

FIAP

NABA



COGNITIVE ENVIRONMENTS

DATA SCIENCE & AI MBA



MODELO GPT

Generative Pre-trained Transformers ou GPT é um tipo de modelo de linguagem em larga escala (ou simplesmente LLM de large language models). Assim como o GPT existem inúmeros outros como **Bard** (Google), **Claude** (Anthropic), **Titan** (AWS), **Llama** (Meta) e até mesmo brasileiro como o Maritaca, dentre inúmeros que estão sendo lançados a cada mês.

Foi treinado utilizando um grande número de parâmetros de dados (175 bilhões no GPT3 e 100 trilhões no GPT4).

Os dados de treinamento reúnem sites de internet incluindo Wikipedia e livros.

Da mesma forma que o Google indexa sites para o mecanismo de busca, os modelos de LLM fazem o mesmo para gerar dados de treinamento.

O arquivo “robots.txt” na raiz dos sites controlam o acesso de diversos bots de LLMs, no caso do GPT o agente é “GPTBot”, do Bard (Google) é “Google-Extended”.

O QUE A AVALIAR NUM MODELO **GPT?**

NÚMERO DE PARÂMETROS

Parâmetros utilizados durante o treinamento de um determinado modelo.

Quanto mais parâmetros, mais robusto o modelo tende a ser.

Para tarefas específicas não podemos deduzir que o número de parâmetros crescente em performance. Por isso a avaliação humana se faz necessária.

JANELA DE TOKENS

Quantidade de tokens que o modelo é capaz de processar na entrada.

Com a evolução das técnicas de RAG (Retrieval-Augmented Generation), da qual enviamos partes de conhecimento para o GPT utilizar como base. Isso requer uma janela cada vez maior, pois modelos com mais parâmetros possuem melhores resultados quando comparamos treinar um novo modelo ou um ajuste fino somente para novas informações.

AVALIAÇÃO HUMANA

Os resultados de modelos de GPT são complexos e querer, para avaliações mais amplas, feedback humano.

Métricas como perplexidade (habilidade de prev), BLEU (Bilingual Evaluation Understudy) para avaliação de texto comparando com referências humanas e ROUGE (Recall-Oriented Understudy for Gisting Evaluation) usado para avaliar a qualidade de resumos automáticos. er a próxima palavra

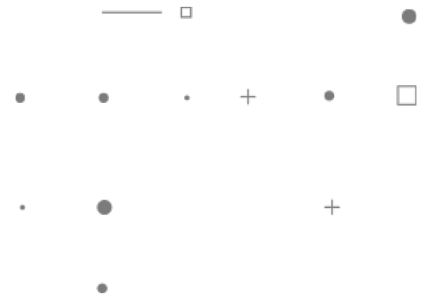


RANKING DOS PRINCIPAIS MODELOS

O ranking considerou principalmente precisão geral em benchmarks padronizados, capacidades multimodais, janela de contexto disponível e velocidade na geração de respostas.

	Modelo	Empresa	Parâmetros	Janela de Contexto	Desempenho Geral	Velocidade	Observações
🥇 1º	GPT-4o	OpenAI	~1T (estimado)	128.000 tokens	65% (LMSYS)	143 tokens/s	Multimodal nativo, alta precisão
🥈 2º	Claude 3 Opus	Anthropic	~130B	200.000 tokens	61,3% (Vellum)	78 tokens/s	Excelência em tarefas longas e análise de documentos
🥉 3º	Gemini 2.5 Pro	Google	Não divulgado	1.000.000 tokens	92% (AIME 2024)	191 tokens/s	Forte em tarefas multimodais e de raciocínio
4º	LLaMA 3 400B	Meta	405B	8.000 tokens	86,1% (MMLU)	Não divulgado	Desempenho competitivo em benchmarks de matemática e conhecimento geral
5º	GPT-4 Turbo	OpenAI	~1T (estimado)	128.000 tokens	88,4% (MMLU)	73 tokens/s	Versão otimizada do GPT-4 com melhor custo-benefício
6º	Claude 3 Sonnet	Anthropic	~130B	200.000 tokens	88,3% (MMLU)	Não divulgado	Equilíbrio entre desempenho e custo
7º	Gemini 1.5 Pro	Google	Não divulgado	1.000.000 tokens	81,9% (MMLU)	Não divulgado	Antecessor do Gemini 2.5, ainda relevante em tarefas específicas
8º	LLaMA 3 70B	Meta	70B	8.000 tokens	82% (MMLU)	Não divulgado	Modelo de código aberto com bom desempenho geral

Desempenho Geral: Refere-se à média de acertos em benchmarks padronizados, como MMLU (Massive Multitask Language Understanding) e AIME (American Invitational Mathematics Examination).



MODELOS DE BASE

[Amazon Bedrock](#) > Modelos de base

Modelos de base

[Informações](#)

[Comprar throughput provisionada](#) [Abrir no playground](#)

[Agrupar por família ...](#)

▼

Claude | by Anthropic (4)

O Claude é um assistente de IA de próxima geração baseada na pesquisa da Anthropic sobre o treinamento de sistemas de IA úteis, honestos e inofensivos.

[Visualizar detalhes do fornecedor](#)

Claude v2.1

Anthropic | Text Modelo | Máx. 200k tokens

Descrição

Uma atualização do Claude 2 que apresenta o dobro da janela de contexto, além de melhorias em confiabilidade, taxas de alucinação e precisão baseadas em evidências em documentos longos e contextos RAG.

Atributos do modelo

Geração de texto, Conversa, Raciocínios e análise complexos

Claude v2

Anthropic | Text Modelo | Máx. 100k tokens

Descrição

O modelo de base mais poderoso da Anthropic, ele se destaca em uma ampla variedade de tarefas, desde diálogos sofisticados e geração de conteúdo criativo até instruções detalhadas.

Atributos do modelo

Geração de texto, Conversa, Raciocínios e análise complexos

Claude v1.3


Anthropic | Text Modelo | Máx. 100k tokens

Descrição

Uma versão anterior dos grandes modelos de linguagem de uso geral da Anthropic.

Atributos do modelo

Geração de texto, Conversa

 Legada

Claude Instant v1.2

Anthropic | Text Modelo | Máx. 100k tokens

Descrição

Um modelo mais rápido e barato, mas ainda assim muito capaz, que pode lidar com uma variedade de tarefas, incluindo diálogo casual, análise de texto, sumarização e resposta a perguntas de documentos.

Atributos do modelo

Geração de texto, Conversa

ACESSO AOS MODELOS DE BASE

A AWS oferece tanto os modelos próprios (Titan) quanto outros no modelo de Marketplace que já existe na plataforma para outros serviços.

Para utilizar os modelos de outros fornecedores, como Anthropic, AI21, etc. é necessário solicitação prévia que dependendo do modelo é liberado imediatamente.

Para acessar as permissões, conceder ou revogar vá em “Acesso a modelo”, no painel à esquerda da plataforma.

▼ Conceitos básicos

Visão geral

Exemplos

Provedores

▼ Modelos de base

[Modelos de base](#)

Modelos personalizados

▼ Playgrounds

Chat

Texto

Imagem

▼ Orquestração

Base de conhecimento

Agentes

▼ Avaliação e implantação

Avaliação do modelo

[Pré-visualização](#)

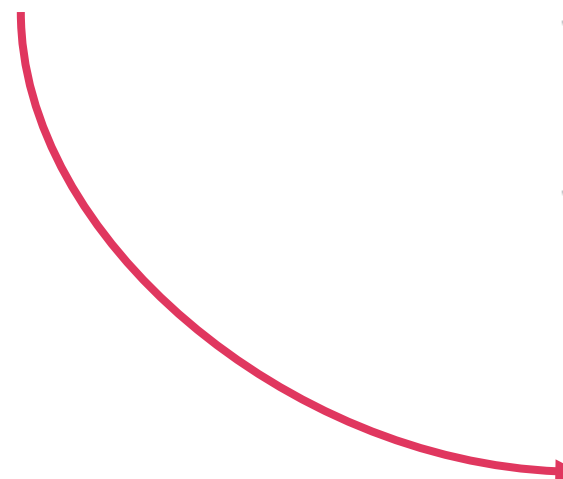
Throughput provisionada

Acesso a modelo

Configurações

Guia do usuário [↗](#)

Termos de Serviço do Bedrock [↗](#)



ACESSO AOS MODELOS DE BASE

Antes de liberar acessos do IAM, o modelo precisa ser contratado pelo Marketplace. Essa contratação é ativada pela liberação de acesso abaixo.

A precificação é por uso de token, comum de mercado começado pela OpenAI.

[Amazon Bedrock](#) > Acesso a modelo

Acesso a modelo

Para usar o Bedrock, é necessário solicitar acesso aos FMs do Bedrock. Para fazer isso, você precisará ter o [Permissões do IAM](#) correto. Para determinados modelos, talvez seja necessário primeiro enviar os detalhes do caso de uso antes de solicitar acesso. Informações adicionais sobre esses modelos estão disponíveis na página [Provedores](#).

Modelos de base (19)

Gerenciar acesso a modelos

Modelos	Status de acesso	Modalidade	EULA
<div>AI21 Labs</div>			
– Jurassic-2 Ultra	✓ Acesso concedido	Texto	EULA
– Jurassic-2 Mid	✓ Acesso concedido	Texto	EULA
<div>Amazon</div>			
– Titan Embeddings G1 - Text	✓ Acesso concedido	Incorporação	EULA
– Titan Text G1 - Lite	✓ Acesso concedido	Texto	EULA
– Titan Text G1 - Express	✓ Acesso concedido	Texto	EULA
– Titan Image Generator G1 Pré-visualização	✓ Acesso concedido	Imagem	EULA
– Titan Multimodal Embeddings G1	⊖ Disponível para solicitação	Incorporação	EULA
<div>Anthropic</div>			
– Claude	⊖ Detalhes do caso de uso necessários	Texto	EULA
– Claude Instant	⊖ Detalhes do caso de uso necessários	Texto	EULA
<div>Cohere</div>			
– Command	✓ Acesso concedido	Texto	EULA
– Command Light	✓ Acesso concedido	Texto	EULA
– Embed English	⊖ Disponível para solicitação	Incorporação	EULA
– Embed Multilingual	⊖ Disponível para solicitação	Incorporação	EULA

PLAYGROUNDS DE MODELOS DE IMAGEM

Para experimentar os diferentes modelos, é recomendável testá-los antes nos playgrounds disponíveis. Voltados para chat, texto ou imagem, neles é possível testar diferentes configurações e ter acesso a especificação de payload dos mesmos ao utilizá-los integrados no AWS SDK.

Amazon Bedrock > Playground de imagens

Playground de imagens

Carregar exemplos

Titan Image Generator G1 v1

Alterar



Mostre a capital de um centro de pesquisa espacial numa realidade alternativa da qual o Brasil fosse uma hegemonia militar, científica e espacial.

Executar

Configurações

Redefinir

▼ Modo

Generate Edit

▼ Prompt negativo

desenho
pessoas
foto

▼ Imagem de inferência

Carregar imagem

▼ Imagem de resposta

Qualidade

Padrão

Orientação

☒ Paisagem ☐ Retrato

Tamanho

512 x 512

Número de imagens

3

▼ Configurações avançadas

PLAYGROUNDS DE MODELOS DE IMAGEM

Os modelos de imagem possuem algumas opções que variam de cada um deles, ou seja, nem todos possuem:

Prompt negativo: termos e palavras que não devem ser consideradas na geração.

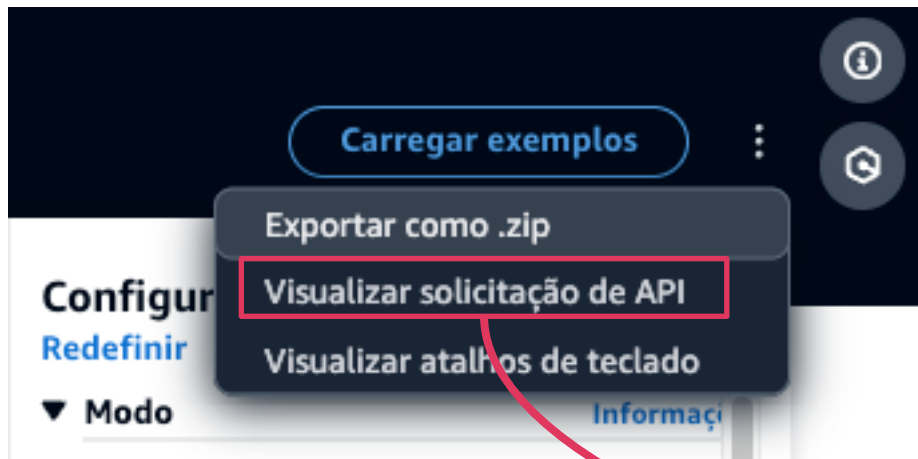
Imagem de referência: imagem de base para o modelo seguir

Tamanho: diferentes resoluções, quanto maior, maior o consumo e o custo

Força do prompt: associado a liberdade do modelo ser influenciado mais ou menos pelo prompt.

Propagação (ou seed): associado a reprodutibilidade do modelo.

PLAYGROUNDS DE MODELOS DE IMAGEM



Para obter as especificações de chamada do modelo, clique no menu de opções ao lado do botão “Carregar exemplos”.

```
{
  "textToImageParams": {
    "text": "gere um fazendeiro que utiliza tratores verdes, em uma
    imagem colorida num campo de milho"
  },
  "taskType": "TEXT_IMAGE",
  "imageGenerationConfig": {
    "cfgScale": 1.1,
    "seed": 1109605337,
    "quality": "standard",
    "width": 512,
    "height": 512,
    "numberOfImages": 3
  }
}
```

Solicitação de API

Essa é a solicitação de API para a API do modelo de invocação. [Saiba mais](#)

AWS CLI

```
aws bedrock-runtime invoke-model \
  --model-id amazon.titan-image-generator-v1 \
  --body '{"textToImageParams":{"text":"gere um fazendeiro que
  --cli-binary-format raw-in-base64-out \
  --region us-east-1 \
  invoke-model-output.txt
```

Copiar

O body de invocação dos modelos varia de modelo para modelo e fabricante para fabricante. É necessário adaptar cada um deles em sua aplicação backend.

PLAYGROUNDS DE MODELOS DE TEXTO

Utilização similar ao playground de imagem. Apresenta as diferentes possibilidades de configurações como “temperatura” “top p”, “comprimento” etc. das quais variam de cada modelo.

The screenshot displays the Amazon Bedrock Playground de texto interface. At the top, the breadcrumb navigation shows "Amazon Bedrock > Playground de texto". The main header includes "Playground de texto" with an "Informações" link and a "Carregar exemplos" button. The model selected is "Titan Text G1 - Lite v1" with an "Alterar" link and an "ODT" icon. The input text is "O que seria uma profissão do futuro quando a inteligência artificial geral estiver disponível?". The generated output is "When artificial general intelligence becomes available, it would revolutionize industries such as manufacturing, healthcare, and transportation." The right sidebar, titled "Configurações", contains a "Redefinir" link and two sections: "Aleatoriedade e diversidade" with sliders for "Temperatura" (set to 1) and "Top P" (set to 1), and "Comprimento" with a "Tamanho da resposta" slider (set to 501) and a "Sequências de parada" field containing "Inserir uma sequência de parada" with a note "Somente | permitido (máximo de 20 caracteres)".

Amazon Bedrock > Playground de texto

Playground de texto [Informações](#) [Carregar exemplos](#)

Titan Text G1 - Lite v1 | [ODT](#) [Alterar](#)

O que seria uma profissão do futuro quando a inteligência artificial geral estiver disponível?

When artificial general intelligence becomes available, it would revolutionize industries such as manufacturing, healthcare, and transportation.

Configurações [Redefinir](#)

▼ **Aleatoriedade e diversidade** [Informações](#)

Temperatura

Top P

▼ **Comprimento** [Informações](#)

Tamanho da resposta

Sequências de parada

Somente | permitido (máximo de 20 caracteres)

PLAYGROUNDS DE MODELOS DE TEXTO

A temperatura, o Top K e o Top P são parâmetros importantes em modelos de geração de linguagem. Eles afetam a forma como o modelo gera sua saída e influenciam a probabilidade de seleção das palavras seguintes.

Temperatura: afeta a distribuição de probabilidade da saída prevista. Escolha um valor baixo para favorecer saídas de maior probabilidade. Escolha um valor alto para favorecer saídas de menor probabilidade.

Uma temperatura mais baixa produz respostas mais determinísticas, enquanto uma temperatura mais alta produz respostas mais aleatórias.

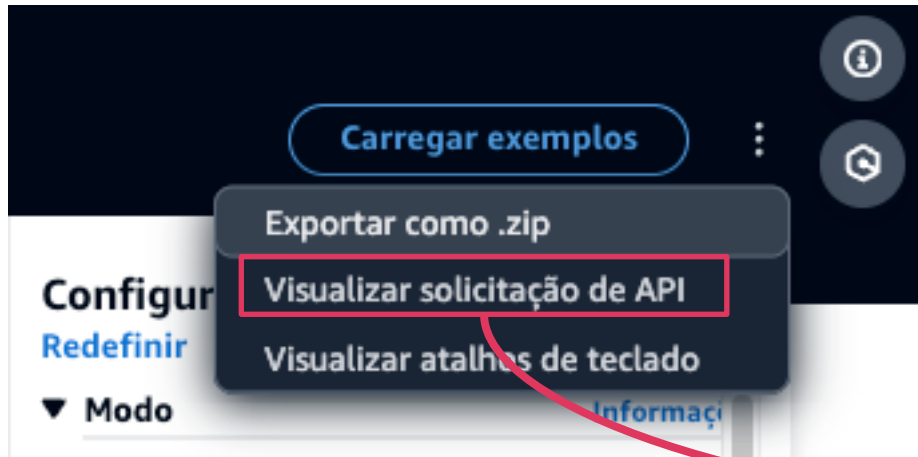
Top K: define o número de candidatos mais prováveis considerados para a próxima palavra. Escolha um valor baixo para limitar as opções às saídas mais prováveis. Escolha um valor alto para ampliar as opções e considerar saídas menos prováveis.

Por exemplo, um valor de 50 seleciona entre os 50 tokens mais prováveis como próximas palavras.

Top P: define a porcentagem dos candidatos mais prováveis considerados para a próxima palavra. Escolha um valor baixo para limitar as opções às saídas mais prováveis. Escolha um valor alto para ampliar as opções e considerar saídas menos prováveis.

Por exemplo, um valor de 0,8 seleciona os 80% melhores da distribuição de probabilidade como próximas palavras.

PLAYGROUNDS DE MODELOS DE TEXTO



disponíveis

```
{
  "inputText": "O que seria uma profissão do futuro quando a
  inteligência artificial geral estiver disponível",
  "textGenerationConfig": {
    "maxTokenCount": 501,
    "stopSequences": [],
    "temperature": 1,
    "topP": 1
  }
}
```

Para obter as especificações de chamada do modelo, clique no menu de opções ao lado do botão “Carregar exemplos”.

Solicitação de API

Essa é a solicitação de API para a API do modelo de invocação. [Saiba mais](#)

AWS CLI

```
aws bedrock-runtime invoke-model \
  --model-id amazon.titan-text-lite-v1 \
  --body '{"inputText":"O que seria uma profissão do futuro quando
  --cli-binary-format raw-in-base64-out \
  --region us-east-1 \
  invoke-model-output.txt
```

 Copiar

O body de invocação dos modelos varia de modelo para modelo e fabricante para fabricante. É necessário adaptar cada um deles em sua aplicação backend.


UTILIZANDO BEDROCK COM O SDK

Vamos listar os modelos existentes para obter os Ids dos mesmos para utilização. Para tanto iremos utilizar um cliente do Bedrock que é diferente da execução.

```
# criando sessão
session = boto3.Session(aws_access_key_id=ACCESS_KEY, aws_secret_access_key=ACCESS_SECRET)
# criando client
client = session.client("bedrock", region_name=REGION)
```

Com o comando abaixo temos acesso a modelos de acordo com os parâmetros filtrados. Por exemplo, por provedor ("Amazon") ou modalidade ("IMAGE").

```
response = client.list_foundation_models(
    byProvider='Amazon',
    byInferenceType='ON_DEMAND',
    byOutputModality='IMAGE',
)
response
```



```
... 'modelSummaries': [{ 'modelArn': 'arn:aws:bedrock:us-east-1::foundation-model/amazon.titan-image-generator-v1:0', 'modelId': 'amazon.titan-image-generator-v1:0', 'modelName': 'Titan Image Generator G1', 'providerName': 'Amazon', 'inputModalities': ['TEXT', 'IMAGE'], 'outputModalities': ['IMAGE'], 'customizationsSupported': ['FINE_TUNING'], 'inferenceTypesSupported': ['ON_DEMAND', 'PROVISIONED'], 'modelLifecycle': {'status': 'ACTIVE'}}, { 'modelArn': 'arn:aws:bedrock:us-east-1::foundation-model/amazon.titan-image-generator-v1', 'modelId': 'amazon.titan-image-generator-v1', 'modelName': 'Titan Image Generator G1', 'providerName': 'Amazon', 'inputModalities': ['TEXT', 'IMAGE'], 'outputModalities': ['IMAGE'], 'customizationsSupported': [], 'inferenceTypesSupported': ['ON_DEMAND'], 'modelLifecycle': {'status': 'ACTIVE'}}}]
```


UTILIZANDO GERAÇÃO DE IMAGEM COM O **SDK**

Criamos o body conforme exemplo de inovação da API.
Note que o cliente agora é de runtime.

```
# novo cliente para execução
client_runtime = session.client("bedrock-runtime", region_name=REGION)

prompt = "Mostre um planeta no espaço, próximo a outra estrela em outro sistema solar, com condições para
habitação humana."

body = json.dumps(
    {
        "taskType": "TEXT_IMAGE",
        "textToImageParams": {
            "text": prompt
        },
        "imageGenerationConfig": {
            "numberOfImages": 1,
            "quality": "standard",
            "cfgScale": 8.0,
            "height": 512,
            "width": 512,
            "seed": 1,
        },
    }
)
```

UTILIZANDO GERAÇÃO DE IMAGEM COM O **SDK**

Configuramos o modelo de execução

```
modelId="amazon.titan-image-generator-v1"  
response = client_runtime.invoke_model(body=body, modelId=modelId)
```

Da resposta, obtemos a imagem que vem no formato base64 (convertida em string).

```
response_body = json.loads(response["body"].read())  
base64_image_data = response_body["images"][0]
```

Convertemos em arquivo e exibimos.

```
image = base64.b64decode(base64_image_data, validate=True)  
file_to_save = "image.png"  
with open(file_to_save, "wb") as f:  
    f.write(image)
```

```
path = "image.png"
```

```
imagem = mpimg.imread(path)  
plt.imshow(imagem)  
plt.axis("off")
```



UTILIZANDO GERAÇÃO DE TEXTO COM O **SDK**

Utilizaremos o mesmo cliente de execução. Configuramos o modelo a ser utilizado e o formato de entrada e saída (JSON)

```
modelId = 'meta.llama2-70b-chat-v1'  
contentType = "application/json"  
accept = "application/json"
```

Definimos o prompt.

```
prompt = "Crie uma história infantil envolvendo elementos de ficção científica. Esta história deve ter 3  
parágrafos apenas."
```

Como escolhemos o Llama, os parâmetros do body são os seguintes (diferem do Titan):

```
body = json.dumps({  
    "prompt": prompt,  
    "max_gen_len": 512,  
    "temperature": 0.5,  
    "top_p": 0.9  
})
```

UTILIZANDO GERAÇÃO DE TEXTO COM O **SDK**

O resultado vem na forma de um streaming. Ou seja, após consumido o resultado não é mais possível acessá-lo.

```
response = client_runtime.invoke_model(body=body, modelId=modelId, accept=accept, contentType=contentType)
response_body = json.loads(response.get('body').read())
```

Agora, exibindo a resposta gerada:

```
print(response_body['generation'])
```

O professor Tecnológico, um velho e louco cientista, inventou uma máquina do tempo. Ele a testou em si mesmo e conseguiu viajar para o futuro. Lá, ele encontrou um mundo futurista e fantástico, cheio de robôs e dispositivos de alta tecnologia. O professor se encantou com aquela vida e decidiu ficar no futuro. Mas, ele logo se lembrou de que havia deixado sua máquina do tempo no passado. Ele precisava de volta para casa, mas não sabia como. Felizmente, um grupo de crianças que estavam brincando na rua, viram o professor em dificuldades e decidiram ajudá-lo. Eles usaram suas habilidades de programação e criaram um aplicativo que poderia controlar a máquina do tempo. Com a ajuda das crianças, o professor conseguiu voltar para o passado e retornar para casa, onde pôde continuar a inventar e descobrir novas coisas incríveis.

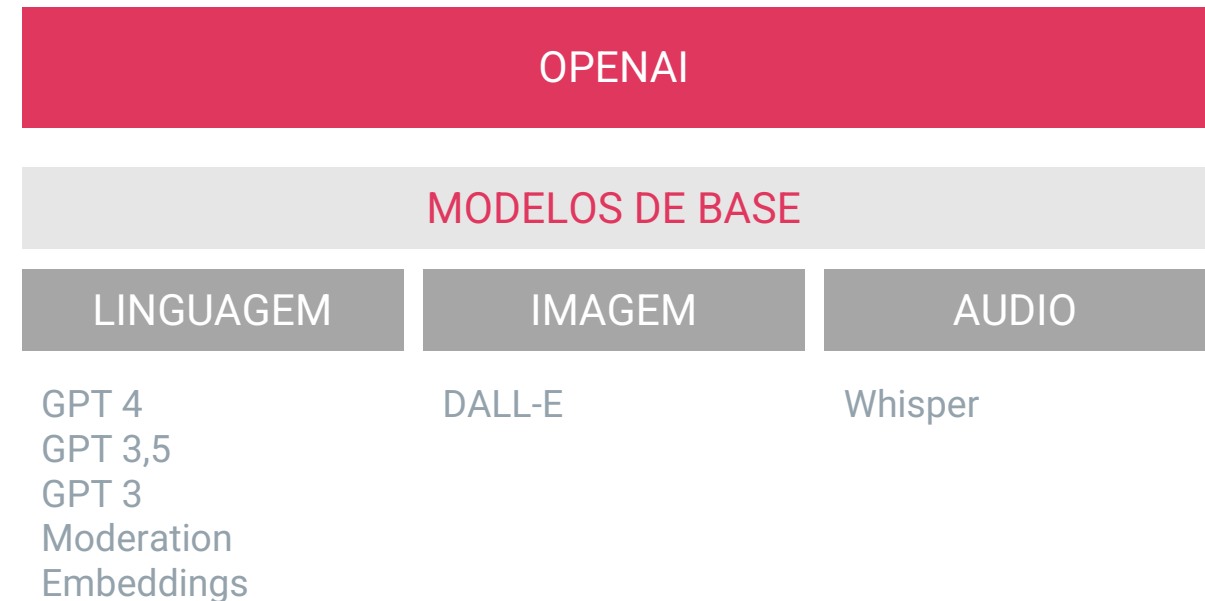


PLATAFORMA **OPENAI**

Responsável pelo serviço ChatGPT que utiliza os modelos mais conhecidos da empresa, o GPT 3,5 e o GPT 4.

Parceira da Microsoft da qual possui um serviço de LLM privado, garantido que informações empresariais não sejam utilizadas para treinamento.

Disponibiliza acesso programático por API para empresas construírem serviços que utilizam os modelos disponibilizado, apesar da maioria ser LLM há outros tipos que envolvem análise de som e imagens.



MODELOS DE BASE: GPT4

Modelo	Descrição	Máximo de Tokens	Dados de Treinamento até
gpt-4	Mais capaz do que qualquer modelo GPT-3.5, capaz de realizar tarefas mais complexas e otimizado para conversação. Será atualizado com nossa última iteração do modelo 2 semanas após o lançamento.	8.192 tokens	Até setembro de 2021
gpt-4-0613	Instantâneo do gpt-4 de 13 de junho de 2023 com dados de chamada de função. Ao contrário do gpt-4, este modelo não receberá atualizações e será descontinuado 3 meses após o lançamento de uma nova versão.	8.192 tokens	Até setembro de 2021
gpt-4-32k	Mesmas capacidades do modelo gpt-4 padrão, mas com 4 vezes o comprimento de contexto. Será atualizado com nossa última iteração do modelo.	32.768 tokens	Até setembro de 2021
gpt-4-32k-0613	Instantâneo do gpt-4-32k de 13 de junho de 2023. Ao contrário do gpt-4-32k, este modelo não receberá atualizações e será descontinuado 3 meses após o lançamento de uma nova versão.	32.768 tokens	Até setembro de 2021

MODELOS DE BASE: **GPT3**

Modelo	Descrição	Máximo de Tokens	Dados de Treinamento até
gpt-3.5-turbo	Modelo GPT-3.5 mais capaz e otimizado para conversação a 1/10 do custo do text-davinci-003. Será atualizado com nossa última iteração do modelo 2 semanas após o lançamento.	4.097 tokens	Até setembro de 2021
gpt-3.5-turbo-16k	Mesmas capacidades do modelo gpt-3.5-turbo padrão, mas com 4 vezes o comprimento de contexto.	16.385 tokens	Até setembro de 2021
gpt-3.5-turbo-instruct	Capacidades similares ao text-davinci-003, mas compatível com o endpoint Completions legado e não com Completions de Chat.	4.097 tokens	Até setembro de 2021
gpt-3.5-turbo-0613	Instantâneo do gpt-3.5-turbo de 13 de junho de 2023 com dados de chamada de função. Ao contrário do gpt-3.5-turbo, este modelo não receberá atualizações e será descontinuado 3 meses após o lançamento de uma nova versão.	4.097 tokens	Até setembro de 2021

O QUE É UM **TOKEN**?

É a granularidade mínima de contagem para determinar tanto a entrada de dados quanto a saída.

Um token não necessariamente representa uma palavra inteira, pois eles podem incluir espaços em branco e também palavras incompletas.

Podem variar para a mesma palavra em diferentes posicionamentos em uma frase.

1 token \sim 4 chars in English

1 token \sim $\frac{3}{4}$ words

100 tokens \sim 75 words

1-2 sentence \sim 30 tokens

1 paragraph \sim 100 tokens

1,500 words \sim 2048 tokens

GPT-3 Codex

Many words map to one token, but some don't: indivisible.

Unicode characters like emojis may be split into many tokens containing the underlying bytes: 🍌

Sequences of characters commonly found next to each other may be grouped together: 1234567890

Clear

Show example

Tokens

64

Characters

252

Many words map to one token, but some don't: indivisible.

Unicode characters like emojis may be split into many tokens containing the underlying bytes: 🍌

Sequences of characters commonly found next to each other may be grouped together: 1234567890

TEXT

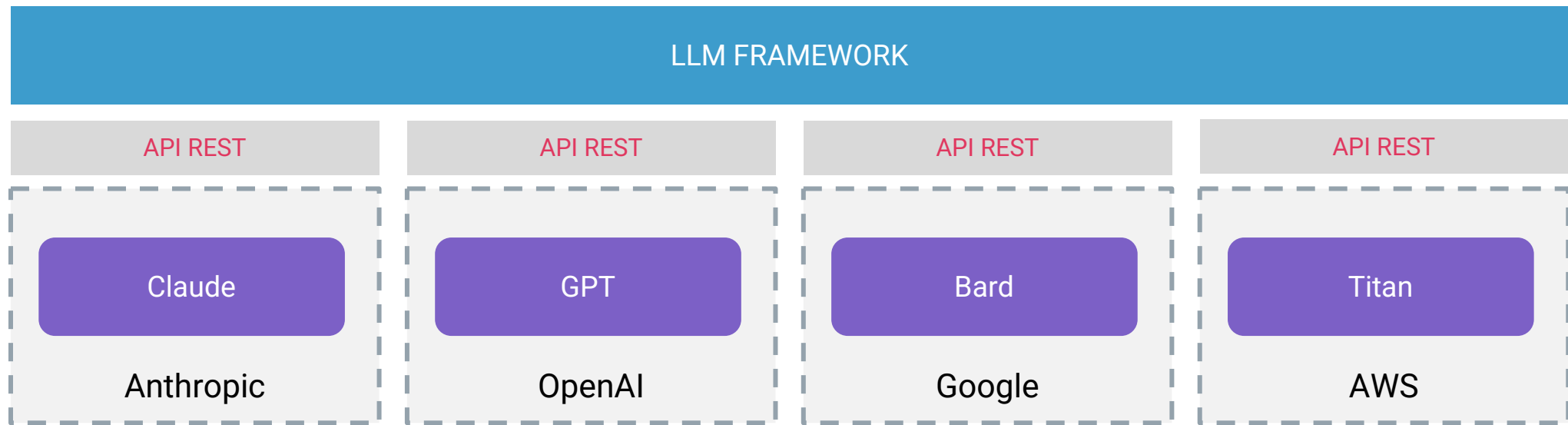
TOKEN IDS



ACESSO PROGRAMÁTICO

Podemos acessar os modelos LLM como GPT de forma direta, pela API da OpenAI ou por um framework especializado.

Utilizando um framework é possível abstrair particularidades de cada modelo de base e realizar testes entre eles mais facilmente.



PROMPT ENGINEERING

Técnica para tirar melhor proveito dos modelos de LLM.

Um dos objetivos é proporcionar a criação de aplicações que utilizem os LLMs de forma mais abstrata, detalhando exatamente o que é necessário.

Apesar de cada demanda poder variar, uma boa prática para construir bons resultados incluem os passos seguintes.

INSTRUÇÃO

Preciso convencer a leitura de livros para um grupo de alunos.

CONTEXTO

Os alunos são crianças de 7 anos que moram em um país que seja o idioma português.

DADOS DE ENTRADA

Considere os 3 livros de maior sucesso no Brasil como referência.

FORMATO DE SAÍDA

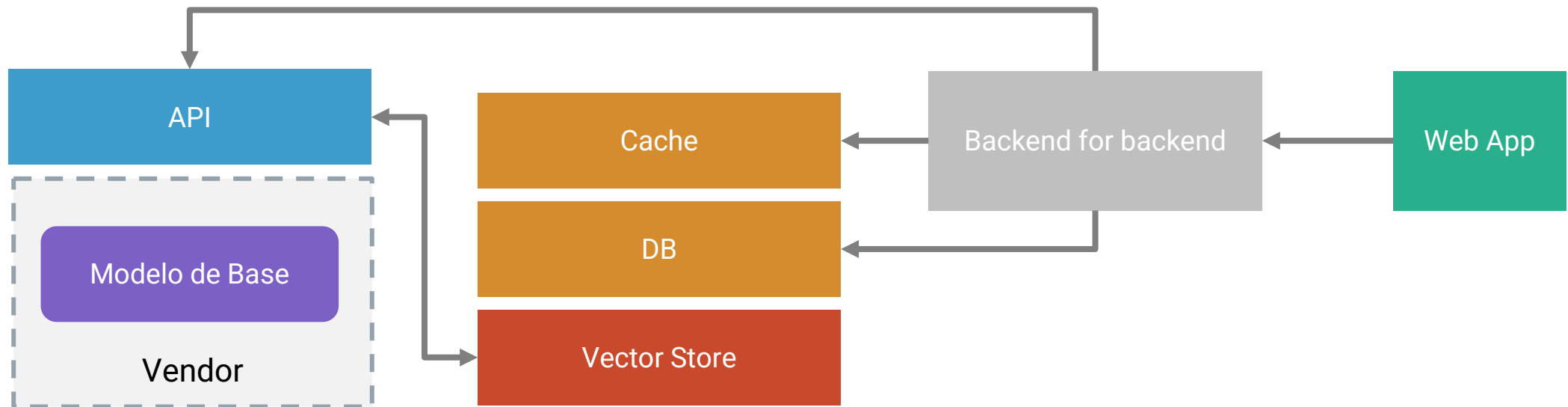
A saída deverá ser em formato JSON, contendo nome do livro e razões.

INTELLIGENT APPS

Técnica para tirar melhor proveito dos modelos de LLM.

Um dos objetivos é proporcionar a criação de aplicações que utilizem os LLMs de forma mais abstrata, detalhando exatamente o que é necessário.

Apesar de cada demanda poder variar, uma boa prática para construir bons resultados incluem os passos seguintes.



UTILIZANDO **API DO OPENAI**

Instalar a biblioteca de acesso a API da OpenAI.

```
!pip install openai
```

Obter uma chave de API da plataforma OpenAI. Novos usuários possuem 30 dias de acesso gratuito.

```
import os  
import openai
```

```
os.environ["OPENAI_API_KEY"] = "chave"  
openai.api_key = os.environ["OPENAI_API_KEY"]
```

Escolha do modelo.

```
model = "gpt-3.5-turbo"
```

UTILIZANDO API DO OPENAI

Agora vamos definir uma lista de mensagens. Cada mensagem pode ser associada a um papel diferente, como usuário, sistema ou assistente. Esses papéis ajudam a compor as principais partes da engenharia de prompt.

Obter uma chave de API da plataforma OpenAI. Novos usuários possuem 30 dias de acesso gratuito.

```
messages=[
    {"role": "system", "content": "Você é um professor de visão computacional que leciona para crianças de até 10 anos."},
    {"role": "user", "content": "Como funciona um detector de bordas de Canny, de forma bem resumida?"},
    {"role": "system", "content": "O resultado precisa ser formatado como Markdown."}
]
```

Na execução do prompt, é necessário configurar o modelo, enviar as mensagens (do prompt) e definir o **temperature**. Este último parâmetro está associado a criatividade das respostas, 0 trará os valores com maiores probabilidades (mais determinístico) associados nos embeddings e quanto maior o número mais randômico e diversa será a resposta.

```
response = openai.ChatCompletion.create(
    model=model,
    messages=messages,
    temperature=0,
)
response
```

UTILIZANDO API DO OPENAI

Este é o resultado final do prompt enviado.

```
{
  "id": "chatcmpl-7zXMp2J2ni8f1hmnbaim3eRlcYHLp",
  "object": "chat.completion",
  "created": 1694900627,
  "model": "gpt-3.5-turbo-0613",
  "response": {
    "role": "assistant",
    "content": "Um detector de bordas de Canny é um algoritmo usado para identificar as bordas em uma imagem. Ele funciona em várias etapas:\n\n1. Suavização: A imagem é suavizada para reduzir o ruído usando um filtro gaussiano.\n\n2. Gradiente: O gradiente da imagem é calculado para determinar a direção e a magnitude das mudanças de intensidade.\n\n3. Supressão de não-máximos: A supressão de não-máximos é aplicada para afinar as bordas, mantendo apenas os pixels que têm a maior magnitude de gradiente em sua vizinhança.\n\n4. Limiarização: Dois limiares são definidos - um limiar inferior e um limiar superior. Os pixels com magnitude de gradiente acima do limiar superior são considerados bordas fortes, os pixels abaixo do limiar inferior são descartados e os pixels entre os dois limiares são considerados bordas fracas.\n\n5. Conexão de bordas: Os pixels de borda fraca são conectados às bordas fortes para formar bordas contínuas.\n\nO resultado final é uma imagem binária, onde os pixels brancos representam as bordas detectadas."
  },
  "finish_reason": "stop",
  "usage": {
    "prompt_tokens": 64,
    "completion_tokens": 271,
    "total_tokens": 335
  }
}
```

No notebook é possível formatar melhor utilizando o seguinte comando para formatar em Markdown.

```
display_markdown(response['choices'][0]['message']['content'], raw=True)
```

PARA SABER **MAIS**

<https://learn.deeplearning.ai/langchain>, Curso sobre Langchain desenvolvido pelo próprio criador

<https://cursos.alura.com.br/ai-generativa-michelpf-1696298768913-p666380>, Trilha de cursos sobre AI Generativa

FIAP