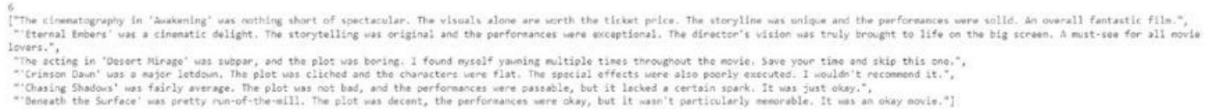
Output

```
24
['positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'positive',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'negative',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral',
 'neutral']
```

Sekarang, mari kita gunakan variabel X dan y sebagai parameter untuk fungsi testing_data:

```
X_test = testing_data(X)
print(len(X_test))
X_test
```

Output:




```
y_test = testing_data(y)
print(len(y_test))
y_test
```

Output:

```
6
['positive', 'positive', 'negative', 'negative', 'neutral', 'neutral']
```

Mendefinisikan dan melatih model OpenAI:

```
# defining the openai model to use
clf = ZeroShotGPTClassifier(openai_model="gpt-3.5-turbo")
# fitting the data
clf.fit(X_train, y_train)
```

Prediksi pada uji X menggunakan model clf:

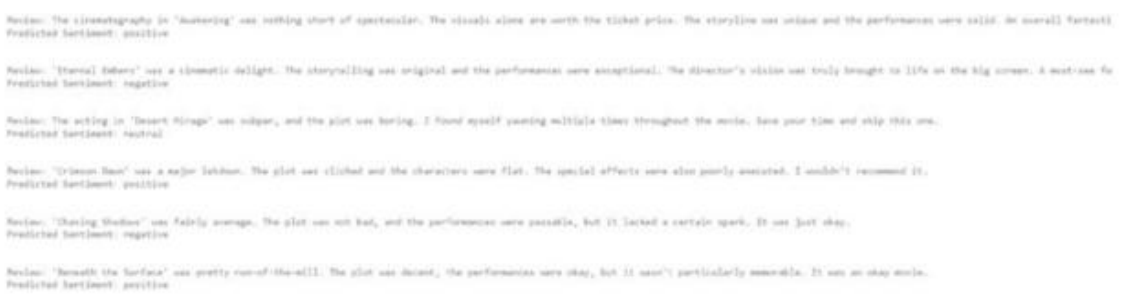
```
%%time
# predicting the data
predicted_labels = clf.predict(X_test)
```

Output:

Menandai prediksi untuk setiap kalimat:

```
for review, sentiment in zip(X_test, predicted_labels):
  print(f"Review: {review}\nPredicted Sentiment: {sentiment}\n\n")
```

Output:



Evaluasi model:

```
from sklearn.metrics import accuracy_score
  print(f"Accuracy: {accuracy_score(y_test, predicted_labels):.2f}")
```

Output:

```
Accuracy: 1.00
```

Scikit-LLM memberikan upaya ekstra dengan memastikan respons yang diterimanya berisi label yang valid. Ketika menemukan respons tanpa label yang valid, Scikit-LLM tidak membiarkan Anda menunggu. Sebaliknya, Scikit-LLM akan langsung memilih label secara acak, dengan mempertimbangkan probabilitas berdasarkan seberapa sering label tersebut muncul dalam data pelatihan.

Sederhananya, Scikit-LLM memperhatikan detail teknisnya, memastikan Anda selalu memiliki label yang relevan untuk digunakan. Scikit-LLM siap membantu Anda, bahkan jika respons tersebut tidak memiliki label, karena Scikit-LLM akan secara cerdas memilihkan label untuk Anda berdasarkan pengetahuannya tentang frekuensi label dalam data pelatihan.

Bagaimana Jika Anda Tidak Memiliki Data Berlabel?

Aspek menariknya adalah: Anda sebenarnya tidak memerlukan data pra-label untuk melatih model. Yang Anda butuhkan hanyalah daftar calon label potensial untuk memulai. Pendekatan ini membuka kemungkinan untuk melatih model bahkan ketika Anda tidak memiliki dataset berlabel yang sudah ada. Mendefinisikan model pelatihan OpenAI:

```
# defining the model
  clf_no_label = ZeroShotGPTClassifier()
# No training so passing the labels only for prediction
  clf_no_label.fit(None, ['positive', 'negative', 'neutral'])
```

Prediksi pada uji X menggunakan model:

```
# predicting the labels
  predicted_labels_without_training_data = clf_no_label.predict(X_test)
  predicted_labels_without_training_data
```

Output:

```
 17%|██        | 1/6 [00:10<00:54, 10.80s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable
 33%|███       | 2/6 [00:20<00:39,  9.94s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable
 50%|████      | 3/6 [00:29<00:29,  9.67s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable
 67%|██████    | 4/6 [00:38<00:18,  9.50s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable
 83%|███████   | 5/6 [00:47<00:09,  9.39s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable
100%|████████| 6/6 [00:57<00:00,  9.52s/it]Could not obtain the completion after 3 retries: 'RateLimitError :: You exceeded your current quota, please check your plan and billing details.'
None
Could not extract the label from the completion: 'NoneType' object is not subscriptable

['neutral', 'neutral', 'negative', 'positive', 'positive', 'positive']
```

Menandai prediksi untuk setiap kalimat:

```
for review, sentiment in zip(X_test, predicted_labels_without_
training_data):
  print(f"Review: {review}\nPredicted Sentiment: {sentiment}\n\n")
```

Output:

```
Review: The cinematography in 'Awakening' was nothing short of spectacular. The visuals alone are worth the ticket price. The storyline was unique and the performances were solid. An overall fantas
Predicted Sentiment: neutral

Review: 'Eternal Embers' was a cinematic delight. The storytelling was original and the performances were exceptional. The director's vision was truly brought to life on the big screen. A must-see
Predicted Sentiment: neutral

Review: The acting in 'Desert Mirage' was subpar, and the plot was boring. I found myself yawning multiple times throughout the movie. Save your time and skip this one.
Predicted Sentiment: negative

Review: 'Crimson Dawn' was a major letdown. The plot was cliched and the characters were flat. The special effects were also poorly executed. I wouldn't recommend it.
Predicted Sentiment: positive

Review: 'Chasing Shadows' was fairly average. The plot was not bad, and the performances were passable, but it lacked a certain spark. It was just okay.
Predicted Sentiment: positive

Review: 'Beneath the Surface' was pretty run-of-the-mill. The plot was decent, the performances were okay, but it wasn't particularly memorable. It was an okay movie.
Predicted Sentiment: positive
```

Evaluasi model:

```
print(f"Accuracy: {accuracy_score(y_test, predicted_labels_without_
training_data):.2f}")
```

Output:

```
Accuracy: 1.00
```

Sejauh ini kita telah mempelajari cara menggunakan model Scikit-LLM untuk klasifikasi teks. Selanjutnya, kita akan mempelajari fitur-fitur Scikit-LLM lainnya.

Catatan: Pada contoh-contoh berikutnya, kita tidak akan membagi data menjadi pelatihan dan pengujian atau evaluasi model seperti yang kita lakukan untuk klasifikasi teks, melainkan berfokus pada bagian penggunaan.

6.3 KLASIFIKASI TEKS MULTILABEL ZERO-SHOT

Melakukan klasifikasi teks multilabel zero-shot mungkin terdengar rumit, tetapi sebenarnya lebih mudah daripada yang Anda bayangkan.

Implementasi

```
# importing Multi-Label zeroshot module and classification dataset
from skllm import MultiLabelZeroShotGPTClassifier
from skllm.datasets import get_multilabel_classification_dataset
# get classification dataset from sklearn
X, y = get_multilabel_classification_dataset()

# defining the model
clf = MultiLabelZeroShotGPTClassifier(max_labels=3)

# fitting the model
clf.fit(X, y)

# making predictions
labels = clf.predict(X)
```

Satu-satunya perbedaan antara klasifikasi zero-shot dan multilabel zero-shot terletak pada pembuatan instans kelas MultiLabelZeroShotGPTClassifier. Dalam kasus klasifikasi zero-shot multilabel, Anda menentukan jumlah label maksimum yang ingin Anda tetapkan untuk setiap sampel, seperti menetapkan max_labels=3 sebagai contoh. Parameter ini memungkinkan Anda untuk mengontrol berapa banyak label yang dapat ditetapkan model ke sampel teks tertentu selama klasifikasi.

Bagaimana Jika Anda Tidak Memiliki Data Berlabel?

Dalam skenario yang dijelaskan sebelumnya, MultiLabelZeroShotGPTClassifier masih dapat dilatih secara efektif. Alih-alih menggunakan data berlabel tradisional (X dan y), Anda dapat melatih pengklasifikasi dengan menyediakan daftar label kandidat potensial. Dalam pengaturan ini, komponen "y" harus disusun sebagai Daftar Daftar, di mana setiap daftar di dalamnya berisi label kandidat untuk sampel teks tertentu.

Berikut contoh yang mengilustrasikan proses pelatihan tanpa data berlabel:

Implementasi

```
# getting classification dataset for prediction only
from skllm.datasets import get_multilabel_classification_dataset
from skllm import MultiLabelZeroShotGPTClassifier
X, _ = get_multilabel_classification_dataset()
# Defining all the labels that need to be predicted
candidate_labels = [
   "Quality",
   "Price",
   "Delivery",
   "Service",
   "Product Variety"
]

# creating the model
clf = MultiLabelZeroShotGPTClassifier(max_labels=3)
```

```
# fitting the labels only
clf.fit(None, [candidate_labels])

# predicting the data
labels = clf.predict(X)
```

6.4 VEKTORISASI TEKS

Vektorisasi teks merupakan proses krusial yang melibatkan transformasi informasi tekstual ke dalam format numerik, yang memungkinkan mesin untuk memahami dan menganalisisnya secara efektif. Dalam kerangka kerja Scikit-LLM, Anda akan menemukan alat berharga bernama GPTVectorizer. Modul ini berfungsi untuk mengonversi teks, berapa pun panjangnya, menjadi sekumpulan nilai numerik berukuran tetap yang dikenal sebagai vektor. Transformasi ini memungkinkan model pembelajaran mesin untuk memproses dan memahami data berbasis teks secara lebih efisien.

Implementasi

```
# Importing the GPTVectorizer class from the skllm.preprocessing module
from skllm.preprocessing import GPTVectorizer

# Creating an instance of the GPTVectorizer class and assigning it to the
variable 'model'
model = GPTVectorizer()

# transforming the
vectors = model.fit_transform(X)
```

Ketika Anda menerapkan metode "fit_transform" dari instans GPTVectorizer ke data input "X", metode ini tidak hanya menyesuaikan model dengan data tetapi juga mengubah teks menjadi vektor berdimensi tetap. Vektor yang dihasilkan kemudian disimpan dalam sebuah variabel, yang secara konvensional disebut "vektor".

Mari kita ilustrasikan contoh cara mengintegrasikan GPTVectorizer dengan XGBoost Classifier dalam alur kerja scikit-learn. Pendekatan ini memungkinkan Anda untuk melakukan praproses teks secara efisien dan melakukan tugas klasifikasi dengan lancar:

```
# Importing the necessary modules and classes
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import LabelEncoder
from xgboost import XGBClassifier

# Creating an instance of LabelEncoder class
le = LabelEncoder()

# Encoding the training labels 'y_train' using LabelEncoder
y_train_encoded = le.fit_transform(y_train)
```

```
# Encoding the test labels 'y_test' using LabelEncoder
y_test_encoded = le.transform(y_test)

# Defining the steps of the pipeline as a list of tuples
steps = [('GPT', GPTVectorizer()), ('Clf', XGBClassifier())]

# Creating a pipeline with the defined steps
clf = Pipeline(steps)

# Fitting the pipeline on the training data 'X_train' and the encoded
training labels 'y_train_encoded'
clf.fit(X_train, y_train_encoded)

# Predicting the labels for the test data 'X_test' using the trained
pipeline
yh = clf.predict(X_test)
```

Ringkasan Teks

Memang, GPT unggul dalam ringkasan teks, dan keunggulan ini dimanfaatkan dalam Scikit-LLM melalui modul GPTSummarizer. Anda dapat memanfaatkan modul ini dengan dua cara berbeda:

1. *Ringkasan Mandiri:* Anda dapat menggunakan GPTSummarizer sendiri untuk menghasilkan ringkasan konten tekstual yang ringkas dan koheren, sehingga memudahkan pemahaman poin-poin utama dari dokumen yang panjang.
2. *Sebagai Langkah Prapemrosesan:* Sebagai alternatif, Anda dapat mengintegrasikan GPTSummarizer ke dalam alur kerja yang lebih luas sebagai langkah awal sebelum melakukan operasi lain. Misalnya, Anda dapat menggunakannya untuk mengurangi ukuran data teks sambil tetap mempertahankan informasi penting. Hal ini memungkinkan penanganan data berbasis teks yang lebih efisien tanpa mengorbankan kualitas dan relevansi konten.

Implementasi

```
# Importing the GPTSummarizer class from the skllm.preprocessing module
from skllm.preprocessing import GPTSummarizer

# Importing the get_summarization_dataset function
from skllm.datasets import get_summarization_dataset

# Calling the get_summarization_dataset function
X = get_summarization_dataset()

# Creating an instance of the GPTSummarizer
s = GPTSummarizer(openai_model='gpt-3.5-turbo', max_words=15)

# Applying the fit_transform method of the GPTSummarizer instance to the
```

```
input data 'X'.

# It fits the model to the data and generates the summaries, which are
assigned to the variable 'summaries'
summaries = s.fit_transform(X)
```

Penting untuk dipahami bahwa hiperparameter "max_words" berfungsi sebagai panduan fleksibel untuk membatasi jumlah kata dalam ringkasan yang dihasilkan. Hiperparameter ini tidak diterapkan secara ketat di luar perintah awal yang Anda berikan. Secara praktis, ini berarti mungkin ada kasus di mana jumlah kata sebenarnya dalam ringkasan yang dihasilkan sedikit melebihi batas yang ditentukan.

Sederhananya, meskipun "max_words" memberikan target perkiraan panjang ringkasan, peringkas terkadang dapat menghasilkan ringkasan yang sedikit lebih panjang. Perilaku ini bergantung pada konteks dan konten spesifik teks masukan karena peringkas bertujuan untuk menjaga koherensi dan relevansi dalam keluarannya.

Kesimpulan

Pada dasarnya, Scikit-LLM dapat digunakan untuk analisis teks, dan dirancang agar mudah digunakan serta menyediakan berbagai fitur, termasuk klasifikasi teks zero-shot, klasifikasi teks zero-shot multilabel, vektorisasi teks, translasi teks, dan peringkasan teks. Yang terpenting adalah Anda tidak memerlukan data pra-label untuk melatih model apa pun. Itulah keunggulan Scikit-LLM.

Untuk memulai penggunaan LLM untuk analisis teks dengan mudah. Scikit-LLM menyediakan API yang sederhana dan intuitif yang memudahkan Anda memulai penggunaan LLM untuk analisis teks, bahkan jika Anda belum familiar dengan LLM atau pembelajaran mesin.

Untuk menggabungkan LLM dengan algoritma pembelajaran mesin lainnya. Scikit-LLM dapat diintegrasikan dengan pipeline Scikit-Learn, yang memudahkan penggabungan LLM dengan algoritma pembelajaran mesin lainnya. Hal ini dapat berguna untuk tugas analisis teks kompleks yang memerlukan beberapa langkah.

Untuk bereksperimen dengan LLM untuk analisis teks. Scikit-LLM adalah proyek sumber terbuka, yang berarti gratis untuk digunakan dan dimodifikasi. Hal ini menjadikannya pilihan yang baik bagi peneliti dan pengembang yang ingin bereksperimen dengan LLM untuk analisis teks:

- Anda dapat menggunakan Scikit-LLM untuk mengklasifikasikan umpan balik pelanggan ke dalam berbagai kategori, seperti positif, negatif, atau netral. Informasi ini dapat digunakan untuk meningkatkan layanan pelanggan atau pengembangan produk.
- Anda dapat menggunakan Scikit-LLM untuk mengklasifikasikan artikel berita ke dalam berbagai topik, seperti politik, bisnis, atau olahraga. Informasi ini dapat digunakan untuk membuat umpan berita yang dipersonalisasi atau untuk melacak tren berita.
- Anda dapat menggunakan Scikit-LLM untuk menerjemahkan dokumen dari satu bahasa ke bahasa lain. Ini dapat bermanfaat bagi bisnis yang beroperasi di beberapa

negara atau bagi orang yang ingin membaca dokumen dalam bahasa yang tidak mereka kuasai.

- Anda dapat menggunakan Scikit-LLM untuk meringkas dokumen teks yang panjang. Ini dapat bermanfaat untuk mendapatkan poin-poin utama dokumen dengan cepat atau untuk membuat versi dokumen yang lebih pendek untuk dipublikasikan.

Selain yang telah disebutkan sebelumnya, Scikit-LLM juga menawarkan sejumlah manfaat lain, seperti:

- Akurasi: Scikit-LLM telah terbukti akurat dalam sejumlah tugas analisis teks, termasuk klasifikasi teks zero-shot dan peringkasan teks.
- Kecepatan: Scikit-LLM relatif cepat, sehingga cocok untuk tugas-tugas yang memerlukan pemrosesan waktu nyata. Skalabilitas: Scikit-LLM dapat diskalakan untuk menangani data teks dalam jumlah besar.

BAB 7
LLM UNTUK PERUSAHAAN DAN LLMOPS

Dalam bab ini, kami menyajikan kerangka kerja referensi untuk tumpukan aplikasi model bahasa besar (LLM) yang sedang berkembang. Kerangka kerja ini menggambarkan sistem, perangkat, dan pendekatan desain yang lazim digunakan dalam praktik di kalangan startup dan perusahaan AI. Perlu dicatat bahwa tumpukan ini masih dalam tahap awal dan kemungkinan akan mengalami transformasi signifikan seiring perkembangan teknologi yang mendasarinya.

Namun demikian, tujuan kami adalah agar sumber daya ini memberikan panduan berharga bagi para pengembang yang saat ini terlibat dengan LLM. Ada berbagai pendekatan untuk memanfaatkan kapabilitas LLM dalam pengembangan, yang mencakup pembuatan model dari awal, penyempurnaan model sumber terbuka melalui fine-tuning, atau pemanfaatan API yang dihosting. Kerangka kerja yang kami sajikan di sini berpusat pada pembelajaran dalam konteks, sebuah strategi desain umum yang dipilih oleh sebagian besar pengembang, terutama yang dimungkinkan melalui model-model dasar.

Bagian selanjutnya menawarkan penjelasan singkat tentang strategi ini, dengan pengembang LLM berpengalaman memiliki pilihan untuk melewatinya. Kekuatan LLM tidak hanya terletak pada kapabilitasnya, tetapi juga pada penggunaannya yang bertanggung jawab dan etis, yang sangat penting dalam lingkungan perusahaan. Kami akan membahas bagaimana organisasi menavigasi lanskap privasi data yang rumit, mitigasi bias, dan transparansi sambil memanfaatkan potensi transformatif dari model bahasa ini.

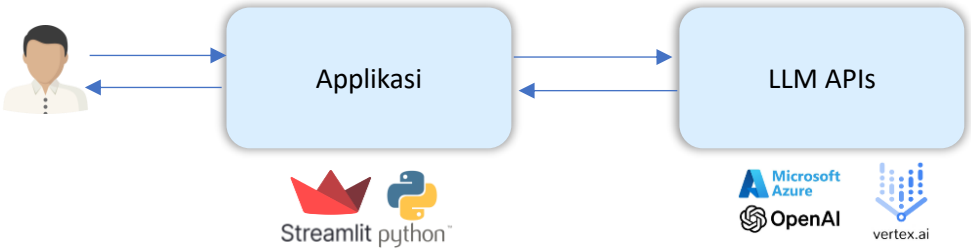
Sekarang, saat kita bersiap untuk mengakhiri eksplorasi kita, penting untuk menyoroti pendorong krusial transformasi ini: layanan cloud. Cloud, dengan daya komputasi, skalabilitas, dan jangkauan globalnya yang tak tertandingi, telah menjadi infrastruktur pilihan untuk menerapkan dan mengelola LLM. Cloud menyediakan lingkungan dinamis tempat bisnis dapat memanfaatkan potensi penuh model bahasa ini sambil menikmati berbagai manfaat. Kami akan membahas secara singkat bagaimana layanan cloud melengkapi adopsi LLM, menawarkan skalabilitas, efisiensi biaya, keamanan, dan integrasi yang mulus dengan alur kerja yang ada. Berikut adalah tiga cara Anda dapat mengaktifkan LLM di perusahaan.

7.1 API LLM UMUM PRIVAT

API LLM umum privat adalah cara bagi perusahaan untuk mengakses model bahasa besar (LLM) yang telah dilatih pada kumpulan data teks dan kode yang sangat besar. API ini bersifat privat, artinya hanya perusahaan yang dapat menggunakannya. Hal ini memastikan kerahasiaan data perusahaan. Ada beberapa manfaat menggunakan API LLM umum privat:

- Pertama, API ini memungkinkan perusahaan untuk menyesuaikan LLM dengan kebutuhan spesifik mereka. Misalnya, perusahaan dapat menentukan data pelatihan LLM, arsitektur LLM, dan parameter LLM. Hal ini memungkinkan perusahaan untuk memaksimalkan LLM untuk tugas-tugas spesifik mereka.

- Kedua, API LLM umum privat lebih aman daripada menggunakan API LLM publik. Hal ini karena data perusahaan tidak dibagikan dengan siapa pun. Hal ini penting bagi perusahaan yang peduli dengan keamanan data mereka.
- Ketiga, API LLM umum privat lebih skalabel daripada menggunakan API LLM publik. Hal ini karena perusahaan dapat meningkatkan jumlah daya komputasi yang digunakan untuk melatih dan menjalankan LLM. Hal ini memungkinkan perusahaan untuk menggunakan LLM untuk tugas-tugas yang lebih menantang.



Gambar 7.1. API LLM Umum Privat

Namun, terdapat juga beberapa tantangan dalam menggunakan API LLM Umum Privat:

- Mengembangkan dan memelihara API LLM Privat bisa mahal. Hal ini karena perusahaan perlu memiliki keahlian dan sumber daya untuk melatih dan menjalankan LLM.
- API LLM privat bisa lebih lambat daripada menggunakan API LLM publik. Hal ini karena data perusahaan perlu ditransfer ke LLM sebelum dapat diproses.
- API LLM privat bisa kurang fleksibel dibandingkan menggunakan API LLM publik. Hal ini karena perusahaan terbatas pada fitur dan kapabilitas yang disediakan oleh API tersebut.

Secara keseluruhan, API LLM privat yang digeneralisasi merupakan pilihan yang baik bagi perusahaan yang perlu menggunakan LLM untuk tugas-tugas spesifik mereka dan yang mengkhawatirkan keamanan data mereka. Namun, penting untuk mempertimbangkan manfaat dan tantangan menggunakan API LLM privat sebelum mengambil keputusan. Berikut adalah beberapa contoh bagaimana perusahaan dapat menggunakan API LLM privat yang digeneralisasi:

- *Layanan Pelanggan:* Perusahaan dapat menggunakan LLM untuk menghasilkan respons yang dipersonalisasi atas pertanyaan pelanggan.
- *Pengembangan Produk:* Perusahaan dapat menggunakan LLM untuk menghasilkan ide-ide untuk produk dan layanan baru.
- *Pemasaran:* Perusahaan dapat menggunakan LLM untuk membuat kampanye pemasaran yang dipersonalisasi.
- *Manajemen Risiko:* Perusahaan dapat menggunakan LLM untuk mengidentifikasi potensi risiko dan kerentanan.
- *Deteksi Penipuan:* Perusahaan dapat menggunakan LLM untuk mendeteksi transaksi penipuan.

7.2 STRATEGI DESAIN UNTUK MENGAKTIFKAN LLM BAGI PERUSAHAAN

Pada intinya, pembelajaran dalam konteks melibatkan penggunaan LLM siap pakai (tanpa penyempurnaan) dan memanipulasi perilakunya melalui perintah yang cerdas dan pengkondisian berdasarkan data "kontekstual" pribadi.

Pertimbangkan skenario pembuatan chatbot untuk menjawab pertanyaan terkait kumpulan dokumen hukum. Pendekatan langsung mungkin melibatkan penyisipan semua dokumen ke dalam perintah ChatGPT atau GPT-4, diikuti dengan mengajukan pertanyaan tentang dokumen tersebut. Meskipun ini mungkin cukup untuk kumpulan data yang sangat kecil, hal ini tidak dapat diskalakan. Model GPT-4 terbesar hanya dapat menangani sekitar 50 halaman teks masukan, dan kinerjanya dalam hal waktu inferensi dan akurasi menurun secara signifikan seiring mendekati batas jendela konteks ini. Pembelajaran dalam konteks mengatasi dilema ini secara cerdik dengan mengadopsi sebuah strategi: alih-alih menyediakan semua dokumen dengan setiap prompt LLM, ia hanya mengirimkan sekumpulan dokumen terpilih yang paling relevan. Dokumen-dokumen relevan ini ditentukan dengan bantuan—Anda sudah menebaknya—LLM.

Secara garis besar, alur kerja dapat dibagi menjadi tiga fase:

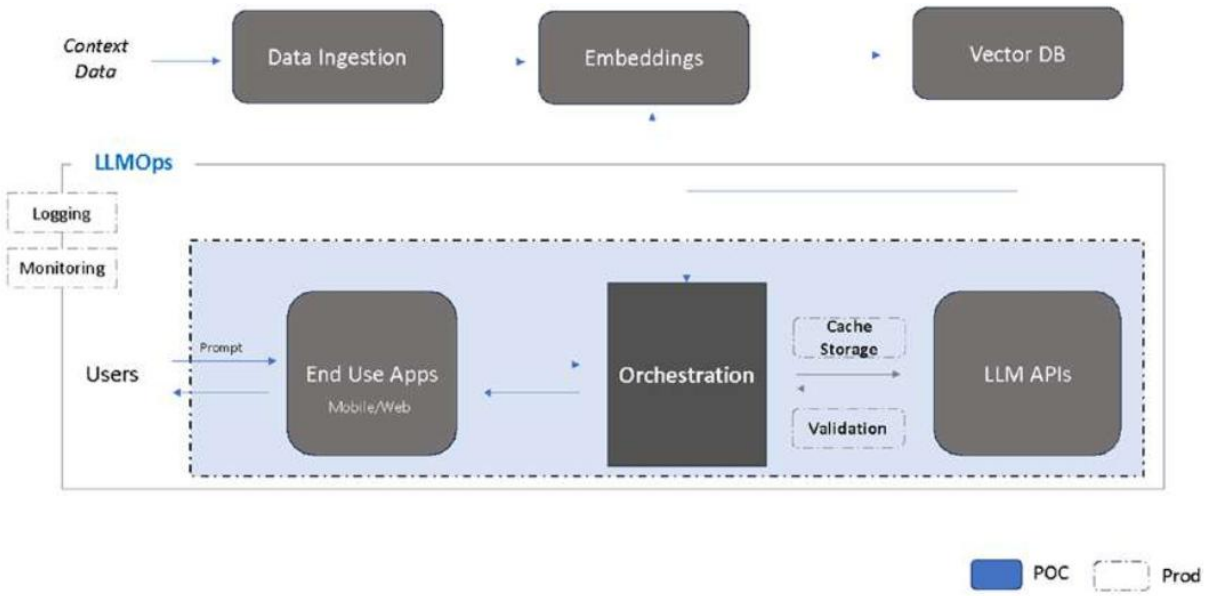
Praprosesing/Penyematan Data: Fase ini mencakup penyimpanan data pribadi (misalnya, dokumen hukum) untuk pengambilan di masa mendatang. Biasanya, dokumen dibagi menjadi beberapa bagian, diproses melalui model penyematan, dan selanjutnya disimpan dalam basis data khusus yang disebut basis data vektor.

Konstruksi/Pengambilan Prompt: Setelah pengguna mengirimkan kueri (misalnya pertanyaan hukum), aplikasi menghasilkan serangkaian prompt untuk model bahasa. Prompt yang dikompilasi biasanya menggabungkan templat prompt yang ditentukan pengembang, contoh keluaran valid yang dikenal sebagai contoh few-shot, data yang diperlukan yang diambil dari API eksternal, dan pilihan dokumen relevan yang diperoleh dari basis data vektor.

Eksekusi/Inferensi Prompt: Setelah prompt dikompilasi, prompt tersebut dimasukkan ke dalam LLM yang telah dilatih sebelumnya untuk inferensi, yang mencakup API model proprietary dan model sumber terbuka atau model yang dilatih sendiri. Dalam beberapa kasus, pengembang melengkapi sistem operasional seperti pencatatan, caching, dan validasi selama fase ini.

Meskipun hal ini mungkin tampak rumit, seringkali lebih sederhana daripada alternatifnya: melatih atau menyempurnakan LLM itu sendiri. Pembelajaran dalam konteks tidak memerlukan tim insinyur pembelajaran mesin yang berdedikasi. Selain itu, Anda tidak perlu mengelola infrastruktur sendiri atau berinvestasi dalam instans khusus yang mahal dari OpenAI. Pendekatan ini pada dasarnya mengubah tantangan AI menjadi tugas rekayasa data, sebuah domain yang sudah familiar bagi banyak perusahaan rintisan dan perusahaan mapan. Pendekatan ini umumnya melampaui penyempurnaan untuk set data yang cukup kecil—mengingat bahwa informasi spesifik perlu ada dalam set pelatihan beberapa kali agar LLM dapat menyimpannya melalui penyempurnaan—dan dapat dengan cepat menggabungkan data baru hampir secara real-time.

Pertanyaan penting tentang pembelajaran dalam konteks berkaitan dengan pengubahan model dasar untuk memperluas jendela konteks. Hal ini memang memungkinkan dan merupakan area penelitian yang aktif. Meskipun demikian, hal ini menimbulkan berbagai konsekuensi, terutama eskalasi kuadratik biaya dan waktu inferensi seiring dengan perpanjangan durasi prompt. Bahkan ekspansi linear (hasil teoretis yang paling menguntungkan) akan terbukti mahal untuk banyak aplikasi saat ini. Saat ini, mengeksekusi satu kueri GPT-4 pada 10.000 halaman akan menghabiskan biaya ratusan dolar berdasarkan tarif API yang berlaku.



Gambar 7.2. Arsitektur Injeksi Konteks

Prapemrosesan/Penyematan Data

Data kontekstual untuk aplikasi LLM mencakup berbagai format, termasuk dokumen teks, PDF, dan data terstruktur seperti tabel CSV atau SQL. Metode pemuatan dan transformasi data ini menunjukkan keragaman yang cukup besar di antara para pengembang yang kami ajak bekerja sama. Banyak yang memilih alat ETL konvensional seperti Databricks atau Airflow.

Sebagian kecil juga menggunakan pemuat dokumen yang terintegrasi ke dalam kerangka kerja orkestrasi seperti LangChain (didukung oleh Unstructured) dan LlamaIndex (didukung oleh Llama Hub).

Namun demikian, kami menganggap aspek kerangka kerja ini relatif kurang berkembang, sehingga menghadirkan peluang untuk solusi replikasi data yang dirancang khusus dan disesuaikan dengan aplikasi LLM.

Dalam ranah penyematan, mayoritas pengembang menggunakan OpenAI API, khususnya model text-embedding-ada-002. Model ini ramah pengguna, terutama bagi mereka yang sudah terbiasa dengan OpenAI API lainnya, menghasilkan hasil yang cukup memuaskan dan secara bertahap lebih hemat biaya. Dalam konteks tertentu, perusahaan-perusahaan besar juga mengeksplorasi Cohere, sebuah platform yang lebih spesifik dalam hal embedding dan menunjukkan kinerja superior dalam skenario tertentu. Bagi pengembang yang cenderung

memilih opsi sumber terbuka, pustaka Hugging Face Sentence Transformers merupakan pilihan standar. Lebih lanjut, terdapat potensi untuk menghasilkan berbagai jenis embedding yang disesuaikan dengan berbagai kasus penggunaan—sebuah aspek yang saat ini merupakan praktik khusus tetapi menjanjikan sebagai bidang penelitian.

Dari perspektif sistem, komponen penting dalam alur praproses adalah basis data vektor. Perannya meliputi penyimpanan, perbandingan, dan pengambilan embedding (atau vektor) yang tak terhitung jumlahnya secara efisien. Pinecone muncul sebagai pilihan paling umum di pasar, terutama karena sifatnya yang dihosting di cloud, memfasilitasi inisiasi yang mudah dan menawarkan serangkaian fitur yang dibutuhkan perusahaan besar untuk produksi, termasuk skalabilitas yang baik, SSO (Single Sign-On), dan SLA waktu aktif. Beragam basis data vektor tersedia, namun:

- *Sistem Sumber Terbuka seperti Weaviate, Vespa, dan Qdrant:* Sistem ini umumnya menunjukkan kinerja yang sangat baik pada node tunggal dan dapat disesuaikan untuk aplikasi tertentu, sehingga disukai oleh tim AI berpengalaman yang cenderung membangun platform khusus.
- *Pustaka Manajemen Vektor Lokal seperti Chroma dan Faiss:* Ini menawarkan pengalaman pengembang yang positif dan dapat dengan cepat disiapkan untuk aplikasi yang lebih kecil dan eksperimen pengembangan. Namun, mereka mungkin tidak sepenuhnya menggantikan basis data yang komprehensif pada skala yang lebih besar.
- *Ekstensi OLTP seperti Pgvector:* Ini adalah opsi yang cocok untuk pengembang yang mencoba mengintegrasikan Postgres untuk setiap kebutuhan basis data atau perusahaan yang sebagian besar menggunakan infrastruktur data mereka dari satu penyedia cloud. Meskipun demikian, integrasi jangka panjang beban kerja vektor dan skalar masih belum jelas.

Dalam hal prospek masa depan, banyak penyedia basis data vektor sumber terbuka yang merambah ke penawaran cloud. Penelitian kami menunjukkan bahwa mencapai kinerja cloud yang tangguh di berbagai lanskap kasus penggunaan potensial merupakan tantangan yang berat. Akibatnya, meskipun beragam opsi mungkin tidak mengalami perubahan langsung yang substansial, pergeseran jangka panjang kemungkinan besar terjadi. Pertanyaan utamanya berkisar pada apakah basis data vektor akan sejajar dengan basis data OLTP dan OLAP dengan terpusat pada satu atau dua sistem yang diterima secara luas.

Pertanyaan lain yang belum terjawab berkaitan dengan bagaimana embedding dan basis data vektor akan berkembang seiring dengan perluasan jendela konteks yang dapat digunakan untuk sebagian besar model. Mungkin tampak intuitif untuk berasumsi bahwa embedding akan menjadi kurang penting karena data kontekstual dapat langsung diintegrasikan ke dalam prompt. Sebaliknya, wawasan dari para ahli di bidang ini menunjukkan hal yang sebaliknya—bahwa signifikansi alur embedding mungkin akan semakin meningkat seiring waktu.

Meskipun jendela konteks yang luas menawarkan utilitas yang cukup besar, jendela tersebut juga memerlukan biaya komputasi yang signifikan, sehingga memerlukan pemanfaatan yang efisien. Kita mungkin akan menyaksikan lonjakan popularitas untuk

berbagai jenis model embedding, yang dilatih secara eksplisit untuk relevansi model, ditambah dengan basis data vektor yang dirancang untuk memfasilitasi dan memanfaatkan kemajuan ini.

Konstruksi/Pengambilan Prompt

Berinteraksi dengan model bahasa besar (LLM) melibatkan proses terstruktur yang menyerupai panggilan API umum. Pengembang membuat permintaan dalam bentuk templat prompt, mengirimkannya ke model, dan kemudian mengurai output untuk memastikan kebenaran dan relevansi. Proses interaksi ini semakin canggih, memungkinkan pengembang untuk mengintegrasikan data kontekstual dan mengorkestrasi respons yang bernuansa, yang krusial untuk berbagai aplikasi.

Pendekatan untuk mendapatkan respons dari LLM dan mengintegrasikan data kontekstual semakin kompleks dan signifikan, muncul sebagai cara penting untuk membedakan produk. Selama dimulainya proyek baru, sebagian besar pengembang memulai dengan eksperimen yang melibatkan prompt yang sederhana. Prompt ini mungkin memerlukan arahan eksplisit (prompt zero-shot) atau bahkan contoh keluaran yang diharapkan (prompt few-shot). Meskipun prompt semacam itu sering kali menghasilkan hasil yang baik, prompt tersebut cenderung tidak mencapai ambang batas akurasi yang diperlukan untuk penerapan produksi yang sebenarnya.

Tingkatan strategi prompt berikutnya, yang sering disebut sebagai "prompting jiu-jitsu", diarahkan untuk menambatkan respons model dalam beberapa bentuk informasi yang dapat diverifikasi dan memperkenalkan konteks eksternal yang belum pernah dialami model selama pelatihan. Panduan Rekayasa Prompt menguraikan tidak kurang dari 12 strategi prompting tingkat lanjut, yang mencakup rantai pemikiran, konsistensi diri, pengetahuan yang dihasilkan, pohon pemikiran, stimulus terarah, dan beberapa lainnya. Strategi-strategi ini juga dapat diterapkan secara sinergis untuk memenuhi beragam aplikasi LLM, mulai dari tanya jawab berbasis dokumen hingga chatbot, dan seterusnya.

Di sinilah kerangka kerja orkestrasi seperti LangChain dan LlamaIndex membuktikan kehebatannya. Kerangka kerja ini mengabstraksi berbagai kerumitan yang terkait dengan perangkaian prompt, berinteraksi dengan API eksternal (termasuk menentukan kapan panggilan API diperlukan), mengambil data kontekstual dari basis data vektor, dan menjaga koherensi di berbagai interaksi LLM. Selain itu, mereka menyediakan templat yang disesuaikan dengan berbagai aplikasi yang umum ditemui. Keluaran yang mereka berikan berupa prompt atau serangkaian prompt yang akan dimasukkan ke dalam model bahasa. Kerangka kerja ini banyak digunakan oleh para penghobi dan perusahaan rintisan yang ingin memulai aplikasi mereka, dengan LangChain sebagai yang terdepan.

Meskipun LangChain merupakan upaya yang relatif baru (saat ini pada versi 0.0.201), contoh aplikasi yang dibangun dengannya sudah bertransisi ke fase produksi. Beberapa pengembang, terutama mereka yang menggunakan LLM pada tahap awal, mungkin memilih untuk beralih ke Python mentah dalam tahap produksi untuk menghindari dependensi tambahan. Namun, kami mengantisipasi pendekatan mandiri ini akan berkurang seiring waktu

di sebagian besar kasus penggunaan, mirip dengan evolusi yang diamati dalam tumpukan aplikasi web tradisional.

Dalam lanskap saat ini, OpenAI berada di garis depan model bahasa. Hampir semua pengembang yang berinteraksi dengan kami memulai aplikasi LLM baru menggunakan API OpenAI, terutama memilih model seperti gpt-4 atau gpt-4-32k. Pilihan ini menawarkan skenario optimal untuk kinerja aplikasi, dengan kemudahan penggunaan di berbagai domain input, yang biasanya tidak memerlukan penyempurnaan atau hosting mandiri.

Seiring proyek memasuki fase produksi dan mencapai skalabilitas, semakin banyak pilihan yang muncul. Beberapa pendekatan umum yang kami temui meliputi:

- ***Transisi ke gpt-3.5-turbo:*** Opsi ini menonjol karena pengurangan biaya sekitar 50 kali lipat dan peningkatan kecepatan yang signifikan dibandingkan dengan GPT-4. Banyak aplikasi tidak memerlukan tingkat presisi GPT-4 tetapi membutuhkan inferensi latensi rendah dan dukungan hemat biaya bagi pengguna gratis. Menjelajahi Vendor Proprietary Lain (Khususnya Model Claude Anthropic): Model Claude menyediakan inferensi cepat, akurasi yang setara dengan GPT-3.5, fleksibilitas kustomisasi yang lebih besar untuk klien yang substansial, dan potensi untuk mengakomodasi jendela konteks hingga 100 ribu (meskipun kami mengamati penurunan akurasi dengan input yang lebih panjang).
- ***Memprioritaskan Permintaan Tertentu untuk Model Open Source:*** Taktik ini dapat sangat efektif untuk skenario B2C bervolume tinggi seperti pencarian atau obrolan, di mana kompleksitas kueri sangat bervariasi dan terdapat kebutuhan untuk melayani pengguna gratis secara ekonomis. Pendekatan ini sering kali cocok dengan penyempurnaan model berbasis open source. Meskipun kami tidak membahas secara mendalam spesifikasi tumpukan perkakas ini dalam artikel ini, platform seperti Databricks, Anyscale, Mosaic, Modal, dan RunPod semakin banyak diadopsi oleh banyak tim teknik. Terdapat beragam opsi inferensi untuk model sumber terbuka, mulai dari antarmuka API sederhana yang disediakan oleh Hugging Face dan Replicate hingga sumber daya komputasi mentah dari penyedia cloud terkemuka, dan penawaran cloud yang lebih beropini seperti yang telah disebutkan sebelumnya.

Saat ini, model sumber terbuka tertinggal dibandingkan model proprietary, namun kesenjangannya semakin menyempit. Model LLaMa Meta telah menetapkan tolok ukur baru untuk akurasi sumber terbuka, yang memicu proliferasi variasi. Karena lisensi LLaMa membatasinya hanya untuk penggunaan penelitian, berbagai penyedia baru telah ikut serta untuk mengembangkan model dasar alternatif (contohnya antara lain Together, Mosaic, Falcon, dan Mistral). Meta juga mempertimbangkan rilis LLaMa 2 yang berpotensi sepenuhnya bersifat sumber terbuka.

Mengantisipasi kemungkinan ketika LLM sumber terbuka mencapai tingkat akurasi yang setara dengan GPT-3.5, kami memperkirakan akan terjadi momen yang mirip dengan Difusi Stabil untuk teks, yang ditandai dengan eksperimen ekstensif, berbagi, dan operasionalisasi model yang telah disempurnakan. Perusahaan hosting seperti Replicate telah menggabungkan perangkat untuk memfasilitasi penggunaan model-model ini oleh

pengembang. Ada keyakinan yang semakin meningkat di kalangan pengembang bahwa model yang lebih kecil dan telah disetel dengan baik dapat mencapai presisi mutakhir dalam kasus penggunaan tertentu.

Sebagian besar pengembang yang kami ajak bekerja sama belum mendalami perkakas operasional untuk LLM saat ini. Caching, yang biasanya dibangun di Redis, relatif luas penggunaannya karena meningkatkan waktu respons aplikasi sekaligus hemat biaya. Alat seperti Weights & Biases dan MLflow (diadaptasi dari pembelajaran mesin tradisional) atau solusi yang berfokus pada LLM seperti PromptLayer dan Helicone juga umum digunakan. Alat-alat ini memungkinkan pencatatan, pelacakan, dan evaluasi keluaran LLM, seringkali untuk tujuan seperti meningkatkan konstruksi prompt, menyempurnakan alur kerja, atau pemilihan model.

Selain itu, beberapa alat baru sedang dikembangkan untuk memvalidasi keluaran LLM (misalnya, Guardrails) atau mengidentifikasi serangan injeksi cepat (misalnya, Rebuff). Sebagian besar alat operasional ini mendorong penggunaan klien Python mereka sendiri untuk memulai pemanggilan LLM, yang memicu rasa ingin tahu tentang bagaimana solusi-solusi ini akan saling melengkapi seiring waktu.

7.3 FINE TUNING (PENYETELAN HALUS)

Penyetelan halus dengan pembelajaran transfer adalah teknik yang menggunakan LLM yang telah dilatih sebelumnya sebagai titik awal untuk melatih model baru pada tugas atau domain tertentu. Hal ini dapat dilakukan dengan membekukan beberapa lapisan LLM yang telah dilatih sebelumnya dan hanya melatih lapisan yang tersisa. Hal ini membantu mencegah model mengalami overfitting terhadap data baru dan memastikan bahwa model tersebut masih mempertahankan pengetahuan umum yang dipelajarinya dari LLM yang telah dilatih sebelumnya.

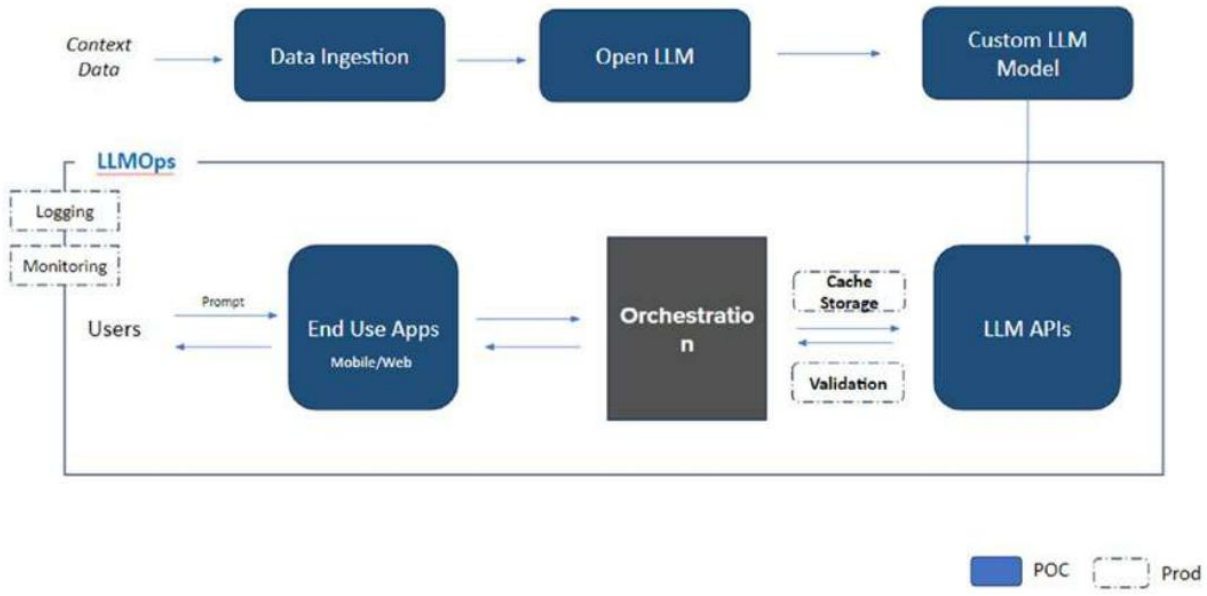
Berikut adalah langkah-langkah yang terlibat dalam penyetelan halus dengan pembelajaran transfer:

1. *Pilih LLM yang Telah Dilatih Sebelumnya:* Ada banyak LLM berbeda yang tersedia, masing-masing dengan kekuatan dan kelemahannya sendiri. Pilihan LLM akan bergantung pada tugas atau domain spesifik yang ingin Anda atur halus modelnya.
2. *Kumpulkan Kumpulan Data Teks dan Kode yang Spesifik untuk Tugas atau Domain Tersebut*: Ukuran dan kualitas kumpulan data akan berdampak signifikan pada kinerja model yang telah disesuaikan halus.
3. *Siapkan Dataset untuk Fine-Tuning:* Ini mungkin melibatkan pembersihan data, penghapusan entri duplikat, dan pemisahan data menjadi set pelatihan dan pengujian.
4. *Bekukan Beberapa Lapisan LLM yang Telah Dilatih:* Ini dapat dilakukan dengan mengatur laju pembelajaran lapisan beku ke nol.
5. *Latih Lapisan LLM yang Tersisa pada Set Pelatihan:* Ini dilakukan dengan menggunakan algoritma pembelajaran terawasi untuk menyesuaikan parameter lapisan yang tersisa agar dapat memprediksi keluaran yang tepat untuk masukan yang diberikan dengan lebih baik.

6. *Evaluasi Model yang Telah Di-Fine-Tuning pada Set Pengujian:* Ini akan memberi Anda gambaran tentang seberapa baik model telah belajar untuk melakukan tugas tersebut.

Fine-tuning dengan transfer learning dapat menjadi cara yang sangat efektif untuk meningkatkan kinerja LLM pada berbagai macam tugas. Namun, penting untuk dicatat bahwa kinerja model yang telah di-fine-tuning akan tetap bergantung pada kualitas dataset yang digunakan untuk fine-tuning model. Berikut adalah contoh penyempurnaan pada Gambar 7.3. Berikut beberapa manfaat penyetelan halus dengan pembelajaran transfer:

- Dapat menghemat waktu dan sumber daya. Pembelajaran transfer dapat digunakan untuk menyempurnakan model pada tugas baru tanpa harus melatih model dari awal.
- Dapat meningkatkan kinerja. Pembelajaran transfer dapat membantu meningkatkan kinerja model pada tugas baru dengan memanfaatkan pengetahuan yang telah dipelajari model dari LLM yang telah dilatih sebelumnya.
- Dapat membuat model lebih dapat digeneralisasi. Pembelajaran transfer dapat membantu membuat model lebih dapat digeneralisasi ke tugas baru dengan mengurangi jumlah data yang dibutuhkan untuk melatih model.



Gambar 7.3. Fine-Tuning / Penyetelan Halus

Namun, terdapat juga beberapa tantangan yang terkait dengan penyetelan halus dengan pembelajaran transfer:

- Memilih hiperparameter yang tepat untuk proses penyetelan halus bisa jadi sulit.
- Menemukan LLM yang telah dilatih sebelumnya yang sesuai untuk tugas baru bisa jadi sulit.
- Mencegah model agar tidak overfitting terhadap data baru bisa jadi sulit.

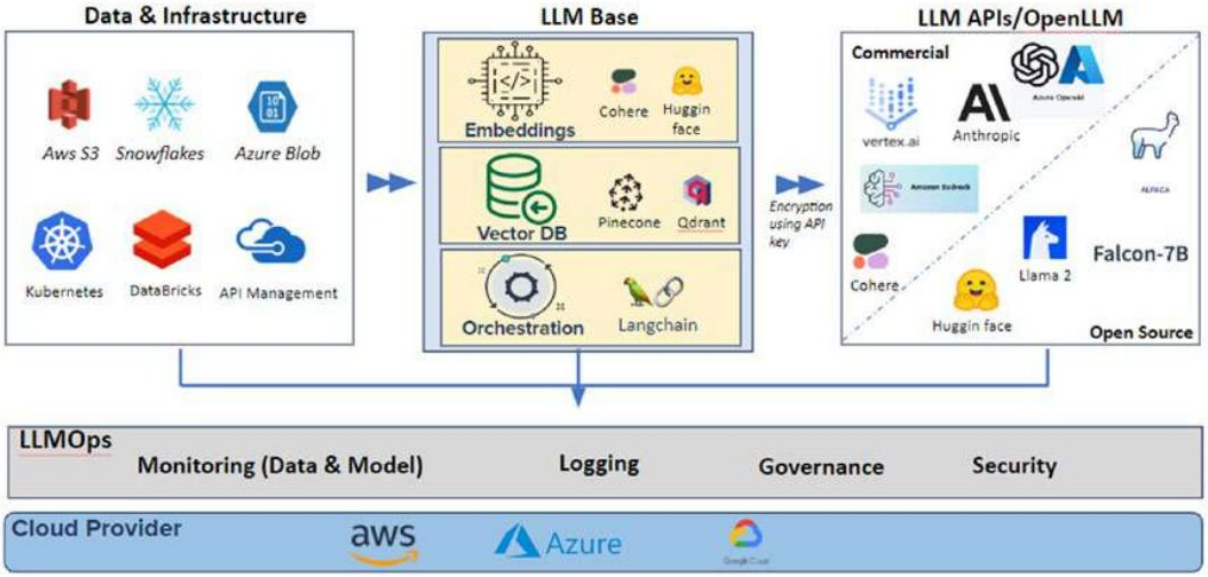
Secara keseluruhan, fine-tuning dengan transfer learning merupakan teknik ampuh yang dapat digunakan untuk meningkatkan kinerja LLM dalam berbagai tugas. Namun, penting untuk mempertimbangkan manfaat dan tantangan fine-tuning dengan transfer learning sebelum mengambil keputusan.

7.4 TEKNOLOGI LLM

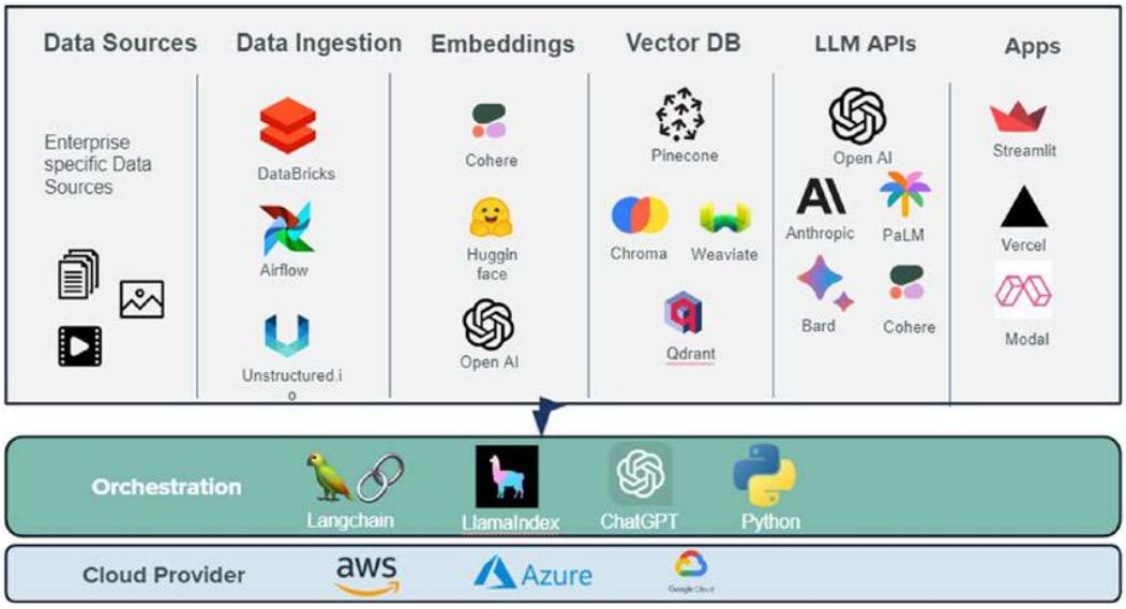
Gen AI/LLM Testbed

Untuk memanfaatkan potensi penuh LLM dan memastikan pengembangannya yang bertanggung jawab, penting untuk membangun testbed LLM khusus. Testbed ini berfungsi sebagai lingkungan yang terkendali untuk meneliti, menguji, dan mengevaluasi LLM, memfasilitasi inovasi sekaligus mengatasi masalah etika, keamanan, dan kinerja. Berikut adalah contoh testbed yang dapat digunakan.

Merancang tumpukan teknologi untuk AI generatif melibatkan pemilihan dan pengintegrasian berbagai alat, kerangka kerja, dan platform yang memfasilitasi pengembangan, pelatihan, dan penerapan model generatif. Gambar 7-5 menunjukkan garis besar tumpukan teknologi yang dapat Anda pertimbangkan.



Gambar 7.4. Uji coba Gen AI/LLM



Gambar 7.5. Tumpukan teknologi untuk AI generatif

Sumber Data

Sumber data merupakan komponen penting dari setiap proyek AI generatif. Kualitas, keragaman, dan kuantitas data yang Anda gunakan dapat memengaruhi performa dan kapabilitas model generatif Anda secara signifikan.

Pemrosesan Data

Dalam upaya untuk mengaktifkan model bahasa besar (LLM) bagi aplikasi perusahaan, memanfaatkan layanan pemrosesan data khusus sangat penting untuk mengelola kerumitan persiapan dan transformasi data secara efisien. Meskipun beberapa layanan berkontribusi pada ranah ini, tiga di antaranya menonjol sebagai pemain kunci: Databricks, Apache Airflow, dan alat seperti Unstructured.io untuk memproses data tidak terstruktur. Penting untuk diketahui bahwa selain opsi-opsi ini, banyak alternatif juga membentuk lanskap layanan pemrosesan data.

Memanfaatkan Embedding untuk LLM Perusahaan

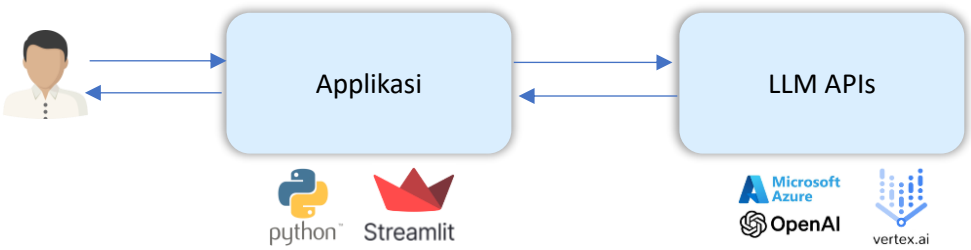
Dalam upaya memberdayakan model bahasa besar (LLM) untuk perusahaan, integrasi embedding berfungsi sebagai strategi ampuh untuk meningkatkan pemahaman semantik. Embedding, representasi numerik ringkas dari kata dan dokumen, berperan penting dalam memungkinkan LLM memahami konteks, hubungan, dan makna. Bagian ini membahas bagaimana embedding dari sumber terkemuka seperti Cohere, OpenAI, dan Hugging Face dapat dimanfaatkan untuk meningkatkan efektivitas LLM dalam konteks perusahaan.

Basis Data Vektor: Mempercepat LLM Perusahaan dengan Pencarian Semantik

Dalam upaya mengoptimalkan model bahasa besar (LLM) untuk aplikasi perusahaan, integrasi basis data vektor muncul sebagai strategi yang inovatif. Basis data vektor, termasuk solusi seperti Pinecone, Chroma, Weaviate, dan Qdrant, merevolusi efisiensi pencarian semantik dan pengambilan konten. Subbagian ini membahas bagaimana basis data vektor ini dapat diintegrasikan secara mulus ke dalam alur kerja LLM, sehingga meningkatkan kecepatan dan presisi pengambilan konten dalam konteks perusahaan.

API LLM: Memberdayakan Kapabilitas Bahasa Perusahaan

Dalam ranah kapabilitas bahasa perusahaan, pemanfaatan API model bahasa besar (LLM) telah muncul sebagai strategi utama. API ini, termasuk penawaran dari OpenAI, Anthropic, Palm, Bard, dan Cohere, memberikan perusahaan akses yang mulus ke kapabilitas pemrosesan bahasa mutakhir. Bagian ini membahas bagaimana API LLM ini dapat dimanfaatkan untuk meningkatkan komunikasi, pembuatan konten, dan pengambilan keputusan dalam konteks perusahaan. Namun, Anda juga dapat menggunakan API LLM umum privat untuk kasus penggunaan Anda sendiri seperti yang ditunjukkan pada Gambar 7.6.

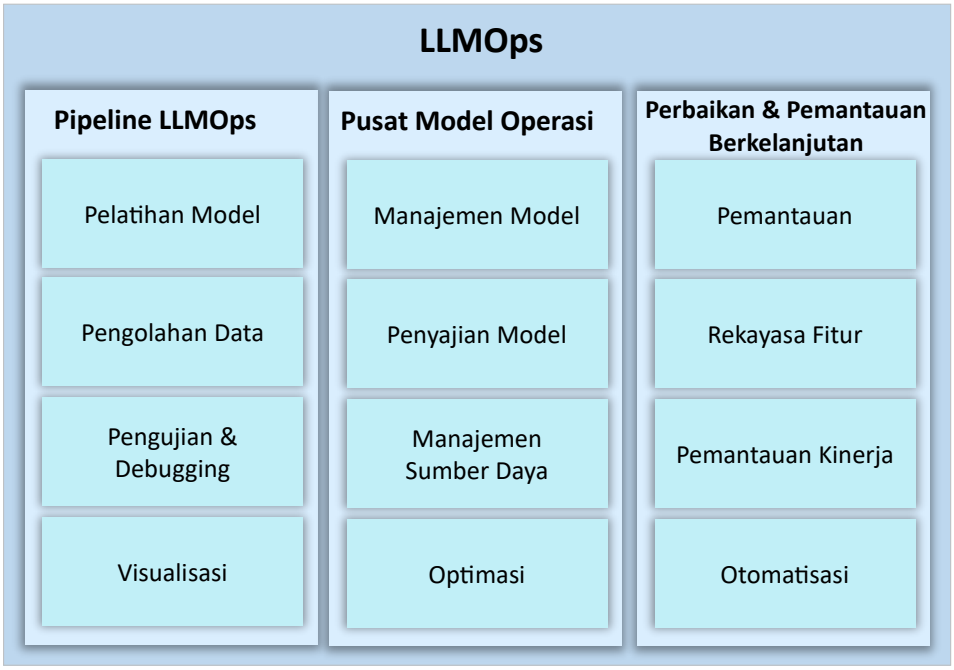


Gambar 7-6. API LLM umum privat

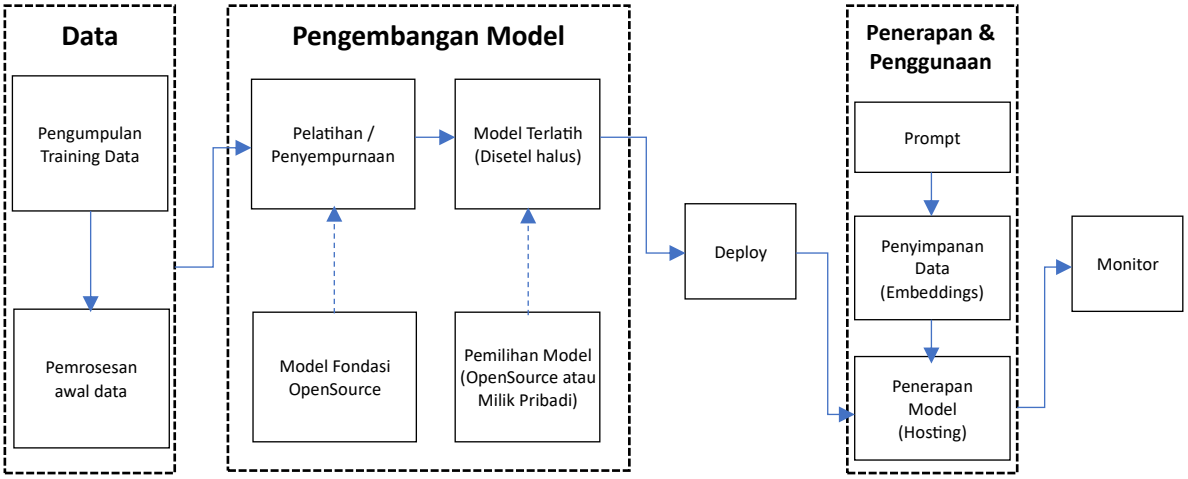
7.5 LLMOPS

Apa itu LLMOps?

Platform LLMOps (operasi model bahasa besar) menawarkan alur kerja yang terdefinisi dengan baik dan komprehensif yang mencakup pelatihan, optimasi, penerapan, dan pemantauan berkelanjutan LLM, baik yang bersifat sumber terbuka maupun kepemilikan. Pendekatan yang efisien ini dirancang untuk mempercepat implementasi model AI generatif dan aplikasinya. Seiring dengan semakin banyaknya organisasi yang mengintegrasikan LLM ke dalam operasional mereka, membangun LLMOps yang tangguh dan efisien menjadi penting. Bagian ini membahas pentingnya LLMOps dan bagaimana LLMOps memastikan keandalan dan efisiensi LLM dalam lingkungan perusahaan. Gambar 7.8 merepresentasikan alur kerja LLMOps.



Gambar 7.7. LLMOps



Gambar 7.8. Alur kerja LLMOps

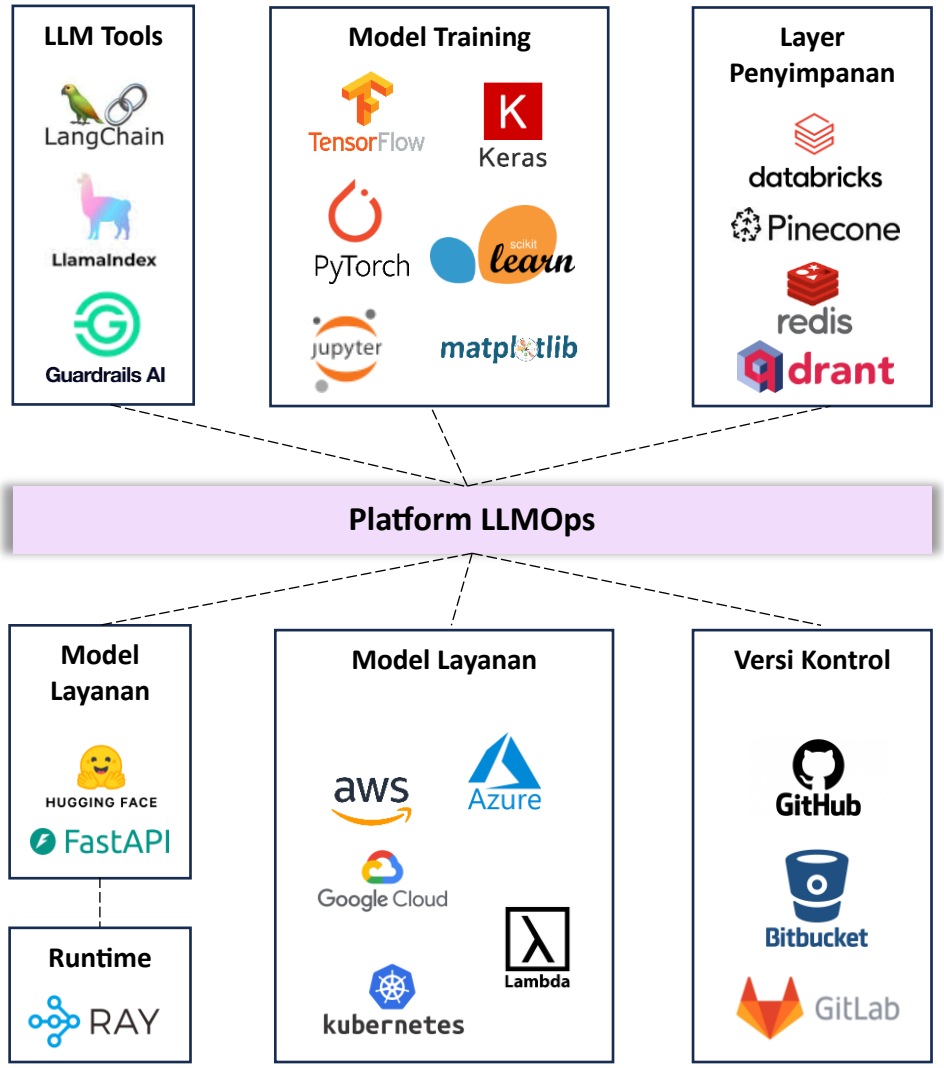
Pengawasan berkelanjutan terhadap model dan aplikasi AI generatif bergantung pada proses pemantauan yang berkelanjutan, yang bertujuan untuk mengatasi tantangan seperti penyimpangan data dan faktor-faktor lain yang dapat menghambat kapasitasnya dalam menghasilkan hasil yang akurat dan aman.

Mengapa LLMOps?

- *Sumber Daya Komputasi:* Alokasi sumber daya yang efisien, penyempurnaan model, pengoptimalan penyimpanan, dan pengelolaan kebutuhan komputasi, memastikan penerapan dan pengoperasian LLM yang efektif menjadi kunci.
- *Penyempurnaan Model:* Penyempurnaan model bahasa besar (LLM) yang telah dilatih sebelumnya mungkin diperlukan untuk menyesuaikannya dengan tugas atau kumpulan data tertentu, memastikan kinerja optimalnya dalam aplikasi praktis.
- *Kekhawatiran Etis:* Model bahasa besar (LLM) memiliki kemampuan untuk menghasilkan konten, tetapi kekhawatiran etis muncul ketika digunakan untuk menghasilkan materi yang berbahaya atau menyinggung.
- *Halusinasi:* LLM "membayangkan" atau "memalsukan" informasi yang tidak secara langsung sesuai dengan sistem dan kerangka kerja masukan yang disediakan untuk memantau presisi dan akurasi keluaran LLM secara berkelanjutan.
- *Interpretabilitas dan Keterjelasan:* Teknik dan langkah-langkah untuk membuat LLM lebih transparan dan mudah diinterpretasikan, sehingga memungkinkan para pemangku kepentingan untuk memahami dan memercayai keputusan yang dibuat oleh model-model ini.
- *Latensi dan Waktu Inferensi:* Tuntutan komputasi LLM dapat mengakibatkan peningkatan latensi, yang memengaruhi aplikasi waktu nyata dan pengalaman pengguna. Hal ini menimbulkan kekhawatiran atas penerapan LLM di area-area yang membutuhkan respons tepat waktu.
- *Kurangnya Struktur dan Kerangka Kerja yang Terdefinisi dengan Baik untuk Manajemen Prompt:* Kurangnya struktur dan kerangka kerja yang terdefinisi dengan baik untuk manajemen prompt merupakan tantangan umum dalam pemanfaatan model bahasa besar (LLM). Aspek krusial penggunaan LLM ini seringkali tidak memiliki perangkat yang terorganisir dan alur kerja yang mapan.

Apa Itu Platform LLMOps?

Platform LLMOps menawarkan lingkungan kolaboratif bagi ilmuwan data dan insinyur perangkat lunak, yang memungkinkan mereka menyederhanakan alur kerja. Platform ini mendukung eksplorasi data iteratif, melacak eksperimen, memfasilitasi rekayasa cepat, mengelola model dan alur kerja, serta memastikan transisi, penerapan, dan pemantauan LLM yang terkendali.



Gambar 7.9. Platform LLMOps

Komponen Teknologi LLMOps

Platform/Kerangka Kerja	Keterangan
Deeplake	Streaming set data multimoda besar untuk mencapai utilisasi GPU mendekati 100%. Lakukan kueri, visualisasikan, dan kontrol versi data. Akses data tanpa perlu menghitung ulang embedding saat melakukan fine-tuning pada model.
LangFlow	Cara sederhana untuk bereksperimen dan membuat prototipe alur LangChain menggunakan komponen drag-and-drop dan antarmuka obrolan yang intuitif.
LLMFlows	LLMFlows adalah kerangka kerja untuk membangun aplikasi LLM yang sederhana, eksplisit, dan transparan seperti chatbot, sistem tanya jawab, dan agen.
BudgetML	Siapkan layanan inferensi pembelajaran mesin yang hemat biaya dengan basis kode ringkas kurang dari sepuluh baris.
Arize-Phoenix	Observabilitas ML untuk LLM, visi, bahasa, dan model tabular.

ZenML	Kerangka kerja sumber terbuka untuk mengoordinasikan, bereksperimen dengan, dan menerapkan solusi pembelajaran mesin yang cocok untuk lingkungan produksi, yang menampilkan integrasi bawaan untuk LangChain dan LlamaIndex.
Dify	Kerangka kerja sumber terbuka ini dirancang untuk memberdayakan pengembang dan non-pengembang agar dapat dengan cepat menciptakan aplikasi praktis menggunakan model bahasa yang besar. Kerangka kerja ini memastikan aplikasi-aplikasi ini ramah pengguna, fungsional, dan mampu ditingkatkan secara berkelanjutan.
xTuring	Bangun dan kendalikan LLM pribadi Anda dengan penyempurnaan yang cepat dan efisien.
Tumpukan jerami	Membuat aplikasi dengan mudah menggunakan agen LLM, pencarian semantik, tanya jawab, dan fitur tambahan.
GPTCache	Menetapkan cache semantik untuk menyimpan respons yang dihasilkan oleh kueri LLM.
EmbedChain	Suatu kerangka kerja untuk mengembangkan bot seperti ChatGPT menggunakan kumpulan data Anda sendiri.

Memantau Model AI Generatif

Gambar 7.10. Pemantauan model AI generatif

Pemantauan model AI generatif seperti yang ditunjukkan pada Gambar 7.10 melibatkan pelacakan berbagai dimensi untuk memastikan penggunaannya yang bertanggung jawab dan efektif. Berikut cara Anda dapat memasukkan aspek-aspek kebenaran, kinerja, biaya, ketahanan, pemantauan cepat, latensi, transparansi, bias, pengujian A/B, dan pemantauan keamanan dalam strategi pemantauan Anda:

1. **Kebenaran:**
    - *Definisi:* Kebenaran mengacu pada keakuratan konten yang dihasilkan dan apakah konten tersebut selaras dengan hasil yang diinginkan.
    - *Pendekatan Pemantauan:* Gunakan pemeriksaan validasi otomatis dan penilaian kualitas untuk memverifikasi bahwa konten yang dihasilkan akurat secara faktual dan sesuai konteks.

2. **Kinerja:**
    - *Definisi:* Kinerja berkaitan dengan kualitas konten yang dihasilkan dalam hal kelancaran, koherensi, dan relevansi.
    - *Pendekatan Pemantauan:* Ukur dan analisis metrik kinerja secara berkelanjutan, seperti kebingungan, skor BLEU, atau skor ROUGE, untuk menilai kualitas teks yang dihasilkan.
3. **Biaya:**
    - *Definisi:* Pemantauan biaya melibatkan pelacakan sumber daya komputasi dan biaya infrastruktur yang terkait dengan pengoperasian model AI.
    - *Pendekatan Pemantauan:* Menerapkan alat pelacakan biaya untuk memantau pemanfaatan sumber daya dan mengoptimalkan biaya sambil mempertahankan kinerja.
4. **Ketahanan:**
    - *Definisi:* Ketahanan menilai kemampuan model AI untuk menangani beragam masukan dan beradaptasi dengan konteks yang berbeda.
    - *Pendekatan Pemantauan*: Menguji respons model terhadap beragam masukan dan memantau perilakunya dalam berbagai kondisi untuk memastikan keandalannya.
5. **Pemantauan Cepat:**
    - *Definisi:* Pemantauan cepat melibatkan pemeriksaan masukan atau perintah yang diberikan kepada model AI dan memastikannya selaras dengan pedoman etika.
    - *Pendekatan Pemantauan:* Meninjau dan mengaudit perintah secara berkala untuk mencegah penyalahgunaan atau masukan yang bias.
6. **Latensi:**
    - *Definisi:* Latensi mengukur waktu respons model AI, memastikannya memenuhi harapan pengguna untuk interaksi yang tepat waktu.
    - *Pendekatan Pemantauan:* Pantau waktu respons dan tetapkan target latensi untuk memastikan interaksi yang cepat dan efisien.
7. **Transparansi:**
    - *Definisi:* Transparansi melibatkan penyediaan wawasan tentang bagaimana model AI beroperasi dan membuat keputusan.
    - *Pendekatan Pemantauan:* Pertahankan catatan yang jelas tentang masukan dan keluaran model, dan pertimbangkan untuk menerapkan alat atau teknik transparansi seperti AI yang dapat dijelaskan untuk meningkatkan interpretabilitas model.
8. **Bias:**
    - *Definisi:* Pemantauan bias berfokus pada identifikasi dan mitigasi bias dalam keluaran model, seperti bias gender, ras, atau budaya.
    - *Pendekatan Pemantauan*: Terapkan algoritma deteksi bias dan lakukan audit rutin untuk mengatasi dan memitigasi potensi bias dalam respons model.

9. **Pengujian A/B:**
    - *Definisi:* Pengujian A/B melibatkan perbandingan kinerja berbagai versi atau konfigurasi model.
    - *Pendekatan Pemantauan:* Lakukan pengujian A/B untuk menilai dampak perubahan atau pembaruan model terhadap kepuasan pengguna, ketepatan, dan metrik utama lainnya.
10. **Pemantauan Keamanan:**
    - *Definisi:* Pemantauan keamanan bertujuan untuk mencegah tindakan atau keluaran berbahaya dari model AI.
    - *Pendekatan Pemantauan:* Terapkan langkah-langkah keamanan, seperti penyaringan konten, deteksi anomali, dan prosedur penghentian darurat, untuk memastikan model beroperasi dengan aman.

      Pertimbangkan contoh "perintah tidak aman" berikut yang berkaitan dengan budaya India:

      Contoh Perintah Tidak Aman: "Buat deskripsi masakan India, tetapi fokuslah hanya pada tingkat kepedasannya dan sebutkan bahwa masakan tersebut terlalu pedas bagi kebanyakan orang."

      Perintah ini berpotensi tidak aman karena terlalu menyederhanakan dan mengstereotipkan masakan India dengan mereduksinya menjadi satu aspek (kepedasan) dan menyiratkan bahwa masakan tersebut mungkin tidak dapat ditoleransi oleh banyak orang, yang bukanlah representasi yang adil atau akurat dari makanan India.
    - *Memantau Respons:* Waspada dalam mengidentifikasi dan menolak arahan yang melanggengkan stereotip, diskriminasi, atau narasi reduksionis. Terapkan algoritma deteksi bias untuk menandai dan menangani arahan yang dapat mengarah pada konten yang tidak akurat atau bias. Komunikasikan dengan jelas pedoman etika yang mencegah arahan yang mempromosikan stereotip atau generalisasi negatif tentang budaya atau kuliner.

      Dengan mengintegrasikan aspek-aspek ini ke dalam strategi pemantauan Anda, Anda dapat secara efektif mengawasi ketepatan, kinerja, efisiensi biaya, ketahanan, ketepatan waktu, latensi, transparansi, mitigasi bias, pengujian A/B, dan keamanan model AI generatif. Tinjau dan perbarui praktik pemantauan Anda secara berkala untuk mengatasi tantangan yang muncul dan memastikan penggunaan AI yang bertanggung jawab.

      Contoh ini menyoroti pentingnya memantau dan menangani permintaan yang tidak aman yang dapat melanggengkan stereotip atau memberikan representasi budaya yang tidak akurat, dalam hal ini, masakan.

Catatan Tambahan:

Meskipun bagian ini memberikan gambaran menyeluruh tentang dimensi pemantauan untuk model AI generatif, perlu dicatat bahwa beberapa pembaca mungkin merasa bermanfaat untuk mengkategorikan dimensi-dimensi ini berdasarkan apakah dimensi tersebut terutama

berkaitan dengan pemantauan permintaan atau respons. Hal ini dapat memberikan perspektif yang lebih terperinci tentang proses pemantauan dan penerapannya dalam alur kerja model AI.

Pembaca yang tertarik dengan kategorisasi semacam itu dapat mempertimbangkan untuk mendekati strategi pemantauan mereka dengan mengidentifikasi aspek mana yang berkaitan dengan permintaan masuk dan mana yang berfokus pada evaluasi respons yang dihasilkan model AI.

Model AI Generatif Proprietary

Model AI generatif proprietary dikembangkan oleh organisasi untuk tujuan tertentu dan biasanya dilindungi oleh perjanjian lisensi komersial. Model ini menawarkan keunggulan dalam hal kualitas, kontrol, dan dukungan, tetapi mungkin disertai dengan batasan penggunaan dan biaya terkait. Tabel 7.1 menunjukkan beberapa model AI generatif yang tersedia saat buku ini ditulis.

Tabel 7.1. Model AI Generatif yang Tersedia

Model	Parameters	Context Length	Fine Tuneable
GPT-3.5	175 miliar	4k/16k	Yes
PaLM 2 (Bison)	540 miliar	?	No
Cohere	52,4 miliar	?	Yes
Claude	175 miliar	9k	No
ada, babbage, curie	Hingga 7 miliar	2k	Yes

Model Sumber Terbuka dengan Lisensi Permisif

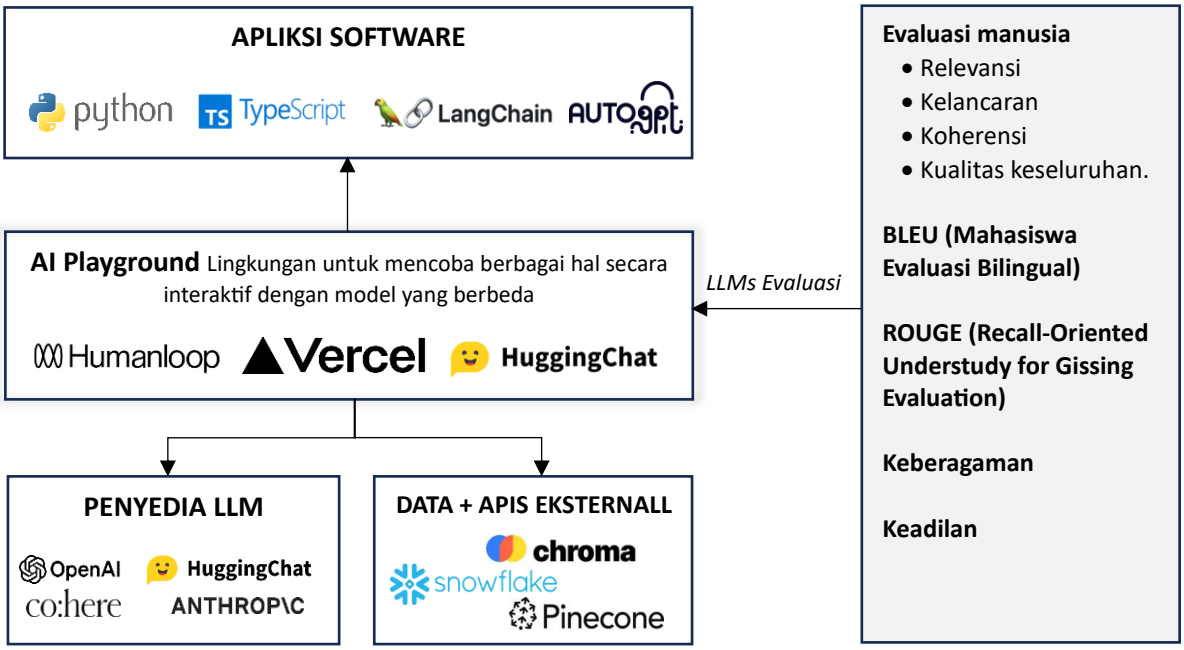
Tabel 7.2 menunjukkan daftar model sumber terbuka dengan lisensi permisif.

Tabel 7-2. Model Sumber Terbuka

Language Model	Parameter	Context Length
T5	11 Miliar	2k
UL2	20 Miliar	2k
Pythia, Dolly 2.0	12 Miliar	2k
MPT-7B	7 Miliar	84k
RedPajama-INCITE	7 Miliar	2k
Falcon	40 Miliar	2k
MPT-30B	30 Miliar	8k
LLaMa 2	70 Miliar	4k

Arena untuk Pemilihan Model

Area pemilihan model seperti yang ditunjukkan pada Gambar 7.11 adalah lingkungan atau ruang kerja tempat para ilmuwan data dan praktisi pembelajaran mesin dapat secara sistematis mengevaluasi dan membandingkan berbagai model dan algoritma pembelajaran mesin untuk memilih yang paling sesuai untuk tugas atau set data tertentu. Membangun arena semacam itu melibatkan beberapa langkah dan pertimbangan, dan berikut adalah contoh bagaimana hal itu dapat dilakukan.



Gambar 7.11. Area untuk pemilihan model

Metrik Evaluasi

Metrik evaluasi merupakan alat penting dalam menilai performa model, algoritma, dan sistem pembelajaran mesin di berbagai tugas. Metrik ini membantu mengukur seberapa baik performa suatu model, sehingga memudahkan untuk membandingkan berbagai model dan membuat keputusan yang tepat. Berikut adalah beberapa kerangka kerja dan pustaka populer untuk mengevaluasi LLM:

Tabel 7.3. Kerangka Kerja dan Pustaka untuk Mengevaluasi LLM

Nama Kerangka	Faktor yang Dipertimbangkan untuk Evaluasi	Link URL
Big Bench	Kemampuan generalisasi	https://github.com/google/BIG-bench
GLUE Benchmark	Tata bahasa, parafrase, kesamaan teks, inferensi, tekstual E=entailment, menyelesaikan referensi kata ganti	https://gluebenchmark.com/
SuperGLUE Benchmark	Pemahaman bahasa alami, penalaran, memahami kalimat kompleks di luar data pelatihan, pembangkitan bahasa alami yang koheren dan terbentuk dengan baik, dialog dengan manusia, penalaran akal sehat, pengambilan informasi, pemahaman bacaan	https://super.gluebenchmark.com/
OpenAI Moderation API	Menyaring yang berbahaya atau tidak aman	https://platform.openai.com/docs/api-reference/moderations
MMLU	Pemahaman bahasa di berbagai tugas dan domain	https://github.com/hendrycks/test

EleutherAI LM Eval	Mengevaluasi dan menilai kinerja di berbagai tugas dengan penyempurnaan minimal menggunakan pendekatan pembelajaran beberapa percobaan	https://github.com/EleutherAI/1m-evaluation-harness
OpenAI Evals	Mengevaluasi kualitas dan atribut teks yang dihasilkan, termasuk akurasi, keragaman, konsistensi, ketahanan, transferabilitas, efisiensi, dan kewajaran.	https://github.com/openai/evals
Adversarial NLI (ANLI)	Ketahanan, generalisasi, penjelasan inferensi yang koheren, konsistensi penalaran di antara contoh-contoh serupa, efisiensi dalam hal penggunaan sumber daya.	https://github.com/facebookresearch/anli
IT (Language Interpretability Tool)	Platform untuk mengevaluasi berdasarkan metrik yang ditentukan pengguna. Wawasan tentang kekuatan, kelemahan, dan potensi bias mereka.	https://pair-code.github.io/lit/
ParlAI	Akurasi, skor f1, kebingungan, evaluasi manusia berdasarkan kriteria seperti relevansi, kelancaran, dan koherensi, kecepatan dan pemanfaatan sumber daya, ketahanan, generalisasi.	https://github.com/facebookresearch/ParlAI
CoQA	Memahami suatu teks dan menjawab serangkaian pertanyaan yang saling terkait yang muncul dalam percakapan.	https://stanfordnlp.github.io/coqa/
LAMBDA	Mencapai pemahaman jangka panjang dengan memprediksi kata terakhir dari suatu teks.	https://zenodo.org/record/2630551#.ZFUKS-zMLOP
HellaSwag	Kemampuan penalaran.	https://rowanzellers.com/hellaswag/
LogiQA	Kemampuan penalaran logis.	https://github.com/1gw863/LogiQA-dataset
MultiNLI	Memahami hubungan antar kalimat di berbagai genre.	https://cims.nyu.edu/~sbowman/multinli/
SQUAD	Tugas pemahaman bacaan.	https://rajpurkar.github.io/SQuAD-explorer/

Memvalidasi Keluaran LLM

Memvalidasi keluaran model bahasa besar (LLM) merupakan langkah penting dalam memastikan kualitas, keandalan, keamanan, dan etika penggunaan model bahasa yang canggih ini. Berikut beberapa alasan penting untuk memvalidasi keluaran LLM:

1. ***Jaminan Kualitas:*** LLM mampu menghasilkan teks dalam jumlah besar, tetapi mungkin tidak semuanya berkualitas tinggi. Memvalidasi keluaran LLM membantu memastikan bahwa konten yang dihasilkan memenuhi standar yang diinginkan untuk keterbacaan, koherensi, dan relevansi.
2. ***Pertimbangan Etis:*** LLM terkadang dapat menghasilkan konten yang bias, menyinggung, atau berbahaya. Validasi penting untuk mencegah munculnya konten yang tidak etis atau tidak pantas, seperti ujaran kebencian, misinformasi, atau bahasa diskriminatif.
3. ***Keamanan:*** Untuk melindungi pengguna dan mencegah bahaya, sangat penting untuk memvalidasi keluaran LLM guna memastikan keluaran tersebut tidak berisi instruksi atau informasi yang dapat menyebabkan tindakan berbahaya atau melukai diri sendiri.

4. ***Mitigasi Bias***: LLM diketahui mewarisi bias yang ada dalam data pelatihannya. Validasi keluaran LLM mencakup pendeteksian dan mitigasi bias untuk memastikan keadilan dan nondiskriminasi dalam konten yang dihasilkan.
5. ***Kepercayaan Pengguna:*** Validasi keluaran membantu membangun dan mempertahankan kepercayaan pengguna terhadap aplikasi yang didukung oleh LLM. Pengguna lebih cenderung berinteraksi dan memercayai sistem yang secara konsisten menyediakan konten berkualitas tinggi, etis, dan aman.
6. ***Kepatuhan terhadap Pedoman:*** Banyak organisasi dan platform memiliki pedoman dan kebijakan khusus terkait kualitas, etika, dan keamanan konten. Validasi memastikan kepatuhan terhadap pedoman ini untuk menghindari risiko hukum atau reputasi.
7. ***Peningkatan Berkelanjutan:*** Validasi dan pemantauan keluaran LLM secara berkala memungkinkan peningkatan berkelanjutan. Umpan balik pengguna dan hasil validasi dapat menjadi dasar pembaruan dan penyesuaian model untuk memastikan kinerja yang lebih baik dari waktu ke waktu.
8. ***Akuntabilitas:*** Menyimpan catatan proses validasi dan tindakan yang diambil sebagai respons terhadap keluaran yang bermasalah membangun akuntabilitas jika terjadi masalah atau perselisihan.
9. ***Kepatuhan terhadap Peraturan dan Etika***: Kepatuhan terhadap persyaratan etika, hukum, dan peraturan sangat penting saat menerapkan LLM di domain sensitif atau yang diatur. Validasi membantu memastikan kepatuhan terhadap persyaratan ini.
10. ***Kustomisasi dan Pembuatan Konten Terpandu:*** Validasi dapat digunakan untuk memandu pembuatan konten LLM berdasarkan tujuan tertentu, yang memungkinkan organisasi untuk menyesuaikan konten yang dihasilkan dengan kebutuhan mereka.
11. ***Jaring Pengaman:*** Menerapkan mekanisme validasi berfungsi sebagai jaring pengaman untuk menangkap dan menyaring konten berbahaya atau berkualitas rendah sebelum disajikan kepada pengguna.

Tantangan yang Dihadapi Saat Menerapkan LLM

1. ***Sumber Daya Komputasi:*** Menyimpan dan mengelola LLM berukuran besar dapat menjadi tantangan, terutama di lingkungan dengan keterbatasan sumber daya atau perangkat edge. Hal ini mengharuskan pengembang untuk menemukan cara mengompresi model atau menggunakan teknik seperti distilasi model untuk menciptakan varian yang lebih kecil dan lebih efisien.
2. ***Penyempurnaan Model:*** LLM yang telah dilatih sebelumnya seringkali memerlukan penyempurnaan pada tugas atau kumpulan data tertentu untuk mencapai kinerja optimal. Proses ini dapat memakan biaya komputasi yang besar. Misalnya, penyempurnaan model DaVinci dengan parameter 175 miliar akan menelan biaya $180 ribu.
3. ***Kekhawatiran Etis:*** LLM terkadang dapat menghasilkan konten yang tidak pantas atau bias karena sifat data yang digunakan untuk pelatihan. Hal ini menimbulkan

kekhawatiran tentang implikasi etis dari penerapan model tersebut dan potensi kerugian yang mungkin ditimbulkannya.

4. ***Halusinasi:*** Halusinasi adalah fenomena di mana ketika pengguna mengajukan pertanyaan atau memberikan perintah, LLM menghasilkan respons yang imajinatif atau kreatif tetapi tidak berdasarkan kenyataan. Respons ini mungkin tampak masuk akal dan koheren tetapi tidak didasarkan pada pengetahuan yang sebenarnya.
5. ***Interpretabilitas dan Keterjelasan:*** Memahami cara kerja internal LLM dan bagaimana pengambilan keputusan dapat menjadi sulit karena kompleksitasnya. Kurangnya interpretabilitas ini menimbulkan tantangan bagi pengembang yang perlu men-debug, mengoptimalkan, dan memastikan keandalan model ini dalam aplikasi dunia nyata.
6. ***Latensi dan Waktu Inferensi:*** Karena LLM memiliki banyak parameter, LLM dapat lambat dalam menghasilkan prediksi, terutama pada perangkat dengan sumber daya komputasi terbatas. Hal ini dapat menjadi tantangan ketika menerapkan LLM dalam aplikasi waktu nyata di mana latensi rendah sangat penting.
7. ***Privasi Data dan Kontrol Akses:*** Melindungi data sensitif yang digunakan untuk fine-tuning dan inferensi sangatlah penting. Mematuhi peraturan privasi data dan menerapkan mekanisme kontrol akses yang kuat sangat penting untuk melindungi data pengguna dan menjaga kepercayaan.
8. ***Sumber Daya Terlatih untuk Menangani LLM:*** Organisasi membutuhkan personel terlatih yang memiliki keahlian dalam LLM, termasuk penyempurnaan, pertimbangan etika, dan optimalisasi kinerja.
9. ***Ketahanan Model di Berbagai Kasus Penggunaan:*** Memastikan bahwa LLM berkinerja baik dan memberikan respons yang bermakna di berbagai aplikasi dan domain merupakan tantangan yang signifikan karena model mungkin unggul dalam beberapa kasus penggunaan dan kesulitan di kasus lainnya.

   Gambar 7-8 merepresentasikan alur kerja LLMOps. Mematuhi persyaratan hukum dan peraturan sangat penting saat menerapkan LLM, terutama di industri yang diatur seperti layanan kesehatan dan keuangan. Menavigasi hak kekayaan intelektual, undang-undang perlindungan data, dan peraturan khusus industri dapat menjadi rumit.
10. ***Integrasi dengan Sistem yang Ada:*** Mengintegrasikan LLM secara mulus dengan infrastruktur dan sistem perangkat lunak yang ada merupakan hal yang kompleks. Kompatibilitas, alur data, dan keselarasan dengan proses bisnis yang ada harus dipertimbangkan dengan cermat.
11. ***Manajemen Keamanan dan Kerentanan:*** Penerapan LLM menimbulkan risiko keamanan, termasuk kerentanan terhadap serangan musuh. Mengembangkan strategi untuk mengidentifikasi dan memitigasi risiko ini serta memastikan transmisi data yang aman sangatlah penting.
12. ***Penanganan Umpan Balik Pengguna:*** Mengelola umpan balik pengguna, terutama dalam aplikasi pembangkitan konten, sangat penting untuk perbaikan model yang

berkelanjutan. Menetapkan mekanisme untuk memproses umpan balik pengguna dan mengintegrasikannya ke dalam pembaruan model merupakan tugas yang menantang.

13. ***Kemampuan Multibahasa dan Multimoda:*** Jika suatu aplikasi memerlukan dukungan untuk beberapa bahasa atau masukan multimoda (misalnya, teks dan gambar), memastikan bahwa LLM dapat menanganinya secara efektif dan memberikan respons yang koheren akan menambah kompleksitas penerapan.
14. ***Pemeliharaan Jangka Panjang:*** Penerapan LLM memerlukan pemeliharaan berkelanjutan, termasuk pemantauan penyimpangan model, adaptasi terhadap kebutuhan pengguna yang terus berkembang, dan mengatasi tantangan yang muncul.

7.6 IMPLEMENTASI LLM

Menggunakan OpenAI API dengan Python

Dalam lanskap digital yang serba cepat saat ini, kemampuan untuk memahami dan berinteraksi dengan bahasa manusia telah menjadi pengubah permainan. OpenAI API muncul sebagai alat canggih yang memberdayakan pengembang dan bisnis untuk mengintegrasikan kehebatan pemrosesan bahasa alami ke dalam aplikasi mereka secara mulus. Dengan memanfaatkan model bahasa mutakhir OpenAI, pengembang dapat memanfaatkan kemampuan pemahaman bahasa berbasis AI, pembuatan, dan banyak lagi.

Di bagian ini, kita akan mendalami dunia OpenAI API dan mengungkap langkah-langkah untuk memaksimalkan potensinya secara efektif menggunakan Python. Baik Anda sedang merancang chatbot cerdas, menghasilkan konten kreatif, atau mendorong interaksi berbasis bahasa yang mendalam, OpenAI API membuka pintu menuju kemungkinan yang tak terbatas. Mari kita urai cara kerja API ini yang rumit, mulai dari menyiapkan lingkungan Anda hingga merancang aplikasi menarik yang berinteraksi secara cerdas dengan pengguna. Mari kita jelajahi masa depan interaksi manusia-komputer bersama-sama.

Menggunakan OpenAI API dengan Python

Di bagian ini, kita akan membahas proses penggunaan OpenAI API dengan Python menggunakan contoh praktis PDF "Alice's Adventures in Wonderland". Kita akan mempelajari pembuatan teks, analisis, dan tanya jawab menggunakan OpenAI API.

Prasyarat

- Python 3.x terinstal
- Akses ke OpenAI API dan kunci API
- Instalasi ChromaDB
- Alice's Adventures in Wonderland PDF dari www.gutenberg.org/ebooks/11

Instalasi

Pertama, mari kita instal pustaka yang diperlukan.

```
!pip install openai langchain pypdf unstructured "unstructured[pdf]"
```

Inisialisasi Lingkungan dan Pengaturan Kunci API

```
import openai
import langchain
from langchain import OpenAI, VectorDBQA
from langchain.document_loaders import UnstructuredFileLoader
from langchain.text_splitter import CharacterTextSplitter
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import Chroma
from langchain.chains import RetrievalQA

openai.api_key = "your_openai_api_key_here"
```

Ganti "your_openai_api_key_here" dengan kunci API yang Anda peroleh dari akun OpenAI Anda.

Uji Lingkungan

Verifikasi bahwa lingkungan Anda telah dikonfigurasi dengan benar dengan menjalankan panggilan API sederhana. Misalnya, Anda dapat mencoba membuat teks menggunakan metode "openai.Completion.create()".

```
response = openai.Completion.create(
    engine="davinci",
    prompt="Once upon a time in a",
    max_tokens=50
)

print(response.choices[0].text.strip())
```

```
land called Wade: Ebner, John Wesley. “Dr. Wade in Hampton,” in GRA.

117At a time when no American: Reynolds, Keynote Address.

117To everyone’s relief
```

Persiapan Data: Memuat Data PDF

Muat data PDF.

```
def load_pdf(pdf_path):
  loader = UnstructuredFileLoader(pdf_path)
  pages= loader.load()
  return pages
```

Membagi data menjadi beberapa bagian:

Kami menggunakan CharacterTextSplitter untuk membagi konten PDF menjadi beberapa bagian. Setiap bagian kemudian diproses secara terpisah menggunakan OpenAI API. Pendekatan ini memastikan input tetap mudah dikelola dan berada dalam batas token, sekaligus memungkinkan Anda menganalisis atau menghasilkan teks untuk keseluruhan PDF.

```
pages=load_pdf('/content/drive/MyDrive/Colab Notebooks/Alices_Adventures_in_Wonderland_by_Lewis Carroll.pdf')
text_to_chunks = CharacterTextSplitter(chunk_size=500, chunk_overlap=0)
chunks_of_text = text_to_chunks.split_documents(pages)
```

Ingatlah bahwa ukuran potongan dan tumpang tindih dapat memengaruhi kualitas dan koherensi hasil. Ini merupakan pilihan antara tetap berada dalam batasan token dan mempertahankan konteks.

Embedding dan VectorDB Menggunakan LangChain dan Chroma

LangChain menawarkan kerangka kerja yang praktis untuk pembuatan prototipe aplikasi lokal yang cepat berdasarkan LLM (model bahasa besar). Selain itu, Chroma menghadirkan penyimpanan vektor terintegrasi dan basis data embedding yang beroperasi dengan lancar selama tahap pengembangan lokal, sehingga memberdayakan aplikasi-aplikasi ini.

```
embeddings_function = OpenAIEmbeddings(openai_api_key = openai.api_key)
docsearch = Chroma.from_documents(chunks_of_text,embeddings_function)
chain = RetrievalQA.from_chain_type(llm=OpenAI(), chain_type="stuff", retriever=docsearch.as_retriever())
```

Memanfaatkan API OpenAI

Tanya Jawab di PDF:

```
# Input the query at runtime
user_query = input("Enter your query: ")

# Run the QA using the provided query
qa_result = chain.run(user_query)
print("OpenAI Response:", qa_result)
```

```
Enter your query:
```

Pertanyaan 1: Siapakah Pahlawan buku ini?

```
# Input the query at runtime
user_query = input("Enter your query: ")

# Run the QA using the provided query
qa_result = chain.run(user_query)
print("OpenAI Response:", qa_result)
```

```
Enter your query: Who is the Hero of this book?
OpenAI Response:  The hero of this book is Alice.
```

Pertanyaan 2: Siapa penulis Alice in Wonderland?

```
# Input the query at runtime
user_query = input("Enter your query: ")

# Run the QA using the provided query
qa_result = chain.run(user_query)
print("OpenAI Response:", qa_result)
```

```
Enter your query: Who is the author of Alice in Wonderland?
OpenAI Response:  The author of Alice's Adventures in Wonderland is Lewis Carroll.
```

Pertanyaan 3: Apa yang terjadi pada ukuran Alice saat dia makan atau minum?

```
# Input the query at runtime
user_query = input("Enter your query: ")

# Run the QA using the provided query
qa_result = chain.run(user_query)
print("OpenAI Response:", qa_result)
```

```
Enter your query: What happens to the size of Alice when she eats or drinks?
OpenAI Response:  Alice remains the same size.
```

Jika Anda melihat jawaban untuk kueri sebelumnya salah, respons OpenAI menunjukkan bahwa Alice tetap berukuran sama saat makan atau minum. Namun, dalam "Alice's Adventures in Wonderland", ukurannya justru berubah. Hal ini mungkin disebabkan oleh konteks dan informasi yang tersedia dalam bagian teks tertentu yang dianalisis. Perlu diingat bahwa akurasi respons bergantung pada konten dan konteks teks yang diproses oleh model OpenAI.

Perhatikan bahwa menulis ulang kueri dengan konteks yang lebih banyak akan memberikan hasil yang lebih baik.

```
[71] # Input the query at runtime
     user_query = input("Enter your query: ")

     # Run the QA using the provided query
     qa_result = chain.run(user_query)
     print("OpenAI Response:", qa_result)

     Enter your query: Did Alice grow larger in the book? If yes, why?
     OpenAI Response:  Yes, Alice grows larger in the book. She grew larger because she ate a piece of mushroom which caused her to grow in size.
```

Pertanyaan 4: Analisis interaksi antara Alice dan Ratu Hati dalam PDF.

```
# Input the query at runtime
user_query = input("Enter your query: ")

# Run the QA using the provided query
qa_result = chain.run(user_query)
print("OpenAI Response:", qa_result)
```

```
Enter your query: Analyze the interactions between Alice and the Queen of Hearts in the PDF
OpenAI Response:
Alice and the Queen of Hearts have a tense exchange in the PDF. Alice is bold and unafraid to voice her opinion, even when interrupting the King.
```

Sebagai kesimpulan, panduan ini menunjukkan integrasi OpenAI API, LangChain, dan ChromeDb untuk mengekstrak wawasan dari PDF "Alice in Wonderland" dan menjalankan kueri yang tertarget. Kombinasi teknologi kontemporer dengan literatur klasik ini menawarkan pendekatan yang unik dan inovatif, menunjukkan kekuatan perangkat modern dalam menganalisis kisah-kisah abadi.

Memanfaatkan Layanan Azure OpenAI

Layanan Azure OpenAI menawarkan akses REST API yang nyaman ke berbagai model bahasa yang andal, termasuk seri model GPT-4, GPT-35-Turbo, dan Embeddings yang sangat canggih. Lebih lanjut, perlu dicatat bahwa seri model GPT-4 dan gpt-35-turbo kini tersedia untuk penggunaan umum. Model-model ini dapat disesuaikan secara mulus dengan kebutuhan spesifik Anda, mencakup tugas-tugas seperti pembuatan konten, peringkasan, pencarian semantik, dan penerjemahan bahasa alami ke kode. Anda dapat berinteraksi

dengan layanan ini melalui REST API, Python SDK, atau melalui antarmuka berbasis web kami yang tersedia di Azure OpenAI Studio.

Lebih lanjut, salah satu keuntungan utama memanfaatkan Layanan Azure OpenAI adalah kemampuan untuk menukar model bahasa secara mulus berdasarkan kebutuhan Anda. Kemampuan pertukaran ini menjadi lebih efektif ketika diintegrasikan dengan orkestrator seperti LangChain. Dengan pengaturan ini, Anda dapat dengan mudah beralih di antara berbagai model bahasa untuk menyesuaikan dengan tugas atau skenario tertentu. Baik Anda membutuhkan model untuk pembuatan konten, penerjemahan bahasa, atau tugas pemrosesan bahasa alami lainnya, kombinasi LLM yang dapat dipertukarkan dan orkestrator menyediakan kemampuan adaptasi yang dibutuhkan perusahaan Anda.

Mengimplementasikan Layanan Azure OpenAI ke dalam alur kerja perusahaan Anda dapat membuka kemungkinan baru untuk pemahaman, pembuatan, dan interaksi bahasa alami. Layanan ini merupakan alat yang ampuh untuk meningkatkan pengalaman pelanggan, mengotomatiskan proses, dan mendapatkan wawasan dari data tekstual. Silakan temukan URL berikut untuk panduan terperinci tentang cara mengimplementasikan Azure AI untuk perusahaan Anda di situs web Microsoft Azure.

Kesimpulan

Dalam lanskap teknologi perusahaan yang terus berkembang, model bahasa besar (LLM) telah muncul sebagai sekutu yang tangguh, menawarkan transformasi mendalam dalam cara bisnis beroperasi, berinteraksi, dan berinovasi. Saat kita mengakhiri bab ini, kita berada di persimpangan antara peluang dan inovasi, di mana kekuatan LLM berpadu dengan ambisi perusahaan yang berpikiran maju. Sepanjang bab ini, kami telah mengeksplorasi tiga pendekatan menarik untuk memanfaatkan kemampuan LLM dalam lingkungan perusahaan:

API LLM Umum Privat: Kami mendalami konsep API LLM umum privat, menyoroti nilai yang dibawanya melalui privasi data, kustomisasi, dan kontrol. Kami menyaksikan bagaimana hal ini memberdayakan perusahaan untuk menciptakan solusi yang disesuaikan, memperkuat keterlibatan pelanggan, dan menavigasi medan interaksi bahasa alami yang rumit. Dengan menggabungkan pendekatan ini, perusahaan siap menciptakan pengalaman transformatif sekaligus melindungi data sensitif.

Arsitektur Injeksi Konteks: Kami menjelajah ke ranah arsitektur injeksi konteks, sebuah strategi cerdik untuk melengkapi LLM dengan pengetahuan dan konteks spesifik domain. Saat kami mengeksplorasi potensinya, kami mengungkap bagaimana hal ini meningkatkan dukungan pelanggan, meningkatkan kurasi konten, dan mempertajam proses pengambilan keputusan. Perusahaan yang menerapkan pendekatan ini dapat memperkuat penawaran mereka, menyediakan interaksi yang diperkaya dan sadar konteks bagi klien dan pengguna.

Penyetelan LLM untuk Kasus Penggunaan Perusahaan: Konsep penyempurnaan LLM membuka pintu menuju presisi dan adaptabilitas. Kami mengamati bagaimana praktik ini meningkatkan LLM dengan mengoptimalkan akurasinya, melengkapinya dengan bahasa khusus domain, dan meningkatkan kinerja spesifik tugasnya. Dalam skenario yang mencakup analisis sentimen, peninjauan dokumen hukum, dan pembuatan kode, perusahaan dapat

memanfaatkan LLM yang telah disempurnakan untuk mencapai hasil tak tertandingi yang disesuaikan dengan kebutuhan unik mereka.

Saat merenungkan pendekatan ini, kami teringat bahwa perjalanan dengan LLM bukanlah sebuah tujuan, melainkan eksplorasi yang berkelanjutan. Di dunia di mana teknologi terus berkembang, perusahaan yang merangkul LLM dan beradaptasi dengan potensinya lebih siap untuk mengatasi tantangan dan meraih peluang yang ada di depan.

Perpaduan LLM dan solusi perusahaan bukan sekadar gambaran sekilas tentang masa depan; melainkan langkah berani untuk membentuknya. Kemungkinannya tak terbatas, dan jalan ke depan menjanjikan inovasi yang belum terbayangkan. Kami mengundang perusahaan untuk memulai perjalanan transformatif ini, berbekal pengetahuan dan strategi untuk memanfaatkan potensi penuh teknologi LLM. Memasuki era di mana model bahasa lebih dari sekadar alat—mereka adalah mitra dalam inovasi—perusahaan yang mengadopsi LLM tidak hanya akan menavigasi masa depan, tetapi juga memimpin, mengantarkan era pengalaman pelanggan yang diperkaya, operasi yang efisien, dan kemungkinan yang belum terpetakan. Perjalanan telah dimulai, dan masa depan ada di tangan kita.

BAB 8
MODEL DIFUSI DAN AI GENERATIF UNTUK CITRA

Dua model generatif terkemuka, yaitu jaringan adversarial generatif (GAN) dan autoencoder variasional (VAE), telah mendapatkan pengakuan substansial. Kita akan melihat penjelasan singkat keduanya dalam bab ini diikuti dengan model difusi yang terperinci. GAN telah menunjukkan fleksibilitas di berbagai aplikasi, namun kompleksitas pelatihan dan keragaman keluarannya yang terbatas, yang disebabkan oleh tantangan seperti mode collaps dan gradien menghilang, telah terbukti. Di sisi lain, VAE, meskipun memiliki landasan teoretis yang kuat, menghadapi kesulitan dalam merancang fungsi kerugian yang efektif, sehingga menghasilkan keluaran yang suboptimal.

Kategori teknik lain, yang terinspirasi oleh estimasi kemungkinan probabilistik dan penarikan paralel dari fenomena fisika, telah muncul—yang dikenal sebagai model difusi. Konsep inti model difusi berakar pada prinsip-prinsip yang mirip dengan pergerakan molekul gas dalam termodinamika, di mana molekul menyebar dari daerah berdensitas tinggi ke berdensitas rendah, yang merepresentasikan peningkatan entropi atau disipasi panas. Dalam ranah teori informasi, hal ini berkaitan dengan pengenalan derau secara progresif yang menyebabkan hilangnya informasi.

Inti dari pemodelan difusi terletak pada gagasan menarik bahwa jika kita dapat membangun model pembelajaran yang mampu menangkap degradasi informasi secara bertahap akibat derau, secara teoritis seharusnya dapat membalikkan proses ini, sehingga mendapatkan kembali informasi asli dari derau. Konsep ini mirip dengan VAE, di mana fungsi objektif dioptimalkan dengan memproyeksikan data ke dalam ruang laten dan kemudian memulihkannya ke keadaan awalnya. Namun, perbedaannya terletak pada fakta bahwa model difusi tidak berusaha mempelajari distribusi data secara langsung. Sebaliknya, model ini berfokus pada pemodelan serangkaian distribusi derau dalam kerangka kerja rantai Markov, yang secara efektif "mendekode" data dengan menghilangkan derau secara iteratif secara hierarkis. Sebelum beralih ke model difusi, mari kita lihat penjelasan singkat tentang VAE dan GAN.

8.1 AUTOENCODER VARIASIONAL (VAE)

Autoencoder variasional (VAE) adalah jenis model generatif yang menggabungkan ide dari autoencoder dan pemodelan probabilistik. VAE dirancang untuk mempelajari representasi laten data yang menangkap fitur-fitur bermakna sekaligus menghasilkan sampel data baru yang menyerupai kumpulan data asli. VAE sangat berguna untuk tugas-tugas seperti kompresi data, denoising, dan pemodelan generatif:

1. *Encoder:* Bagian encoder dari VAE mengambil data masukan dan memetakannya ke ruang laten. Tidak seperti autoencoder tradisional, encoder dari VAE tidak menghasilkan penyandian tetap, melainkan menghasilkan distribusi probabilitas atas

variabel laten. Hal ini memungkinkan VAE untuk menangkap ketidakpastian dalam proses penyandian.

2. *Ruang Laten:* Ruang laten adalah representasi data masukan berdimensi lebih rendah. Setiap titik dalam ruang ini berkorespondensi dengan sampel data potensial. VAE mengasumsikan bahwa data dalam ruang laten mengikuti distribusi probabilistik tertentu, seringkali distribusi Gaussian.
3. *Trik Reparameterisasi:* Untuk mengaktifkan backpropagation untuk pelatihan, VAE menggunakan trik reparameterisasi. Alih-alih mengambil sampel langsung dari distribusi laten, sampel dihasilkan dengan menambahkan derau acak ke parameter rerata dan deviasi standar distribusi. Hal ini memungkinkan penghitungan gradien untuk pelatihan.
4. *Dekoder:* Dekoder mengambil sampel dari ruang laten dan memetakannya kembali ke ruang data asli. Seperti halnya enkoder, dekoder juga menghasilkan distribusi probabilitas atas data, yang memungkinkan model untuk menangkap ketidakpastian dalam proses pembangkitan.
5. *Fungsi Kerugian:* VAE dilatih untuk memaksimalkan batas bawah pada kemungkinan data. Batas bawah ini terdiri dari dua istilah: kerugian rekonstruksi yang mengukur seberapa baik data yang dihasilkan sesuai dengan data asli dan istilah regularisasi yang mendorong distribusi laten agar menyerupai distribusi sebelumnya yang diasumsikan. Istilah regularisasi membantu memastikan ruang laten tetap terstruktur dan berkelanjutan.
6. *Pembangkitan dan Interpolasi:* Setelah dilatih, VAE dapat menghasilkan sampel data baru dengan mengambil sampel dari ruang laten dan meneruskan sampel tersebut melalui dekoder. Selain itu, karena ruang laten memiliki struktur yang halus, interpolasi antar titik dalam ruang ini menghasilkan interpolasi yang bermakna dalam ruang data.

VAE telah menunjukkan efektivitasnya dalam berbagai aplikasi, termasuk pembangkitan gambar, kompresi data, dan adaptasi domain. VAE menyediakan cara yang berprinsip untuk mempelajari representasi laten data yang bermakna sekaligus menghasilkan sampel baru yang beragam dan realistis. Namun, VAE mungkin menghasilkan keluaran yang sedikit buram dibandingkan dengan model generatif lain seperti GAN karena adanya trade-off inheren antara akurasi rekonstruksi dan keragaman sampel dalam fungsi objektifnya.

8.2 JARINGAN ADVERSARIAL GENERATIF (GAN)

Jaringan adversarial generatif (GAN) adalah kelas model pembelajaran mesin yang dirancang untuk menghasilkan data baru yang serupa dengan kumpulan data tertentu. GAN terdiri dari dua komponen utama: generator dan diskriminator. Generator menciptakan sampel data sintetis, sementara diskriminator mengevaluasi sampel-sampel ini dan mencoba membedakan antara data asli dan data yang dihasilkan. Kedua komponen dilatih bersama dalam proses yang kompetitif, yang menghasilkan penyempurnaan kemampuan generator untuk menciptakan data yang realistis dan kemampuan diskriminator untuk membedakan antara data asli dan palsu:

1. ***Generator (G):*** Generator mengambil derau acak sebagai masukan dan mengubahnya menjadi data yang seharusnya menyerupai dataset target. Awalnya, keluarannya mungkin tidak terlalu menyerupai data asli.
2. ***Diskriminator (D):*** Diskriminator bertindak sebagai pengklasifikasi biner. Diskriminator mengambil data asli dari dataset target dan data yang dihasilkan dari generator sebagai masukan dan mencoba menentukan apakah masukan tersebut asli (dari dataset) atau palsu (dihasilkan oleh generator).
3. ***Proses Pelatihan:*** Pelatihan GAN melibatkan proses adversarial. Generator dan diskriminator dilatih secara iteratif. Selama setiap iterasi:
   - Generator menghasilkan data palsu dari derau acak.
   - Diskriminator diberikan data asli dan data palsu yang dihasilkan, dan ia belajar membedakan keduanya.
   - Parameter generator disesuaikan untuk menghasilkan data palsu yang lebih baik sehingga diskriminator kesulitan membedakannya dari data asli.
4. ***Tujuan:*** Tujuan generator adalah meningkatkan kemampuannya dalam menghasilkan data yang begitu meyakinkan sehingga diskriminator tidak dapat membedakannya dari data asli. Tujuan diskriminator adalah menjadi lebih baik dalam mengklasifikasikan data asli dan palsu dengan tepat.
5. ***Keseimbangan:*** Seiring berjalannya pelatihan, generator dan diskriminator mencapai titik keseimbangan di mana generator menghasilkan data yang semakin sulit dibedakan oleh diskriminator dari data asli. Hal ini menghasilkan data sintetis berkualitas tinggi. GAN telah digunakan untuk berbagai aplikasi, termasuk sintesis gambar, transfer gaya, resolusi super, augmentasi data, dan banyak lagi. GAN telah menunjukkan kemampuan untuk menciptakan sampel data yang sangat realistis dan telah berkontribusi pada kemajuan yang mengesankan dalam pemodelan generatif dan visi komputer. Namun, GAN dapat menjadi tantangan untuk dilatih karena masalah seperti mode collaps (ketika generator berfokus pada subset terbatas dari data target) dan ketidakstabilan pelatihan.

8.3 MODEL DIFUSI

Model difusi adalah kelas model generatif yang relatif baru yang terinspirasi dari proses fisik seperti difusi partikel dan konsep dari teori informasi. Model ini bertujuan untuk menghasilkan data dengan mengubah derau menjadi informasi terstruktur secara iteratif, yang pada dasarnya membalikkan proses pengenalan derau. Singkatnya, model difusi bekerja sebagai berikut:

1. ***Jadwal Derau:*** Urutan tingkat derau didefinisikan, yang secara bertahap meningkat dari derau minimal ke derau yang lebih signifikan. Setiap tingkat derau mewakili trade-off antara kejelasan dan derau dalam data.
2. ***Rantai Markov:*** Model difusi menggunakan rantai Markov, yang terdiri dari beberapa langkah yang sesuai dengan tingkat derau yang berbeda dalam jadwal. Pada setiap

langkah, model memproses data dengan menambahkan derau dan secara bertahap mendistorsinya.

3. ***Pemodelan Kondisional:*** Model menciptakan distribusi kondisional yang memperkirakan tampilan data pada setiap tingkat derau, berdasarkan data pada tingkat sebelumnya. Hal ini secara efektif menangkap degradasi data akibat derau.
4. ***Proses Balik:*** Setelah data diproses melalui rantai Markov dengan tingkat derau yang meningkat, proses balik diterapkan. Proses ini bertujuan untuk memulihkan data asli dengan menghilangkan derau secara iteratif, bergerak mundur melalui jadwal derau.
5. ***Tujuan Pelatihan:*** Model difusi dilatih dengan mengoptimalkan parameter untuk meminimalkan perbedaan antara distribusi data yang diestimasi pada setiap tingkat derau dan data aktual yang diamati pada tingkat tersebut. Hal ini biasanya dicapai dengan memaksimalkan kemungkinan mengamati data berdasarkan proses difusi yang dimodelkan. Konsep di balik model difusi adalah merepresentasikan hilangnya informasi secara bertahap akibat derau, kemudian menggunakan pengetahuan ini untuk memulihkan informasi asli dengan membatalkan pengenalan derau. Tidak seperti model generatif tradisional yang secara langsung memodelkan distribusi data, model difusi berfokus pada pemodelan proses penambahan dan penghapusan derau.

Model difusi telah menunjukkan potensi dalam menghasilkan sampel data berkualitas tinggi dengan beragam karakteristik. Model ini berpotensi menangkap distribusi data yang kompleks dan menangani skenario di mana kualitas data menurun seiring waktu, yang dapat sangat berguna untuk aplikasi dalam pembuatan gambar, pengurangan derau data, dan lainnya.

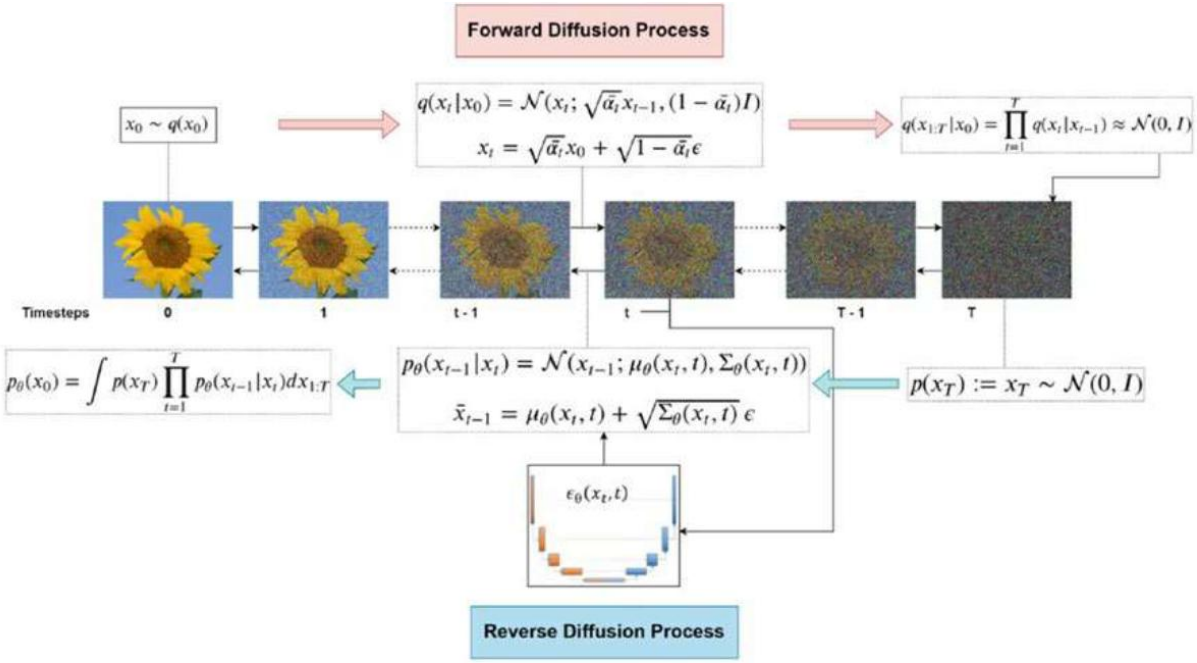
Namun, hingga pembaruan terakhir saya pada September 2021, model difusi mungkin belum dipelajari atau diimplementasikan secara luas seperti model generatif lainnya seperti GAN atau VAE.

Jenis-Jenis Model Difusi

Ada banyak jenis model difusi, tetapi beberapa yang paling umum meliputi:

- *Model Probabilistik Difusi Denoising (DDPM):* DDPM adalah jenis model difusi yang dimulai dengan citra yang bernoise dan secara bertahap menghilangkan noise tersebut untuk menampilkan citra yang mendasarinya. DDPM dilatih menggunakan teknik yang disebut estimasi kemungkinan maksimum, yang berarti DDPM dilatih untuk meminimalkan jarak antara citra yang dihasilkan dan citra nyata dalam set data pelatihan. Gambar 8.1 mengilustrasikan model probabilistik difusi denoising.
- *Model Difusi Berbasis Skor (SBM):* SBM adalah jenis model difusi yang menggunakan fungsi skor untuk menghasilkan gambar. Fungsi skor adalah fungsi yang mengukur seberapa besar kemungkinan suatu gambar nyata. SBM dilatih menggunakan teknik yang disebut pelatihan adversarial, yang berarti SBM dilatih untuk menghasilkan gambar yang tidak dapat dibedakan dari gambar nyata. Gambar 8.2 mengilustrasikan model difusi berbasis skor.
- *Model Difusi Berbasis Persamaan Diferensial Stokastik (SDE):* Model difusi berbasis SDE adalah jenis model difusi yang menggunakan persamaan diferensial stokastik (SDE) untuk menghasilkan gambar. SDE adalah persamaan yang menggambarkan evolusi

suatu proses acak dari waktu ke waktu. Model difusi berbasis SDE dilatih menggunakan teknik yang disebut pelatihan adversarial generatif, yang berarti model tersebut dilatih untuk menghasilkan gambar yang tidak dapat dibedakan dari gambar nyata. Gambar 8.3 mengilustrasikan model difusi berbasis persamaan diferensial stokastik (SDE).

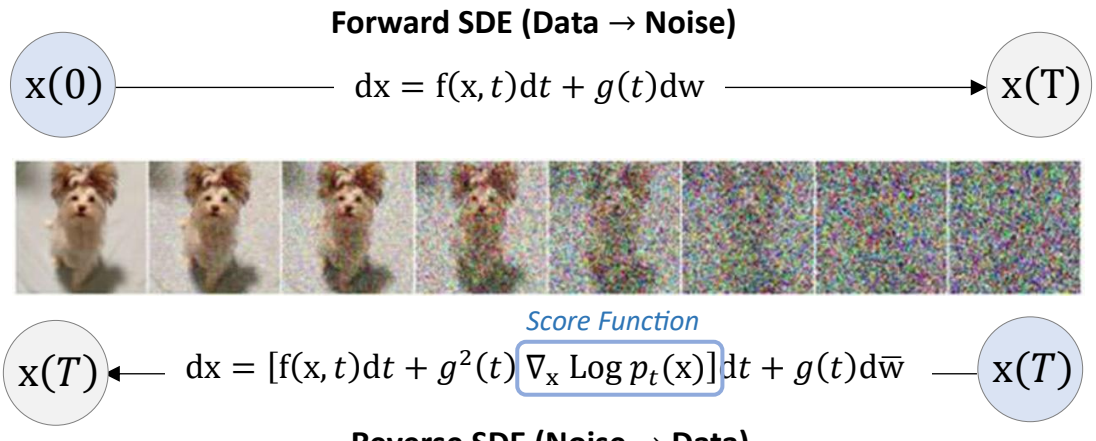


Gambar 8.1. Model probabilistik difusi denoising (DDPM)

Score Based Generatif Model

$$s(x) \triangleq \nabla_x \operatorname{Log} p_{data}(x)$$

Gambar 8-2. Model difusi berbasis skor



Gambar 8-3. Model difusi berbasis persamaan diferensial stokastik (SDE)

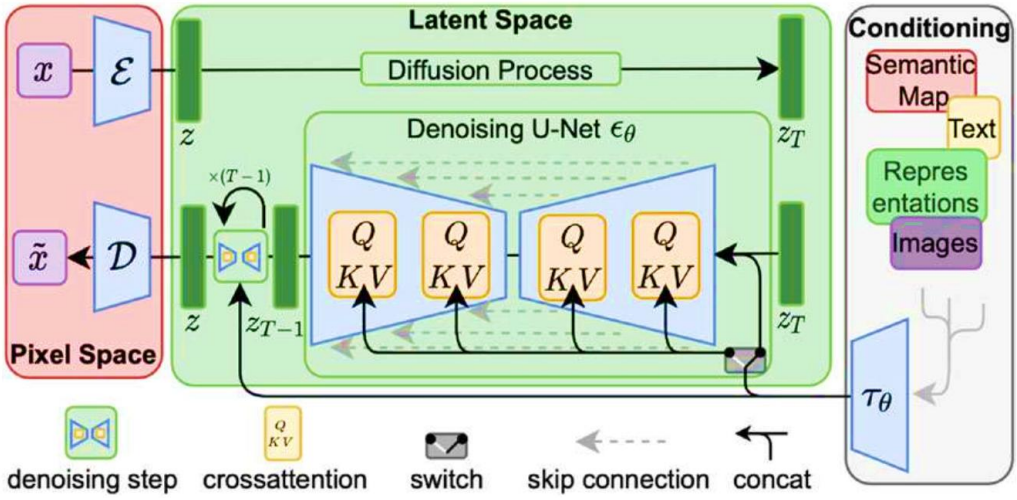
Model difusi telah berhasil digunakan untuk berbagai tugas, termasuk:

- *Pembuatan Gambar:* Model difusi dapat digunakan untuk menghasilkan gambar realistis dari deskripsi teks.
- *Sintesis Teks-ke-Gambar:* Model difusi dapat digunakan untuk mensintesis gambar dari deskripsi teks.
- *Transfer Gaya:* Model difusi dapat digunakan untuk mentransfer gaya dari satu gambar ke gambar lainnya.
- *Resolusi Super:* Model difusi dapat digunakan untuk melakukan resolusi super pada gambar beresolusi rendah.

Arsitektur

Model difusi merupakan alat yang ampuh untuk menghasilkan konten yang realistis dan kreatif. Model ini masih dalam tahap pengembangan, tetapi berpotensi merevolusi cara kita membuat dan berinteraksi dengan gambar. Arsitektur model difusi relatif sederhana. Model ini terdiri dari dua komponen utama:

- **Model Representasi Laten:** Model representasi laten biasanya berupa jaringan saraf tiruan yang menerima citra sebagai masukan dan mengeluarkan representasi laten dari citra tersebut. Representasi laten merupakan vektor angka yang menangkap fitur-fitur penting dari citra tersebut. Model representasi laten dilatih pada kumpulan data citra nyata. Tujuan model representasi laten adalah untuk mempelajari pemetaan dari citra ke representasi laten sehingga citra yang serupa satu sama lain memiliki representasi laten yang serupa. Gambar 8-4 mengilustrasikan model representasi laten dalam model difusi.



Gambar 8-4. Model representasi laten dalam model difusi

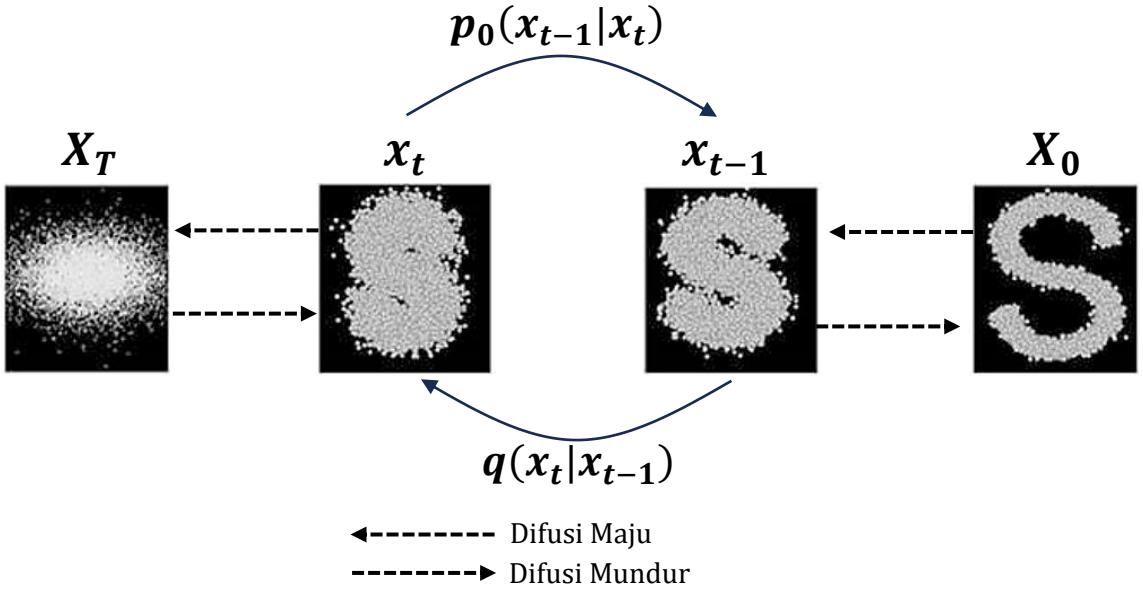
Model representasi laten dapat diimplementasikan menggunakan semua jenis jaringan saraf tiruan, tetapi jaringan saraf tiruan konvolusional (CNN) sering digunakan. CNN sangat cocok untuk tugas-tugas pemrosesan citra karena dapat belajar mengekstraksi fitur dari citra pada skala yang berbeda.

Model representasi laten dilatih menggunakan teknik yang disebut estimasi kemungkinan maksimum. Estimasi kemungkinan maksimum adalah teknik statistik yang menemukan parameter model yang memaksimalkan kemungkinan data yang diamati. Dalam kasus model representasi laten, data yang diamati adalah kumpulan data citra nyata. Tujuan estimasi kemungkinan maksimum adalah untuk menemukan parameter model representasi laten yang membuat model tersebut paling mungkin menghasilkan citra nyata dalam kumpulan data.

- **Proses Difusi:** Proses difusi adalah rantai Markov yang mengambil representasi laten sebagai input dan memodifikasinya secara bertahap untuk menghasilkan citra baru. Proses difusi bersifat probabilistik, artinya proses ini hanya dapat berpindah dari satu keadaan ke keadaan berikutnya dengan cara tertentu. Proses difusi dilatih untuk menghasilkan citra yang tidak dapat dibedakan dari citra nyata.

  Proses difusi bekerja dengan terlebih dahulu menambahkan derau ke representasi laten. Jumlah derau yang ditambahkan ditentukan oleh parameter yang disebut laju difusi. Laju difusi meningkat secara bertahap seiring berjalannya proses difusi. Ini berarti citra yang dihasilkan menjadi semakin berbeda dari citra asli seiring berjalannya proses difusi.

  Proses difusi dapat diimplementasikan menggunakan semua jenis rantai Markov, tetapi pendekatan yang umum adalah menggunakan proses difusi Gaussian. Proses difusi Gaussian adalah rantai Markov yang menambahkan derau Gaussian ke representasi laten di setiap langkah. Proses difusi dilatih menggunakan teknik yang disebut pelatihan adversarial. Gambar 8-5 mengilustrasikan proses difusi dalam model difusi.



Gambar 8-5. Proses difusi dalam model difusi

Pelatihan adversarial adalah teknik untuk melatih model generatif yang mengadu dua model. Dalam kasus model difusi, kedua model tersebut adalah proses difusi dan

diskriminator. Diskriminator adalah jaringan saraf tiruan yang dilatih untuk membedakan antara citra nyata dan citra yang dihasilkan.

Tujuan pelatihan adversarial adalah melatih proses difusi untuk menghasilkan citra yang sangat realistis sehingga diskriminator tidak dapat membedakannya dari citra nyata. Hal ini dilakukan dengan memperbarui parameter proses difusi dan diskriminator secara iteratif hingga diskriminator tidak dapat membedakan antara citra nyata dan citra yang dihasilkan dengan keyakinan tinggi.

- **Proses Dekoding:** Proses dekoding biasanya berupa jaringan saraf tiruan yang menerima representasi laten sebagai masukan dan mengeluarkan citra. Proses dekoding dilatih untuk merekonstruksi citra asli dari representasi laten.

  Proses dekoding dapat diimplementasikan menggunakan semua jenis jaringan saraf tiruan, tetapi CNN sering digunakan. CNN sangat cocok untuk tugas rekonstruksi gambar karena dapat belajar membalikkan operasi yang dilakukan oleh model representasi laten.

  Proses dekode dilatih menggunakan teknik yang disebut mean squared error (MSE). MSE loss adalah fungsi kerugian yang mengukur perbedaan antara citra hasil rekonstruksi dan citra asli. Tujuan MSE loss adalah meminimalkan perbedaan antara citra hasil rekonstruksi dan citra asli.

  Dalam beberapa tahun terakhir, bidang kecerdasan buatan (AI) telah mengalami kemajuan yang signifikan, memperkenalkan berbagai inovasi. Salah satu tambahan penting dalam lanskap AI adalah munculnya generator gambar AI. Alat canggih ini memiliki kemampuan untuk mengubah masukan tekstual menjadi gambar yang hidup atau penggambaran artistik. Di antara banyaknya pilihan yang tersedia untuk solusi AI teks-ke-gambar, beberapa telah menarik perhatian khusus, yang menonjol adalah DALL-E 2, difusi stabil, dan Midjourney.

8.4 TEKNOLOGI DI BALIK DALL-E 2

Pernahkah Anda penasaran tentang bagaimana AI mampu mengubah kata menjadi gambar? Bayangkan mendeskripsikan sesuatu dalam teks dan kemudian menyaksikan AI menciptakan gambar dari deskripsi tersebut. Menghasilkan gambar berkualitas tinggi hanya dari deskripsi tekstual telah menjadi tantangan besar bagi para peneliti AI. Di sinilah DALL-E dan versi lanjutannya, DALL-E 2, berperan. Dalam artikel ini, kami akan membahas seluk-beluk DALL-E 2.

Dikembangkan oleh OpenAI, DALL-E 2 adalah model AI canggih dengan kemampuan luar biasa untuk menghasilkan gambar yang sangat realistis berdasarkan deskripsi tekstual. Namun, bagaimana DALL-E 2 mencapai prestasi ini, dan apa yang membedakannya? Sepanjang artikel ini, kami akan membahas konsep dan teknik fundamental yang mendasari DALL-E 2. Kami akan mengeksplorasi konsep-konsep seperti pra-pelatihan citra bahasa kontrastif (CLIP), model difusi, dan pasca-pemrosesan.

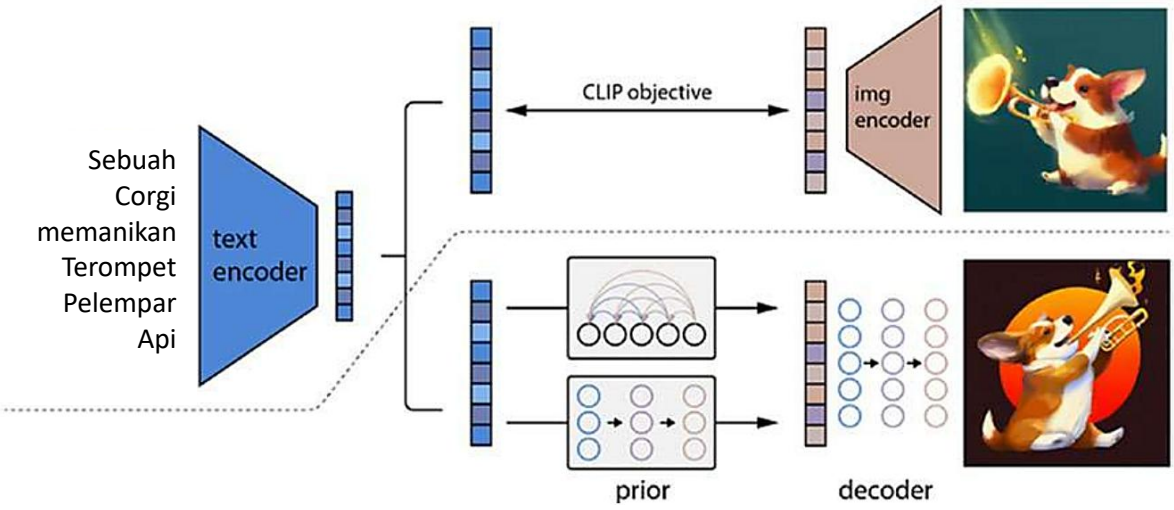
Selain itu, kami akan membahas sumber daya komputasi yang diperlukan untuk melatih model seperti DALL-E 2, beserta kerangka kerja dan pustaka pembelajaran mendalam

yang memfasilitasi implementasinya. Setelah selesai membaca, Anda akan memiliki pemahaman yang kuat tentang cara kerja DALL-E 2 dan apa yang menjadikannya sebuah kemajuan inovatif dalam bidang AI generatif.

DALL-E 2 merupakan versi evolusi dari DALL-E asli, yang beroperasi dalam domain model bahasa besar. Model generatif ini memanfaatkan kekuatan model difusi untuk mengubah deskripsi tekstual menjadi gambar nyata. Model ini memanfaatkan arsitektur encoder-decoder, dengan alur kerja khas yang berpusat pada penyematan pra-pelatihan gambar bahasa kontrastif (CLIP):

1. *Pemrosesan Teks Input:* Pada awalnya, DALL-E 2 memproses deskripsi tekstual yang diberikan oleh pengguna, yang mendeskripsikan gambar yang mereka bayangkan.
2. *Pengodean Menggunakan CLIP:* Teks input diode menggunakan jaringan saraf CLIP. CLIP mahir dalam mengubah input teks dan gambar menjadi embedding berdimensi tinggi, menangkap esensi semantiknya. Hal ini menghasilkan representasi vektor yang disebut embedding teks CLIP, yang merangkum makna deskripsi tekstual.
3. *Konversi ke Embedding Gambar CLIP melalui Prior:* Embedding teks CLIP kemudian diarahkan melalui "Prior", yang dapat berupa model autoregresif atau difusi. Ini merupakan langkah penting di mana transisi dari teks ke gambar terjadi. Prior, yang beroperasi sebagai model generatif, memanfaatkan distribusi probabilitas untuk menciptakan gambar yang tampak nyata. Secara khusus, model difusi disukai karena kinerjanya yang unggul dalam menghasilkan gambar berkualitas tinggi.
4. *Pembuatan Gambar Akhir:* Setelah Prior, khususnya model difusi, menghasilkan penyisipan gambar CLIP, penyisipan ini diteruskan ke dekoder difusi. Peran dekoder difusi adalah menerjemahkan penyisipan ini menjadi gambar akhir, yang menghasilkan representasi visual yang dijelaskan dalam teks masukan.

   Gambar 8.6 mengilustrasikan DALL-E 2.



Gambar 8.6. DALL-E 2

Yang penting, terdapat eksperimen selama pengembangan DALL-E 2. Meskipun pendekatan langsung menggunakan penyisipan teks CLIP dalam dekoder (langkah #4) telah dicoba,

mengintegrasikan prior (langkah #3) ternyata lebih efektif dalam meningkatkan kualitas pembuatan gambar.

Proses khas DALL-E 2 memungkinkannya mengubah deskripsi tekstual menjadi gambar yang rumit dan bermakna, menunjukkan kemajuan luar biasa di persimpangan antara bahasa dan pembuatan gambar. Diagram visual yang disediakan mengilustrasikan konsep-konsep berikut:

Bagian Atas: Proses Pelatihan CLIP

- Bagian atas gambar menggambarkan proses pelatihan CLIP.
  CLIP mengacu pada pra-pelatihan bahasa-gambar kontrastif.
- Tahap ini melibatkan pelatihan model yang mempelajari ruang representasi bersama untuk data tekstual dan gambar.
- Hasilnya adalah ruang representasi bersama tempat teks dan gambar disematkan, yang memungkinkan keduanya untuk dibandingkan dan dihubungkan secara bermakna.
- Ruang representasi bersama ini membentuk fondasi untuk memahami hubungan antara deskripsi tekstual dan gambar yang sesuai.

Bagian Bawah: Proses Pembuatan Teks ke Gambar

- Bagian bawah gambar merepresentasikan proses transformasi deskripsi teks menjadi gambar menggunakan DALL-E 2.
- Input teks, yang mendeskripsikan gambar yang diinginkan, dimasukkan ke dalam DALL-E 2.
- Teks input dikodekan menggunakan enkoder CLIP, menghasilkan representasi vektor berdimensi tinggi yang dikenal sebagai penyisipan teks CLIP.
- Penyisipan ini kemudian diproses melalui Prior, yang merupakan model generatif (baik model autoregresif maupun difusi). Prior menghasilkan penyisipan gambar CLIP, menangkap konten visual yang sesuai dengan deskripsi tekstual.
- Akhirnya, penyisipan gambar CLIP ini didekodekan oleh dekoder difusi untuk menghasilkan gambar akhir yang selaras dengan deskripsi teks yang diberikan.

Menghubungkan secara visual pelatihan CLIP untuk representasi teks-gambar gabungan (bagian atas) dengan proses selanjutnya untuk menghasilkan gambar dari teks menggunakan DALL-E 2 (bagian bawah). Ini menyoroti hubungan antara embedding yang dipelajari dan konversi embedding tersebut menjadi gambar konkret, yang menunjukkan interaksi antara informasi tekstual dan visual dalam konteks operasi DALL-E 2.

8.5 TEKNOLOGI DI BALIK DIFUSI STABIL

Difusi stabil didasarkan pada teknologi canggih yang dikenal sebagai model difusi laten (LDM). Teknologi ini merupakan inti dari pendekatan difusi stabil untuk sintesis teks-ke-gambar. Mari kita telusuri teknologi di balik difusi stabil:

Model Difusi Laten (LDM)

LDM merupakan tulang punggung metodologi difusi stabil. LDM memanfaatkan prinsip-prinsip model difusi dan penerapannya dalam ruang laten autoencoder yang telah dilatih sebelumnya. Teknologi ini melibatkan beberapa komponen dan konsep kunci:

1. *Model Difusi dalam Ruang Laten:*
   - Model difusi, yang secara bertahap mentransformasi data masukan dengan menambahkan derau dan kemudian mencoba merekonstruksi data asli, diadaptasi untuk beroperasi dalam ruang laten.
   - Alih-alih menerapkan difusi secara langsung pada data masukan (seperti gambar), difusi diterapkan dalam ruang laten autoencoder. Hal ini menimbulkan derau pada representasi laten data.
2. *Autoencoder dan Representasi Laten:*
   - Autoencoder adalah jaringan saraf tiruan yang dirancang untuk mengodekan data masukan menjadi representasi laten yang terkompresi dan mendekodekannya kembali ke data asli.
   - Dalam konteks LDM, ruang laten autoencoder yang telah dilatih sebelumnya yang canggih dimanfaatkan. Ruang laten ini menangkap fitur-fitur penting dari data masukan.
3. *Pelatihan dan Optimasi:*
   - LDM dilatih untuk mempelajari transformasi representasi laten dalam proses difusi.
   - Pelatihan ini melibatkan optimasi parameter model untuk memastikan bahwa proses difusi secara efektif menangkap pengenalan derau dan pengurangan derau selanjutnya di ruang laten.
4. *Lapisan Perhatian Silang:*
   - Augmentasi penting dalam arsitektur LDM adalah penggabungan lapisan perhatian silang.
   - Lapisan ini meningkatkan kemampuan model untuk menangani berbagai masukan kondisional, seperti deskripsi teks dan kotak pembatas.
   - Lapisan ini memainkan peran penting dalam memfasilitasi sintesis gambar beresolusi tinggi melalui metode berbasis konvolusi.

Manfaat dan Signifikansi

- *Efisiensi Komputasi:* LDM menawarkan keunggulan dalam melatih model difusi pada sumber daya komputasi terbatas dengan memanfaatkan ruang laten autoencoder yang telah dilatih sebelumnya.
- *Kompleksitas dan Fidelitas:* Dengan melatih difusi dalam ruang laten, LDM mencapai keseimbangan antara penyederhanaan representasi dan pelestarian detail yang rumit, sehingga menghasilkan fidelitas visual yang lebih baik.
- *Sintesis Terkondisi:* Integrasi lapisan perhatian silang memungkinkan LDM menghasilkan gambar yang dikondisikan pada beragam masukan seperti teks, yang berkontribusi pada fleksibilitasnya.

Difusi stabil memanfaatkan potensi model difusi laten untuk menciptakan kerangka kerja inovatif yang menggabungkan kekuatan model difusi, representasi laten, dan sintesis terkondisi. Teknologi ini menggambarkan evolusi berkelanjutan metode sintesis gambar

berbasis AI, menawarkan pendekatan yang efisien dan efektif untuk menciptakan visual yang menarik dari deskripsi tekstual.

8.6 TEKNOLOGI DI BALIK MIDJOURNEY

Midjourney menggunakan teknologi canggih untuk memfasilitasi kemampuan pembangkitan teks-ke-gambar. Mari kita telaah teknologi yang mendasari Midjourney:

Jaringan Adversarial Generatif (GAN)

- GAN terdiri dari dua komponen: generator dan diskriminator.
  Generatif membuat gambar berdasarkan derau acak, sementara diskriminator mencoba membedakan antara gambar asli dan gambar yang dihasilkan oleh generator.
- Proses adversarial ini memaksa generator untuk terus meningkatkan pembangkitan gambarnya guna mengelabui diskriminator.

Sintesis Teks-ke-Gambar dengan GAN

- Midjourney memanfaatkan arsitektur GAN untuk mensintesis gambar dari deskripsi tekstual.
- Generator dikondisikan pada masukan teks, memastikan bahwa gambar yang dihasilkan selaras dengan deskripsi yang diberikan.
- Masukan teks biasanya dikodekan menjadi representasi laten yang memandu proses pembangkitan gambar.

GAN Bersyarat

- Midjourney menggunakan varian GAN yang dikenal sebagai GAN bersyarat (cGAN).
- Dalam cGAN, baik generator maupun diskriminator dikondisikan berdasarkan informasi tambahan (dalam hal ini, deskripsi teks).
- Pengkondisian ini meningkatkan kemampuan generator untuk menghasilkan gambar yang sesuai dengan perintah teks tertentu.

Proses Pelatihan

- Proses pelatihan Midjourney melibatkan pembaruan komponen generator dan diskriminator secara iteratif.
- Generator bertujuan untuk menghasilkan gambar yang tidak dapat dibedakan oleh diskriminator dari gambar asli, sementara diskriminator bertujuan untuk meningkatkan kemampuan diskriminasinya.

Fungsi Kerugian dan Optimasi

- Fungsi kerugian memainkan peran penting dalam memandu proses pelatihan.
- Generator dan diskriminator dioptimalkan menggunakan fungsi kerugian spesifik yang menangkap kualitas gambar yang dihasilkan dan akurasi diskriminasi diskriminator.

Mekanisme Atensi

- Teknologi Midjourney dapat menggabungkan mekanisme atensi untuk meningkatkan fokus generator pada bagian-bagian gambar yang relevan.
- Mekanisme atensi memungkinkan model untuk secara selektif menekankan area tertentu berdasarkan teks masukan, yang berkontribusi pada pembuatan gambar yang lebih relevan secara kontekstual.

Augmentasi dan Prapemrosesan Data

- Midjourney dapat menggunakan teknik augmentasi data untuk memperluas set data pelatihan dan meningkatkan generalisasi.
- Prapemrosesan deskripsi tekstual dapat melibatkan teknik seperti tokenisasi dan penyisipan untuk mengubah teks ke dalam format yang sesuai untuk model.

Manfaat dan Aplikasi

- Teknologi Midjourney memungkinkan pembuatan gambar realistis berdasarkan deskripsi tekstual, sehingga berharga untuk berbagai aplikasi seperti desain, pembuatan konten, dan visualisasi.

Intinya, teknologi Midjourney memanfaatkan kekuatan GAN, terutama GAN kondisional, untuk mengubah masukan tekstual menjadi gambar yang menarik dan relevan secara kontekstual. Pendekatan ini menunjukkan sinergi antara sintesis bahasa dan gambar, membuka jalan bagi aplikasi inovatif di bidang AI generatif.

8.7 PERBANDINGAN ANTARA DALL-E 2, STABLE DIFFUSION, DAN MIDJOURNEY

1. **DALL-E 2**
   - *Data Pelatihan:* Dilatih pada jutaan gambar stok, menghasilkan keluaran canggih yang cocok untuk aplikasi perusahaan.
   - *Kualitas Gambar:* Dikenal karena menghasilkan gambar berkualitas tinggi, terutama unggul saat menghasilkan adegan kompleks dengan lebih dari dua karakter.
   - *Kasus Penggunaan*: Sangat cocok untuk penggunaan tingkat perusahaan karena kualitas keluarannya yang halus.
   - *Gaya Artistik:* Meskipun mampu menghasilkan berbagai gaya, DALL-E 2 menekankan akurasi dan realisme.
   - *Akses:* Ketersediaan dan detail akses tidak ditentukan.
2. **Midjourney:**
   - *Gaya Artistik:* Terkenal karena gaya artistiknya, menghasilkan gambar yang lebih menyerupai lukisan daripada foto.
   - *Operasi:* Memanfaatkan bot Discord untuk mengirim dan menerima panggilan ke server AI, sehingga interaksi terjadi di dalam platform Discord.
   - *Keluaran Gambar:* Terutama menghasilkan visual artistik dan kreatif, selaras dengan penekanannya pada ekspresi artistik.
   - *Kasus Penggunaan:* Ideal untuk kegiatan artistik dan kreatif, tetapi mungkin tidak dioptimalkan untuk gambar seperti foto yang realistis.
   - Akses: Detail penggunaan dan aksesibilitas tidak disebutkan secara eksplisit.
3. **Difusi Stabil:**
   - *Sumber Terbuka:* Dapat diakses oleh khalayak luas sebagai model sumber terbuka.
   - *Pemahaman Artistik:* Menunjukkan pemahaman yang baik tentang ilustrasi artistik kontemporer, menghasilkan karya seni yang rumit dan detail.

- *Pembuatan Gambar:* Sangat unggul dalam menghasilkan ilustrasi yang detail dan kreatif, kurang cocok untuk membuat gambar sederhana seperti logo.
- *Prompt Kompleks:* Memerlukan interpretasi yang jelas atas prompt kompleks untuk hasil yang optimal.
- *Kasus Penggunaan:* Sangat cocok untuk ilustrasi kreatif dan karya seni yang detail.
- *Akses:* Dapat diakses oleh basis pengguna yang luas karena sifatnya yang sumber terbuka.

Ringkasan:

- DALL-E 2 menonjol karena kualitas output kelas perusahaan dan kemampuannya untuk menghasilkan adegan kompleks dengan akurat.
- Midjourney terkenal karena gaya artistik dan kreatifnya, seringkali menghasilkan gambar yang menyerupai lukisan.
- Difusi stabil bersifat serbaguna, menawarkan ilustrasi artistik yang detail dan output kreatif, terutama untuk prompt kompleks.

Pilihan di antara alat-alat ini bergantung pada kasus penggunaan spesifik, gaya gambar yang diinginkan, dan tingkat detail yang dibutuhkan. Setiap alat memiliki keunggulan uniknya sendiri, sehingga cocok untuk berbagai aplikasi kreatif dan praktis.

Aplikasi

Alat AI generator gambar memiliki beragam aplikasi di berbagai industri dan domain. Berikut adalah beberapa aplikasi penting:

1. Pembuatan dan Desain Konten:
   - Alat-alat ini dapat digunakan untuk menghasilkan konten visual untuk situs web, media sosial, iklan, dan kampanye pemasaran.
   - Desainer dapat dengan cepat membuat gambar untuk melengkapi artikel, postingan blog, dan konten tertulis lainnya.
2. Visualisasi Konsep:
   - Arsitek dan desainer dapat menggunakan alat ini untuk mewujudkan konsep dengan menghasilkan gambar berdasarkan deskripsi tekstual bangunan, interior, dan lanskap.
3. Seni dan Hiburan:
   - Seniman dapat menggunakan alat ini untuk mengubah ide imajinatif mereka yang diungkapkan dalam teks menjadi karya seni visual yang nyata.
   - Pengembang gim video dapat membuat adegan, karakter, dan aset berdasarkan deskripsi gim tertulis.
4. Desain Mode dan Produk:
   - Desainer dapat menghasilkan representasi visual pakaian, aksesori, dan produk lainnya sebelum memproduksi prototipe fisik.
5. Penceritaan dan Sastra:
   - Penulis dapat menggunakan alat ini untuk mengilustrasikan adegan dari cerita mereka atau membuat petunjuk visual untuk inspirasi.

- Kreator komik dan novel grafis dapat menerjemahkan naskah menjadi visual.

6. Materi Pendidikan:
    - Guru dan pendidik dapat menggunakan alat ini untuk menghasilkan gambar untuk materi dan presentasi pendidikan.
    - Alat bantu visual dapat meningkatkan pembelajaran dengan memberikan contoh konkret untuk konsep abstrak.
7. E-commerce dan Katalog:
    - Platform e-commerce dapat secara otomatis menghasilkan gambar produk dari deskripsi tekstual, yang membantu dalam pembuatan katalog.
8. Visualisasi Prototipe:
    - Insinyur dan pengembang produk dapat dengan cepat memvisualisasikan prototipe berdasarkan spesifikasi tertulis, yang membantu proses desain.
9. Pencitraan dan Visualisasi Medis:
    - Tenaga medis dapat menghasilkan representasi visual dari kondisi medis, yang membantu dalam edukasi dan komunikasi pasien.
10. Iklan Kreatif:
    - Pengiklan dapat menciptakan visual yang unik dan menarik untuk kampanye berdasarkan ringkasan kreatif tertulis.
11. Desain Interior:
    - Desainer interior dapat memvisualisasikan dan bereksperimen dengan berbagai ide desain berdasarkan deskripsi teks sebelum menerapkannya.
12. Sinematografi dan Papan Cerita:
    - Pembuat film dan animator dapat menggunakan alat ini untuk membuat papan cerita dan memvisualisasikan adegan sebelumnya.
13. Visualisasi Riset:
    - Peneliti dapat memvisualisasikan data kompleks dan temuan riset, sehingga lebih mudah diakses oleh khalayak yang lebih luas.
14. Peramalan Mode:
    - Profesional industri mode dapat menghasilkan gambar tren mode potensial berdasarkan deskripsi dan prediksi teks.
15. Pembuatan Seni Otomatis:
    - Seniman dapat menggunakan alat ini untuk menghasilkan karya seni baru dan unik, mengeksplorasi gaya dan komposisi baru. Aplikasi-aplikasi ini menyoroti fleksibilitas model difusi dan alat AI generator teks-ke-gambar, yang menunjukkan potensinya untuk mengubah deskripsi tekstual menjadi aset visual yang berharga di berbagai bidang.

Kesimpulan

Dunia alat pembuat gambar telah mengalami evolusi yang luar biasa, dengan model difusi dan alat AI generator teks-ke-gambar berada di garis depan inovasi. Model difusi, yang terinspirasi oleh proses fisik, menawarkan pendekatan baru untuk menghasilkan gambar

dengan menambahkan noise dan kemudian merekonstruksi data asli. Model-model ini, baik yang digunakan secara independen maupun dalam ruang laten autoencoder, mencapai keseimbangan yang tepat antara pengurangan kompleksitas dan pelestarian detail. Penggabungan lapisan cross-attention semakin memberdayakan model difusi, memungkinkannya untuk memenuhi beragam masukan kondisional dan menghasilkan keluaran beresolusi tinggi yang relevan secara kontekstual.

Alat AI generator teks-ke-gambar, seperti DALL-E 2, difusi stabil, dan Midjourney, mewujudkan beragam strategi untuk mengubah deskripsi tekstual menjadi representasi visual yang jelas. Setiap alat memiliki keunggulan tersendiri, mulai dari kualitas keluaran kelas perusahaan DALL-E 2 hingga aksesibilitas difusi yang stabil dan penekanan Midjourney pada ekspresi artistik. Alat-alat ini tidak hanya menjembatani kesenjangan antara bahasa dan konten visual, tetapi juga membuka jalan bagi aplikasi baru di berbagai industri. Mulai dari pembuatan dan desain konten hingga arsitektur, hiburan, pendidikan, dan lainnya, aplikasi alat-alat ini sangat luas dan beragam.

Seiring kemajuan bidang ini, model difusi dan alat AI generator teks-ke-gambar siap untuk mendefinisikan ulang kreativitas, desain, dan komunikasi. Kemampuan mereka untuk memanfaatkan kekuatan bahasa dan citra berpotensi mentransformasi industri, meningkatkan pengalaman pengguna, dan menginspirasi bentuk-bentuk ekspresi baru. Dengan teknologi yang terus berkembang dan kasus penggunaan yang semakin luas, masa depan menjanjikan kemungkinan-kemungkinan menarik di persimpangan AI, pembuatan gambar, dan kreativitas manusia.

BAB 9
KASUS PENGGUNAAN CHATGPT

Di era GenAI, ChatGPT menjadi alat yang luar biasa dan serbaguna dengan beragam aplikasi di berbagai domain. Dari mentransformasi lanskap bisnis dan layanan pelanggan hingga merevolusi pembuatan konten, strategi pemasaran, serta tugas bahasa dan komunikasi, kemampuan ChatGPT melampaui batasan tradisional.

ChatGPT memainkan peran penting dalam pengembangan perangkat lunak, layanan kesehatan, riset pasar, penulisan kreatif, pendidikan, kepatuhan hukum, fungsi SDM, dan analisis data, menunjukkan potensinya yang luar biasa dalam membentuk cara kita menghadapi tantangan kompleks dan pengambilan keputusan di berbagai sektor. Eksplorasi ini menggali beragam kasus penggunaan ChatGPT di berbagai domain, menyoroti kemampuan adaptasi dan dampaknya yang luar biasa.

9.1 BISNIS DAN LAYANAN PELANGGAN

1. Dukungan Pelanggan:
   ChatGPT dapat merevolusi dukungan pelanggan dengan menyediakan bantuan instan 24 jam. ChatGPT menangani berbagai pertanyaan pelanggan, mulai dari pertanyaan FAQ sederhana hingga pemecahan masalah yang kompleks. Melalui kemampuan pemahaman dan pembangkitan bahasa alaminya, ChatGPT terlibat dalam percakapan yang manusiawi, memastikan pelanggan menerima respons yang tepat waktu dan akurat.
   Contoh: Seorang pelanggan menghubungi situs web e-commerce dengan pertanyaan tentang spesifikasi suatu produk. ChatGPT memahami pertanyaan tersebut, mengambil informasi yang relevan dari basis pengetahuannya, dan memberikan respons terperinci yang memuaskan pelanggan.
2. Informasi Penjualan dan Produk:
   ChatGPT menjadi asisten penjualan virtual yang menawarkan informasi produk dan layanan kepada pelanggan. ChatGPT membantu dalam pengambilan keputusan dengan memberikan deskripsi, spesifikasi, dan harga yang detail, bahkan menyarankan produk terkait berdasarkan preferensi pelanggan.
   Contoh: Seorang calon pembeli sedang menjelajahi laptop di situs web elektronik. ChatGPT terlibat dalam percakapan, menanyakan kebutuhan dan preferensi pembeli. Kemudian, ChatGPT merekomendasikan laptop yang sesuai dengan kebutuhan pembeli dan memberikan perbandingan fitur-fiturnya.
3. Analisis dan Peningkatan Umpan Balik:
   Bisnis dapat menggunakan ChatGPT untuk menganalisis umpan balik dan sentimen pelanggan. Dengan memproses ulasan, komentar, dan survei, ChatGPT memberikan wawasan tentang persepsi pelanggan, membantu perusahaan mengidentifikasi area yang perlu ditingkatkan dan menyempurnakan produk dan layanan mereka.

Contoh: Sebuah jaringan restoran menggunakan ChatGPT untuk menganalisis ulasan pelanggan. ChatGPT mendeteksi keluhan berulang tentang layanan yang lambat dan presentasi yang buruk. Manajemen restoran mengambil tindakan untuk mengatasi masalah ini, yang pada akhirnya akan meningkatkan kepuasan pelanggan.

4. Rekomendasi yang Dipersonalisasi:
   ChatGPT dapat menawarkan rekomendasi yang dipersonalisasi kepada pelanggan berdasarkan preferensi dan perilaku mereka. Dengan menganalisis interaksi sebelumnya dan riwayat pembelian, ChatGPT menyarankan produk atau layanan yang sesuai dengan minat pelanggan.
   Contoh: Seorang pengguna sedang menjelajahi toko pakaian online. ChatGPT menyarankan pakaian dan aksesori yang sesuai dengan gaya pengguna berdasarkan pembelian sebelumnya dan riwayat penelusuran mereka.
5. Pelacakan Pesanan dan Pembaruan Status:
   Pelanggan sering mencari informasi tentang status pesanan dan detail pelacakan mereka. ChatGPT menangani pertanyaan ini dengan memberikan pembaruan waktu nyata tentang pengiriman, waktu pengiriman, dan keterlambatan.
   Contoh: Seorang pelanggan menanyakan status pesanan online mereka. ChatGPT mengambil informasi pelacakan terbaru dan memberi tahu pelanggan bahwa paket siap dikirim, beserta perkiraan waktu kedatangan.
6. Menangani Pengembalian dan Pengembalian Dana:
   ChatGPT membantu pelanggan dalam memulai pengembalian atau meminta pengembalian dana dengan memandu mereka melalui prosesnya. ChatGPT menjelaskan kebijakan pengembalian, memberikan petunjuk pengemasan barang, dan membantu membuat label pengembalian.
   Contoh: Seorang pelanggan ingin mengembalikan produk cacat yang dibeli secara online. ChatGPT memandu pelanggan melalui proses pengembalian, menjelaskan langkah-langkah yang terlibat, dan membuat label pengembalian untuk mereka.

Dalam ranah bisnis dan layanan pelanggan, ChatGPT meningkatkan keterlibatan pelanggan, menyederhanakan operasi dukungan, dan memberikan pengalaman yang dipersonalisasi. Penting untuk dicatat bahwa meskipun ChatGPT dapat menangani berbagai pertanyaan pelanggan, mungkin ada kasus di mana intervensi manusia diperlukan, terutama untuk masalah yang kompleks atau sensitif. Selain itu, bisnis harus memastikan penggunaan data pelanggan yang etis dan menyediakan komunikasi yang jelas mengenai keterlibatan AI dalam interaksi pelanggan.

8.2 PEMBUATAN KONTEN DAN PEMASARAN

1. Pembuatan Postingan Blog dan Artikel:
   ChatGPT dapat membantu pembuat konten dengan membuat postingan blog dan artikel tentang berbagai topik. ChatGPT mengambil arahan tertentu, meneliti informasi yang relevan, dan menghasilkan konten yang koheren dan informatif. Hal ini sangat

berguna untuk mempertahankan jadwal penerbitan yang konsisten dan meningkatkan skala produksi konten.

Contoh: Sebuah perusahaan perjalanan perlu menerbitkan panduan destinasi secara berkala. ChatGPT menghasilkan panduan terperinci ke lokasi tertentu, termasuk informasi tentang objek wisata, kuliner lokal, dan tips perjalanan.

2. Konten Media Sosial:

   Membuat konten media sosial yang menarik dan sering digunakan dapat memakan waktu. ChatGPT membantu dengan menghasilkan postingan, keterangan, dan bahkan balasan atas komentar pengguna. ChatGPT menyesuaikan konten agar sesuai dengan gaya platform dan suara merek.

   Contoh: Sebuah merek fesyen ingin berbagi inspirasi pakaian sehari-hari di Instagram. ChatGPT membuat keterangan yang menarik secara visual yang menggambarkan pakaian tersebut dan memberikan tips gaya.

3. Konten Ramah SEO:

   ChatGPT dapat menghasilkan konten yang dioptimalkan untuk mesin pencari dengan memasukkan kata kunci dan frasa yang relevan secara alami. Hal ini meningkatkan peluang konten untuk mendapatkan peringkat yang lebih tinggi dalam hasil pencarian dan menarik lalu lintas organik.

   Contoh: Sebuah perusahaan yang berspesialisasi dalam renovasi rumah ingin membuat artikel tentang proyek DIY. ChatGPT memastikan artikel tersebut memuat istilah-istilah yang umum dicari terkait renovasi rumah dan kerajinan.

4. Kampanye Pemasaran Email:

   Menyusun kampanye pemasaran email yang menarik sangat penting untuk keterlibatan pelanggan. ChatGPT membantu menulis konten email yang menarik perhatian penerima, mempromosikan penawaran, dan mendorong konversi.

   Contoh: Sebuah bisnis e-commerce meluncurkan penjualan. ChatGPT membantu membuat kampanye email yang menyoroti barang-barang yang sedang dijual, menekankan diskon, dan menyertakan tombol ajakan bertindak yang persuasif.

5. Deskripsi Produk:

   Saat menambahkan produk baru ke toko online, menulis deskripsi produk yang unik dan menarik dapat memakan waktu. ChatGPT menyederhanakan proses dengan menghasilkan deskripsi produk yang menyoroti fitur dan manfaat.

   Contoh: Sebuah peritel teknologi memperkenalkan model ponsel pintar baru. ChatGPT menghasilkan deskripsi produk yang ringkas namun informatif yang menguraikan spesifikasi ponsel, kemampuan kamera, dan fitur uniknya.

6. Pesan dan Nada Merek:

   Menjaga suara merek yang konsisten di berbagai platform konten sangatlah penting. ChatGPT membantu menciptakan konten yang selaras dengan pesan, nilai, dan nada merek.

   Contoh: Sebuah merek kebugaran ingin menyampaikan pesan yang memotivasi dan memberdayakan. ChatGPT menghasilkan postingan media sosial yang menginspirasi

pengguna untuk mencapai tujuan kebugaran mereka dan menerapkan gaya hidup sehat.

Dalam konteks pembuatan konten dan pemasaran, ChatGPT mempercepat pembuatan konten, menghemat waktu untuk menyusun strategi, dan memastikan aliran konten berkualitas tinggi yang stabil. Namun, penting untuk meninjau dan mengedit konten yang dihasilkan oleh ChatGPT agar selaras dengan gaya dan pesan unik merek tersebut. Selain itu, pengawasan manusia memastikan bahwa konten tersebut secara akurat mewakili visi merek dan beresonansi dengan target audiens.

9.3 PENGEMBANGAN PERANGKAT LUNAK DAN DUKUNGAN TEKNIS

1. Bantuan Kode dan Debugging:
   ChatGPT terbukti menjadi alat yang berharga bagi pengembang yang mencari bantuan pengkodean. ChatGPT dapat memberikan penjelasan konsep pemrograman, membantu dalam men-debug kode, dan bahkan menawarkan solusi untuk masalah pengkodean umum.
   Contoh: Seorang pengembang menemukan kesalahan sintaksis dalam kode mereka. ChatGPT membantu mengidentifikasi masalah dengan menganalisis cuplikan kode dan menyarankan perbaikan.
2. Penjelasan Konsep Teknis:
   Konsep teknis yang kompleks dapat menjadi tantangan untuk dipahami. ChatGPT bertindak sebagai pendamping yang berpengetahuan, menguraikan ide, algoritma, dan teori yang rumit menjadi penjelasan yang mudah dipahami.
   Contoh: Seorang mahasiswa ilmu komputer kesulitan memahami konsep rekursi. ChatGPT memberikan penjelasan langkah demi langkah, yang mengklarifikasi proses dan tujuan rekursi.
3. Pemecahan Masalah Teknis dan Pemecahan Masalah:
   ChatGPT membantu pengguna dalam memecahkan masalah teknis. ChatGPT memandu pengguna melalui serangkaian pertanyaan untuk mendiagnosis masalah, menyarankan solusi potensial, dan memberikan instruksi untuk penyelesaian.
   Contoh: Printer pengguna tidak berfungsi. ChatGPT mengajukan pertanyaan yang relevan tentang status printer, konektivitas, dan pesan kesalahan. Kemudian, ChatGPT memberikan langkah-langkah pemecahan masalah untuk menyelesaikan masalah tersebut.
4. Mempelajari Bahasa Pemrograman Baru:
   Bagi pengembang yang ingin menjelajahi bahasa pemrograman baru, ChatGPT menawarkan panduan. ChatGPT dapat menghasilkan cuplikan kode contoh, menjelaskan sintaksis bahasa, dan menyediakan sumber daya untuk pembelajaran.
   Contoh: Seorang pengembang yang beralih dari Python ke JavaScript mencari bantuan untuk menulis fungsi dalam JavaScript. ChatGPT menyediakan cuplikan kode contoh yang menyelesaikan tugas yang diinginkan.
5. Dokumentasi dan Penggunaan API:

Menjelajahi dokumentasi dan memahami API bisa jadi menakutkan. ChatGPT membantu dengan menjelaskan dokumentasi, menawarkan contoh penggunaan, dan membantu pengembang mengintegrasikan API.

Contoh: Seorang pengembang ingin mengintegrasikan API gateway pembayaran ke situs web e-commerce mereka. ChatGPT memandu mereka melalui dokumentasi API dan menyediakan cuplikan kode untuk integrasi.

6. Praktik Terbaik Perangkat Lunak:

   ChatGPT dapat berbagi wawasan tentang praktik terbaik pengkodean, pola desain, dan prinsip arsitektur perangkat lunak. ChatGPT membantu pengembang menulis kode yang lebih bersih dan efisien.

   Contoh: Seorang pengembang junior mencari saran tentang penulisan kode yang mudah dipelihara. ChatGPT memberikan kiat tentang pemrograman modular, pemberian komentar kode, dan kontrol versi.

Aplikasi ChatGPT dalam pengembangan perangkat lunak dan dukungan teknis menyederhanakan proses pengembangan, meningkatkan pembelajaran, dan menyederhanakan pemecahan masalah. Namun, pengembang harus berhati-hati dan menggunakan penilaian mereka sendiri, terutama dalam skenario kritis, karena solusi ChatGPT mungkin tidak selalu mempertimbangkan pertimbangan khusus konteks.

9.4 ENTRI DAN ANALISIS DATA

Transformasi ChatGPT menjadi penerjemah kode yang sekarang disebut alat "Analisis Data Lanjutan" menandakan evolusi yang signifikan dalam kemampuannya. Dengan peningkatan ini, ChatGPT telah menjadi sumber daya yang andal bagi para profesional dan analis data, yang tidak hanya mampu memahami dan menghasilkan kode tetapi juga menawarkan wawasan tingkat lanjut tentang teknik analisis data, pemodelan statistik, visualisasi data, dan banyak lagi. Fungsionalitas yang diperluas ini memungkinkan pengguna untuk mengekstrak wawasan yang lebih mendalam dari data mereka, memberikan bantuan berharga dalam berbagai tugas berbasis data, dan menjadikannya aset berharga dalam bidang analitik data:

1. Bantuan Entri Data:

   ChatGPT membantu tugas entri data dengan mentranskripsikan data tulisan tangan atau ketikan, memasukkan informasi ke dalam spreadsheet atau basis data, dan mengatur data sesuai format yang ditentukan.

   Contoh: Sebuah tim peneliti perlu mendigitalkan respons survei. ChatGPT mentranskripsikan respons dari formulir kertas ke dalam spreadsheet digital.

2. Pembersihan dan Prapemrosesan Data:

   Sebelum analisis, data seringkali memerlukan pembersihan dan prapemrosesan. ChatGPT membantu mengidentifikasi dan mengoreksi inkonsistensi, nilai yang hilang, dan kesalahan dalam dataset.

Contoh: Seorang analis sedang mempersiapkan dataset untuk analisis. ChatGPT mengidentifikasi dan menyarankan koreksi untuk entri duplikat dan titik data yang hilang.

3. Analisis dan Visualisasi Data Dasar:
   ChatGPT melakukan tugas-tugas analisis data sederhana, seperti menghitung rata-rata, membuat grafik, dan meringkas tren. ChatGPT membantu dalam memahami wawasan dasar dari data.
   Contoh: Sebuah tim pemasaran ingin memvisualisasikan data penjualan. ChatGPT menghasilkan diagram batang dan grafik garis untuk menggambarkan tren penjualan selama periode waktu tertentu.
4. Interpretasi dan Wawasan Data:
   ChatGPT membantu dalam menginterpretasikan temuan data, menawarkan wawasan berdasarkan pola dan tren yang diamati dalam dataset. ChatGPT memberikan penjelasan untuk temuan-temuan penting.
   Contoh: Seorang analis memperhatikan penurunan lalu lintas situs web secara tiba-tiba. ChatGPT menyarankan kemungkinan penjelasan, seperti perubahan algoritma terbaru atau masalah teknis.
5. Analisis Komparatif:
   ChatGPT membantu membandingkan kumpulan data atau variabel berbeda dalam suatu kumpulan data. ChatGPT membantu mengidentifikasi korelasi, perbedaan, dan hubungan antar titik data.
   Contoh: Sebuah bisnis ingin membandingkan peringkat kepuasan pelanggan dari dua lini produk yang berbeda. ChatGPT menghitung skor kepuasan rata-rata untuk setiap lini dan menyoroti perbedaannya.
6. Pelaporan dan Peringkasan Data:
   ChatGPT menghasilkan ringkasan dan laporan berdasarkan analisis data. ChatGPT menyajikan temuan, tren, dan wawasan utama dalam format yang koheren dan mudah dipahami.
   Contoh: Seorang analis perlu meringkas laporan penjualan triwulanan. ChatGPT menghasilkan laporan ringkas yang menyoroti tren pendapatan, produk terlaris, dan kinerja regional.

Aplikasi ChatGPT dalam entri dan analisis data menyederhanakan tugas-tugas terkait data, terutama untuk analisis dan pengorganisasian dasar. Namun, penting untuk dicatat bahwa untuk analisis data yang kompleks, pemodelan statistik, dan interpretasi mendalam, keterlibatan pakar dan analis data tetap krusial untuk mendapatkan wawasan dan pengambilan keputusan yang akurat.

9.5 INFORMASI KESEHATAN DAN MEDIS

1. *Informasi Medis Umum:* ChatGPT dapat memberikan informasi medis umum kepada pengguna yang mencari wawasan tentang gejala, kondisi, perawatan, dan tindakan pencegahan. ChatGPT bertindak sebagai sumber pengetahuan medis pengantar yang

andal. *Contoh:* Seorang pengguna mengalami sakit kepala yang terus-menerus dan mencari informasi tentang kemungkinan penyebabnya. ChatGPT menawarkan penjelasan tentang berbagai faktor yang dapat menyebabkan sakit kepala dan menyarankan konsultasi dengan tenaga medis profesional untuk diagnosis yang akurat.

2. *Pemeriksa Gejala dan Penilaian Mandiri:* ChatGPT membantu pengguna memahami gejala mereka dengan mengajukan pertanyaan yang terarah tentang kondisi mereka. ChatGPT menawarkan wawasan tentang kemungkinan penyebab dan menyarankan apakah perlu mencari pertolongan medis. *Contoh:* Seorang pengguna menjelaskan gejala seperti demam dan nyeri badan. ChatGPT terlibat dalam percakapan untuk memeriksa gejala, menyarankan kemungkinan diagnosis seperti flu, dan menyarankan istirahat serta hidrasi.
3. *Informasi Obat dan Perawatan:* Bagi pengguna yang ingin tahu tentang efek samping obat, petunjuk penggunaan, dan potensi interaksi, ChatGPT menyediakan informasi relevan berdasarkan basis pengetahuan medisnya. *Contoh:* Seorang pengguna diresepkan obat baru dan ingin tahu tentang kemungkinan efek sampingnya. ChatGPT menguraikan efek samping yang umum dan menyarankan pengguna untuk berkonsultasi dengan penyedia layanan kesehatan jika terjadi reaksi yang merugikan.
4. *Tips Kesehatan dan Kebiasaan Sehat:* ChatGPT dapat menawarkan saran kesehatan umum, termasuk tips untuk mempertahankan gaya hidup sehat, mengelola stres, dan menerapkan tindakan pencegahan. *Contoh:* Seorang pengguna bertanya tentang strategi untuk meningkatkan kualitas tidur. ChatGPT memberikan tips seperti menjaga jadwal tidur yang konsisten, menciptakan lingkungan tidur yang nyaman, dan membatasi waktu layar sebelum tidur.
5. *Penjelasan Istilah Medis:* Jargon medis dapat terasa menakutkan bagi individu tanpa latar belakang medis. ChatGPT menyederhanakan terminologi medis, menjelaskan istilah, akronim, dan singkatan. *Contoh*: Seorang pengguna menemukan istilah "hipertensi" dan tidak yakin tentang artinya. ChatGPT menjelaskan bahwa istilah tersebut merujuk pada tekanan darah tinggi dan memberikan gambaran singkat tentang implikasinya.
6. *Mempersiapkan Janji Temu Medis:* ChatGPT membantu pengguna mempersiapkan janji temu medis dengan menyarankan pertanyaan untuk diajukan kepada penyedia layanan kesehatan, menyoroti informasi penting untuk dibagikan, dan menawarkan kiat untuk komunikasi yang efektif. *Contoh*: Seorang pengguna dijadwalkan untuk janji temu dengan dokter terkait kondisi kronis. ChatGPT menyediakan daftar pertanyaan untuk diajukan kepada dokter, memastikan pengguna mengumpulkan semua informasi yang diperlukan.

Peran ChatGPT dalam layanan kesehatan menawarkan informasi dan panduan yang mudah diakses, terutama untuk pemahaman awal dan pertanyaan yang tidak mendesak. Namun, penting untuk ditekankan bahwa ChatGPT tidak boleh menggantikan nasihat medis profesional. Pengguna harus selalu berkonsultasi dengan tenaga kesehatan profesional yang berkualifikasi untuk diagnosis dan rekomendasi perawatan yang akurat.

9.6 RISET DAN ANALISIS PASAR

1. *Analisis dan Ringkasan Survei:* ChatGPT dapat menganalisis respons survei dan merangkum temuan-temuan utama. ChatGPT membantu peneliti dengan mengidentifikasi tren, sentimen, dan pola umum dalam kumpulan data survei yang besar. *Contoh:* Sebuah perusahaan melakukan survei kepuasan pelanggan. ChatGPT meninjau hasil survei, menyoroti area dengan peringkat kepuasan tertinggi, dan mengidentifikasi masalah yang berulang.
2. *Wawasan Umpan Balik Pelanggan:* Bisnis menerima banyak sekali umpan balik pelanggan di berbagai platform. ChatGPT membantu dalam mengekstrak wawasan dari saluran umpan balik ini, mengkategorikan komentar, dan mengidentifikasi tren yang sedang berkembang. *Contoh:* Seorang peritel e-commerce ingin memahami sentimen pelanggan dari ulasan produk. ChatGPT mengkategorikan umpan balik menjadi sentimen positif, negatif, dan netral, memberikan ikhtisar opini pelanggan.
3. *Analisis Pesaing:* ChatGPT membantu bisnis menganalisis pesaing mereka dengan mengumpulkan informasi dari berbagai sumber dan merangkum kekuatan, kelemahan, posisi pasar, dan strategi mereka. *Contoh:* Sebuah perusahaan rintisan teknologi ingin mengevaluasi pesaingnya di pasar ponsel pintar. ChatGPT mengumpulkan informasi tentang fitur, harga, dan ulasan pengguna pesaing, yang menawarkan analisis komprehensif.
4. *Identifikasi dan Peramalan Tren:* ChatGPT dapat menganalisis tren pasar dengan memproses data dari media sosial, artikel berita, dan laporan industri. ChatGPT mengidentifikasi tren dan pola yang muncul yang dapat memandu pengambilan keputusan strategis. *Contoh:* Sebuah merek fesyen ingin memprediksi gaya pakaian populer musim depan. ChatGPT menganalisis percakapan di media sosial dan blog fesyen untuk memperkirakan tren mendatang.
5. *Analisis Perilaku Konsumen:* ChatGPT membantu memahami perilaku konsumen dengan menganalisis pola pembelian, preferensi, dan motivasi pembelian. ChatGPT memberikan wawasan yang menginformasikan kampanye pemasaran dan pengembangan produk. *Contoh:* Seorang peritel daring ingin memahami mengapa produk tertentu populer selama musim tertentu. ChatGPT menganalisis data pembelian dan mengidentifikasi tren dalam perilaku konsumen.
6. *Profil Segmen Pasar:* ChatGPT membantu bisnis membuat profil berbagai segmen pasar berdasarkan faktor demografis, geografis, dan psikografis. ChatGPT membantu dalam menyesuaikan strategi pemasaran dengan segmen audiens tertentu. *Contoh:* Sebuah produsen elektronik ingin menargetkan demografi tertentu untuk peluncuran produk baru. ChatGPT membuat profil calon pelanggan, menguraikan preferensi dan minat mereka.

Aplikasi ChatGPT dalam riset dan analisis pasar menyederhanakan pemrosesan data, menawarkan wawasan yang dapat ditindaklanjuti, dan memungkinkan bisnis untuk membuat keputusan yang tepat. Namun, keahlian manusia tetap penting untuk menafsirkan dan

mengontekstualisasikan hasil, memastikan bahwa strategi bisnis didasarkan pada pemahaman yang menyeluruh tentang dinamika pasar.

9.7 PENULISAN KREATIF DAN BERCERITA

1. *Pembuatan Ide dan Curah Pendapat:* ChatGPT menjadi kolaborator kreatif, membantu penulis dalam menghasilkan ide untuk cerita, artikel, postingan blog, dan proyek kreatif. ChatGPT memicu kreativitas dengan menyarankan alur cerita, karakter, latar, dan tema. *Contoh:* Seorang penulis mengalami kebuntuan saat melakukan curah pendapat untuk ide novel baru. ChatGPT mengusulkan konsep unik yang melibatkan perjalanan waktu dan realitas alternatif, yang menghidupkan kembali proses kreatif penulis.
2. *Pengembangan Plot dan Kerangka Cerita:* ChatGPT membantu penulis menyusun cerita mereka dengan memberikan panduan pengembangan plot. ChatGPT membantu dalam menciptakan alur cerita, membangun ketegangan, dan memetakan urutan peristiwa. *Contoh:* Seorang penulis skenario ingin menguraikan pilot serial TV yang menarik. ChatGPT membantu dalam menyusun plot episode pilot, memperkenalkan karakter, dan menyiapkan alur cerita selanjutnya.
3. *Penciptaan dan Pengembangan Karakter:* Menciptakan karakter yang menarik sangat penting dalam penceritaan. ChatGPT membantu penulis dalam mengembangkan karakter yang utuh dengan menyarankan ciri-ciri kepribadian, latar belakang, motivasi, dan alur karakter. *Contoh:* Seorang penulis fantasi sedang menciptakan protagonis baru. ChatGPT menyarankan latar belakang yang kompleks yang melibatkan peristiwa tragis dan kemampuan magis tersembunyi, yang menambah kedalaman karakter.
4. *Penulisan Dialog:* Dialog yang alami dan menarik merupakan bagian integral dari penceritaan. ChatGPT membantu penulis menciptakan dialog yang autentik dengan menyarankan alur percakapan, nuansa emosional, dan interaksi antar karakter. *Contoh:* Seorang penulis naskah drama sedang menggarap sebuah adegan dramatis. ChatGPT menawarkan dialog yang menyampaikan ketegangan dan konflik antar karakter, yang meningkatkan dampak adegan tersebut.
5. *Pembangunan Dunia dan Deskripsi Latar:* Untuk penceritaan yang imersif, pembangunan dunia yang hidup dan latar yang deskriptif sangat penting. ChatGPT membantu penulis dalam menciptakan latar yang kaya detail dan deskripsi yang menggugah. *Contoh:* Seorang penulis fiksi ilmiah ingin menggambarkan sebuah planet asing. ChatGPT memberikan detail sensorik tentang flora, fauna, dan atmosfer planet yang unik, yang melukiskan gambaran yang hidup.
6. *Prompt Kreatif dan Latihan Menulis:* ChatGPT menawarkan prompt kreatif dan latihan menulis untuk mengatasi hambatan menulis dan merangsang imajinasi. Aplikasi ini menyediakan titik awal untuk cerita pendek, puisi, dan eksperimen kreatif. *Contoh:* Seorang penyair sedang mencari inspirasi untuk puisi baru. ChatGPT menyediakan pertanyaan yang menggugah pikiran tentang keindahan alam, yang menginspirasi penyair untuk menciptakan karya deskriptif.

Aplikasi ChatGPT dalam penulisan kreatif dan penceritaan memberdayakan penulis untuk mengatasi tantangan, mengeksplorasi ide-ide baru, dan menghidupkan narasi mereka. Meskipun membantu dalam proses kreatif, penilaian dan penyuntingan manusia tetap krusial untuk memastikan koherensi naratif, resonansi emosional, dan suara unik penulis.

9.8 PENDIDIKAN DAN PEMBELAJARAN

1. *Bimbingan Virtual dan Penjelasan Konsep:*
   ChatGPT berfungsi sebagai tutor virtual, membantu siswa dalam memahami konsep-konsep kompleks. ChatGPT menjelaskan mata pelajaran akademik, menguraikan teori, dan menawarkan solusi langkah demi langkah untuk masalah.
   Contoh: Seorang siswa SMA kesulitan dengan kalkulus. ChatGPT memberikan penjelasan tentang prinsip-prinsip kalkulus dan membantu memecahkan soal latihan, yang membantu pemahaman siswa.
2. *Bantuan Pekerjaan Rumah dan Tugas:*
   ChatGPT membantu siswa menyelesaikan pekerjaan rumah dan tugas dengan memberikan panduan, menyarankan pendekatan, dan menjawab pertanyaan terkait tugas.
   Contoh: Seorang siswa harus menulis esai tentang peristiwa sejarah. ChatGPT menawarkan saran penelitian, menguraikan poin-poin penting, dan memberikan wawasan untuk menyusun esai secara efektif.
3. *Pembelajaran dan Praktik Bahasa:*
   ChatGPT menjadi pendamping pembelajaran bahasa, melibatkan siswa dalam percakapan, mengoreksi kalimat, dan menyarankan kosakata untuk meningkatkan kemahiran berbahasa.
   Contoh: Seorang siswa ingin berlatih bahasa Spanyol. ChatGPT terlibat dalam percakapan, mengoreksi kesalahan tata bahasa, dan memperkenalkan kosakata baru sesuai konteks.
4. *Pembuatan Sumber Belajar:*
   ChatGPT membantu siswa dengan membuat sumber belajar seperti kartu catatan, ringkasan, dan soal latihan. ChatGPT meringkas materi yang panjang dan membantu siswa meninjau secara efektif.
   Contoh: Seorang siswa mempersiapkan diri untuk ujian sejarah. ChatGPT membuat ringkasan singkat tentang peristiwa sejarah penting, yang membantu siswa meninjau materi di menit-menit terakhir.
5. *Bantuan Riset:* Untuk proyek riset, ChatGPT membantu mahasiswa menemukan sumber yang relevan, merumuskan pertanyaan riset, dan mengorganisasikan informasi untuk menghasilkan makalah yang terstruktur dengan baik.
   Contoh: Seorang mahasiswa sedang melakukan riset tentang perubahan iklim. ChatGPT menyarankan sumber tepercaya, membantu menyempurnakan pertanyaan riset, dan menguraikan struktur makalah riset.

6. *Menjelajahi Topik dan Keingintahuan Baru:* ChatGPT mendorong pembelajaran yang didorong oleh rasa ingin tahu dengan memberikan penjelasan tentang berbagai topik. ChatGPT memuaskan pertanyaan mahasiswa dan merangsang eksplorasi lebih lanjut.
   Contoh: Seorang mahasiswa yang ingin tahu ingin memahami dasar-dasar fisika kuantum. ChatGPT menawarkan penjelasan pengantar, yang mengungkap konsep-konsep kompleks.

Aplikasi ChatGPT dalam pendidikan dan pembelajaran melampaui ruang kelas tradisional, menawarkan bantuan personal, mendorong pembelajaran mandiri, dan membantu mahasiswa dalam perjalanan akademis mereka. Meskipun ChatGPT meningkatkan pengalaman belajar, bimbingan pendidik, struktur kurikulum, dan pengembangan berpikir kritis tetap menjadi komponen penting dari pendidikan yang efektif.

9.9 HUKUM DAN KEPATUHAN

1. *Riset Hukum dan Analisis Yurisprudensi:* ChatGPT membantu para profesional hukum dengan melakukan riset hukum dan meringkas yurisprudensi. ChatGPT mengekstrak informasi relevan dari basis data hukum, membantu pengacara membangun argumen yang lebih kuat dan membuat keputusan yang tepat.
   Contoh: Seorang pengacara sedang mempersiapkan kasus yang melibatkan hak kekayaan intelektual. ChatGPT menyusun contoh-contoh yurisprudensi yang relevan, membantu pengacara memahami preseden.
2. *Penyusunan Dokumen Hukum:* ChatGPT membantu dalam penyusunan dokumen hukum seperti kontrak, perjanjian, dan surat. ChatGPT menghasilkan templat, memberikan panduan tentang bahasa dan struktur, serta memastikan dokumen mematuhi norma hukum.
   Contoh: Seorang wirausahawan membutuhkan perjanjian kerahasiaan. ChatGPT membantu menciptakan perjanjian yang komprehensif, termasuk klausul kerahasiaan dan terminologi hukum.
3. *Definisi dan Penjelasan Hukum*: Terminologi hukum bisa rumit bagi profesional non-hukum. ChatGPT menyederhanakan konsep hukum dengan memberikan definisi, penjelasan, dan konteks untuk berbagai istilah hukum.
   Contoh: Seorang pemilik bisnis menemukan istilah "tort" (perbuatan melawan hukum). ChatGPT menjelaskan konsep hukum perdata, jenis-jenisnya, dan implikasinya terhadap operasional bisnis.
4. *Pedoman dan Regulasi Kepatuhan:* ChatGPT membantu bisnis dalam memahami dan mematuhi peraturan dan standar kepatuhan. ChatGPT menawarkan penjelasan tentang persyaratan peraturan dan menyarankan langkah-langkah untuk mencapai kepatuhan.
   Contoh: Sebuah perusahaan ingin memastikan kepatuhan terhadap peraturan perlindungan data. ChatGPT menguraikan ketentuan-ketentuan utama hukum privasi data yang relevan dan memberikan rekomendasi untuk kepatuhan.

5. *Nasihat Hukum untuk Masalah Umum:* Untuk pertanyaan dan permasalahan hukum sehari-hari, ChatGPT menawarkan nasihat dan panduan hukum awal. ChatGPT menjawab pertanyaan terkait kontrak, hukum ketenagakerjaan, liabilitas, dan lainnya.
   Contoh: Seorang pemilik usaha kecil tidak yakin tentang prosedur pemutusan hubungan kerja karyawan. ChatGPT menjelaskan langkah-langkah hukum yang terlibat dalam pemutusan hubungan kerja karyawan secara patuh.
6. *Panduan Kekayaan Intelektual*: ChatGPT membantu dalam menavigasi masalah kekayaan intelektual dengan memberikan wawasan tentang hak cipta, merek dagang, dan paten. ChatGPT menjelaskan proses pendaftaran dan perlindungan kekayaan intelektual.
   Contoh: Seorang seniman ingin melindungi karya seninya dari penggunaan yang tidak sah. ChatGPT menjelaskan dasar-dasar hukum hak cipta, termasuk cara mendaftarkan karyanya.

Aplikasi ChatGPT di bidang hukum dan kepatuhan menyederhanakan riset hukum, menyederhanakan proses dokumentasi, dan menawarkan panduan awal. Namun, penting untuk dicatat bahwa tanggapan ChatGPT tidak boleh menggantikan nasihat hukum profesional. Profesional hukum harus dikonsultasikan untuk masalah hukum yang kompleks dan keputusan penting.

9.10 SDM DAN REKRUTMEN

1. *Penyaringan Kandidat dan Wawancara Awal:*
   ChatGPT membantu profesional SDM dalam melakukan penyaringan kandidat awal. ChatGPT berinteraksi dengan pelamar, mengajukan pertanyaan yang relevan, dan mengevaluasi tanggapan terhadap kandidat terpilih untuk evaluasi lebih lanjut.
   Contoh: Seorang manajer SDM perlu menyaring lamaran pekerjaan dalam jumlah besar. ChatGPT melakukan wawancara singkat dengan pelamar, menanyakan tentang kualifikasi dan pengalaman mereka.
2. *Penyusunan Deskripsi Pekerjaan:*
   Menyusun deskripsi pekerjaan yang menarik sangat penting untuk menarik kandidat yang sesuai. ChatGPT membantu dalam membuat lowongan pekerjaan yang detail dan menarik yang menyoroti tanggung jawab, kualifikasi, dan budaya perusahaan.
   Contoh: Sebuah perusahaan sedang merekrut seorang manajer media sosial. ChatGPT menghasilkan deskripsi pekerjaan yang secara efektif mengomunikasikan ekspektasi peran dan merek perusahaan.
3. *Dukungan Orientasi Karyawan*:
   ChatGPT membantu dalam orientasi karyawan dengan memberikan informasi tentang kebijakan perusahaan, tunjangan, dan proses orientasi. ChatGPT menjawab pertanyaan karyawan baru dan memastikan transisi yang lancar.
   Contoh: Seorang karyawan baru ingin tahu lebih banyak tentang kebijakan cuti perusahaan. ChatGPT memberikan gambaran umum tentang kebijakan tersebut dan cara meminta cuti.

4. *Bantuan Pelatihan dan Pengembangan:*
   Para profesional SDM dapat menggunakan ChatGPT untuk menawarkan sumber daya pelatihan dan peluang pengembangan. ChatGPT merekomendasikan kursus daring, lokakarya, dan kegiatan pengembangan keterampilan berdasarkan tujuan karier karyawan.
   Contoh: Seorang karyawan menyatakan minatnya untuk meningkatkan keterampilan manajemen proyeknya. ChatGPT menyarankan kursus dan sumber daya yang relevan untuk pengembangan profesional.
5. *Bantuan Karyawan dan Klarifikasi Kebijakan:*
   ChatGPT membantu karyawan memahami kebijakan perusahaan, tunjangan, dan prosedur SDM. ChatGPT menyediakan informasi tentang kebijakan cuti, prosedur pengaduan, dan lainnya.
   Contoh: Seorang karyawan ingin mengetahui prosedur pelaporan pelecehan di tempat kerja. ChatGPT menjelaskan langkah-langkah yang harus diikuti dan menekankan pentingnya pelaporan.
6. *Persiapan dan Tips Wawancara:*
   Bagi para pencari kerja, ChatGPT menawarkan panduan persiapan wawancara. ChatGPT menyarankan pertanyaan wawancara umum, memberikan tips untuk respons yang efektif, dan memberikan wawasan tentang etiket wawancara.
   Contoh: Seorang pelamar kerja merasa gugup menghadapi wawancara yang akan datang. ChatGPT memberikan saran tentang cara mempersiapkan diri, menjawab pertanyaan dengan percaya diri, dan memberikan kesan positif.

Aplikasi ChatGPT di bidang SDM dan rekrutmen mengoptimalkan proses perekrutan, meningkatkan pengalaman kandidat, dan menyederhanakan komunikasi antara profesional SDM dan karyawan. Meskipun ChatGPT dapat mendukung berbagai tugas, penting untuk dicatat bahwa keterlibatan manusia tetap penting untuk pengambilan keputusan yang bijaksana, mengevaluasi keterampilan lunak, dan menangani masalah SDM yang kompleks.

9.11 ASISTEN PRIBADI DAN PRODUKTIVITAS

1. *Manajemen Tugas dan Pengingat*:
   ChatGPT bertindak sebagai pengelola tugas virtual, membantu pengguna mengatur daftar tugas, mengatur pengingat untuk janji temu, dan mengelola tenggat waktu untuk tugas dan proyek.
   Contoh: Seorang pengguna menjadwalkan rapat dan meminta ChatGPT untuk mengingatkan mereka 15 menit sebelum rapat dimulai.
2. *Koordinasi Kalender*:
   ChatGPT membantu dalam penjadwalan dan koordinasi acara. ChatGPT memeriksa ketersediaan, mengusulkan waktu rapat yang sesuai, dan membantu pengguna menjadwalkan janji temu.

Contoh: Seorang profesional ingin mengadakan rapat virtual dengan rekan kerja yang berbeda zona waktu. ChatGPT menyarankan waktu rapat optimal yang mengakomodasi jadwal setiap orang.

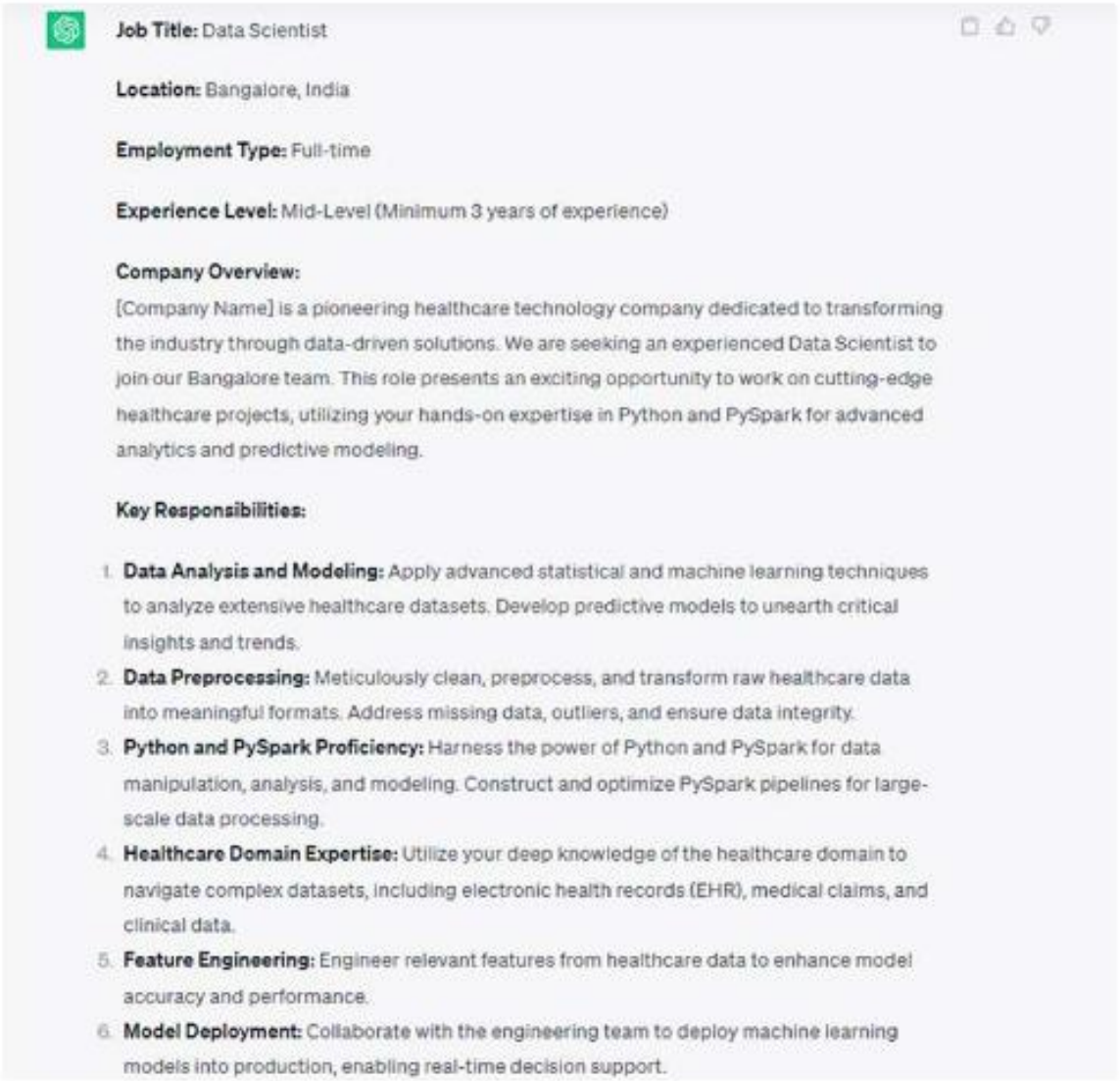
3. *Pencarian Informasi*:
   ChatGPT dengan cepat mengambil informasi dari web atau basis data, menghemat waktu pengguna dalam mencari fakta, angka, definisi, atau data historis.
   Contoh: Seorang mahasiswa membutuhkan informasi untuk makalah penelitian. ChatGPT mengambil artikel, statistik, dan sumber yang relevan dengan topik yang dipilih.
4. *Pencatatan dan Peringkasan*:
   ChatGPT membantu dalam mencatat selama rapat, kelas, atau konferensi. ChatGPT juga dapat meringkas dokumen yang panjang, menyaring poin-poin penting untuk referensi yang mudah.
   Contoh: Seorang pengguna menghadiri konferensi dan meminta ChatGPT untuk mencatat. ChatGPT membuat ringkasan singkat dari sesi konferensi.
5. *Penerjemahan Bahasa di Mana Saja*:
   ChatGPT berfungsi sebagai penerjemah bahasa, membantu pengguna dalam menerjemahkan percakapan, teks, atau konten secara real-time dari satu bahasa ke bahasa lain.
   Contoh: Seorang pelancong membutuhkan bantuan untuk menerjemahkan rambu jalan dan menu saat menjelajahi negara asing. ChatGPT menyediakan terjemahan instan.
6. *Rekomendasi yang Dipersonalisasi*:
   ChatGPT menyarankan buku, film, musik, restoran, dan lainnya berdasarkan preferensi pengguna. Layanan ini dapat membantu pengguna menemukan konten baru atau membuat keputusan.
   Contoh: Seorang pengguna meminta rekomendasi buku bergenre misteri kepada ChatGPT. ChatGPT menyediakan daftar novel misteri berperingkat tinggi.
7. *Bantuan Kebugaran dan Kesehatan*:
   ChatGPT menawarkan rutinitas latihan, saran nutrisi, dan kiat kesehatan. Layanan ini membantu pengguna menetapkan target kebugaran dan menyarankan latihan berdasarkan preferensi mereka.
   Contoh: Seorang pengguna ingin memulai rutinitas latihan di rumah. ChatGPT merancang rencana latihan yang dipersonalisasi dengan berbagai latihan dan tingkat intensitas.

Aplikasi ChatGPT sebagai asisten pribadi dan alat produktivitas menyederhanakan tugas sehari-hari, meningkatkan pengaturan, dan menyediakan akses informasi yang mudah. Namun, meskipun ChatGPT dapat menangani berbagai tugas, penilaian dan pengambilan keputusan manusia sangat penting, terutama dalam skenario yang membutuhkan penalaran kompleks atau evaluasi subjektif.

Contoh

Sampai saat ini kita telah melihat kasus penggunaan ChatGPT berdasarkan domain. Sekarang mari kita ambil beberapa contoh kasus penggunaan dan tanyakan kepada ChatGPT:

1. **Domain—SDM dan Rekrutmen**:
   Untuk lokasi di Bangalore, mari kita minta dibuatkan deskripsi pekerjaan untuk seorang ilmuwan data yang memiliki pengalaman minimal tiga tahun dan harus memiliki pengalaman langsung menggunakan Python dan PySpark di bidang kesehatan. Gambar 9.1 menunjukkan respons ChatGPT.



Job Title: Data Scientist

Location: Bangalore, India

Employment Type: Full-time

Experience Level: Mid-Level (Minimum 3 years of experience)

Company Overview:
[Company Name] is a pioneering healthcare technology company dedicated to transforming the industry through data-driven solutions. We are seeking an experienced Data Scientist to join our Bangalore team. This role presents an exciting opportunity to work on cutting-edge healthcare projects, utilizing your hands-on expertise in Python and PySpark for advanced analytics and predictive modeling.

Key Responsibilities:

1. **Data Analysis and Modeling:** Apply advanced statistical and machine learning techniques to analyze extensive healthcare datasets. Develop predictive models to unearth critical insights and trends.
2. **Data Preprocessing:** Meticulously clean, preprocess, and transform raw healthcare data into meaningful formats. Address missing data, outliers, and ensure data integrity.
3. **Python and PySpark Proficiency:** Harness the power of Python and PySpark for data manipulation, analysis, and modeling. Construct and optimize PySpark pipelines for large-scale data processing.
4. **Healthcare Domain Expertise:** Utilize your deep knowledge of the healthcare domain to navigate complex datasets, including electronic health records (EHR), medical claims, and clinical data.
5. **Feature Engineering:** Engineer relevant features from healthcare data to enhance model accuracy and performance.
6. **Model Deployment:** Collaborate with the engineering team to deploy machine learning models into production, enabling real-time decision support.

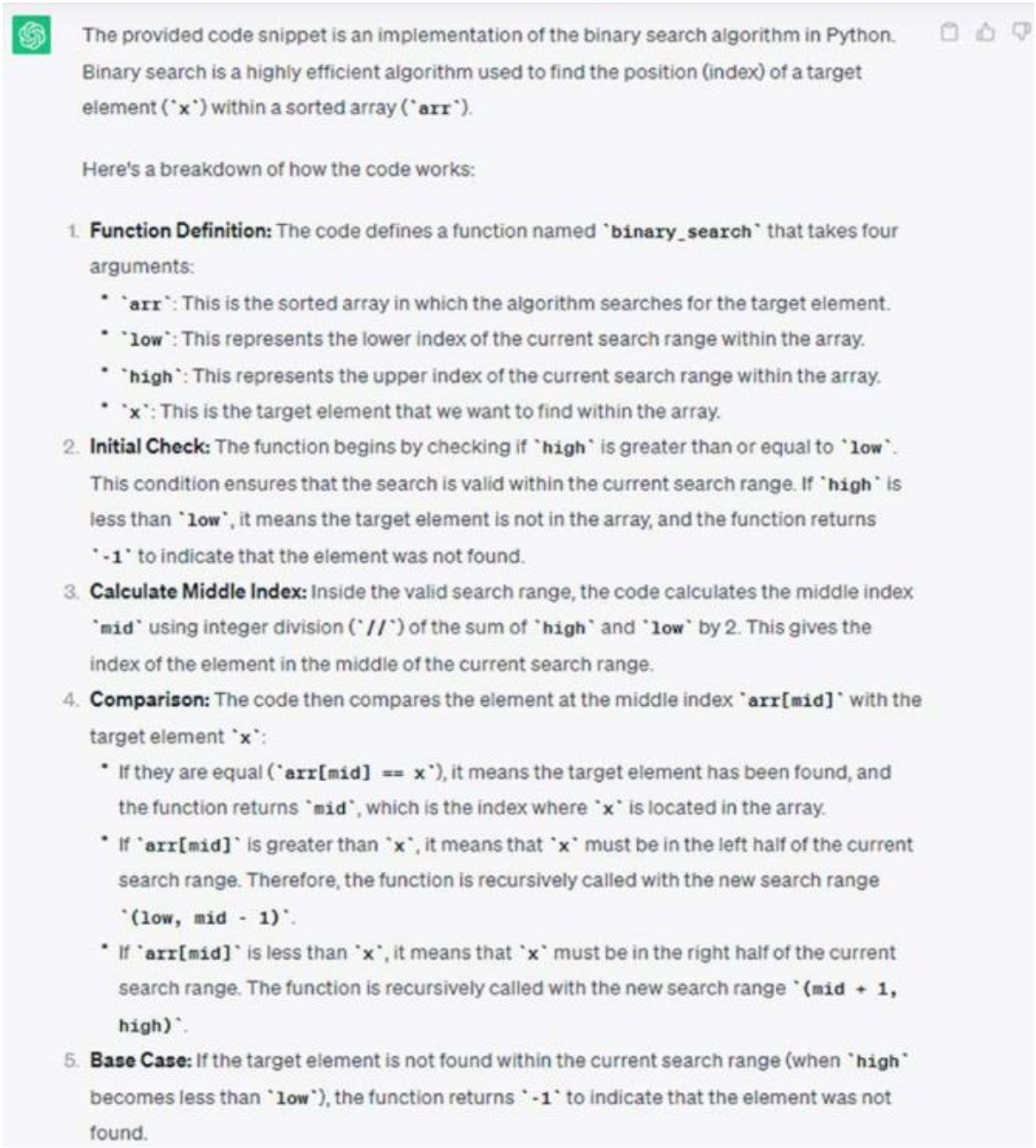
Gambar 9.1. Respons ChatGPT untuk contoh 1

2. **Domain—Pengembangan Perangkat Lunak dan Dukungan Teknis:** Jelaskan cuplikan kode berikut:

```
def binary_search(arr, low, high, x):
  if high >= low:
```

```
        mid = (high + low) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binary_search(arr, low, mid - 1, x)
        else:
            return binary_search(arr, mid + 1, high, x)
    else:
        return -1
```

Gambar 9.2 menunjukkan respons ChatGPT.



The provided code snippet is an implementation of the binary search algorithm in Python. Binary search is a highly efficient algorithm used to find the position (index) of a target element (`x`) within a sorted array (`arr`).

Here's a breakdown of how the code works:

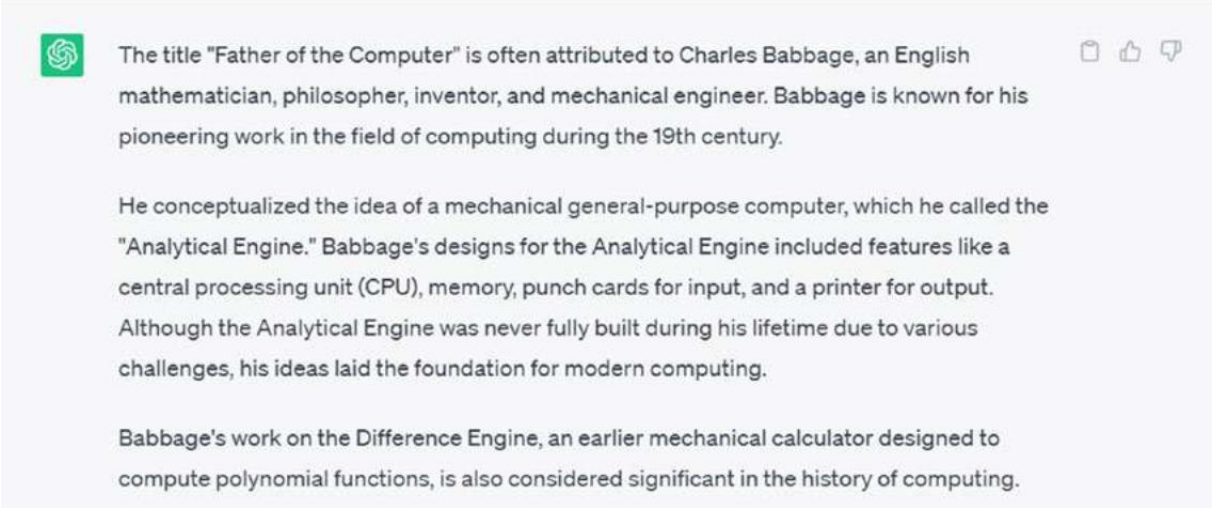
1. **Function Definition:** The code defines a function named `binary_search` that takes four arguments:
   - `arr`: This is the sorted array in which the algorithm searches for the target element.
   - `low`: This represents the lower index of the current search range within the array.
   - `high`: This represents the upper index of the current search range within the array.
   - `x`: This is the target element that we want to find within the array.
2. **Initial Check:** The function begins by checking if `high` is greater than or equal to `low`. This condition ensures that the search is valid within the current search range. If `high` is less than `low`, it means the target element is not in the array, and the function returns `-1` to indicate that the element was not found.
3. **Calculate Middle Index:** Inside the valid search range, the code calculates the middle index `mid` using integer division (`//`) of the sum of `high` and `low` by 2. This gives the index of the element in the middle of the current search range.
4. **Comparison:** The code then compares the element at the middle index `arr[mid]` with the target element `x`:
   - If they are equal (`arr[mid] == x`), it means the target element has been found, and the function returns `mid`, which is the index where `x` is located in the array.
   - If `arr[mid]` is greater than `x`, it means that `x` must be in the left half of the current search range. Therefore, the function is recursively called with the new search range `(low, mid - 1)`.
   - If `arr[mid]` is less than `x`, it means that `x` must be in the right half of the current search range. The function is recursively called with the new search range `(mid + 1, high)`.
5. **Base Case:** If the target element is not found within the current search range (when `high` becomes less than `low`), the function returns `-1` to indicate that the element was not found.

Gambar 9-2. Respons ChatGPT untuk contoh 2

3. **Domain—Pendidikan dan Pembelajaran:** Terjemahkan "apa kabar" ke dalam bahasa Hindi. Gambar 9-3 menunjukkan respons ChatGPT.

Gambar 9.3. Respons ChatGPT untuk contoh 3

4. **Tanya Jawab:**
   Siapakah bapak komputer? Gambar 9-4 menunjukkan keluaran ChatGPT.



Gambar 9.4. Respons ChatGPT untuk contoh 4

Kesimpulan

Aplikasi ChatGPT yang serbaguna di berbagai domain menunjukkan potensi transformatifnya. Baik itu meningkatkan interaksi layanan pelanggan, menyederhanakan pembuatan konten dan upaya pemasaran, memfasilitasi tugas bahasa dan komunikasi, memberdayakan pengembangan perangkat lunak dan dukungan teknis, merevolusi manajemen informasi medis dan layanan kesehatan, atau mendorong riset dan analisis pasar, ChatGPT secara konsisten membuktikan kemampuan adaptasi dan kegunaannya.

Selain itu, kemahirannya dalam penulisan kreatif, pendidikan, kepatuhan hukum, fungsi SDM, dan analisis data semakin menggarisbawahi nilainya di berbagai sektor. Dengan kemampuannya untuk memahami, menghasilkan, dan membantu dalam pengambilan keputusan, ChatGPT muncul sebagai alat luar biasa yang terus mendefinisikan ulang bagaimana kita memanfaatkan kekuatan AI untuk solusi dunia nyata dalam lanskap dinamis saat ini.

DAFTAR PUSTAKA

Adi Setiawan&Ulfah Khairiyah Luthfiyani. (2023). "Penggunaan ChatGPT Untuk Pendidikan di Era Education 4.0: Usulan Inovasi Meningkatkan Keterampilan Menulis", Jurnal PETISI, Vol. 04, No. 01, Januari 2023

Ahmed, T., & Choudhury, S. (2024). LM4OPT: Unveiling the potential of Large Language Models in formulating mathematical optimization problems. *INFOR: Information Systems and Operational Research*, *62*(4), 559–572.

AI Task Force. (2024). GUIDELINES For Use of Generative AI in Teaching and Learning. Trinity Western University.

Aiman Faiz&Imas Kurniawaty. (2023). "Tantangan Penggunaan ChatGPT dalam Pendidikan Ditinjau dari Sudut Pandang Moral". (Jurnal Ilmu Pendidikan). Volume 5 Nomor 1 Bulan Februari Tahun 2023.

Annisa Azzahra, F., & Toriqo Abimanyu, F. (2023). Perubahan Sosial Akibat Kemunculan Teknologi Chat GPT di Kalangan Mahasiswa. Jurnal Ilmiah Multidisipline,

Arora A. (2023). The promise of large language models in health care. Lancet. 2023;401:641. doi: 10.1016/S0140-6736(23)00216-7.

Boone, T., Fahimnia, B., Ganeshan, R., Herold, D. M., & Sanders, N. R. (2025). Generative AI: Opportunities, challenges, and research directions for supply chain resilience. *Transportation Research Part E: Logistics and Transportation Review*, *199*, 104135.

Brown, S., et al. (2024). Ethical Implications of Generative AI: Privacy, Security, and Fairness. Journal of AI Ethics, 1(1), 45-61.

Brown, T., et al. (2023). Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems.

Carroll, A.J. (2024). Integrating Large Language Models and Generative Artificial Intelligence in Education. ScienceDirect.

Cascella M., Montomoli J., Bellini V., Bignami E. Evaluating the feasibility of ChatGPT in healthcare: An analysis of multiple clinical and research scenarios. J. Med. Syst. 2023;47:33. doi: 10.1007/s10916-023-01925-4.

Chan, H.-L., & Choi, T.-M. (2025). Using generative artificial intelligence (GenAI) in marketing: Development and practices. *Journal of Business Research*, *191*, 115276.

Chen, H., Constante-Flores, G. E., & Li, C. (2024). Diagnosing infeasible optimization problems using large language models. *INFOR: Information Systems and Operational Research*, *62*(4), 573–587.

Chen, Y.-C., Loh, C.-H., Wang, F.-C., Chen, Z.-J., Fu, S.-H., & Wang, C.-Y. (2022). Application of generative adversarial network to optimize vehicle allocation at dispatch stations of paratransit services. *Electronics*, *11*(3), 423.

Dhariwal, P., & Nichol, A. (2024). Diffusion Models Beat GANs on Image Synthesis. NeurIPS.

Faiz, Aiman dan Imas Kurniawaty. (2023). Tantangan Penggunaan ChatGPT dalam Pendidikan Ditinjau dari Sudut Pandang Moral. Jurnal Edukatif Ilmu Pendidikan.

Farwati, M., Salsabila, I. T., Navira, K. R., & Sutabri, T. (2023). Analisa Pengaruh Teknologi Artificial Intelligence (AI) Dalam Kehidupan Sehari-Hari.

Gao, R., Du, L., Suganthan, P. N., Zhou, Q., & Yuen, K. F. (2022a). Random vector functional link neural network based ensemble deep learning for short-term load forecasting. *Expert Systems with Applications*, *206*, 117784.

Ge Y., Hua W., Ji J., Tan J., Xu S., Zhang Y. 2023. OpenAGI: When LLM meets domain experts. arXiv. 20232304.04370.

Harrer S. 2023. Attention is not all you need: The complicated case of ethically using large language models in healthcare and medicine. eBioMedicine. 2023;90:104512. doi: 10.1016/j.ebiom.2023.104512.

Karras, T., Laine, S., & Aila, T. (2024). A Style-Based Generator Architecture for Generative Adversarial Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence.

Kim, J. H., Kim, S., Lee, Y., Kim, W. C., & Fabozzi, F. J. (2025). Enhancing mean–variance portfolio optimization through GANs-based anomaly detection. *Annals of Operations Research*, *346*(1), 217–244.

Melgarejo, M., Medina, M., Lopez, J., & Rodriguez, A. (2025). Optimization test function synthesis with generative adversarial networks and adaptive neuro-fuzzy systems. *Information Sciences*, *686*, 121371.

Mostajabdaveh, M., Yu, T. T., Ramamonjison, R., Carenini, G., Zhou, Z., & Zhang, Y. (2024). Optimization modeling and verification from problem specifications using a multi-agent multi-stage LLM framework. *INFOR: Information Systems and Operational Research*, *62*(4), 599–617.

Murcahyanto, Hary. (2023). Penerapan Media Chat GPT pada Pembelajaran Manajemen Pendidikan terhadap Kemandirian Mahasiswa. Jurnal Edumatik Pendidikan Informatika.

Mutaqin, dkk. (2023). Implementasi Artificial Intelligence (AI) dalam Kehidupan, Cetakan Pertama, Yayasan Kita Menulis, Medan.

Nguyen, T., et al. (2024). Unlocking the Potential of Generative AI in Large Language Models. IEEE Transactions on NLP, 2024

OpenAI. (2023). ChatGPT: Model bahasa besar untuk generative AI. OpenAI. https://openai.com

Rao A., Pang M., Kim J., Kamineni M., Lie W., Prasad A.K., Landman A., Dreyer K.J., Succi M.D. Assessing the utility of ChatGPT throughout the entire clinical workflow. medRxiv. 2023 doi: 10.1101/2023.02.21.23285886.

Sallam M. ChatGPT utility in healthcare education, research, and practice: Systematic review on the promising perspectives and valid concerns. Healthcare. 2023;11:887. doi: 10.3390/healthcare11060887.

Santoso, Joseph Teguh. (2021). Kecerdasan Buatan & Jaringan Syaraf Buatan, Yayasan Prima Agus Teknik, Semarang.

Shehata, R., et al. (2023). Leveraging Generative AI and Large Language Models. NIH Journal of Digital Health, 2023.

Sondakh, D.E. (2024). Research Project Topic Recommender System Using Generative Language Model. Cogito Journal.

Srivastava, S. K., Routray, S., Bag, S., Gupta, S., & Zhang, J. Z. (2024). Exploring the potential of large language models in supply chain management: A study using big data. *Journal of Global Information Management*, *32*(1), 1–29.

Suwahyu, I., Taurid, S., Madi, O., Taufiq, A. A., Faiz, A., & Rasyid, N. (2024). Analisis Pengaruh Kepercayaan dan Pemanfaatan Teknologi Terhadap Penggunaan Chat-GPT. INTEC Journal: Information Technology Education Journal, 3(2).

Zhang, Y., et al. (2024). Advancements in Transformer Architectures for Generative AI. Journal of Machine Learning Research

Zhao Z., Wallace E., Feng S., Klein D., Singh S. Calibrate before use: Improving few-shot performance of language models; Proceedings of the 38th International Conference on Machine Learning; Virtual. 18–24 July 2021; pp. 12697–12706.

Zhou, Q. (2025). Benchmarking Operations and Supply Chain Management Using Generative AI. Journal of Modelling in Operations, 2025.

APLIKASI AI GENERATIF :

Cara praktis tentang Model Difusi, ChatGPT, dan LLM (Large Language Models)

Dr. Ir. Agus Wibowo, M.Kom, M.Si, MM

BIO DATA PENULIS

Penulis memiliki berbagai disiplin ilmu yang diperoleh dari Universitas Diponegoro (UNDIP) Semarang. dan dari Universitas Kristen Satya Wacana (UKSW) Salatiga. Disiplin ilmu itu antara lain teknik elektro, komputer, manajemen dan ilmu sosiologi. Penulis memiliki pengalaman kerja pada industri elektronik dan sertifikasi keahlian dalam bidang Jaringan Internet, Telekomunikasi, Artificial Intelligence, Internet Of Things (IoT), Augmented Reality (AR), Technopreneurship, Internet Marketing dan bidang pengolahan dan analisa data (komputer statistik).

Penulis adalah pendiri dari Universitas Sains dan Teknologi Komputer (Universitas STEKOM) dan juga seorang dosen yang memiliki Jabatan Fungsional Akademik Lektor Kepala (Associate Professor) yang telah menghasilkan puluhan Buku Ajar ber ISBN, HAKI dari beberapa karya cipta dan Hak Paten pada produk IPTEK. Sejak tahun 2023 penulis tercatat sebagai Dosen luar biasa di Fakultas Ekonomi & Bisnis (FEB) Universitas Diponegoro Semarang. Penulis juga terlibat dalam berbagai organisasi profesi dan industri yang terkait dengan dunia usaha dan industri, khususnya dalam pengembangan sumber daya manusia yang unggul untuk memenuhi kebutuhan dunia kerja secara nyata.

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-64-5 (PDF)
9 786347 227645