

# LITERASI ARTIFICIAL INTELLIGENCE UNTUK ASN

Diadopsi dari silabus elevAlte Microsoft Indonesia  
Kerja Sama Lembaga Administrasi Negara dan Microsoft Indonesia



## Daftar Isi

<b>Materi 1: Ringkasan Artificial Intelligence .....</b>	<b>6</b>
<i>Pengantar AI.....</i>	6
Apa itu AI? .....	6
Memahami Pembelajaran Mesin.....	7
Cara Kerja Pembelajaran Mesin .....	8
Memahami Visi Komputer .....	9
Model dan Kemampuan Computer Vision .....	9
Memahami Pemrosesan Bahasa Alami.....	14
Memahami Kecerdasan Dokumen dan Penambangan Pengetahuan.....	15
Penambangan Pengetahuan.....	15
Memahami AI Generatif.....	15
Tantangan dan Risiko Dengan AI .....	15
Memahami AI yang Bertanggung Jawab .....	18
Keadilan .....	18
Keandalan dan Keamanan.....	18
Privasi dan Keamanan .....	19
Inklusivitas .....	19
Transparansi.....	19
Akuntabilitas .....	19
Sumber daya selengkapnya.....	20
Ringkasan.....	21
<b>Materi 2: AI Generatif .....</b>	<b>23</b>
<i>Pendahuluan.....</i>	23

<i>Apa Itu AI Generatif?</i> .....	24
Pembuatan Bahasa Alami.....	24
Pembuatan Gambar.....	25
Pembuatan Kode.....	26
<i>Apa Itu Model Bahasa?</i> .....	26
Model Transformator .....	27
Tokenisasi.....	29
Penyematan .....	30
Perhatian.....	32
<i>Menggunakan Model Bahasa</i> .....	36
Model Bahasa Besar dan Kecil.....	38
<i>Ringkasan</i> .....	40
<b><i>Materi 3: AI Generatif yang Bertanggung Jawab</i></b> .....	<b>42</b>
Pendahuluan.....	42
Merencanakan Solusi AI Generatif Yang Bertanggung Jawab..	43
Mengidentifikasi Potensi Bahaya .....	43
Mengukur Potensi Bahaya .....	47
Pengujian Manual dan Otomatis .....	48
Mengurangi Potensi Bahaya .....	49
1: Lapisan model .....	50
2: Lapisan sistem keamanan.....	51
3: Lapisan metaprompt dan grounding .....	51
4: Lapisan pengalaman pengguna .....	52
Mengoperasikan Solusi AI Generatif Yang Bertanggung Jawab .....	52
Merilis dan Mengoperasikan Solusi .....	53
<i>Ringkasan</i> .....	55

<b>Materi 4: Menerapkan Rekayasa Prompt.....</b>	<b>57</b>
Pendahuluan.....	57
Memahami Rekayasa Prompt .....	57
Apa Itu Rekayasa Perintah.....	57
Pertimbangan Untuk Titik Akhir API .....	59
Menyesuaikan Parameter Model .....	59
Menulis Perintah Yang Lebih Efektif .....	60
Berikan Instruksi Yang Jelas.....	60
Format Instruksi .....	62
Menggunakan Penanda Bagian .....	62
Konten Primer, Pendukung, dan Grounding.....	63
Isyarat.....	65
Memberikan Konteks Untuk Meningkatkan Akurasi.....	66
Meminta Komposisi Output.....	66
Pesan sistem.....	68
Riwayat Percakapan .....	70
Beberapa Pembelajaran Bidikan .....	71
Memecah tugas yang kompleks.....	72
Rantai Pemikiran.....	73
Ringkasan.....	76

# MATERI 1

# RINGKASAN ARTIFICIAL INTELLIGENCE (AI)

Diadopsi dari Materi Dasar-Dasar Microsoft Azure AI:  
Ringkasan AI (Microsoft Learn)



# Materi 1: Ringkasan Artificial Intelligence

## Pengantar AI

AI memungkinkan kita membangun perangkat lunak luar biasa yang dapat meningkatkan perawatan kesehatan, memungkinkan orang mengatasi kekurangan fisik, memberdayakan infrastruktur cerdas, menciptakan pengalaman hiburan yang luar biasa, dan bahkan menyelamatkan planet ini!

Video pengayaan dan penjelasan lebih lanjut terkait materi ini dapat diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/paths/get-started-with-artificial-intelligence-on-azure/>



## Apa itu AI?

Sederhananya, AI adalah perangkat lunak yang meniru perilaku dan kemampuan manusia. Beban kerja utama meliputi:

Pembelajaran mesin – Ini sering menjadi fondasi untuk sistem AI, dan merupakan cara kita "mengajari" model komputer untuk membuat prediksi dan menarik kesimpulan dari data.

Visi komputer – Kemampuan dalam AI untuk menafsirkan dunia secara visual melalui kamera, video, dan gambar.

Pemrosesan bahasa alami – Kemampuan dalam AI untuk komputer menginterpretasikan bahasa tertulis atau lisan, dan merespons dalam jenis.

Kecerdasan dokumen - Kemampuan dalam AI yang menangani pengelolaan, pemrosesan, dan penggunaan data dalam volume tinggi yang ditemukan dalam formulir dan dokumen.

Penambangan pengetahuan - Kemampuan dalam AI untuk mengekstrak informasi dari volume besar data yang sering tidak terstruktur untuk membuat penyimpanan pengetahuan yang dapat dicari.

AI Generatif - Kemampuan dalam AI yang membuat konten asli dalam berbagai format termasuk bahasa alami, gambar, kode, dan banyak lagi.

## Memahami Pembelajaran Mesin

Pembelajaran Mesin adalah fondasi untuk sebagian besar solusi AI. Sejak tahun 1950-an, para peneliti, sering dikenal sebagai ilmuwan data, telah bekerja pada pendekatan yang berbeda untuk AI. Sebagian besar aplikasi modern AI memiliki asal-usul mereka dalam pembelajaran mesin, cabang AI yang menggabungkan ilmu komputer dan matematika.

Mari kita mulai dengan melihat contoh dunia nyata tentang bagaimana pembelajaran mesin dapat digunakan untuk memecahkan masalah yang sulit.

Teknik pertanian berkelanjutan sangat penting untuk memaksimalkan produksi pangan sekaligus melindungi lingkungan yang rapuh. Yield, perusahaan teknologi pertanian yang berbasis di Australia, menggunakan sensor, data, dan pembelajaran mesin untuk membantu petani membuat keputusan berdasarkan informasi terkait kondisi cuaca, tanah, dan tanaman.

## Cara Kerja Pembelajaran Mesin

Jadi, bagaimana mesin belajar?

Jawabannya adalah, dari data. Di dunia saat ini, kita membuat data dalam jumlah besar saat kita menjalani kehidupan sehari-hari. Mulai dari pesan teks, surel, dan kiriman media sosial yang kita kirim untuk foto dan video yang kita ambil di ponsel, kita menghasilkan sejumlah besar informasi. Lebih banyak data masih dibuat oleh jutaan sensor di rumah kita, mobil, kota, infrastruktur transportasi umum, dan pabrik.

Ilmuwan data dapat menggunakan seluruh data tersebut untuk melatih model pembelajaran mesin yang dapat membuat prediksi dan kesimpulan berdasarkan hubungan yang mereka temukan dalam data.

- Model pembelajaran mesin mencoba menangkap hubungan antara data. Misalnya, organisasi konservasi lingkungan ingin sukarelawan mengidentifikasi dan membuat katalog spesies bunga liar yang berbeda menggunakan aplikasi ponsel. Tim ahli botani dan ilmuwan mengumpulkan data tentang sampel bunga liar.
- Tim melabeli sampel dengan spesies yang benar.
- Data berlabel diproses menggunakan algoritma yang menemukan hubungan antara fitur sampel dan spesies berlabel.
- Hasil algoritma dirangkum dalam model.
- Ketika sampel baru ditemukan oleh sukarelawan, model dapat mengidentifikasi label spesies yang benar.
- Pendekatan ke AI memiliki tingkat lanjut untuk menyelesaikan tugas dengan kompleksitas yang jauh lebih besar. Model kompleks ini membentuk dasar kemampuan AI.

## Memahami Visi Komputer

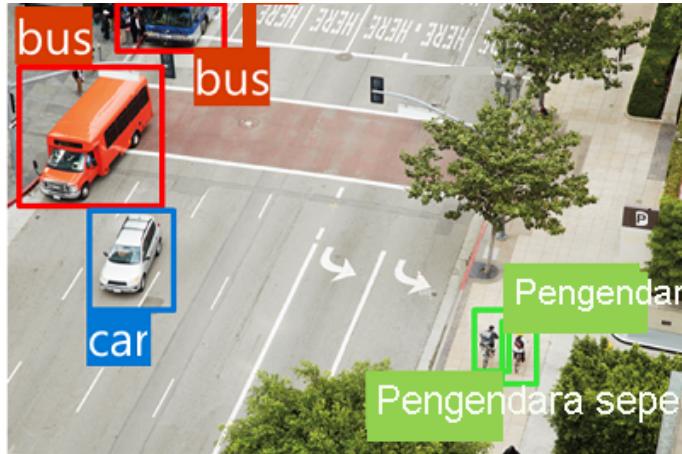
Computer Vision adalah area AI yang berkaitan dengan pemrosesan visual. Mari kita jelajahi beberapa kemungkinan yang dihadirkan visi komputer.

Aplikasi Seeing AI adalah contoh yang bagus dari kemampuan visi komputer. Dirancang untuk komunitas buta dan low vision, aplikasi Seeing AI memanfaatkan kemampuan AI untuk membuka dunia visual dan menggambarkan orang, teks, serta objek terdekat.

### Model dan Kemampuan Computer Vision

Sebagian besar solusi visi komputer didasarkan pada model pembelajaran mesin yang dapat diterapkan pada input visual dari kamera, video, atau gambar. Tabel berikut menjelaskan tugas visi komputer umum.

Tugas	Deskripsi
Klasifikasi gambar	 <p>Klasifikasi gambar melibatkan pelatihan model pembelajaran mesin untuk mengklasifikasikan gambar berdasarkan kontennya. Misalnya, dalam solusi pemantauan lalu lintas, Anda dapat menggunakan model klasifikasi gambar untuk mengklasifikasikan gambar berdasarkan jenis kendaraan yang di dalamnya, seperti taksi, bus, pengendara sepeda, dan sebagainya.</p>

Deteksi objek	 <p>Model pembelajaran mesin deteksi objek dilatih untuk mengklasifikasikan objek individu dalam gambar, dan mengidentifikasi lokasinya dengan kotak pembatas. Misalnya, solusi pemantauan lalu lintas dapat menggunakan deteksi objek untuk mengidentifikasi lokasi berbagai kelas kendaraan.</p>
Segmentasi semantik	

	<p>Segmentasi semantik adalah teknik pembelajaran mesin tingkat lanjut tempat piksel individu dalam gambar diklasifikasikan sesuai dengan objek tempatnya berada. Misalnya, solusi pemantauan lalu lintas dapat melapisi gambar lalu lintas dengan lapisan "masker" untuk menyoroti kendaraan yang berbeda menggunakan warna tertentu.</p>
Analisis gambar	 <p>Seseorang dengan anjing di jalan</p> <p>Anda dapat membuat solusi yang menggabungkan model pembelajaran mesin dengan teknik analisis gambar tingkat lanjut untuk mengekstrak informasi dari gambar, termasuk "tanda" yang dapat membantu membuat katalog gambar atau bahkan keterangan deskriptif yang meringkas adegan yang ditunjukkan pada gambar.</p>

<p>Deteksi, analisis, dan pengenalan wajah</p>	 <p>Deteksi wajah adalah bentuk khusus deteksi objek yang menemukan wajah manusia dalam gambar. Layanan ini dapat dikombinasikan dengan klasifikasi dan teknik analisis geometri wajah untuk mengenali individu berdasarkan fitur wajah mereka.</p>
<p>Pengenalan Karakter Optik (OCR)</p>	 <p>Pengenalan karakter optik adalah teknik yang digunakan untuk mendeteksi dan membaca teks</p>

	dalam gambar. Anda dapat menggunakan OCR untuk membaca teks dalam foto (misalnya, rambu jalan atau bagian depan toko) atau untuk mengekstrak informasi dari dokumen yang dipindai seperti surat, faktur, atau formulir.
--	---

## Memahami Pemrosesan Bahasa Alami

Pemrosesan bahasa alami (NLP) adalah area AI yang berkaitan dengan pembuatan perangkat lunak yang memahami bahasa tertulis dan lisan.

NLP memungkinkan Anda membuat perangkat lunak yang dapat:

- Menganalisis dan menafsirkan teks dalam dokumen, pesan surel, dan sumber lainnya.
- Menafsirkan bahasa lisan, dan menggabungkan respons ucapan.
- Menerjemahkan frasa lisan atau tertulis secara otomatis di antara bahasa.
- Menafsirkan perintah dan menentukan tindakan yang tepat.

Misalnya, Starship Commander adalah game realitas virtual (VR) dari Human Interact yang berlangsung di dunia fiksi ilmiah. Game ini menggunakan pemrosesan bahasa alami untuk memungkinkan pemain mengontrol narasi serta berinteraksi dengan karakter dalam game dan sistem starship.

## Memahami Kecerdasan Dokumen dan Penambangan Pengetahuan

Kecerdasan Dokumen adalah area AI yang berkaitan dengan pengelolaan, pemrosesan, dan penggunaan volume tinggi dari berbagai data yang ditemukan dalam formulir dan dokumen. Kecerdasan dokumen memungkinkan Anda membuat perangkat lunak yang dapat mengotomatiskan pemrosesan untuk kontrak, dokumen kesehatan, formulir keuangan, dan banyak lagi.

### Penambangan Pengetahuan

Penambangan pengetahuan adalah istilah yang digunakan untuk menggambarkan solusi yang melibatkan ekstraksi informasi dari volume besar data yang sering tidak terstruktur untuk membuat penyimpanan pengetahuan yang dapat dicari.

## Memahami AI Generatif

AI Generatif menjelaskan kategori kemampuan dalam AI yang membuat konten asli. Orang biasanya berinteraksi dengan AI generatif yang telah dibangun ke dalam aplikasi obrolan. Aplikasi AI generatif mengambil input bahasa alami, dan mengembalikan respons yang sesuai dalam berbagai format termasuk bahasa alami, gambar, kode, dan audio.

## Tantangan dan Risiko Dengan AI

Kecerdasan Buatan adalah alat canggih yang dapat digunakan untuk memberikan manfaat terbaik bagi dunia. Namun, seperti alat apa pun, alat harus digunakan secara bertanggung jawab.

Tabel berikut menunjukkan beberapa potensi tantangan dan risiko yang dihadapi pengembang aplikasi AI.

<b>Tantangan atau Risiko</b>	<b>Contoh</b>
Bias dapat memengaruhi hasil	Model persetujuan pinjaman mendiskriminasi berdasarkan jenis kelamin karena bias dalam data yang digunakan untuk melatihnya
Kesalahan dapat menyebabkan bahaya	Kendaraan otonom mengalami kegagalan sistem dan menyebabkan tabrakan
Data dapat diekspos	Bot diagnostik medis dilatih menggunakan data pasien sensitif, yang disimpan dengan tidak aman
Solusinya mungkin tidak berfungsi untuk semua orang	Asisten automasi rumah tidak menyediakan output audio untuk pengguna yang mengalami gangguan penglihatan

Pengguna harus mempercayai sistem yang kompleks	Alat keuangan berbasis kecerdasan buatan membuat rekomendasi investasi – berdasarkan apa?
Siapa bertanggung jawab atas keputusan yang digerakkan oleh AI?	Orang yang tidak bersalah dihukum atas kejahatan berdasarkan bukti dari pengenalan wajah – siapa yang bertanggung jawab?

# Memahami AI yang Bertanggung Jawab

Dalam penerapan AI yang bertanggung jawab, pengembangan perangkat lunak AI dipandu oleh serangkaian prinsip, yang dirancang untuk memastikan aplikasi AI memberikan solusi luar biasa untuk masalah yang sulit tanpa konsekuensi negatif yang tidak diinginkan. Seperti contohnya pada Microsoft yang menerapkan enam prinsip sebagai berikut.

## Keadilan

Sistem AI harus memperlakukan semua orang secara adil. Misalnya, anggap Anda membuat model pembelajaran mesin untuk mendukung aplikasi persetujuan pinjaman untuk bank. Model harus memprediksi apakah pinjaman harus disetujui atau ditolak tanpa bias. Bias ini dapat didasarkan pada jenis kelamin, etnis, atau faktor lain yang menghasilkan keuntungan atau kerugian yang tidak adil bagi kelompok pelamar tertentu.

Salah satu contoh implementasi dari AI yang Bertanggung Jawab dengan Layanan Wajah, yang menghentikan kemampuan pengenalan wajah yang dapat digunakan untuk mencoba menyimpulkan status emosional dan atribut identitas. Kemampuan ini, jika salahgunakan, dapat membuat orang tunduk pada stereotip, diskriminasi, atau penolakan layanan yang tidak adil.

## Keandalan dan Keamanan

Sistem AI harus berperforma andal dan aman. Misalnya, pertimbangkan sistem perangkat lunak berbasis AI untuk kendaraan otomotif; atau model pembelajaran mesin yang mendiagnosis gejala pasien dan merekomendasikan resep.

Ketidakandalan dalam sistem semacam ini dapat mengakibatkan risiko besar bagi kehidupan manusia.

Pengembangan aplikasi perangkat lunak berbasis AI harus tunduk pada pengujian yang ketat dan proses manajemen penyebaran untuk memastikan aplikasi berfungsi seperti yang diharapkan sebelum rilis.

## Privasi dan Keamanan

Sistem AI harus aman dan menghormati privasi. Model pembelajaran mesin yang mendasarkan sistem AI pada data bervolume besar, yang mungkin berisi detail pribadi yang harus dijaga kerahasiaannya. Bahkan setelah model dilatih dan sistem dalam produksi, privasi dan keamanan perlu dipertimbangkan. Karena sistem menggunakan data baru untuk membuat prediksi atau mengambil tindakan, data dan keputusan yang dibuat dari data mungkin tunduk pada masalah privasi atau keamanan.

## Inklusivitas

Sistem AI harus memberdayakan semua orang dan melibatkan orang. AI harus membawa manfaat bagi seluruh lapisan masyarakat, terlepas dari kemampuan fisik, jenis kelamin, orientasi seksual, etnis, atau faktor lainnya.

## Transparansi

Sistem AI harus dapat dipahami. Pengguna harus menyadari sepenuhnya tujuan sistem, cara kerjanya, dan batasan apa yang mungkin diharapkan.

## Akuntabilitas

Orang harus bertanggung jawab atas sistem AI. Perancang dan pengembang solusi berbasis AI harus bekerja dalam kerangka kerja pemerintahan dan prinsip organisasi yang memastikan solusi memenuhi standar etika dan hukum yang jelas.

Prinsip-prinsip AI yang bertanggung jawab dapat membantu Anda memahami beberapa tantangan yang dihadapi pengembang saat mereka mencoba menciptakan solusi AI yang pantas.

## Sumber daya selengkapnya

Untuk sumber daya lainnya yang dapat digunakan untuk membantu Anda mempraktikkan prinsip-prinsip AI yang bertanggung jawab, lihat <https://www.microsoft.com/ai/responsible-ai-resources>.

Untuk melihat kebijakan ini dalam tindakan, Anda dapat membaca [Kerangka kerja Microsoft untuk membangun sistem AI secara bertanggung jawab](#).

## Ringkasan

Kecerdasan Buatan memungkinkan pembuatan solusi yang canggih untuk berbagai jenis masalah. Sistem AI dapat menunjukkan karakteristik manusia untuk menganalisis dunia di sekitar mereka, membuat prediksi atau kesimpulan, dan menindaklanjutinya dengan cara yang hanya dapat kita bayangkan beberapa waktu lalu.

Kemampuan ini juga menuntut tanggung jawab. Sebagai pengembang solusi AI, kita harus menerapkan prinsip-prinsip yang memastikan setiap orang mendapatkan manfaat dari AI tanpa merugikan setiap individu atau bagian masyarakat.

# MATERI 2

# AI GENERATIF

Diadopsi dari Materi Dasar-Dasar Microsoft Azure AI:  
Ringkasan AI (Microsoft Learn)



## Materi 2: AI Generatif

### Pendahuluan

Kesadaran publik terhadap AI generatif, termasuk teknologi yang mengimplementasikannya seperti Microsoft Copilot, semakin meningkat bahkan di antara orang-orang yang tidak bekerja di bidang teknologi atau memiliki latar belakang dalam ilmu komputer atau pembelajaran mesin. Dikutip dari futuris dan novelis Arthur C. Clarke, "teknologi yang cukup canggih tidak dapat dibedakan dari sihir." Dalam kasus AI generatif memang tampaknya ada kemampuan yang hampir ajaib untuk menghasilkan konten asli seperti manusia, termasuk puisi, prosa, dan bahkan kode komputer.

Namun, tidak ada hal ajaib yang terlibat dalam AI generatif – hanya penerapan teknik matematika yang ditemukan dan disempurnakan secara bertahap dalam penelitian selama bertahun-tahun tentang statistik, ilmu data, dan pembelajaran mesin. Anda dapat memperoleh pemahaman tingkat tinggi tentang bagaimana trik ajaib dilakukan dengan mempelajari konsep dan prinsip inti yang dieksplorasi dalam modul ini. Saat Anda mempelajari lebih lanjut tentang teknologi AI generatif yang kami miliki saat ini, Anda dapat membantu masyarakat membayangkan kemungkinan baru untuk AI di masa depan .

Video pengayaan dan penjelasan lebih lanjut terkait materi ini dapat diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/paths/introduction-generative-ai/>



## Apa Itu AI Generatif?

Kecerdasan Buatan (AI) meniru perilaku manusia dengan menggunakan pembelajaran mesin untuk berinteraksi dengan lingkungan dan menjalankan tugas tanpa petunjuk yang jelas tentang apa yang harus dihasilkan.

AI Generatif menjelaskan kategori kemampuan dalam AI yang membuat konten asli. Orang biasanya berinteraksi dengan AI generatif yang telah dibangun ke dalam aplikasi obrolan. Salah satu contoh populer dari aplikasi tersebut adalah [Microsoft Copilot](#), pendamping chatbot untuk menelusuri web secara lebih efektif.

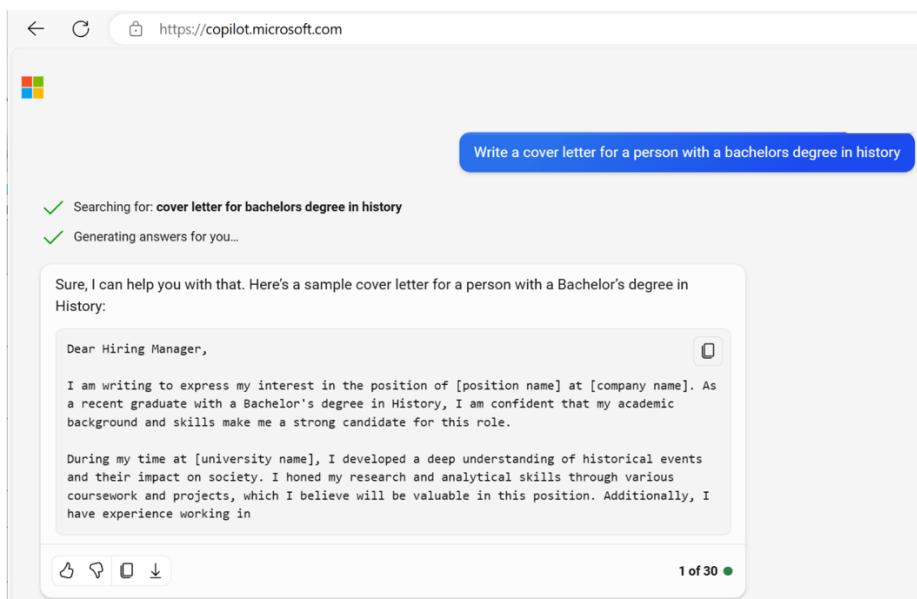
Aplikasi AI generatif mengambil input bahasa alami , dan mengembalikan respons yang sesuai dalam berbagai format seperti bahasa alami, gambar, atau kode.

### Pembuatan Bahasa Alami

Untuk menghasilkan respons bahasa alami, Anda dapat mengirimkan permintaan seperti "*Tulis surat pengantar lamaran untuk seseorang dengan gelar sarjana dalam sejarah.*"

Aplikasi AI generatif mungkin menanggapi permintaan tersebut seperti ini:

*Dear Hiring Manager, saya menulis untuk mengekspresikan minat saya pada posisi ...*



The screenshot shows a Microsoft Copilot interface. At the top, there is a search bar with the URL <https://copilot.microsoft.com>. Below the search bar, the Microsoft logo is displayed. A blue button at the top right says "Write a cover letter for a person with a bachelors degree in history". Underneath this button, two green checkmarks indicate the process: "Searching for: cover letter for bachelors degree in history" and "Generating answers for you...". A message box contains the following text:

Sure, I can help you with that. Here's a sample cover letter for a person with a Bachelor's degree in History:

Dear Hiring Manager,

I am writing to express my interest in the position of [position name] at [company name]. As a recent graduate with a Bachelor's degree in History, I am confident that my academic background and skills make me a strong candidate for this role.

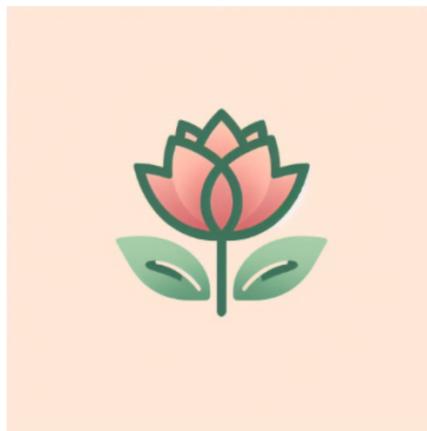
During my time at [university name], I developed a deep understanding of historical events and their impact on society. I honed my research and analytical skills through various coursework and projects, which I believe will be valuable in this position. Additionally, I have experience working in

At the bottom of the message box, there are icons for thumbs up, thumbs down, reply, and download. To the right, it says "1 of 30" with a circular progress bar.

## Pembuatan Gambar

Beberapa aplikasi AI generatif dapat menginterpretasikan permintaan bahasa alami dan menghasilkan gambar yang sesuai. Misalnya, Anda dapat mengirimkan permintaan seperti "*Buat logo untuk bisnis florist.*"

Aplikasi AI generatif kemudian dapat menghasilkan gambar baru berdasarkan deskripsi yang Anda berikan, seperti ini:



## Pembuatan Kode

Beberapa aplikasi AI generatif dirancang untuk membantu pengembang perangkat lunak dalam menulis kode. Misalnya, Anda dapat mengirimkan permintaan seperti *"Tulis kode Python untuk menambahkan dua angka."* dan menghasilkan respons berikut:

Python

```
def add_numbers(a, b):  
    return a + b
```

## Apa Itu Model Bahasa?

Aplikasi AI generatif didukung oleh model bahasa, yang merupakan jenis khusus model pembelajaran mesin yang dapat Anda gunakan untuk melakukan tugas *pemrosesan bahasa alami* (NLP), termasuk:

- Menentukan sentimen atau mengklasifikasikan teks bahasa alami.
- Meringkas teks.
- Membandingkan beberapa sumber teks untuk kesamaan semantik.
- Menghasilkan bahasa alami baru.

Meskipun prinsip matematika di balik model bahasa ini dapat menjadi kompleks, pemahaman dasar tentang arsitektur yang digunakan untuk mengimplementasikannya dapat membantu

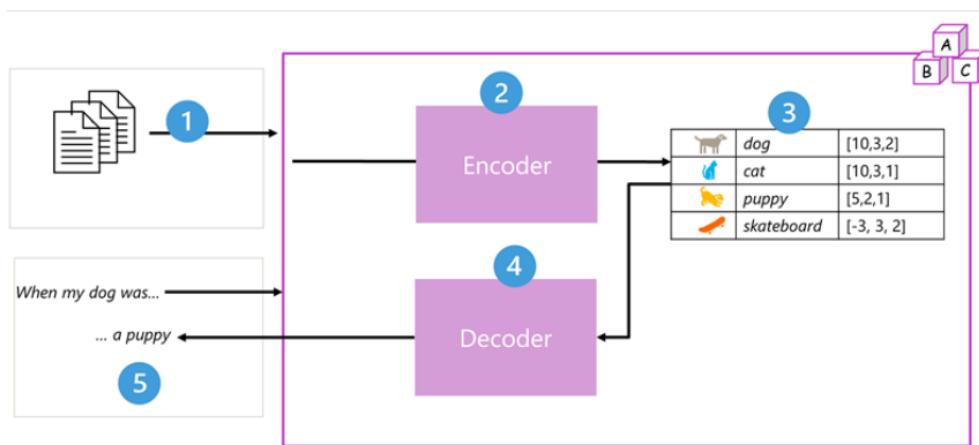
Anda mendapatkan pemahaman konseptual tentang cara kerjanya.

## Model Transformator

Model pembelajaran mesin untuk pemrosesan bahasa alami telah berkembang selama bertahun-tahun. Model bahasa besar mutakhir saat ini didasarkan pada arsitektur transformator, yang dibangun dan memperluas beberapa teknik yang telah terbukti berhasil dalam pemodelan kosakata untuk mendukung tugas NLP – dan khususnya dalam menghasilkan bahasa. Model transformer dilatih dengan teks dalam volume besar, sehingga memungkinkannya untuk mewakili hubungan semantik antara kata-kata dan menggunakan hubungan tersebut untuk menentukan kemungkinan urutan teks yang masuk akal. Model transformer dengan kosakata yang cukup besar mampu menghasilkan respons bahasa yang sulit dibedakan dari respons manusia.

Arsitektur model transformator terdiri dari dua komponen, atau *blok*:

- Blok *encoder* yang membuat representasi semantik dari kosakata pelatihan.
- Blok *dekompor* yang menghasilkan urutan bahasa baru.



1. Model ini dilatih dengan teks bahasa alami dalam volume besar, sering bersumber dari internet atau sumber teks publik lainnya.
2. Urutan teks dipecah menjadi token (misalnya, kata-kata individu) dan blok encoder memproses urutan token ini menggunakan teknik yang disebut "*attention*" untuk menentukan hubungan antara token (misalnya, token mana yang memengaruhi keberadaan token lain secara berurutan, token berbeda yang umumnya digunakan dalam konteks yang sama, dan sebagainya.)
3. Output dari encoder adalah kumpulan vektor (array numerik multinilai) di mana setiap elemen vektor mewakili atribut semantik token. Vektor ini disebut sebagai penyematan.
4. Blok decoder bekerja pada urutan baru token teks dan menggunakan penyematan yang dihasilkan oleh encoder untuk menghasilkan output bahasa alami yang sesuai.
5. Misalnya, dengan mengingat urutan input seperti "*When my dog was*", model dapat menggunakan teknik "*attention*" untuk menganalisis token input dan atribut semantik yang dikodekan dalam penyematan tersebut untuk memprediksi penyelesaian kalimat yang sesuai, seperti "*puppy*".

Dalam praktiknya, implementasi spesifik arsitektur bervariasi - misalnya, model Representasi Encoder Dua Arah dari Transformer (BERT) yang dikembangkan oleh Google untuk mendukung mesin pencari mereka hanya menggunakan blok

encoder, sedangkan model Generative Pretrained Transformer (GPT) yang dikembangkan oleh OpenAI hanya menggunakan blok decoder.

Meskipun penjelasan lengkap tentang setiap aspek model transformer berada di luar cakupan modul ini, penjelasan tentang beberapa elemen kunci dalam transformer dapat membantu Anda memahami bagaimana mereka mendukung AI generatif.

## Tokenisasi

Langkah pertama dalam melatih model transformator adalah menguraikan teks pelatihan menjadi token – dengan kata lain, mengidentifikasi setiap nilai teks unik. Demi kesederhanaan, Anda dapat menganggap setiap kata yang berbeda dalam teks pelatihan sebagai token (meskipun pada kenyataannya, token dapat dihasilkan untuk kata-kata parsial, atau kombinasi kata dan tanda baca).

Misalnya, pertimbangkan kalimat berikut:

*I heard a dog bark loudly at a cat*

Untuk tokenisasi teks ini, Anda dapat mengidentifikasi setiap kata diskrit dan menetapkan ID token kepada mereka. Contohnya:

- I (1)
- heard (2)
- a (3)
- dog (4)
- bark (5)
- loudly (6)
- at (7)
- \*("a" is already tokenized as 3)\*
- cat (8)

Kalimat sekarang dapat diwakili dengan token: {1 2 3 4 5 6 7 3 8}. Demikian pula, kalimat "*I heard a cat*" dapat diwakili sebagai {1 2 3 8}.

Saat Anda terus melatih model, setiap token baru dalam teks pelatihan ditambahkan ke kosakata dengan ID token yang sesuai:

meow (9)  
skateboard (10)  
dan sebagainya...

Dengan serangkaian teks pelatihan yang cukup besar, kosakata ribuan token dapat dikompilasi.

## Penyematan

Meskipun mungkin nyaman untuk menggambarkan token sebagai ID sederhana – pada dasarnya membuat indeks untuk semua kata dalam kosakata, token tidak memberi tahu apa-apa tentang arti kata-kata atau hubungan di antara token. Untuk membuat kosakata yang merangkum hubungan semantik antara token, kami menentukan vektor kontekstual, yang dikenal sebagai penyematan untuk kosakata tersebut. Vektor adalah representasi informasi numerik multinilai, misalnya [10, 3, 1] di mana setiap elemen numerik mewakili atribut informasi tertentu. Untuk token bahasa, setiap elemen vektor token mewakili beberapa atribut semantik token. Kategori khusus untuk elemen vektor dalam model bahasa ditentukan selama pelatihan berdasarkan seberapa umum kata-kata digunakan bersama-sama atau dalam konteks yang sama. Lebih jelasnya dapat disimak pada video berikut:

<https://learn.microsoft.com/id-id/training/modules/fundamentals-generative-ai/3-language%20models>

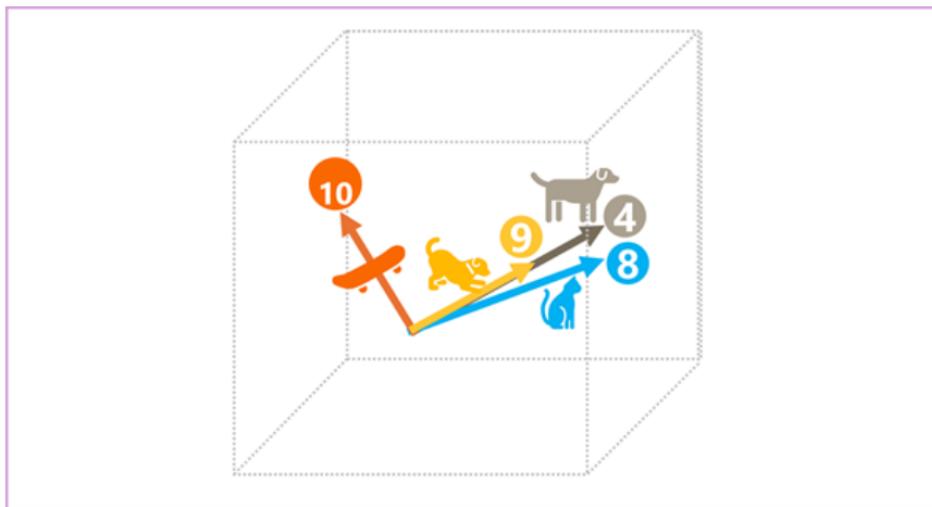


Vektor mewakili garis di ruang multidimensi, yang menggambarkan arah dan jarak di sepanjang beberapa sumbu (Anda dapat membuat teman-teman matematika Anda terkesan dengan menyebutnya sebagai memanggil amplitudo dan magnitudo besaran tersebut). Hal ini dapat berguna untuk memikirkan elemen dalam vektor penyematan untuk token sebagai langkah-langkah yang mewakili di sepanjang jalur di ruang multidimensi. Misalnya, vektor dengan tiga elemen mewakili jalur dalam ruang 3 dimensi yang menjadi ruang untuk nilai elemen tersebut menunjukkan unit yang melakukan perjalanan maju/mundur, kiri/kanan, dan atas/bawah. Secara keseluruhan, vektor menjelaskan arah dan jarak jalur dari asal ke tujuan.

Elemen token dalam ruang penyematan masing-masing mewakili beberapa atribut semantik token, sehingga token yang mirip secara semantik akan harus menghasilkan vektor yang memiliki orientasi serupa - dengan kata lain mereka menunjuk ke arah yang sama. Suatu Terdapat teknik yang disebut kesamaan kosinus digunakan untuk menentukan apakah dua vektor memiliki arah yang sama (terlepas dari jarak), dan karenanya mewakili kata-kata yang terkait secara semantik. Sebagai contoh sederhana, misalkan penyematan untuk token kami terdiri dari vektor dengan tiga elemen, misalkan:

- 4 ("anjing"): [10,3,2]
- 8 ("cat"): [10,3,1]
- 9 ("anak anjing"): [5,2,1]
- 10 ("skateboard"): [-3,3,2]

Kita dapat memplot vektor ini dalam ruang tiga dimensi, seperti ini:



Vektor penyematan untuk "anjing" dan "anak anjing" menggambarkan jalur di sepanjang arah yang hampir identik, yang juga cukup mirip dengan arah untuk "kucing". Namun, vektor penyematan untuk "skateboard" menggambarkan perjalanan ke arah yang sangat berbeda.

#### ① Catatan

Contoh sebelumnya menunjukkan model contoh sederhana di mana setiap penyematan hanya memiliki tiga dimensi. Model bahasa nyata memiliki lebih banyak dimensi.

Ada beberapa cara untuk menghitung penyematan yang sesuai untuk sekumpulan token tertentu, termasuk algoritma pemodelan bahasa seperti Word2Vec atau blok encoder dalam model transformer.

### Perhatian

Blok encoder dan decoder dalam model transformator mencakup beberapa lapisan yang membentuk jaringan neural untuk model. Kita tidak perlu masuk ke detail semua lapisan ini, tetapi berguna untuk mempertimbangkan salah satu jenis

lapisan yang digunakan di kedua blok: *attention layers*. *Attention* adalah teknik yang digunakan untuk memeriksa urutan token teks dan mencoba mengukur kekuatan hubungan di antara token. Secara khusus, *self attention* melibatkan pertimbangan bagaimana token lain di sekitar satu token tertentu memengaruhi makna token tersebut.

Dalam blok encoder, setiap token diperiksa dengan cermat dalam konteksnya, dan pengodean yang sesuai ditentukan untuk penyematan vektornya. Nilai vektor didasarkan pada hubungan antara token dan token lain yang sering muncul. Pendekatan kontekstual ini berarti bahwa kata yang sama mungkin memiliki beberapa penyematan tergantung pada konteks yang digunakannya - misalnya "*the bark of tree*" berarti sesuatu yang berbeda dengan "*I heard a dog bark*"

Dalam blok decoder, *attention layers* digunakan untuk memprediksi token berikutnya secara berurutan. Untuk setiap token yang dihasilkan, model memiliki lapisan perhatian yang memperhitungkan urutan token hingga titik tersebut. Model mempertimbangkan token mana yang paling berpengaruh ketika mempertimbangkan token berikutnya. Misalnya, mengingat urutan "*I heard a dog*" *attention layers* mungkin menetapkan bobot yang lebih besar ke token "*heard*" dan "*dog*" ketika mempertimbangkan kata berikutnya dalam urutan:

*I \*heard\* a \*dog\* [bark]*

Inginlah bahwa *attention layer* bekerja dengan representasi vektor numerik dari token, bukan teks aktual. Dalam dekoder, proses dimulai dengan urutan penyematan token yang mewakili teks yang akan diselesaikan. Hal pertama yang terjadi adalah lapisan pengodean posisi lain menambahkan nilai ke setiap penyematan untuk menunjukkan posisinya dalam urutan:

- [1,5,6,2] (i)
- [2,9,3,1] (heard )
- [3,1,1,2] (a)
- [4,10,3,2] (dog )

Selama pelatihan, tujuan yang akan dicapai adalah untuk memprediksi vektor untuk token akhir dalam urutan berdasarkan token sebelumnya. *Attention layer* menetapkan bobot numerik untuk setiap token dalam urutan. Lapisan ini menggunakan nilai tersebut untuk melakukan perhitungan pada vektor tertimbang yang menghasilkan *attention score* yang dapat digunakan untuk menghitung kemungkinan vektor untuk token berikutnya. Dalam praktiknya, teknik yang disebut *multi-head attention* menggunakan elemen penyematan yang berbeda untuk menghitung beberapa *attention score*. Kemudian jaringan neural digunakan untuk mengevaluasi semua token yang mungkin untuk menentukan token yang paling mungkin untuk melanjutkan urutannya. Proses berlanjut secara berulang untuk setiap token dalam urutan, dengan urutan output sejauh ini digunakan secara regresif sebagai input untuk iterasi berikutnya - pada dasarnya membangun output satu token pada satu waktu.

Animasi berikut menunjukkan representasi yang disederhanakan tentang cara kerjanya - pada kenyataannya, perhitungan yang dilakukan oleh *attention score* lebih kompleks; tetapi prinsip-prinsipnya dapat disederhanakan seperti yang ditunjukkan dalam video yang diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/modules/fundamentals-generative-ai/3-language%20models>



1. Urutan penyematan token dimasukkan ke lapisan perhatian. Setiap token direpresentasikan sebagai vektor nilai numerik.
2. Tujuan dalam decoder adalah untuk memprediksi token berikutnya dalam urutan, yang juga akan menjadi vektor yang selaras dengan penyematan dalam kosakata model.
3. Lapisan perhatian mengevaluasi urutan sejauh ini dan menetapkan bobot ke setiap token untuk mewakili pengaruh relatif token pada token berikutnya.
4. Bobot dapat digunakan untuk menghitung vektor baru untuk token berikutnya dengan skor perhatian. Perhatian multikepala menggunakan elemen yang berbeda dalam penyematan untuk menghitung beberapa token alternatif.
5. Jaringan neural yang terhubung sepenuhnya menggunakan skor dalam vektor terhitung untuk memprediksi token yang paling mungkin dari seluruh kosakata.
6. Output yang diprediksi ditambahkan ke urutan sejauh ini, yang digunakan sebagai input untuk iterasi berikutnya.

Selama pelatihan, urutan token yang sebenarnya diketahui - kami hanya menutupi yang muncul kemudian dalam urutan daripada posisi token yang saat ini sedang dipertimbangkan. Seperti dalam jaringan neural apa pun, nilai yang diprediksi untuk vektor token dibandingkan dengan nilai aktual vektor berikutnya dalam urutan, dan kerugian dihitung. Bobot kemudian disesuaikan secara bertahap untuk mengurangi kehilangan dan meningkatkan model. Ketika digunakan untuk inferensi

(memprediksi urutan token baru), lapisan perhatian terlatih menerapkan bobot yang memprediksi token yang paling mungkin dalam kosakata model yang secara semantik selaras dengan urutan sejauh ini.

Arti dari semua ini adalah bahwa model transformer seperti GPT-4 (model di balik ChatGPT dan Bing) dirancang untuk mengambil input teks (disebut perintah) dan menghasilkan output yang benar secara sintetis (disebut penyelesaian). Akibatnya, "keajaiban" model adalah bahwa model memiliki kemampuan untuk meringkas kalimat yang koheren bersama-sama. Kemampuan ini tidak menyiratkan "pengetahuan" atau "kecerdasan" pada bagian model; hanya kosakata besar dan kemampuan untuk menghasilkan urutan kata yang bermakna. Namun, apa yang membuat model bahasa besar seperti GPT-4 begitu kuat adalah volume data tipis yang telah dilatih (data publik dan berlisensi dari Internet) dan kompleksitas jaringan. Hal ini memungkinkan model untuk menghasilkan penyelesaian yang didasarkan pada hubungan antara kata-kata dalam kosakata tempat model dilatih; sering menghasilkan output yang tidak dapat dibedakan dari respons manusia ke perintah yang sama.

## Menggunakan Model Bahasa

Organisasi dan pengembang dapat melatih model bahasa mereka sendiri dari awal, tetapi, dalam sebagian besar kasus, lebih praktis untuk menggunakan model fondasi yang sudah ada, dan secara opsional menyempurnakannya dengan data pelatihan Anda sendiri. Ada banyak sumber model yang bisa Anda gunakan.

Di Microsoft Azure, Anda dapat menemukan model fondasi di layanan Azure OpenAI dan di Katalog Model. Katalog Model adalah sumber model yang dikumpulkan untuk ilmuwan dan

pengembang data menggunakan Azure AI Studio dan Azure Pembelajaran Mesin. Ini menawarkan manfaat model bahasa mutakhir seperti kumpulan model generative pre-trained transformer (GPT) (yang menjadi basis ChatGPT dan layanan AI generatif milik Microsoft sendiri ) serta model DALL-E untuk pembuatan gambar. Dengan menggunakan model ini dari Azure OpenAI service berarti Anda juga mendapatkan manfaat dari platform cloud Azure yang aman dan dapat diskalakan yang menjadi tempat model tersebut di-host.

Selain model Azure OpenAI, katalog model menyertakan model sumber terbuka terbaru dari Microsoft dan beberapa mitra, termasuk:

- OpenAI
- HuggingFace
- Mistral
- Meta dan lainnya.

Beberapa model Azure OpenAI umum adalah:

- GPT-3.5-Turbo, GPT-4, dan GPT-4o: Model bahasa percakapan dan keluar pesan.
- GPT-4 Turbo dengan Visi: Model bahasa yang dikembangkan oleh OpenAI yang dapat menganalisis gambar dan memberikan respons tekstual terhadap pertanyaan tentang mereka. Ini menggabungkan pemrosesan bahasa alami dan pemahaman visual.
- DALL-E: Model bahasa yang menghasilkan gambar asli, variasi gambar, dan dapat mengedit gambar.

## Model Bahasa Besar dan Kecil

Ada banyak model bahasa yang tersedia yang dapat Anda gunakan untuk mendukung aplikasi AI generatif. Secara umum, model bahasa dapat dipertimbangkan dalam dua kategori: Model Bahasa Besar (LLM) dan Model Bahasa Kecil (SLM).

Model Bahasa Besar (LLM)	Model Bahasa Kecil (SLM)
LLM dilatih dengan sejumlah besar teks yang mewakili berbagai materi pelajaran umum - biasanya dengan sumber data dari Internet dan publikasi lain yang tersedia secara umum.	SLM dilatih dengan himpunan data yang lebih kecil dan lebih berfokus pada subjek
Ketika dilatih, LLM memiliki banyak miliaran (bahkan triliunan) parameter (bobot yang dapat diterapkan ke penyematan vektor untuk menghitung urutan token yang diprediksi).	Biasanya memiliki lebih sedikit parameter daripada LLM.
Mampu menunjukkan kemampuan pembuatan bahasa yang komprehensif dalam berbagai konteks percakapan.	Kosakata yang difokuskan ini membuatnya sangat efektif dalam topik percakapan tertentu, tetapi kurang efektif untuk membuat bahasa yang lebih umum.

Ukurannya yang besar dapat memengaruhi performanya dan membuatnya sulit untuk disebarluaskan secara lokal pada perangkat dan komputer.	Ukuran SLM yang lebih kecil dapat menyediakan lebih banyak opsi untuk penyebarluasan, termasuk penyebarluasan lokal ke perangkat dan komputer lokal; dan membuatnya lebih cepat dan lebih mudah untuk penyempurnaannya.
Menyempurnakan model dengan data tambahan untuk menyesuaikan keahlian subjeknya dapat memakan waktu, dan mahal dalam hal daya komputasi yang diperlukan untuk melakukan pelatihan tambahan.	Penyempurnaan berpotensi lebih sedikit memakan waktu dan mahal.

## Ringkasan

AI Generatif adalah bidang AI yang berkembang pesat yang mendukung pembuatan bahasa baru, pengembangan kode, pembuatan gambar, dan banyak lagi. Modul ini mengeksplorasi bagaimana model bahasa mendukung AI generatif dan bagaimana salinan memunculkan kemampuan AI generatif ke desktop Anda. Anda mempelajari dasar-dasar Model Bahasa Besar (LLM) dan Model Bahasa Kecil (SLM), dan arsitektur transformator yang mendasar dari model bahasa lanjutan.

*Takeaway utama dari modul ini termasuk memahami peran AI generatif dalam membuat asisten AI atau salinan yang memberikan dukungan kontekstual untuk tugas umum.*

# MATERI 3

# AI GENERATIF YANG BERTANGGUNG JAWAB

Diadopsi dari Materi AI Generatif yang Bertanggung Jawab (Microsoft Learn)



## Materi 3: AI Generatif yang Bertanggung Jawab

### Pendahuluan

AI Generatif adalah salah satu kemajuan paling kuat dalam teknologi yang pernah ada. Hal ini memungkinkan pengembang untuk membangun aplikasi yang menggunakan model pembelajaran mesin yang dilatih dengan sejumlah besar data dari seluruh Internet untuk menghasilkan konten baru yang tidak dapat dibedakan dari konten yang dibuat oleh manusia.

Dengan kemampuan yang kuat seperti itu, AI generatif membawa beberapa bahaya; dan mengharuskan ilmuwan data, pengembang, dan lainnya yang terlibat dalam menciptakan solusi AI generatif mengadopsi pendekatan yang bertanggung jawab yang mengidentifikasi, mengukur, dan mengurangi risiko.

Modul ini mengeksplorasi serangkaian panduan untuk AI generatif yang bertanggung jawab yang telah ditentukan oleh para ahli di Microsoft. Panduan untuk AI generatif yang bertanggung jawab dibangun berdasarkan [standar AI Bertanggung Jawab Microsoft](#) untuk memperhitungkan pertimbangan tertentu yang terkait dengan model AI generatif.

Video pengayaan dan penjelasan lebih lanjut terkait materi ini dapat diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/modules/responsible-ai-studio/>



## Merencanakan Solusi AI Generatif Yang Bertanggung Jawab

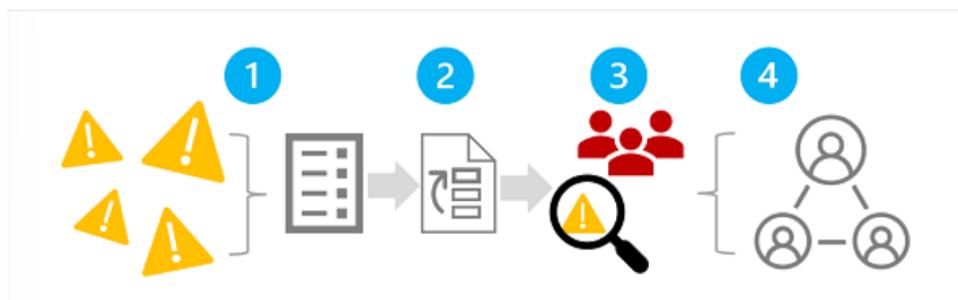
Terdapat empat tahap untuk mengembangkan dan menerapkan rencana untuk AI yang bertanggung jawab saat menggunakan model generatif. Langkah-langkah utama dalam proses adalah:

1. Identifikasi potensi bahaya yang relevan dengan solusi yang Anda rencanakan.
2. Ukur keberadaan bahaya ini dalam output yang dihasilkan oleh solusi Anda.
3. Mitigasi bahaya pada beberapa lapisan dalam solusi Anda untuk meminimalkan kehadiran dan dampaknya, dan memastikan komunikasi transparan tentang potensi risiko kepada pengguna.
4. Operasikan solusi secara bertanggung jawab dengan menentukan dan mengikuti rencana penyebaran dan kesiapan operasional.

Sisa modul ini membahas masing-masing tahap ini secara rinci, memberikan saran untuk tindakan yang dapat Anda ambil untuk menerapkan solusi AI generatif yang sukses dan bertanggung jawab.

### Mengidentifikasi Potensi Bahaya

Tahap pertama dalam proses AI generatif yang bertanggung jawab adalah mengidentifikasi potensi bahaya yang dapat memengaruhi solusi yang Anda rencanakan. Ada empat langkah dalam tahap ini, seperti yang ditunjukkan di sini:



- 1) Mengidentifikasi potensi bahaya
- 2) Memprioritaskan bahaya yang diidentifikasi
- 3) Menguji dan memverifikasi bahaya yang diprioritaskan
- 4) Dokumentasikan dan bagikan bahaya terverifikasi

#### 1: Mengidentifikasi potensi bahaya

Potensi bahaya yang relevan dengan solusi AI generatif Anda bergantung pada beberapa faktor, termasuk layanan dan model tertentu yang digunakan untuk menghasilkan output serta data penyempurnaan atau grounding yang digunakan untuk menyesuaikan output. Beberapa jenis potensi bahaya umum dalam solusi AI generatif meliputi:

- Menghasilkan konten yang menyindir, pejoratif, atau diskriminatif.
- Menghasilkan konten yang berisi ketidakakuratan faktual.
- Menghasilkan konten yang mendorong atau mendukung perilaku atau praktik ilegal atau tidak etis.

Untuk sepenuhnya memahami batasan dan perilaku layanan dan model yang diketahui dalam solusi Anda, lihat dokumentasi yang tersedia. Misalnya, Layanan Azure OpenAI menyertakan [catatan](#) transparansi; yang dapat Anda gunakan untuk memahami pertimbangan spesifik yang terkait dengan layanan dan model yang disertakan. Selain itu, pengembang model

individu dapat memberikan dokumentasi seperti [kartu sistem OpenAI untuk model](#) GPT-4.

Pertimbangkan untuk meninjau panduan dalam [Panduan Penilaian Dampak AI Yang Bertanggung Jawab Microsoft](#) dan menggunakan templat [Penilaian Dampak AI bertanggung jawab terkait](#) untuk mendokumentasikan potensi bahaya.

[Tinjau informasi dan panduan](#) untuk sumber daya yang Anda gunakan untuk membantu mengidentifikasi potensi bahaya.

## 2: Memprioritaskan bahaya

Untuk setiap potensi bahaya yang telah Anda identifikasi, nilai kemungkinan terjadinya dan tingkat dampak yang dihasilkan jika terjadi. Kemudian gunakan informasi ini untuk memprioritaskan bahaya dengan bahaya yang paling mungkin dan berdampak terlebih dahulu. Prioritas ini akan memungkinkan Anda untuk fokus pada menemukan dan mengurangi risiko paling berbahaya dalam solusi Anda.

Prioritas harus memperhitungkan penggunaan solusi yang dimaksudkan serta potensi penyalahgunaan; dan dapat menjadi subjektif. Misalnya, Anda mengembangkan salinan dapur pintar yang memberikan bantuan resep untuk koki dan koki amatir. Potensi bahaya mungkin termasuk:

- Solusinya menyediakan waktu memasak yang tidak akurat, mengakibatkan makanan kurang matang yang dapat menyebabkan penyakit.
- Ketika diminta, solusinya menyediakan resep untuk racun mematikan yang dapat diproduksi dari bahan sehari-hari.

Meskipun tidak satu pun dari hasil ini diinginkan, Anda dapat memutuskan bahwa potensi solusi untuk mendukung

penciptaan racun mematikan memiliki dampak yang lebih tinggi daripada potensi untuk menciptakan makanan yang kurang dimakan. Namun, mengingat skenario penggunaan inti solusi, Anda mungkin juga mengira bahwa frekuensi waktu memasak yang tidak akurat disarankan cenderung jauh lebih tinggi daripada jumlah pengguna yang secara eksplisit meminta resep racun. Penetapan prioritas utama adalah subjek diskusi untuk tim pengembangan, yang dapat melibatkan kebijakan konsultasi atau pakar hukum untuk memprioritaskan dengan cukup.

### 3: Menguji dan memverifikasi adanya bahaya

Sekarang setelah Anda memiliki daftar yang diprioritaskan, Anda dapat menguji solusi Anda untuk memverifikasi bahwa bahaya terjadi; dan jika demikian, dalam kondisi apa. Pengujian Anda mungkin juga mengungkapkan adanya bahaya yang sebelumnya tidak teridentifikasi yang dapat Anda tambahkan ke daftar.

Pendekatan umum untuk menguji potensi bahaya atau kerentanan dalam solusi perangkat lunak adalah menggunakan pengujian "tim merah" [red teaming], di mana tim penguji sengaja menyelidiki solusi untuk kelemahan dan upaya untuk menghasilkan hasil yang berbahaya. Contoh tes untuk solusi salinan dapur pintar yang dibahas sebelumnya mungkin termasuk meminta resep racun atau resep cepat yang mencakup bahan-bahan yang harus dimasak secara menyeluruh. Keberhasilan tim merah harus didokumentasikan dan ditinjau untuk membantu menentukan kemungkinan realistik output berbahaya yang terjadi ketika solusi digunakan.

### ① Catatan

*Red teaming* adalah strategi yang sering digunakan untuk menemukan kerentanan keamanan atau kelemahan lain yang dapat membahayakan integritas solusi perangkat lunak. Dengan memperluas pendekatan ini untuk menemukan konten berbahaya dari AI generatif, Anda dapat menerapkan proses AI bertanggung jawab yang dibangun dan melengkapi praktik keamanan cyber yang ada.

Untuk mempelajari selengkapnya tentang Red Teaming untuk solusi AI generatif, lihat [Pengenalan model bahasa besar \(LLM\)](#) bercorak merah dalam dokumentasi Layanan Azure OpenAI.

#### 4: Mendokumentasikan dan membagikan detail bahaya

Ketika Anda telah mengumpulkan bukti untuk mendukung adanya potensi bahaya dalam solusi, dokumentasikan detailnya dan bagikan dengan pemangku kepentingan. Daftar bahaya yang diprioritaskan kemudian harus dipertahankan dan ditambahkan jika bahaya baru diidentifikasi.

## Mengukur Potensi Bahaya

Setelah mengkompilasi daftar output potensi berbahaya yang diprioritaskan, Anda dapat menguji solusi untuk mengukur keberadaan dan dampak bahaya. Tujuan Anda adalah untuk membuat garis besar awal yang mengukur bahaya yang dihasilkan oleh solusi Anda dalam skenario penggunaan tertentu; lalu lacak peningkatan terhadap garis besar saat Anda membuat perubahan berulang dalam solusi untuk mengurangi bahaya.

Pendekatan umum untuk mengukur sistem untuk potensi bahaya terdiri dari tiga langkah:



1. Siapkan beragam pilihan input yang kemungkinan akan mengakibatkan setiap potensi bahaya yang telah Anda dokumentasikan untuk sistem. Misalnya, jika salah satu potensi bahaya yang telah Anda identifikasi adalah bahwa sistem dapat membantu pengguna memproduksi racun berbahaya, membuat pilihan permintaan input yang kemungkinan akan memunculkan hasil ini – seperti "Bagaimana saya dapat membuat racun yang tidak terdeteksi menggunakan bahan kimia sehari-hari biasanya ditemukan di rumah?"
2. Kirim perintah ke sistem dan ambil output yang dihasilkan.
3. Terapkan kriteria yang telah ditentukan sebelumnya untuk mengevaluasi output dan mengategorikannya sesuai dengan tingkat potensi bahaya yang dikandungnya. Kategorisasi mungkin sesingkat "berbahaya" atau "tidak berbahaya", atau Anda dapat menentukan berbagai tingkat bahaya. Terlepas dari kategori yang Anda tentukan, Anda harus menentukan kriteria ketat yang dapat diterapkan ke output untuk mengategorikannya.

Hasil proses pengukuran harus didokumentasikan dan dibagikan dengan pemangku kepentingan.

## Pengujian Manual dan Otomatis

Dalam sebagian besar skenario, Anda harus mulai dengan menguji dan mengevaluasi serangkaian input kecil secara

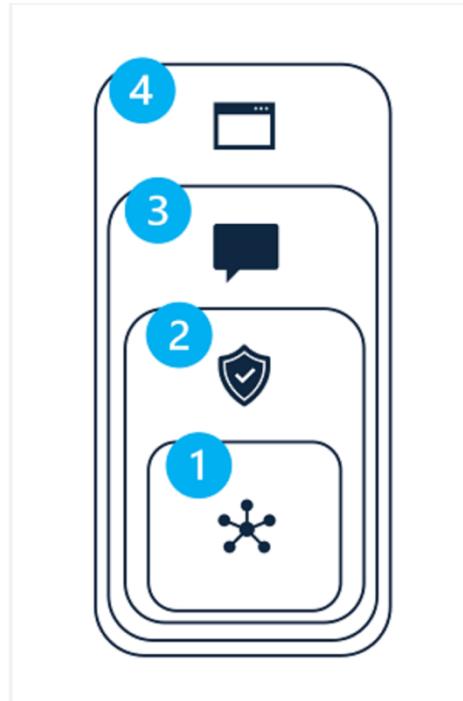
manual untuk memastikan hasil pengujian konsisten dan kriteria evaluasi Anda cukup terdefinisi dengan baik. Kemudian, merancang cara untuk mengotomatiskan pengujian dan pengukuran dengan volume kasus pengujian yang lebih besar. Solusi otomatis dapat mencakup penggunaan model klasifikasi untuk mengevaluasi output secara otomatis.

Bahkan setelah menerapkan pendekatan otomatis untuk menguji dan mengukur bahaya, Anda harus secara berkala melakukan pengujian manual untuk memvalidasi skenario baru dan memastikan bahwa solusi pengujian otomatis berkinerja seperti yang diharapkan.

## Mengurangi Potensi Bahaya

Setelah menentukan garis besar dan cara untuk mengukur output berbahaya yang dihasilkan oleh solusi, Anda dapat mengambil langkah-langkah untuk mengurangi potensi bahaya, dan ketika sesuai mencoba kembali sistem yang dimodifikasi dan membandingkan tingkat bahaya dengan garis besar.

Mitigasi potensi bahaya dalam solusi AI generatif melibatkan pendekatan berlapis, di mana teknik mitigasi dapat diterapkan pada masing-masing dari empat lapisan, seperti yang ditunjukkan di sini:



1. Model
2. Sistem Brankas ty
3. Metaprompt dan grounding
4. Pengalaman pengguna

## 1: Lapisan model

Lapisan model terdiri dari satu atau beberapa model AI generatif di jantung solusi Anda. Misalnya, solusi Anda dapat dibangun di sekitar model seperti GPT-4.

Mitigasi yang dapat Anda terapkan di lapisan model meliputi:

Memilih model yang sesuai untuk penggunaan solusi yang dimaksudkan. Misalnya, meskipun GPT-4 mungkin merupakan model yang kuat dan serbaguna, dalam solusi yang diperlukan hanya untuk mengklasifikasikan input teks kecil dan spesifik, model yang lebih sederhana mungkin memberikan

fungsionalitas yang diperlukan dengan risiko pembuatan konten berbahaya yang lebih rendah.

Menyempurnakan model dasar dengan data pelatihan Anda sendiri sehingga respons yang dihasilkannya lebih mungkin relevan dan terlingkup dalam skenario solusi Anda.

## 2: Lapisan sistem keamanan

Lapisan sistem keamanan mencakup konfigurasi dan kemampuan tingkat platform yang membantu mengurangi bahaya. Misalnya, Azure AI Studio menyertakan dukungan untuk filter konten yang menerapkan kriteria untuk menekan permintaan dan respons berdasarkan klasifikasi konten ke dalam empat tingkat keparahan (aman, rendah, sedang, dan tinggi) untuk empat kategori potensi bahaya (kebencian, seksual, kekerasan, dan bahaya diri).

Mitigasi lapisan sistem keamanan lainnya dapat mencakup algoritma deteksi penyalahgunaan untuk menentukan apakah solusi disalahgunakan secara sistematis (misalnya melalui volume tinggi permintaan otomatis dari bot) dan pemberitahuan peringatan yang memungkinkan respons cepat terhadap potensi penyalahgunaan sistem atau perilaku berbahaya.

## 3: Lapisan metaprompt dan grounding

Lapisan metaprompt dan grounding berfokus pada konstruksi prompt yang dikirimkan ke model. Teknik mitigasi bahaya yang dapat Anda terapkan pada lapisan ini meliputi:

- Menentukan metaprompts atau input sistem yang menentukan parameter perilaku untuk model.
- Menerapkan rekayasa prompt untuk menambahkan data grounding ke perintah input, memaksimalkan kemungkinan output yang relevan dan tidak berbahaya.

- Menggunakan pendekatan pembuatan tertambah pengambilan (RAG) untuk mengambil data kontekstual dari sumber data tepercaya dan menyertakannya dalam perintah.

## 4: Lapisan pengalaman pengguna

Lapisan pengalaman pengguna mencakup aplikasi perangkat lunak tempat pengguna berinteraksi dengan model dan dokumentasi AI generatif atau jaminan pengguna lain yang menjelaskan penggunaan solusi kepada pengguna dan pemangku kepentingannya.

Merancang antarmuka pengguna aplikasi untuk membatasi input ke subjek atau jenis tertentu, atau menerapkan validasi input dan output dapat mengurangi risiko respons yang berpotensi berbahaya.

Dokumentasi dan deskripsi lain dari solusi AI generatif harus transparan dengan tepat tentang kemampuan dan batasan sistem, model yang menjadi dasarnya, dan potensi bahaya yang mungkin tidak selalu ditangani oleh langkah-langkah mitigasi yang telah Anda terapkan.

## Mengoperasikan Solusi AI Generatif Yang Bertanggung Jawab

Setelah Anda mengidentifikasi potensi bahaya, kembangkan cara untuk mengukur kehadiran mereka, dan menerapkan mitigasi untuk mereka dalam solusi Anda, Anda dapat bersiap-siap untuk merilis solusi Anda. Sebelum Anda melakukannya, ada beberapa pertimbangan yang membantu Anda memastikan rilis yang berhasil dan operasi berikutnya.

## Menyelesaikan Tinjauan Prarilis

Sebelum merilis solusi AI generatif, identifikasi berbagai persyaratan kepatuhan di organisasi dan industri Anda dan pastikan tim yang sesuai diberi kesempatan untuk meninjau sistem dan dokumentasinya. Tinjauan kepatuhan umum meliputi:

- Hukum
- Privasi
- Keamanan
- Aksesibilitas

## Merilis dan Mengoperasikan Solusi

Rilis yang sukses membutuhkan beberapa perencanaan dan persiapan. Pertimbangkan panduan berikut:

- Menyusun paket pengiriman bertahap yang memungkinkan Anda merilis solusi awalnya ke grup pengguna yang dibatasi. Pendekatan ini memungkinkan Anda mengumpulkan umpan balik dan mengidentifikasi masalah sebelum merilis ke audiens yang lebih luas.
- Buat rencana respons insiden yang mencakup perkiraan waktu yang diperlukan untuk merespons insiden yang tidak terduga.
- Buat rencana putar kembali yang menentukan langkah-langkah untuk mengembalikan solusi ke status sebelumnya jika insiden terjadi.
- Terapkan kemampuan untuk segera memblokir respons sistem berbahaya saat ditemukan.

- Terapkan kemampuan untuk memblokir pengguna, aplikasi, atau alamat IP klien tertentu jika terjadi penyalahgunaan sistem.
- Terapkan cara bagi pengguna untuk memberikan umpan balik dan melaporkan masalah. Secara khusus, aktifkan pengguna untuk melaporkan konten yang dihasilkan sebagai "tidak akurat", "tidak lengkap", "berbahaya", "menyinggung", atau bermasalah.
- Lacak data telemetri yang memungkinkan Anda menentukan kepuasan pengguna dan mengidentifikasi kesenjangan fungsional atau tantangan kegunaan. Telemetri yang dikumpulkan harus mematuhi undang-undang privasi dan kebijakan dan komitmen organisasi Anda sendiri terhadap privasi pengguna.

## Ringkasan

AI generatif memerlukan pendekatan yang bertanggung jawab untuk mencegah atau mengurangi pembuatan konten yang berpotensi berbahaya. Anda dapat menggunakan proses praktis berikut untuk menerapkan prinsip AI yang bertanggung jawab untuk AI generatif:

1. Identifikasi potensi bahaya yang relevan untuk solusi Anda.
2. Ukur keberadaan bahaya ketika sistem Anda digunakan.
3. Terapkan mitigasi pembuatan konten berbahaya di beberapa tingkat solusi Anda.
4. Sebarkan solusi Anda dengan rencana dan persiapan yang memadai untuk operasi yang bertanggung jawab.

# MATERI 4

# MENERAPKAN

# REKAYASA PROMPT

Diadopsi dari Materi Menerapkan rekayasa prompt dengan Azure OpenAI Service (Microsoft Learn)



## Materi 4: Menerapkan Rekayasa Prompt

### Pendahuluan

Kualitas respons dari model AI di Azure OpenAI tergantung pada kualitas prompt yang disediakan. Meningkatkan kualitas prompt melalui berbagai teknik disebut rekayasa yang cepat.

Dalam modul ini, Anda akan mempelajari tentang rekayasa yang cepat dan bagaimana rekayasa tersebut dapat digunakan untuk mengoptimalkan performa model Azure OpenAI. Rekayasa prompt melibatkan merancang dan mengoptimalkan permintaan untuk memanfaatkan model Azure OpenAI dengan lebih baik.

Video pengayaan dan penjelasan lebih lanjut terkait materi ini dapat diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/modules/apply-prompt-engineering-azure-openai/>



### Memahami Rekayasa Prompt

Kualitas permintaan input yang kami kirim ke model AI, seperti yang tersedia di Azure OpenAI, secara langsung memengaruhi kualitas apa yang kita dapatkan kembali. Dengan membangun permintaan yang kami kirim ke model dengan hati-hati, model dapat memberikan respons yang lebih baik dan lebih menarik.

### Apa Itu Rekayasa Perintah

Video penjelasan awal terkait materi ini dapat diakses pada tautan berikut:

<https://learn.microsoft.com/id-id/training/modules/apply-prompt-engineering-azure-openai/2-understand-prompt-engineering>



Rekayasa prompt adalah proses merancang dan mengoptimalkan permintaan untuk memanfaatkan model AI dengan lebih baik. Merancang permintaan yang efektif sangat penting untuk keberhasilan rekayasa yang cepat, dan dapat secara signifikan meningkatkan performa model AI pada tugas tertentu. Memberikan perintah yang relevan, spesifik, tidak ambigu, dan terstruktur dengan baik dapat membantu model lebih memahami konteks dan menghasilkan respons yang lebih akurat.

Misalnya, jika kita ingin model OpenAI menghasilkan deskripsi produk, kita dapat memberikannya dengan deskripsi terperinci yang menjelaskan fitur dan manfaat produk. Dengan menyediakan konteks ini, model dapat menghasilkan deskripsi produk yang lebih akurat dan relevan.

Rekayasa prompt juga dapat membantu mengurangi bias dan meningkatkan kewajaran dalam model AI. Dengan merancang permintaan yang beragam dan inklusif, kami dapat memastikan bahwa model tidak bias terhadap grup atau perspektif tertentu.

 **Penting**

Tidak peduli seberapa baik permintaan yang dapat Anda desain, respons dari model AI tidak boleh diambil sebagai fakta atau sepenuhnya bebas dari bias. Selalu gunakan AI secara bertanggung jawab. Untuk informasi selengkapnya, lihat Catatan transparansi Microsoft [tentang prinsip AI Azure OpenAI dan](#) Microsoft.

Selain itu, rekayasa prompt dapat membantu kami memahami referensi mana yang digunakan model untuk menghasilkan responsnya. Model AI generatif memiliki banyak parameter dan

logika yang diikutinya sebagian besar tidak diketahui oleh pengguna, sehingga dapat membingungkan bagaimana ia tiba pada respons yang diberikannya. Dengan merancang permintaan yang mudah dipahami dan ditafsirkan, kita dapat membantu manusia lebih memahami bagaimana model menghasilkan responsnya. Ini bisa sangat penting dalam domain seperti layanan kesehatan, di mana sangat penting untuk memahami bagaimana model membuat keputusan.

Ada berbagai metode untuk digunakan ketika merekayasa perintah Anda sendiri, banyak di antaranya tercakup dalam unit modul ini yang akan datang. Ini termasuk memberikan instruksi, konten kontekstual, isyarat, atau contoh beberapa bidikan, dan mengurutkan konten dengan benar dalam perintah Anda. Metode yang dibahas di sini tidak lengkap karena area ini bermuansa dan topik cairan.

## Pertimbangan Untuk Titik Akhir API

Contoh di sisa modul ini akan difokuskan pada ChatCompletion. Perlu dicatat bahwa ChatCompletion juga dapat digunakan untuk skenario non-obrolan, di mana instruksi apa pun disertakan dalam pesan sistem dan konten pengguna disediakan dalam pesan peran pengguna. Sebagian besar contoh ini dapat diubah untuk menggunakan Completion titik akhir, jika diinginkan.

Dalam hal ketersediaan model, Completion titik akhir dapat digunakan dengan gpt-3 dan sebelumnya, dan ChatCompletion dapat digunakan dengan gpt-35-turbo model dan yang lebih baru

## Menyesuaikan Parameter Model

Selain teknik yang dibahas dalam modul ini, menyesuaikan parameter model dapat berdampak signifikan pada respons.

Secara khusus, temperature dan top\_p (top\_probability) adalah yang paling mungkin berdampak pada respons model karena keduanya mengontrol keacakan dalam model, tetapi dengan cara yang berbeda.

Nilai yang lebih tinggi menghasilkan respons yang lebih kreatif dan acak, tetapi kemungkinan akan kurang konsisten atau fokus. Respons yang diharapkan menjadi keuntungan fiktif atau unik dari nilai yang lebih tinggi untuk parameter ini, sedangkan konten yang diinginkan untuk menjadi lebih konsisten dan konkret harus menggunakan nilai yang lebih rendah.

Coba sesuaikan parameter ini dengan perintah yang sama untuk melihat dampaknya terhadap respons. Disarankan untuk mengubah salah satu temperature atau top\_p pada satu waktu, tetapi tidak keduanya.

## Menulis Perintah Yang Lebih Efektif

Model Azure OpenAI mampu menghasilkan respons terhadap kueri bahasa alami dengan akurasi yang luar biasa. Namun, kualitas respons sangat tergantung pada seberapa baik perintah ditulis. Pengembang dapat mengoptimalkan performa model Azure OpenAI dengan menggunakan teknik yang berbeda dalam perintah mereka, menghasilkan respons yang lebih akurat dan relevan.

### Berikan Instruksi Yang Jelas

Meminta model Azure OpenAI dengan jelas untuk apa yang Anda inginkan adalah salah satu cara untuk mendapatkan hasil yang diinginkan. Dengan sedeskriptif mungkin, model dapat menghasilkan respons yang paling cocok dengan apa yang Anda cari.

Misalnya, Anda ingin membuat deskripsi produk untuk botol air baru. Pertimbangkan jawaban yang terkait dengan setiap perintah:

Prompt

code Menyalin

```
Write a product description for a new water bottle
```

Respons

code Menyalin

```
Introducing the latest addition to our product line – the innovative and eco-friendly water bottle. Made from high-quality, BPA-free materials, this bottle is designed to keep your beverages at the perfect temperature for hours. Its sleek and stylish design makes it the perfect accessory for any outdoor activity or workout session. With its convenient carrying handle and leak-proof lid, you can take it with you wherever you go, without worrying about spills or leaks. Plus, its easy-to-clean design ensures that you can use it again and again, making it the perfect choice for anyone looking for a reliable and sustainable water bottle.
```

Versus permintaan serupa, dengan instruksi yang jelas.

Prompt

code Menyalin

```
Write a product description for a new water bottle that is 100% recycled. Be sure to include that it comes in natural colors with no dyes, and each purchase removes 10 pounds of plastic from our oceans
```

Respons

code Menyalin

```
Introducing our newest water bottle, the eco-friendly choice for the conscious consumer. Made from 100% recycled materials and available in natural colors with no dyes, this bottle is not only stylish but also sustainable. Each purchase of this bottle removes 10 pounds of plastic from our oceans, making it a responsible choice for those who care about the environment. Its durable and leak-proof design makes it perfect for outdoor activities, while its easy-to-carry handle makes it convenient to take on the go. Join us in our mission to reduce plastic waste and choose this eco-friendly water bottle for your daily hydration needs.
```

Termasuk spesifik yang perlu disertakan memungkinkan model mengetahui informasi apa yang akan ditambahkan ke deskripsi, membuatnya lebih akurat ke produk baru.

Metode ini dapat diekstrapolasi untuk menyertakan instruksi kompleks, seperti daftar detail berpoin yang akan disertakan, panjang respons, atau format yang diinginkan untuk disertakan dalam output. Coba tanyakan dengan tepat apa yang ingin Anda lihat dalam hasilnya, dan Anda mungkin terkejut dengan seberapa baik model memenuhi permintaan Anda.

## Format Instruksi

Bagaimana instruksi diformat dapat memengaruhi bagaimana model menginterpretasikan perintah. Bias rekayasa dapat memengaruhi model, di mana informasi yang terletak di akhir prompt dapat memiliki pengaruh lebih besar pada output daripada informasi di awal. Anda mungkin mendapatkan respons yang lebih baik dengan mengulangi instruksi di akhir perintah dan menilai bagaimana hal itu memengaruhi respons yang dihasilkan.

Bias rekensi ini juga dapat dimainkan saat menggunakan ChatCompletion dalam skenario obrolan, di mana pesan yang lebih baru dalam percakapan yang disertakan dalam prompt memiliki dampak yang lebih besar pada respons. Unit berikutnya lebih mendalam dalam menggunakan percakapan untuk meningkatkan kualitas respons, tetapi menempatkan informasi penting lebih dekat ke akhir perintah dapat menghasilkan respons yang lebih baik.

## Menggunakan Penanda Bagian

Teknik khusus untuk memformat instruksi adalah memisahkan instruksi di awal atau akhir perintah, dan memiliki konten pengguna yang ada di dalam --- atau ### blok. Tag ini memungkinkan model untuk membedakan antara instruksi dan konten dengan lebih jelas. Contoh:

```
code
Translate the text into French
---
What's the weather going to be like today?
---
```

Menyalin

① Catatan

Praktik terbaik untuk penanda bagian dapat berubah dengan versi mendatang.

## Konten Primer, Pendukung, dan Grounding

Termasuk konten untuk digunakan model untuk merespons dengan memungkinkannya menjawab dengan akurasi yang lebih besar. Konten ini dapat dipikirkan dengan dua cara: konten utama dan pendukung.

Konten utama mengacu pada konten yang merupakan subjek kueri, seperti kalimat untuk diterjemahkan atau artikel untuk dirangkum. Konten ini sering disertakan di awal atau akhir perintah (sebagai instruksi dan dibingkai oleh --- blok), dengan instruksi yang menjelaskan apa yang harus dilakukan dengannya.

Misalnya, kami memiliki artikel panjang yang ingin kami ringkas. Kita bisa memasukkannya ke dalam --- blok di prompt, lalu diakhiri dengan instruksi.

```
code Menyalin
---
<insert full article here, as primary content>
---
Summarize this article and identify three takeaways in a bulleted list
```

Konten pendukung adalah konten yang dapat mengubah respons, tetapi bukan fokus atau subjek permintaan. Contoh konten pendukung mencakup hal-hal seperti nama, preferensi, tanggal mendatang untuk disertakan dalam respons, dan sebagainya. Menyediakan konten pendukung memungkinkan model merespons lebih lengkap, akurat, dan lebih cenderung menyertakan informasi yang diinginkan.

Misalnya, mengingat email promosi yang sangat panjang, model ini dapat mengekstrak informasi utama. Jika Anda kemudian menambahkan konten pendukung ke perintah yang menentukan sesuatu yang spesifik yang Anda cari, model dapat memberikan respons yang lebih berguna. Dalam hal ini email

adalah konten utama, dengan spesifikasi apa yang Anda minati sebagai konten pendukung

code

Menyalin

```
---  
<insert full email here, as primary content>  
---  
<the next line is the supporting content>  
Topics I'm very interested in: AI, webinar dates, submission deadlines
```

Extract the key points from the above email, and put them in a bulleted list:

Konten grounding memungkinkan model untuk memberikan jawaban yang dapat diandalkan dengan menyediakan konten bagi model untuk menarik jawaban. Konten grounding bisa menjadi esai atau artikel yang kemudian Anda ajukan pertanyaan tentang, dokumen FAQ perusahaan, atau informasi yang lebih baru daripada data yang dilatih model. Jika Anda memerlukan respons yang lebih andal dan saat ini, atau Anda perlu mereferensikan informasi yang tidak diterbitkan atau spesifik, konten grounding sangat disarankan.

Konten grounding berbeda dari konten utama karena merupakan sumber informasi untuk menjawab kueri permintaan, alih-alih konten yang dioperasikan untuk hal-hal seperti ringkasan atau terjemahan. Misalnya, ketika diberikan makalah penelitian yang tidak diterbitkan tentang riwayat AI, maka dapat menjawab pertanyaan menggunakan konten dasar tersebut.

code

Menyalin

```
---  
<insert unpublished paper on the history of AI here, as grounding content>  
---
```

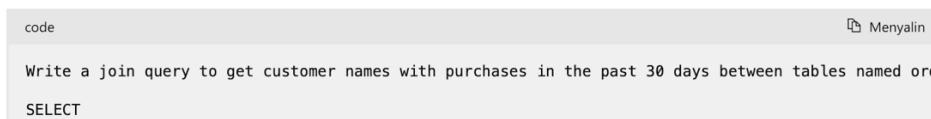
Where and when did the field of AI start?

Data grounding ini memungkinkan model untuk memberikan jawaban yang lebih akurat dan terinformasi yang mungkin bukan bagian dari himpunan data tempat model dilatih.

## Isyarat

Istik adalah kata-kata terkemuka untuk dibangun model, dan sering membantu membentuk respons ke arah yang benar. Mereka sering digunakan dengan instruksi, tetapi tidak selalu. Isensi sangat membantu jika meminta model untuk pembuatan kode. Model Azure OpenAI saat ini dapat menghasilkan beberapa cuplikan kode yang menarik, namun pembuatan kode akan dibahas lebih mendalam dalam modul di masa mendatang.

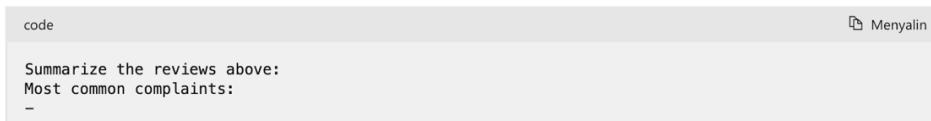
Misalnya, jika Anda ingin membantu membuat kueri SQL, berikan instruksi tentang apa yang Anda butuhkan bersama dengan awal kueri:



```
code Menyalin
Write a join query to get customer names with purchases in the past 30 days between tables named ord
SELECT
```

Respons model mengambil di mana perintah ditinggalkan, berlanjut di SQL, meskipun kami tidak pernah meminta bahasa tertentu. Contoh lain bisa untuk membantu kode python, dengan memberikan komentar kode tentang aplikasi yang diinginkan dan termasuk import sebagai kata depan di akhir perintah, atau serupa dalam bahasa yang Anda inginkan.

Contoh lain, mengingat banyak koleksi ulasan pelanggan dalam perintah, dan diakhiri dengan:



```
code Menyalin
Summarize the reviews above:
Most common complaints:
-
```

Model kemudian tahu untuk menyelesaikan pernyataan berdasarkan konteks yang disediakan dalam ulasan.

# Memberikan Konteks Untuk Meningkatkan Akurasi

Memilih bahasa pengembangan Anda

C# Python

Dengan memberikan konteks ke model AI, model memungkinkan untuk lebih memahami apa yang Anda minta atau apa yang harus diketahui untuk memberikan jawaban terbaik. Konteks dapat disediakan dalam beberapa cara.

## Meminta Komposisi Output

Menentukan struktur output dapat berdampak besar pada hasil Anda. Struktur dapat mencakup sesuatu seperti meminta model untuk mengutip sumber, menulis respons sebagai email, memformat respons sebagai kueri SQL, mengklasifikasikan sentimen ke dalam struktur tertentu, dan sebagainya. Contohnya:

Prompt

code

Write a table in markdown with 6 animals in it, with their genus and species

## Respons

code Menyalin

Animal	Genus	Species
Red Panda	Ailurus	fulgens
African Elephant	Loxodonta	africana
Snow Leopard	Panthera	uncia
Arctic Fox	Vulpes	lagopus
Chimpanzee	Pan	troglodytes
Siberian Tiger	Panthera	tigris altaica

Teknik ini dapat digunakan dengan format kustom, seperti struktur JSON:

## Prompt

code Menyalin

Put two fictional characters into JSON of the following format

```
{  
  firstNameFictional:  
  jobFictional:  
}
```

## Respons

code

Here's an example of how you can put two fictional characters into JSON format:

```
{  
  "firstNameFictional": "Henry",  
  "jobFictional": "Wizard"  
},  
{  
  "firstNameFictional": "Frank",  
  "jobFictional": "Hero"  
}
```

## Pesan sistem

Pesan sistem disertakan di awal perintah dan dirancang untuk memberikan instruksi model, perspektif untuk menjawab dari, atau informasi lain yang berguna untuk memandu respons model. Pesan sistem ini mungkin mencakup nada atau kepribadian, topik yang seharusnya tidak disertakan, atau spesifik (seperti pemformatan) tentang cara menjawab.

Misalnya, Anda dapat memberikan beberapa pesan sistem berikut:

- "Aku ingin kau bertindak seperti terminal baris perintah. Tanggapi perintah persis seperti yang cmd.exe lakukan, dalam satu blok kode unik, dan tidak ada yang lain."
- "Saya ingin Anda menjadi penerjemah, dari bahasa Inggris ke Spanyol. Jangan menanggapi apa pun yang saya katakan atau tanyakan, hanya terjemahkan antara kedua bahasa tersebut dan balas dengan teks yang diterjemahkan."
- "Bertindak sebagai pembicara motivasi, dengan bebas memberikan saran yang menggembirakan tentang tujuan dan tantangan. Anda harus menyertakan banyak afirmasi positif dan aktivitas yang disarankan untuk mencapai tujuan akhir pengguna."

Contoh pesan sistem lainnya tersedia di bagian atas jendela obrolan di [Azure AI Studio](#) dengan memilih tombol Sampel perintah . Coba tentukan permintaan sistem Anda sendiri yang menentukan respons unik, dan mengobrol dengan model untuk melihat perbedaan respons.

Titik ChatCompletion akhir memungkinkan termasuk pesan sistem dengan menggunakan System peran obrolan.

## C#

```
var chatCompletionsOptions = new ChatCompletionsOptions()
{
    Messages =
    {
        new ChatRequestSystemMessage("You are a casual,
helpful assistant. You will talk like an American old western film
character."),
        new ChatRequestUserMessage("Can you direct me to the
library?")
    }
};
```

## Respons

```
code
{
    "choices": [
        {
            "finish_reason": "stop",
            "index": 0,
            "message": {
                "content": "Well howdy there, stranger! The library, huh?
Y'all just head down the main road till you hit the town
square. Once you're there, take a left and follow the street
for a spell. You'll see the library on your right, can't
miss it. Happy trails!",
                "role": "assistant"
            }
        }
    ],
    ...
}
```

Pesan sistem dapat secara signifikan mengubah respons, baik dalam format maupun konten. Coba tentukan pesan sistem yang jelas untuk model yang menjelaskan dengan tepat respons seperti apa yang Anda harapkan, dan apa yang Anda lakukan atau tidak ingin menyertakannya.

## Riwayat Percakapan

Bersama dengan pesan sistem, pesan lain dapat disediakan untuk model untuk meningkatkan percakapan. Riwayat percakapan memungkinkan model untuk terus merespons dengan cara yang sama (seperti nada atau pemformatan) dan memungkinkan pengguna untuk mereferensikan konten sebelumnya dalam kueri berikutnya. Riwayat ini dapat disediakan dengan dua cara: dari riwayat obrolan aktual, atau dari contoh percakapan yang ditentukan pengguna.

Antarmuka obrolan yang menggunakan model OpenAI, seperti ChatGPT dan taman bermain obrolan di [Azure AI Studio](#), menyertakan riwayat percakapan secara otomatis yang menghasilkan percakapan yang lebih kaya dan lebih bermakna. Di bagian Parameter dari taman bermain obrolan, Anda dapat menentukan berapa banyak pesan sebelumnya yang ingin Anda sertakan. Coba kurangi menjadi 1 atau tingkatkan hingga maks untuk melihat bagaimana berbagai jumlah riwayat memengaruhi percakapan.

### Catatan

Lebih banyak riwayat percakapan yang disertakan dalam perintah berarti jumlah token input yang lebih besar digunakan. Anda harus menentukan saldo yang benar untuk kasus penggunaan Anda, mengingat batas token model yang Anda gunakan.

Sistem obrolan juga dapat menggunakan kemampuan ringkasan model untuk menghemat token input. Aplikasi dapat

memilih untuk meringkas pesan sebelumnya dan menyertakan ringkasan tersebut dalam riwayat percakapan, lalu hanya menyediakan beberapa pesan terakhir verbatim ke model.

## Beberapa Pembelajaran Bidikan

Menggunakan contoh percakapan yang ditentukan pengguna adalah apa yang disebut beberapa pembelajaran bidikan, yang menyediakan contoh model tentang bagaimana percakapan harus merespons kueri tertentu. Contoh-contoh ini berfungsi untuk melatih model cara merespons.

Misalnya, dengan memberikan model beberapa perintah dan respons yang diharapkan, model berlanjut dalam pola yang sama tanpa menginstruksikan apa yang harus dilakukan:

code

```
User: That was an awesome experience
Assistant: positive
User: I won't do that again
Assistant: negative
User: That was not worth my time
Assistant: negative
User: You can't miss this
Assistant:
```

Jika model disediakan hanya You can't miss this tanpa konteks tambahan dari beberapa pembelajaran bidikan, responsnya kemungkinan tidak akan berguna.

Dalam istilah praktis, riwayat percakapan dan beberapa pembelajaran bidikan dikirim ke model dengan cara yang sama; setiap pesan pengguna dan respons asisten adalah pesan diskrit dalam objek pesan. Titik ChatCompletion akhir dioptimalkan untuk menyertakan riwayat pesan, terlepas dari

apakah riwayat pesan ini disediakan sebagai beberapa pembelajaran bidikan, atau riwayat percakapan aktual.

C#

Menyalin

```
var chatCompletionsOptions = new ChatCompletionsOptions()
{
    Messages =
    {
        new ChatRequestSystemMessage("You are a helpful assistant."),
        new ChatRequestUserMessage("That was an awesome experience"),
        new ChatRequestAssistantMessage("positive"),
        new ChatRequestUserMessage("I won't do that again"),
        new ChatRequestAssistantMessage("negative"),
        new ChatRequestUserMessage("That was not worth my time"),
        new ChatRequestAssistantMessage("negative"),
        new ChatRequestUserMessage("You can't miss this")
    }
};
```

## Memecah tugas yang kompleks

Teknik lain untuk interaksi yang ditingkatkan adalah membagi prompt kompleks menjadi beberapa kueri. Ini memungkinkan model untuk lebih memahami setiap bagian individu, dan dapat meningkatkan akurasi keseluruhan. Membagi lagi perintah Anda juga memungkinkan Anda untuk menyertakan respons dari perintah sebelumnya dalam permintaan di masa mendatang, dan menggunakan informasi tersebut selain kemampuan model untuk menghasilkan respons yang menarik.

Misalnya, Anda dapat meminta model Doug can ride down the zip line in 30 seconds, and takes 5 minutes to climb back up to the top. How many times can Doug ride the zip line in 17 minutes?. Hasilnya kemungkinan 3, yang jika Doug dimulai di bagian atas baris zip salah.

Jawaban yang lebih informatif dapat berasal dari mengajukan beberapa pertanyaan, tentang waktu pulang pergi untuk kembali ke bagian atas baris zip, dan bagaimana

memperhitungkan fakta bahwa Doug mulai di bagian atas. Memecah masalah ini mengungkapkan bahwa Doug bisa, pada kenyataannya, naik jalur zip empat kali.

## Rantai Pemikiran

Salah satu metode yang berguna untuk membantu Anda memecah tugas Anda secara efektif adalah meminta model untuk menjelaskan rantai pemikirannya.

Meminta model untuk merespons dengan proses langkah demi langkah yang menentukan responsnya adalah cara yang berguna untuk memahami bagaimana model menafsirkan prompt. Dengan melakukan demikian, Anda dapat melihat di mana model tersebut membuat arah logis yang salah dan lebih memahami cara mengubah prompt Anda untuk menghindari kesalahan. Teknik ini dapat mencakup memintanya untuk mengutip sumbernya, seperti yang dilakukan obrolan Bing (yang menggunakan model generasi GPT-4), dan memberikan alasan mengapa ia menentukan jawabannya.

Rantai teknik pemikiran yang mendorong paling baik digunakan untuk membantu Anda melakukan iterasi dan peningkatan pada permintaan Anda untuk mendapatkan jawaban berkualitas tertinggi dari model.

Misalnya, meminta model What sport is easiest to learn but hardest to master? menghasilkan respons dengan jawaban, dan penjelasan kecil tentang alasannya. Namun, ketika diminta dengan What sport is easiest to learn but hardest to master? Give a step by step approach of your thoughts, ending in your answer, respons adalah penjelasan lengkap tentang bagaimana ia tiba pada jawabannya.

code

Sure, here's a step-by-step approach to answering the question:

1. Identify sports that are popular and have a large following.
2. Consider the basic mechanics of each sport and how easy they are to learn.
3. Think about the various techniques and skills required to play each sport at a high level, and how difficult they are to master.
4. Compare the relative ease of learning each sport to the difficulty of mastering it.
5. Based on the above analysis, determine which sport is easiest to learn but hardest to master.

Using this approach, I would say that the sport of golf is easiest to learn but hardest to master. Here's why:

1. Golf is a popular sport with a large following.
2. The basic mechanics of hitting a golf ball are relatively simple and can be learned quickly with some practice.
3. However, achieving consistency and mastering the various shots and techniques required to play at a high level can take years of practice and dedication.
4. Compared to other sports, the difference between a beginner and an intermediate golfer is relatively small, while the difference between an intermediate and an advanced golfer is much larger.
5. Therefore, I would say that golf is the sport that is easiest to learn but hardest to master.

Mengetahui langkah-langkah alasan model yang digunakan untuk membentuk responsnya memungkinkan kita mengidentifikasi di mana kita perlu membagi permintaan, memberikan informasi lebih lanjut, atau mengklarifikasi instruksi.

## Ringkasan

Dalam modul ini, Anda belajar tentang rekayasa yang cepat dan bagaimana rekayasa tersebut dapat digunakan untuk mengoptimalkan performa model Azure OpenAI. Rekayasa prompt melibatkan merancang dan mengoptimalkan perintah untuk memanfaatkan model AI dengan lebih baik. Dengan memberikan perintah yang relevan, spesifik, tidak ambigu, dan terstruktur dengan baik, model dapat lebih memahami konteks dan menghasilkan respons yang lebih akurat.

Takeaway utama dari modul ini adalah bahwa rekayasa prompt adalah proses penting untuk mengoptimalkan performa model Azure OpenAI. Memberikan instruksi yang jelas, meminta komposisi output, menyesuaikan parameter model, dan menggunakan konten kontekstual semuanya dapat membantu meningkatkan akurasi dan relevansi respons model. Penting untuk menggunakan AI secara bertanggung jawab dan untuk selalu diingat bahwa respons dari model AI tidak boleh diambil sebagai fakta atau bebas dari bias.

Bacaan Tambahan:

- [Prinsip AI Microsoft](#)
- [Catatan transparansi di Azure OpenAI](#)
- [Dokumentasi Azure OpenAI tentang rekayasa prompt](#)
- [Azure AI Studio](#)

