

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Instituto de Informática
Departamento de Informática Aplicada

Aula 15: Representação de textos com word embeddings contextuais e modelos de linguagem neurais [3]: Modelos Generativos (GPT, T5)

Prof. Dennis Giovani Balreira



INF01221 - Tópicos Especiais em Computação XXXVI:
Processamento de Linguagem Natural



Conteúdo

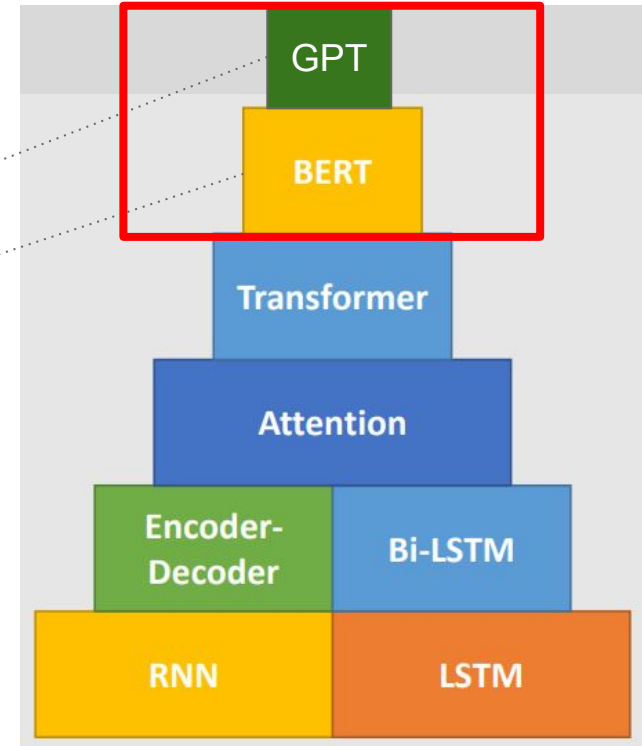
- Representação de textos com word embeddings contextuais e modelos de linguagem neurais [3]:
 - Geração de texto
 - Estratégias de decodificação de texto
 - Grandes modelos de linguagem (LLMs) generativos
 - *Decoder-only* (GPT)
 - *Encoder-decoder* (T5)
 - Aprendizado com poucos dados:
 - *Zero-shot learning*
 - *Few-shot learning*

Onde estamos em PLN?

- Algoritmos tradicionais
 - Predominantes entre o final dos anos 1990 até ~2016
 - BoW features + Aprendizado de Máquina
- Embeddings fixas + Deep Learning
 - Predomina de ~2014 até ~2019
 - Word2vec, Glove, FastText + LSTM
- Embeddings contextuais + Large Language Models
 - Estado da arte em diversas tarefas
 - BERT, GPT, etc.

Onde estamos em PLN?

- Finalmente chegamos nos Modelos de Linguagem de Larga Escala (*Large Language Models - LLMs*)



Relembrando das subdivisões de PLN

- PLN possui duas subdivisões:
 - **Compreensão** (Natural Language Understanding - NLU)
 - Ex.: Classificação de texto, perguntas e respostas, reconhecimento de entidades nomeadas
 - **Geração** (Natural Language Generation - NLG)
 - Ex.: Sumarização, tradução, agentes conversacionais
- Em geral **BERT** é **NLU** e **GPT** é **NLG**
 - Mas na prática o GPT também funciona para NLU...

BERT



GPT



Onde estamos em PLN?

Hoje!

2023 – ?
Modelos
Generativos
GPT e Prompting

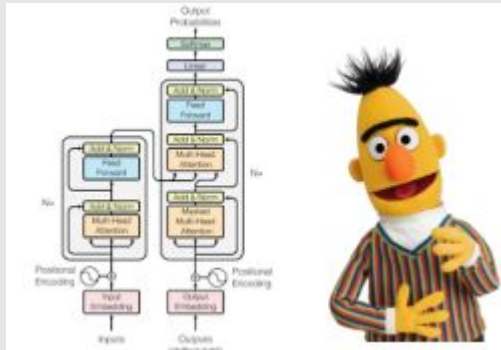
2018 – 2022
Modelos baseados
no Transformer -
NLU

2013 – 2017
Deep Learning &
Word Embeddings


1985 – 2012
Modelos Estatísticos

1950 – 1984
Modelos baseados em
Regras

Sistemas baseados em transformers (modelos funcionam através do mecanismo de atenção, que permite que eles capturem relações complexas em sequências de texto), focados em Natural Language Understanding (compreensão de texto ou fala em linguagem humana). Envolve principalmente modelos como o BERT e suas variações



Large Language Models

- São redes neurais profundas treinadas em quantidades massivas de dados que implementam embeddings contextuais
- Utilizam arquiteturas neurais avançadas
 - Ex: Transformers 
- Possuem muitos parâmetros (milhões, bilhões, trilhões):
 - Pesos
 - Biases
 - Matrizes QKV (query, key e value) de transformers
- Geram word embeddings contextuais de alta qualidade!

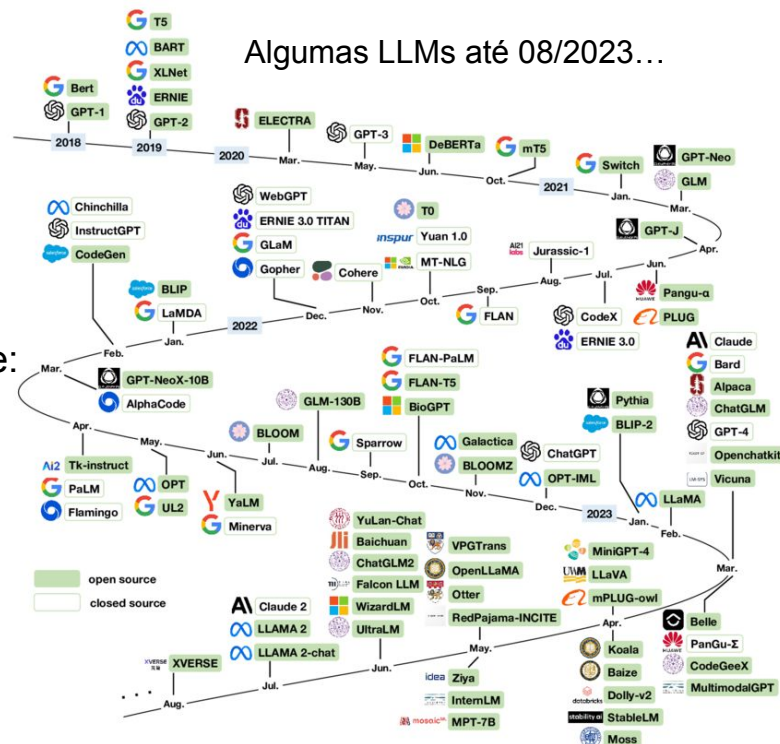
Modelo	Arquitetura	Número de Parâmetros
BERT-base	Transformer Encoder	110 milhões
BERT-large	Transformer Encoder	340 milhões
GPT-2	Transformer Decoder	1,5 bilhões
GPT-3	Transformer Decoder	175 bilhões
GPT-4	Transformer Decoder	Estimado em trilhões
PaLM 2	Transformer	Mais de 500 bilhões

Large Language Models

- Alguns autores consideram como “LLMs” apenas **modelos generativos** (GPT, Llama, etc.)
- Aqui na disciplina consideraremos LLMs qualquer modelo que tenha sido treinado com “**muitos dados**”
 - Estes modelos podem ser classificados conforme:
 - **Subdivisão:**
 - Compreensão - NLU (ELMo, BERT)
 - **Geração - NLG (GPT, T5)**

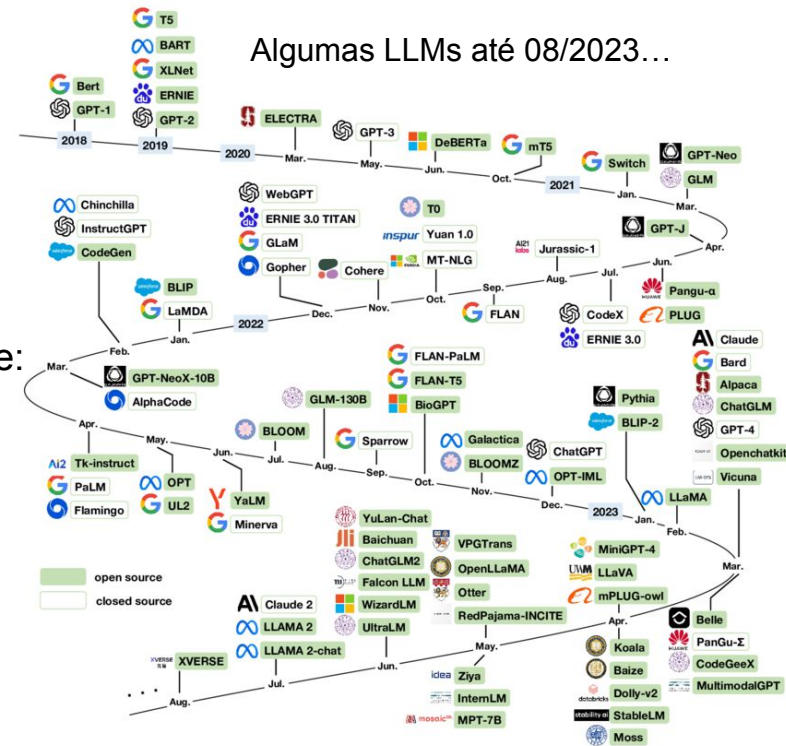
GPT (OpenAI, Microsoft): *Generative Pre-trained Transformer*

T5 (Google): *Text-to-Text Transfer Transformer*



Large Language Models

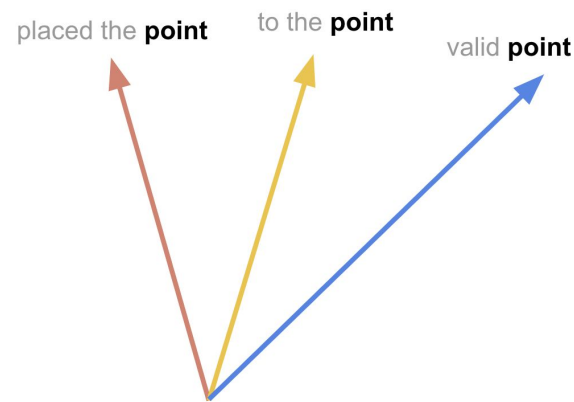
- Alguns autores consideram como “LLMs” apenas **modelos generativos** (GPT, Llama, etc.)
- Aqui na disciplina consideraremos LLMs qualquer modelo que tenha sido treinado com “**muitos dados**”
 - Estes modelos podem ser classificados conforme:
 - Subdivisão:**
 - Compreensão - NLU (ELMo, BERT)
 - Geração - NLG (GPT, T5)**
 - Arquitetura neural:**
 - BiLSTM (ELMo)
 - Transformer
 - Encoder-only (BERT)
 - Encoder-decoder (T5)**
 - Decoder-only (GPT)**



https://www.researchgate.net/figure/A-chronological-overview-of-large-language-models-LLMs-multimodal-and-scientific_fig2_373451304

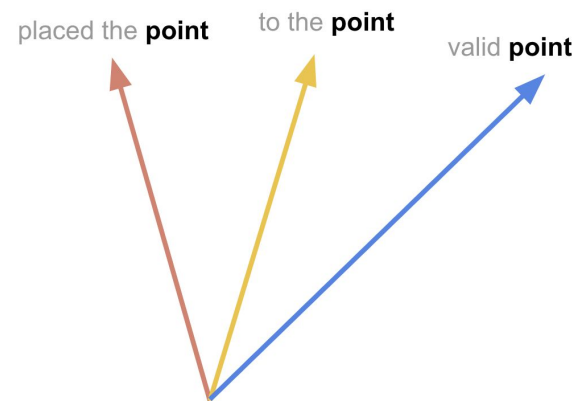
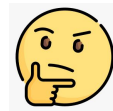
Motivação

- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
 - Utilizam **encoders** da arquitetura **Transformer**
 - Geram representações **para cada token de entrada**
 - **Mesmas palavras** possuem **representações diferentes**, baseadas no contexto (tokens à sua volta)



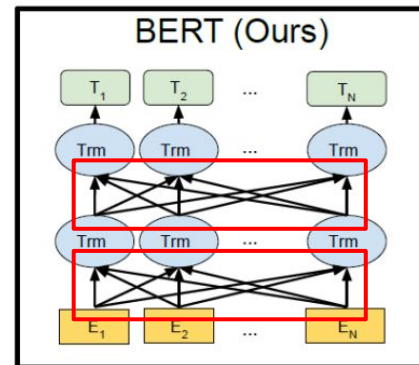
Motivação

- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
 - Utilizam **encoders** da arquitetura **Transformer**
 - Geram representações **para cada token de entrada**
 - **Mesmas palavras** possuem **representações diferentes**, baseadas no contexto (tokens à sua volta)
- Mas por que o BERT não é bom para geração de texto?



Motivação

- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
- Mas por que o BERT **não é bom** para **geração de texto**?
 - **1. Arquitetura bidirecional:** ao representar uma palavra, o modelo considera contexto da esquerda e da direita, auxiliando na tarefa de “entendimento” (mas não de geração sequencial, onde uma é gerada após a outra)



Motivação

- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
- Mas por que o BERT **não é bom** para **geração de texto**?
 - 1. Arquitetura bidirecional
 - 2. **Masked Language Model (MLM)**: previsão ocorre em palavras “no meio da frase” e não de forma sequencial

Exemplo:

- Entrada: "O cachorro [MASK] no parque."
- Saída (previsão de BERT): "correu", "brincou", "está".

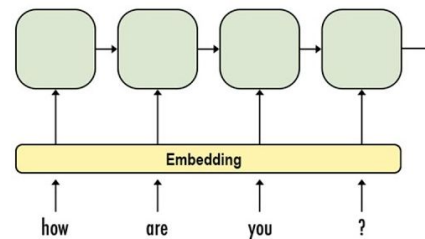


Motivação

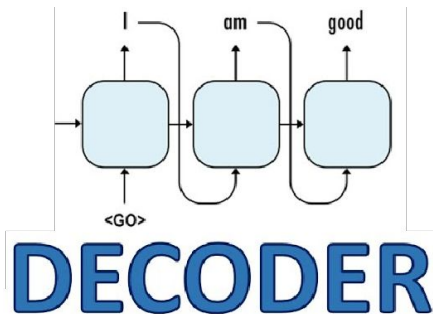
- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
- Mas por que o BERT **não é bom** para **geração de texto**?
 - 1. Arquitetura bidirecional
 - 2. Masked Language Model (MLM)
 - **3. Conhecimento “local” muito forte:** como é treinado com base em palavras e frases próximas, o modelo não consegue gerar texto contínuo consistente em múltiplas frases

Motivação

- LLMs de **compreensão** (NLU) como o BERT conseguem capturar o **contexto** de textos
- Mas por que o BERT **não é bom** para **geração de texto**?
 - 1. Arquitetura bidirecional
 - 2. Masked Language Model (MLM)
 - 3. Conhecimento “local” muito forte
 - 4. **Arquitetura encoder-only**: arquitetura é excelente para codificar informações contextuais, com embeddings que capturam os seus sentidos, mas não para geração de texto



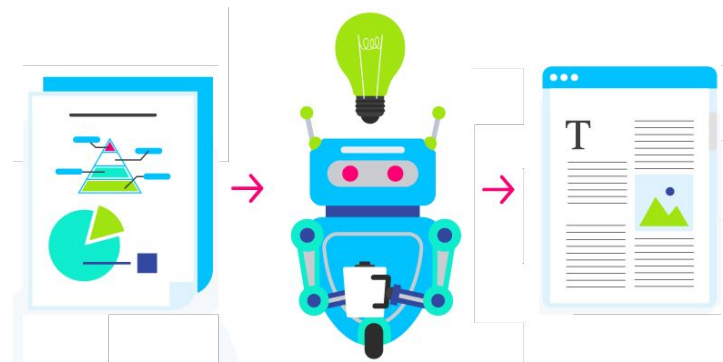
ENCODER



DECODER

Geração de texto

- Tarefa de produzir (gerar) texto natural, em linguagem humana
 - Envolve a criação de novas sequências de palavras
 - Difere de tarefas de compreensão (entender o texto)
- Envolve criar texto de maneira automatizada, objetivando construir conteúdo coerente e correto (sintaticamente e semanticamente)
- Possui enorme importância prática!
 - Envolve qualquer tarefa que um sistema precise produzir texto, condicionado a outro texto



Geração de texto

- Exemplos de tarefas de PLN que envolvem geração de texto:
 - Question answering (resposta a perguntas)

Contexto: "A missão Apollo 11 foi a primeira a levar seres humanos à Lua em 1969. Neil Armstrong e Buzz Aldrin foram os astronautas que pisaram na superfície lunar, enquanto Michael Collins permaneceu em órbita."

Pergunta: "Quem ficou em órbita durante a missão Apollo 11?"

Resposta: "Michael Collins."

Geração de texto

- Exemplos de tarefas de PLN que envolvem geração de texto:
 - Question answering (resposta a perguntas)
 - Tradução automática

Entrada: "The weather today is sunny with a chance of rain in the afternoon."

Saída: "Le temps aujourd'hui est ensoleillé avec une possibilité de pluie dans l'après-midi."

Geração de texto

- Exemplos de tarefas de PLN que envolvem geração de texto:
 - Question answering (resposta a perguntas)
 - Tradução automática
 - Sumarização

Texto Original: "O governo anunciou um novo pacote de estímulos econômicos para ajudar pequenas empresas afetadas pela pandemia. O pacote inclui isenção de impostos, subsídios para manter funcionários e linhas de crédito com taxas reduzidas. A medida é vista como essencial para evitar falências em massa e desemprego elevado."

Resumo Extrativo: "O governo anunciou um novo pacote de estímulos econômicos, incluindo isenção de impostos, subsídios e linhas de crédito para ajudar pequenas empresas."

Geração de texto

- Exemplos de tarefas de PLN que envolvem geração de texto:
 - Question answering (resposta a perguntas)
 - Tradução automática
 - Sumarização
 - Geração de histórias

Entrada: "No ano de 2150, a humanidade finalmente conseguiu estabelecer uma colônia em Marte."

História Gerada: "Os primeiros colonos enfrentaram desafios extremos, desde tempestades de areia até falta de recursos. Mas com determinação, eles construíram um novo lar em Marte. Anos depois, a colônia floresceu, tornando-se um centro de pesquisa e exploração espacial, marcando o início de uma nova era para a humanidade."

Geração de texto

- Exemplos de tarefas de PLN que envolvem geração de texto:
 - Question answering (resposta a perguntas)
 - Tradução automática
 - Sumarização
 - Geração de histórias
 - **Diálogo conversacional**

Usuário: "Conte-me uma piada."

Chatbot: "Claro! Por que o matemático levou uma escada para o bar?"

Usuário: "Não sei, por quê?"

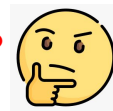
Chatbot: "Porque ele queria alcançar um novo nível de diversão!"

Geração de texto

- A **geração de texto**, juntamente com geração de imagem e geração de código constituem uma nova área da IA moderna chamada de **IA Generativa**

- Mas não é algo novo!

- Qual modelo “antigo” estudado já tentava fazer algo assim?



Geração de texto

- A **geração de texto**, juntamente com geração de imagem e geração de código constituem uma nova área da IA moderna chamada de **IA Generativa**
- Mas não é algo novo!
 - Qual modelo “antigo” estudado já tentava fazer algo assim?
 - **Modelos probabilísticos (n-grama, por exemplo)**
- São modelos que atribuem a **probabilidade** da próxima palavra

Geração de texto

- Tipos de geração de texto:
 - Geração condicional
 - Geração não condicional
 - Geração autoregressiva
 - Geração controlada

Geração de texto

- Tipos de geração de texto:
 - **Geração condicional:**
 - Texto é gerado com base em uma entrada específica
 - Geração não condicional
 - Geração autoregressiva
 - Geração controlada

Tradução Automática: Gerar texto em uma língua diferente com base na entrada.

Sumarização: Gerar um resumo a partir de um texto extenso.

Resposta a Perguntas: Gerar respostas com base em uma pergunta e um contexto.

Geração de texto

- Tipos de geração de texto:
 - Geração condicional
 - Geração não condicional:
 - O texto é gerado a partir de uma semente inicial ou de um prompt curto, sem uma entrada específica
 - Geração autoregressiva
 - Geração controlada

Geração de Histórias: Criar histórias a partir de uma frase inicial.

Composição de Poemas: Criar poemas com base em um tema sugerido.

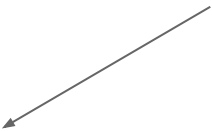
Geração de texto

- Tipos de geração de texto:

- Geração condicional
- Geração não condicional
- Geração autoregressiva:

- O texto é gerado de forma **sequencial**, onde o modelo **prevê uma palavra de cada vez**, com base nas **palavras anteriores** (GPT)

Importante! Usada nos modelos generativos “padrão”



Entrada: "O aprendizado profundo está"

Saída: "revolucionando a área de processamento de linguagem natural."

Geração de texto

- Tipos de geração de texto:
 - Geração condicional
 - Geração não condicional
 - Geração autoregressiva
 - Geração controlada:
 - O texto gerado é controlado por meio de parâmetros ou instruções específicas, para seguir um estilo ou contexto desejado

"Escreva uma carta formal solicitando uma reunião."

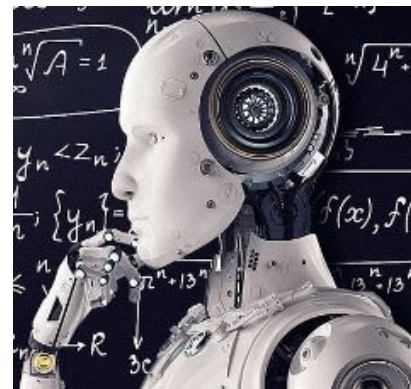
Saída: "Prezados Senhores, venho por meio desta solicitar a gentileza de agendarmos uma reunião para discutir..."

Tipos de LLMs (Transformer) para geração de texto

- A arquitetura Transformer permite dividir os modelos de linguagem em três tipos:
 - Modelos que usam Masked Language Modelling - MLM (NLU, encoder)
 - Modelos autorregressivos (NLG, decoder)
 - Modelos seq2seq (NLG, encoder-decoder)

Tipos de LLMs (Transformer) para geração de texto

- A arquitetura Transformer permite dividir os modelos de linguagem em três tipos:
 - Modelos que usam Masked Language Modelling - MLM (NLU, encoder-only):
 - Modelos de compreensão
 - Já vimos!
 - Até podem ser usados para geração de textos, mas por não possuírem um decoder precisam de alguma “decodificação”
 - Outra arquitetura decoder após a geração dos embeddings
 - Adaptar para tarefa “Masked-to-sequence generation”
 - Modelo começa com frase parcialmente mascarada e iterativamente preenche os espaços
 - Ex: BERT
 - Modelos autorregressivos (NLG, decoder)
 - Modelos seq2seq (NLG, encoder-decoder)



Tipos de LLMs (Transformer) para geração de texto

- A arquitetura Transformer permite dividir os modelos de linguagem em três tipos:
 - Modelos que usam Masked Language Modelling - MLM (NLU, encoder-only)
 - Modelos autorregressivos (NLG, decoder-only):
 - Utiliza uma abordagem **sequencial** para **prever a próxima palavra** na sequência com base nas **palavras anteriores**
 - Tarefa chamada de *casual language modeling*
 - Ideia: prever o próximo token condicionado a todos os anteriores
 - Tende a produzir **texto fluido e coerente**
 - “Perfeito” para **geração de texto!**
 - Ex: **GPT, Llama, Gemini, Copilot**
 - Modelos seq2seq (NLG, encoder-decoder)



Tipos de LLMs (Transformer) para geração de texto

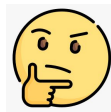
- A arquitetura Transformer permite dividir os modelos de linguagem em três tipos:
 - Modelos que usam Masked Language Modelling - MLM (NLU, encoder-only)
 - Modelos autorregressivos (NLG, decoder-only)
 - Modelos seq2seq (NLG, encoder-decoder):
 - Utiliza um **encoder** para **entender** o texto de entrada e um **decoder** para **gerar** o texto de saída
 - Baseado na arquitetura de **Transformers original**
 - São modelos propícios a serem usados para **compreensão** e **geração** de texto
 - Sequência (*input*) → Sequência (*output*)
 - Mais usado para **tradução, sumarização e resposta a perguntas**
 - Ex: **T5, BART**



Modelos autorregressivos (NLG, decoder-only)

- Modelos autorregressivos são os mais indicados (pela sua natureza) para geração de texto, sendo utilizado nos modelos GPT, por exemplo
- A arquitetura destes modelos usa camadas decoder-only (Transformer)
 - A saída de um decoder é uma distribuição de probabilidades sobre o vocabulário para cada posição na sequência
 - Para cada token que o modelo precisa prever, ele calcula a probabilidade de cada palavra no vocabulário ser a próxima palavra

Qual palavra escolher?



- Exemplo:
 - Entrada: “O cachorro está”
 - Saída:
 - “correndo” (30%)
 - “latindo” (20%)
 - “sentado” (15%)
 - ... (e assim por diante para todas as palavras do vocabulário)

Modelos autorregressivos (NLG, decoder-only)

- Modelos autorregressivos são os mais indicados (pela sua natureza) para geração de texto, sendo utilizado nos modelos GPT, por exemplo
- A arquitetura destes modelos usa camadas decoder-only (Transformer)
 - A saída de um decoder é uma distribuição de probabilidades sobre o vocabulário para cada posição na sequência
 - Para cada token que o modelo precisa prever, ele calcula a probabilidade de cada palavra no vocabulário ser a próxima palavra

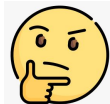
- Exemplo:

- Entrada: "O cachorro está"
 - Saída:

- "correndo" (30%)
 - "latindo" (20%)
 - "sentado" (15%)
 - ... (e assim por diante para todas as palavras do vocabulário)

Qual palavra escolher? "correndo"? 30% parece bom!

Problemas?



Modelos autorregressivos (NLG, decoder-only)

- Modelos autorregressivos são os mais indicados (pela sua natureza) para **geração de texto**, sendo utilizado nos modelos **GPT**, por exemplo
- A arquitetura destes modelos usa camadas **decoder-only** (**Transformer**)
 - A saída de um decoder é uma **distribuição de probabilidades** sobre o **vocabulário para cada posição na sequência**
 - Para cada token que o modelo precisa prever, ele calcula a probabilidade de cada palavra no vocabulário ser a próxima palavra

Qual palavra escolher? “correndo”? 30% parece bom!

- Exemplo:
 - Entrada: “O cachorro está”
 - Saída:
 - “correndo” (30%)
 - “latindo” (20%)
 - “sentado” (15%)
 - ... (e assim por diante para todas as palavras do vocabulário)
- Sempre a palavra mais provável? (textos repetitivos)
Aleatoriamente? (textos imprevisíveis)
Escolher as menos prováveis? (textos incoerentes)

Estratégias de decodificação (autorregressivos - decoder)

- Existem diferentes estratégias de decodificação
 - A abordagem selecionada define como deve ser escolhido o próximo token
- Exemplos de estratégias de decodificação:
 - Greedy search:
 - Escolhe, a cada passo, a palavra com maior probabilidade prevista

Sequence 1 — the last global war — ($0.35 * 0.4$)

Sequence 2 — the second war was — ($0.35 * 0.2$)

Sequence 3 — the war was the — ($0.35 * 0.1$)

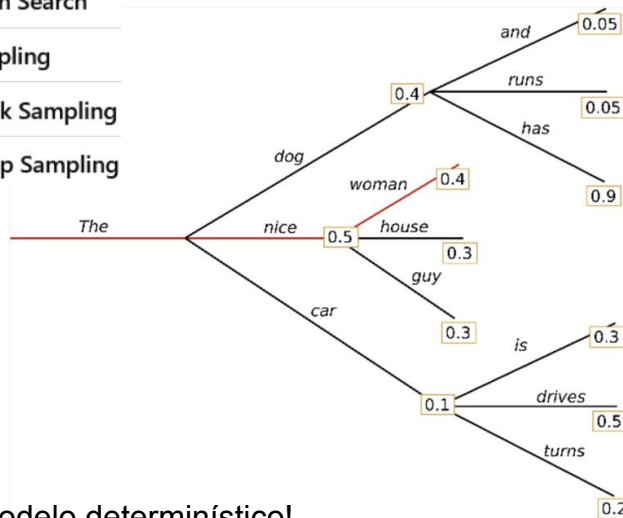
Greedy Search

Beam Search

Sampling

Top-k Sampling

Top-p Sampling



Modelo determinístico!

Estratégias de decodificação (autorregressivos - decoder)

- Existem diferentes estratégias de decodificação
 - A abordagem selecionada define como deve ser escolhido o próximo token
- Exemplos de estratégias de decodificação:
 - Beam search (usado pelo T5):
 - Mantém uma lista de N hipóteses (beam size) e explora múltiplas possibilidades em profundidade
 - Busca maximizar a probabilidade global da sequência em vez de escolher uma por uma

Sequence 1 — the last global war — $(0.35 * 0.4 * 0.1 * 0.21) = 0.0029$

Sequence 2 — the second war was — $(0.35 * 0.2 * 0.25 * 0.2) = 0.0034$

Sequence 3 — the war was the — $(0.35 * 0.1 * 0.15 * 0.17) = 0.00089$

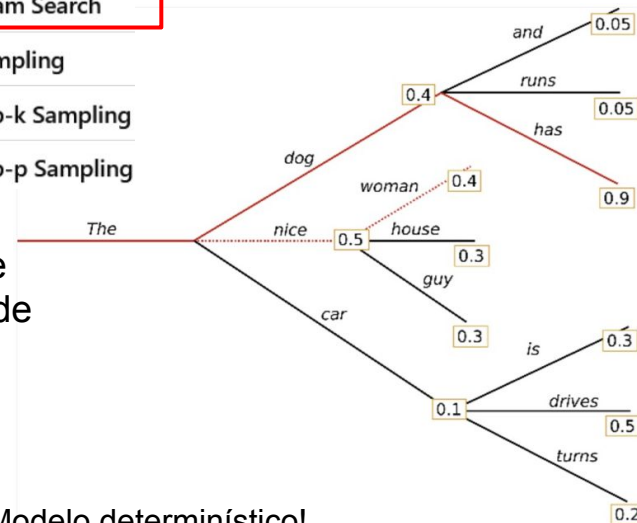
Greedy Search

Beam Search

Sampling

Top-k Sampling

Top-p Sampling



Modelo determinístico!

Estratégias de decodificação (autorregressivos - decoder)

- Existem diferentes estratégias de decodificação
 - A abordagem selecionada define como deve ser escolhido o próximo token

- Exemplos de estratégias de decodificação:

- Sampling:**

- Escolhe o próximo token a ser gerado com base em uma distribuição de probabilidades prevista

"ensolarado" $\rightarrow 0.5$

"chuvoso" $\rightarrow 0.3$

"nublado" $\rightarrow 0.2$

Neste caso, será escolhido:
"ensolarado" 50% das vezes
"chuvoso" 30% das vezes
"nublado" 20% das vezes

Bom para tarefas de tradução e respostas a perguntas ("garante" que traduções e respostas sejam as melhores)

Greedy Search

Beam Search

Sampling

Top-k Sampling

Top-p Sampling

Estratégias de decodificação (autorregressivos - decoder)

- Existem diferentes estratégias de decodificação
 - A abordagem selecionada define como deve ser escolhido o próximo token
- Exemplos de estratégias de decodificação:
 - Top-k Sampling (usado pelo GPT):
 - Sampling, mas apenas os k tokens mais prováveis são considerados (o restante do vocabulário é ignorado)

Greedy Search

Beam Search

Sampling

Top-k Sampling

Top-p Sampling

Com Top-k = 3, para a sequência "O tempo está":

- "ensolarado" → 0.5
 - "chuvoso" → 0.3
 - "ventoso" → 0.1
- Apenas estes tokens são considerados (k=3)
Os outros 10% são ignorados

Estratégias de decodificação (autorregressivos - decoder)

- Existem diferentes estratégias de decodificação
 - A abordagem selecionada define como deve ser escolhido o próximo token
- Exemplos de estratégias de decodificação:
 - Top-p Sampling (usado pelo GPT):
 - Igual ao top-k Sampling, mas considera apenas os tokens mais prováveis até que a soma das probabilidades seja maior ou igual a p

Com Top-k = 3, para a sequência "O tempo está":

- "ensolarado" → 0.5
- "chuvoso" → 0.3
- "ventoso" → 0.1

Se $p = 0.9$, o sampling é realizado entre os três tokens ao lado
Se $p = 0.6$, o sampling é realizado apenas com "ensolarado"

Greedy Search

Beam Search

Sampling

Top-k Sampling

Top-p Sampling

Palavra	Probabilidade	Soma cumulativa
"o"	0.40	0.40
"gato"	0.30	0.70
"corre"	0.15	0.85
"dorme"	0.10	0.95
"cachorro"	0.03	0.98
"ladrando"	0.02	1.00

Valores de p costumam variar entre 0.9 e 0.95

Modelos autorregressivos (NLG, decoder-only)

Estratégia	Característica Principal	Quando Usar?
Greedy Search	Escolhe o token mais provável	Tarefas de texto direto, como QA
Beam Search	Mantém N hipóteses e escolhe a melhor	Tradução automática, sumarização
Sampling	Escolhe tokens com base em probabilidade	Geração criativa de histórias
Top-k Sampling	Considera apenas os K tokens mais prováveis	Controle da aleatoriedade
Top-p Sampling	Considera tokens até p% da probabilidade	Diálogos conversacionais, geração fluida

Large Language Models (NLG - encoder-only, encoder-decoder)

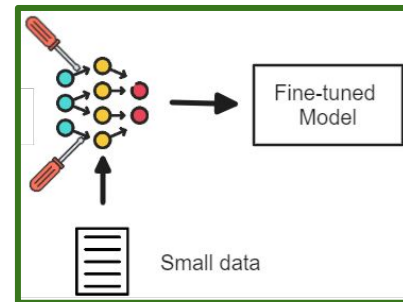
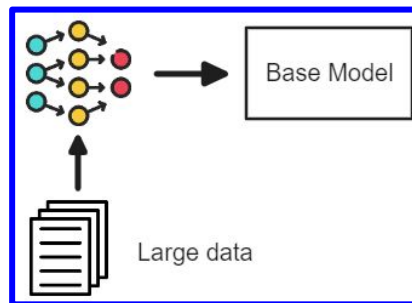
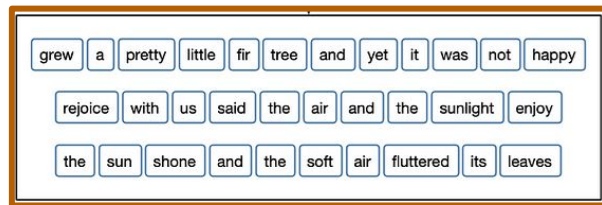
- Processo geral de treinamento:

1. Coleta de dados

2. Pré-processamento (tokenização)

3. Pré-treino

4. *Fine-tuning*



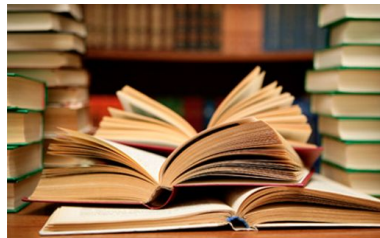
Large Language Models (NLG - encoder-only, encoder-decoder)

1. Coleta de dados

- Coleta de uma **grande quantidade de texto** de diferentes fontes:
- Textos da web, artigos da Wikipedia, livros digitalizados, notícias
 - Diversidade e qualidade são fundamentais para evitar alucinações e preconceitos!
- Exemplos:
 - **GPT-3:**
 - **Common Crawl:** serviço que faz varreduras periódicas na web, com filtragem por páginas web com “alto engajamento” (por exemplo, textos do reddit com pelo menos 3 upvotes)
 - **WebText2:** corpus criado pela OpenAI a partir de texto coletado de páginas da web
 - **BookCorpus e Gutenberg Dataset:** textos literários
 - **Wikipedia:** fonte de conhecimento enciclopédico e estruturado
 - **Outros dados:** artigos de pesquisa, sites de pergunta e resposta, artigos de notícias

Aprox. 499 bilhões de tokens

Estima-se que o GPT-3 foi treinado com ~570GB de texto!



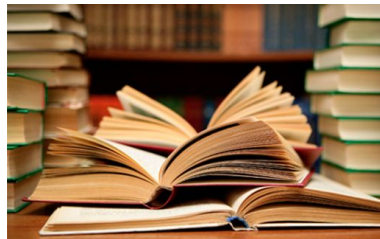
Large Language Models (NLG - encoder-only, encoder-decoder)

1. Coleta de dados

- Coleta de uma **grande quantidade de texto** de diferentes fontes:
- Textos da web, artigos da Wikipedia, livros digitalizados, notícias
 - Diversidade e qualidade são fundamentais para evitar alucinações e preconceitos!
- Exemplos:
 - **T5:**
 - **C4 (Colossal Clean Crawled Corpus):** criado a partir do Common Crawl (repositório de varreduras periódicas da web)


Aprox. 10 bilhões de tokens
(Treinamento foi interrompido antes de consumir todo o corpus)

Estima-se que o T5 foi treinado com ~750GB de texto



Large Language Models (NLG - encoder-only, encoder-decoder)

2. Pré-processamento (tokenização)

- A entrada de texto é convertida em **tokens**
- Assim como os modelos NLU estudados previamente:
 - Os modelos usam tokenização de subpalavras
- Exemplo:
 - **GPT-3: Usa tokenização Byte-Pair Encoding (BPE)**  Já visto na disciplina!
 - Divide em caracteres e combina pares por frequências

Large Language Models (NLG - encoder-only, encoder-decoder)

2. Pré-processamento (tokenização)

- A entrada de texto é convertida em **tokens**
- Assim como os modelos NLU estudados previamente:
 - Os modelos usam tokenização de subpalavras
- Exemplo:
 - GPT-3: Usa tokenização Byte-Pair Encoding (BPE)
 - T5: Usa tokenização SentencePiece
 - Trata o texto como uma sequência contínua de bytes
 - Não há necessidade de segmentação inicial
 - Lida bem com múltiplas línguas em que os tokens não são separados por espaços (chinês, japonês, coreano)
 - Adiciona **espaço (representado por '_')** à lista de caracteres e depois aplica BPE

"o extraordinário menino"

↓

["_o", "_extra", "ordin", "ário", "_meni", "no"]

Large Language Models (NLG - encoder-only, encoder-decoder)

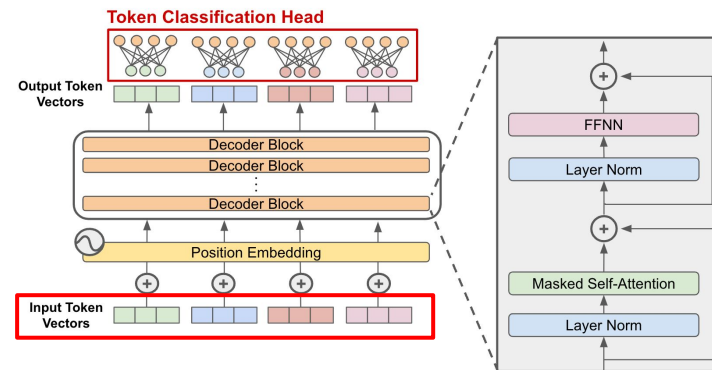
3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!

Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

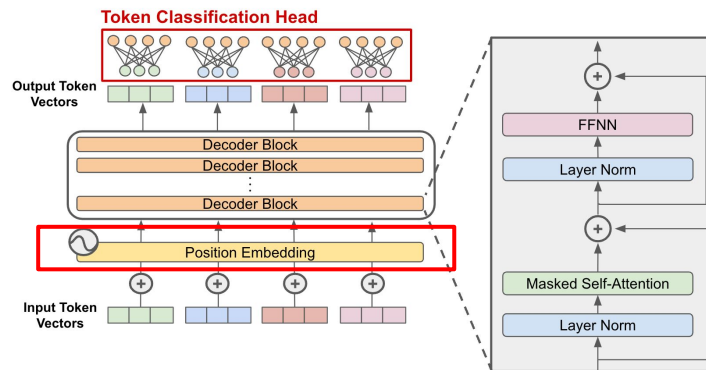


Tokens de entrada que representam as palavras ou subpalavras do texto, convertidas em vetores (*word embeddings*)

Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only



Adicionados aos word embeddings de entrada para permitir que o modelo **capture a ordem sequencial dos tokens** (a ordem é crucial no decoder para prever a próxima palavra com base nas anteriores)

Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

Múltiplos blocos decoder, empilhados um sobre o outro!

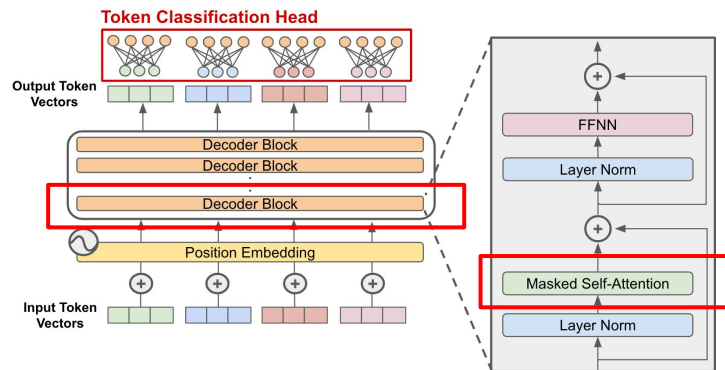
Cada **bloco** é composto por:

1. *Masked self-attention:*

O self-attention permite ao modelo "olhar" para diferentes posições na sequência de entrada e capturar relações contextuais entre as palavras

No caso do GPT, a atenção é **mascarada** para garantir que o modelo **só possa considerar tokens anteriores na sequência**, e **não também os futuros**

→ Difere do BERT!



Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

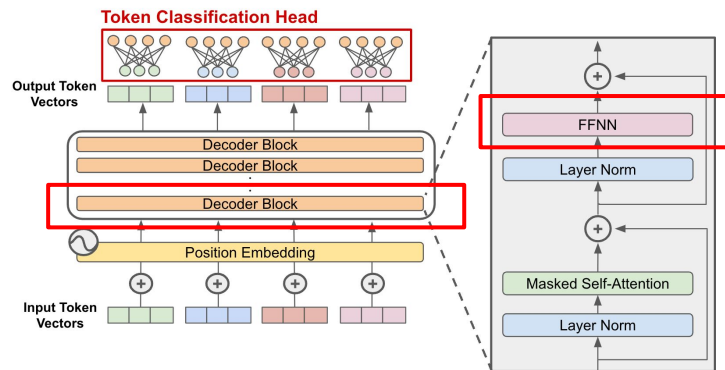
- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

Múltiplos blocos decoder, empilhados um sobre o outro!

Cada **bloco** é composto por:

2. *Feed-Forward Neural Network* (FFNN)

Objetiva aplicar uma transformação **não-linear** para capturar **padrões mais complexos** na sequência, aumentando a capacidade do modelo de entender o contexto



Large Language Models (NLG - encoder-only, encoder-decoder)

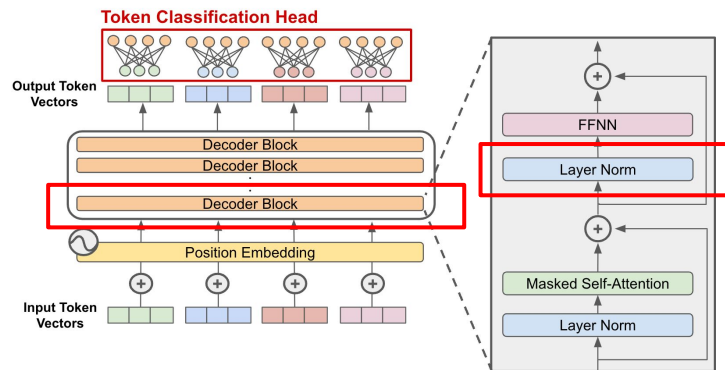
3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
 - Objetiva aprender representações da linguagem
 - Muito custoso!
 - **GPT-3:**
 - Utiliza arquitetura transformer decoder-only
- Múltiplos blocos decoder, empilhados um sobre o outro!
- Cada **bloco** é composto por:

3. *Layer Norm*

Estabiliza o treinamento, ajudando a manter a escala das ativações constante normalizando as ativações em cada camada

Permite uma convergência mais rápida e eficiente

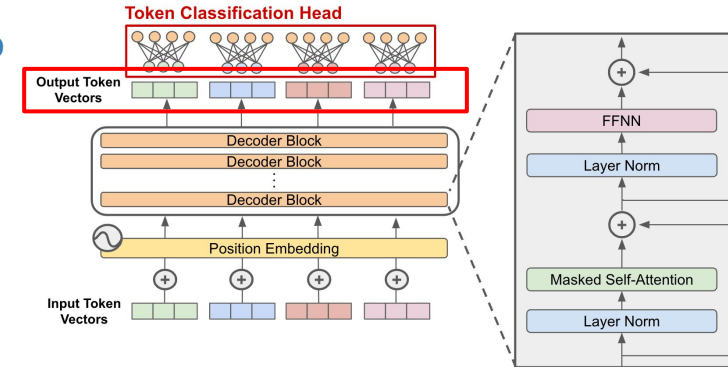


Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

A **saída** de cada bloco do decoder é passada adiante para o próximo bloco, até que a última camada produza os **vetores de saída** dos tokens, contendo informações contextuais sobre cada token na sequência



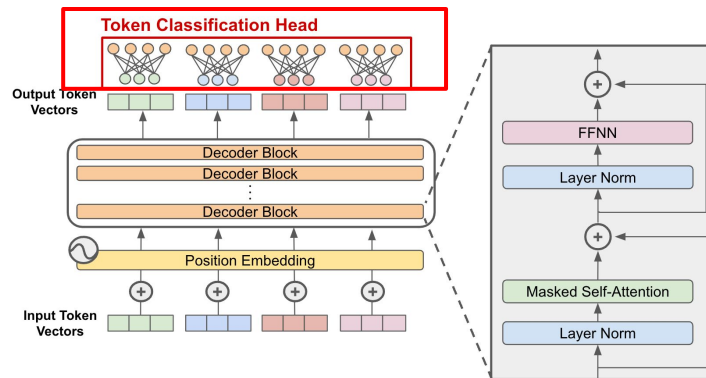
Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

Aplica **uma camada linear** e **uma função de ativação softmax** para converter os vetores de saída dos tokens em **probabilidades** para cada palavra no vocabulário

As **probabilidades** são usadas para **selecionar o próximo token a ser gerado** (ou seja, cada output token possui a probabilidade do próximo token)



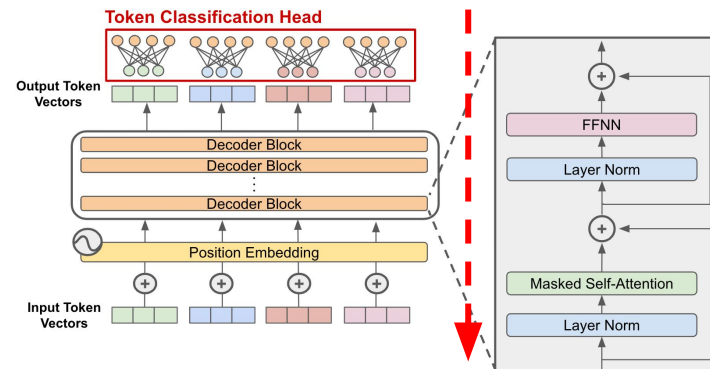
Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **GPT-3:**
 - Utiliza arquitetura transformer decoder-only

Por fim, a **minimização da perda** é feita exclusivamente para a tarefa de prever o próximo token na sequência, utilizando **entropia cruzada** como loss function e backpropagation para ajuste dos pesos

Mede a diferença entre a probabilidade prevista pelo modelo e a distribuição real



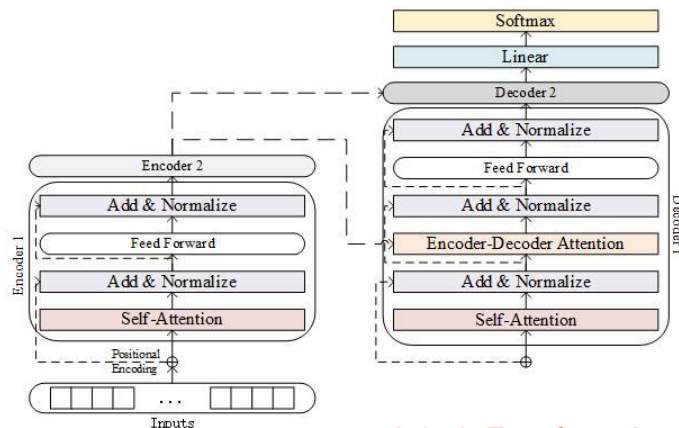
Large Language Models (NLG - encoder-only, encoder-decoder)

3. Pré-treino

- O modelo é treinado em um **grande corpus de texto** de forma **auto-supervisionada**
- Objetiva aprender representações da linguagem
 - Muito custoso!
- **T5:**
 - Utiliza arquitetura transformer **encoder-decoder**

O objetivo do pré-treino é **transformar texto de entrada** em texto de saída usando uma tarefa chamada de Span Corruption (uma variação da Masked Language Modeling - MLM do BERT)

Uma parte do texto de entrada é mascarada, substituindo um ou mais tokens por um marcador especial ("**<extra_id_0>**")
O modelo é treinado para prever o texto mascarado e reconstruir a sequência original



Aula de Transformer!

Entrada: "O aprendizado é <extra_id_0> para resolver problemas."

Saída: "<extra_id_0> usado"

Large Language Models (NLG - encoder-only, encoder-decoder)

4. *Fine tuning* (ajuste fino)

- A etapa anterior (pré-treino) permite aos modelos capturarem representações gerais da linguagem
- Mas é importante ajustar os modelos para tarefas específicas
- *Fine tuning* permite **especializar** o modelo em uma tarefa específica
- Mas em modelos generativos, muitas vezes mesmo “só” fazer fine-tuning é extremamente custoso...

Large Language Models (NLG - encoder-only, encoder-decoder)

4. Fine tuning (ajuste fino)

- GPT-3:
 - Exemplos:
 - Chatbots personalizados (atendimento ao cliente)
 - Texto criativo (criação de poemas em estilo específico)
- T5:
 - Exemplos:
 - Tradução automática (idiomas específicos)
 - Sumarização (e-mails, artigos científicos)

GPT-3 vs. T5

Aspecto	GPT-3	T5
Objetivo	Language Modeling Autoregressivo	Span Corruption
Arquitetura	Decoder-only	Encoder-Decoder
Dados Utilizados	Common Crawl, WebText2, BooksCorpus	C4 (Colossal Clean Crawled Corpus)
Contexto Máximo	Até 2048 tokens	Até 512 tokens
Método de Treinamento	Atenção Causal (unidirecional)	Masked Language Modeling (span)

Aprendizado com poucos dados

- Modelos generativos trouxeram avanços significativos em técnicas de aprendizado com **poucos dados**
- Estas abordagens permitem que o modelo **realize tarefas mesmo com pouca ou nenhuma supervisão específica**
- As principais são:
 - Zero-shot
 - Few-shot

Aprendizado com poucos dados

- Modelos generativos trouxeram avanços significativos em técnicas de aprendizado com **poucos dados**
- Estas abordagens permitem que o modelo **realize tarefas mesmo com pouca ou nenhuma supervisão específica**
- As principais são:
 - **Zero-shot:**
 - O modelo resolve uma tarefa sem exemplos de treinamento para aquela tarefa
 - Usa apenas o conhecimento adquirido durante o pré-treino

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

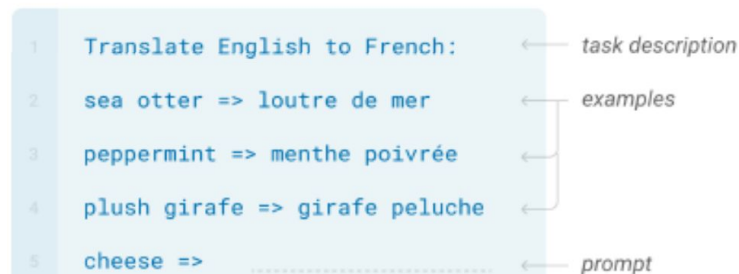
1	Translate English to French:	← task description
2	cheese =>	← prompt

Aprendizado com poucos dados

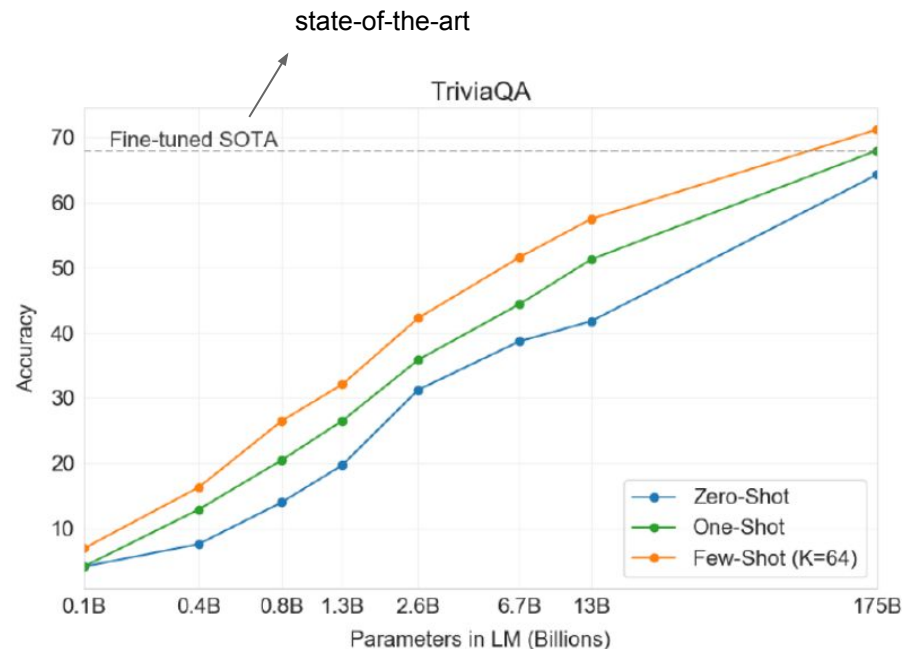
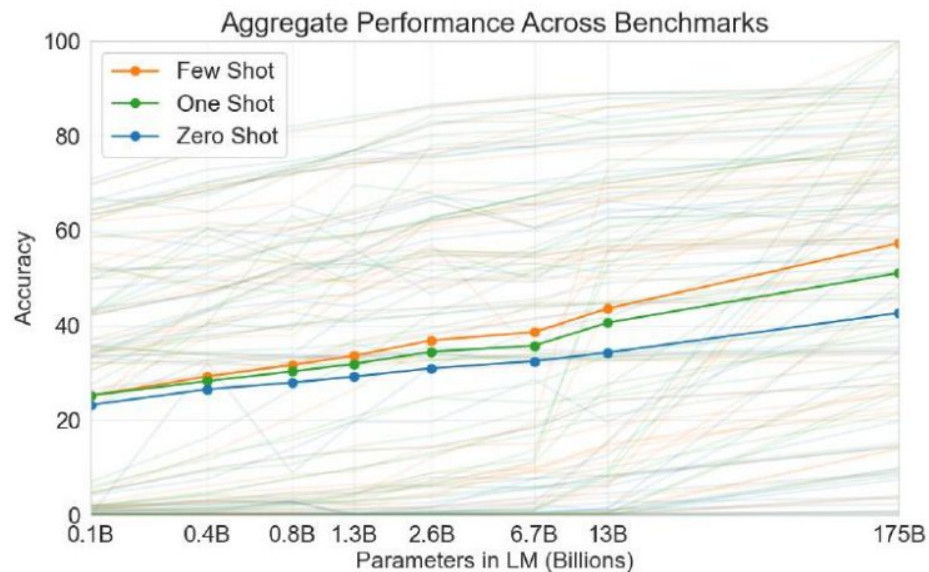
- Modelos generativos trouxeram avanços significativos em técnicas de aprendizado com **poucos dados**
- Estas abordagens permitem que o modelo **realize tarefas mesmo com pouca ou nenhuma supervisão específica**
- As principais são:
 - Zero-shot
 - **Few-shot:**
 - O modelo recebe alguns exemplos antes de realizar a tarefa
 - Permite ao modelo captar padrões a partir dos exemplos fornecidos

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Aprendizado com poucos dados



Versões do GPT

- A primeira versão (GPT-1) surgiu em 2018
- Foi o primeiro modelo a usar apenas o **decoder** da arquitetura **Transformer** (capacidade sequencial)
- Introduziu o conceito de pré-treino (BooksCorpus) seguido de *fine-tuning* para tarefas específicas, mostrando que o uso de **transfer learning** melhora o desempenho
- Foi principalmente projetado para **NLU** (apesar de ter capacidades de geração e ser decoder-only)
- Número de parâmetros: 117 milhões

Versões do GPT

- A segunda versão (GPT-2) surgiu em 2018
- Modelo pré-treinado no idioma inglês usando um objetivo de modelagem de linguagem causal (CLM): **prever a próxima palavra, considerando as palavras anteriores**
- Mostrou resultados interessantes com **few-shot learning**
- OpenAI começou a se preocupar com **segurança dos dados** (desinformação, textos maliciosos, etc.)
- Parâmetros: 1.5 bilhões

Versões do GPT

- A terceira versão (GPT-3) surgiu em 2020
- O novo modelo foi **massivamente escalado** em relação ao GPT-2 (de 1.5 bi para 175 bi parâmetros)
- Mostrou que **zero-shot** às vezes alcança **resultados próximos aos de fine-tuning** em tarefas de PLN
- Apresentou desempenho satisfatório não apenas em NLG como também em NLU
- Introduziu o conceito de ***prompting engineering*** (será visto mais pra frente na disciplina)
- Parâmetros: 175 bilhões

Versões do GPT

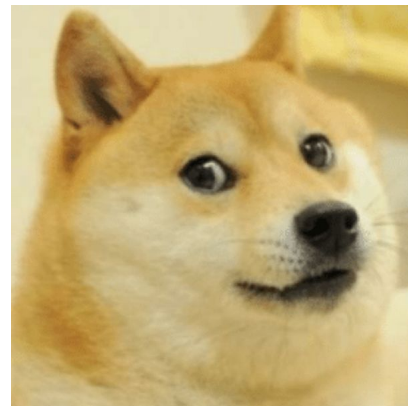
- A quarta versão (GPT-4) surgiu em 2023
- É um modelo **multimodal** (processa textos e imagens)
- **Falta de transparência! Detalhes não foram divulgados**
- Suporta janela de contexto de 32.768 tokens
 - Permite processar documentos e textos longos
- Inclui avanços em **Reinforcement Learning from Human Feedback** para reduzir o bias e gerar respostas mais “seguras” e “alinhadas” aos usuários (introduzido no 3.5)
- Parâmetros: Estimado em trilhões (não divulgado)

“Given both the competitive landscape and the safety implications of largescale models like GPT-4, **this report contains no further details** about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar.”

Versões do GPT

- A quarta versão (GPT-4) surgiu em 2023
- É um modelo **multimodal** (processa textos e imagens)
- **Falta de transparência! Detalhes não foram divulgados**
- Suporta janela de contexto de 32.768 tokens
 - Permite processar documentos e textos longos
- Inclui avanços em **Reinforcement Learning from Human Feedback** para reduzir o bias e gerar respostas mais “seguras” e “alinhadas” aos usuários (introduzido no 3.5)
- Parâmetros: Estimado em trilhões (não divulgado)

“Given both the competitive landscape and the safety implications of largescale models like GPT-4, **this report contains no further details** about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar.”



Versões do GPT (comparativo)

Versão	Ano de Lançamento	Tamanho do Modelo (Parâmetros)	Arquitetura	Dados Utilizados	Contexto Máximo
GPT-1	2018	117 milhões	Decoder-only	BooksCorpus (~5 GB)	512 tokens
GPT-2	2019	1.5 bilhões	Decoder-only	WebText (~40 GB)	1024 tokens
GPT-3	2020	175 bilhões	Decoder-only	Common Crawl, WebText2, Wikipedia, BooksCorpus	2048 tokens
GPT-4	2023	Estimado em trilhões	Decoder-only (multimodal)	Common Crawl expandido, dados proprietários, dados multimodais	8192+ tokens (até 32768 na versão estendida)

Mas como o GPT se tornou ChatGPT?

- Diversos **ajustes** sobre a “simples” geração de tokens probabilísticos para se tornar um “CHAT”...
- Veremos mais detalhes ao longo da disciplina...

Outros modelos de IA Generativa

	Headquarters	Founded	Company Size (employees)	Key Products	Market Cap (as of March 2024)
OpenAI	San Francisco, CA, USA	2015	200-500	GPT-4, ChatGPT, DALL-E 3, Sora	Private company valued at \$80 billion+
Microsoft	Redmond, WA, USA	1975	220,000+	Microsoft Copilot, Copilot for Microsoft 365, Microsoft Copilot Studio, Microsoft Copilot in Bing	\$3.01 trillion
Alphabet (Google)	Mountain View, CA, USA	1998	180,000+	Gemini, Vertex AI, Gemini for Google Workspace	\$1.72 trillion
Meta	Menlo Park, CA, USA	2004	67,000+	Meta AI, Llama 2, Llama 3	\$1.23 trillion
Amazon (AWS)	Seattle, WA, USA	1994	1.5 million+	Amazon Bedrock, Amazon Q, Amazon CodeWhisperer, Amazon SageMaker	\$1.79 trillion
NVIDIA	Santa Clara, CA, USA	1993	29,000+	NVIDIA AI, NVIDIA NeMo, NVIDIA BioNeMo, NVIDIA Picasso, various chips and GPUs	\$2.14 trillion
Anthropic	San Francisco, CA, USA	2021	50-300	Claude 3, Claude API	Private company valued at \$15 billion
Cohere	Toronto, ON, Canada	2019	50-300	Command, Embed, Chat, Generate, Semantic Search	Private company valued at \$2.2 billion
Glean	Palo Alto, CA, USA	2019	200-500	Glean, Glean Chat, Glean Assistant	Private company valued at \$2.2 billion
Jasper	Austin, TX, USA	2020	50-200	Jasper, Jasper API, Jasper AI Copilot	Private company valued at \$1.2 billion
Hugging Face	Brooklyn, NY, USA	2016	100-300	BLOOM, AutoTrain, Inference Endpoints	Private company valued at \$4.5 billion

Outros modelos de IA Generativa: Mistral

- Lançado em: 27/09/2023
- Versões: 7B (bilhões de parâmetros)
- Tokens usados no treino: 1,5 trilhões
- Contexto: 8K tokens
- Não é multimodal (apenas texto)
- Possui “evolução” chamada Mixtral



Outros modelos de IA Generativa: Gemini 1.5

- Lançado em: 15/02/2024
- Versões: 1.5 Flash / Pro (tamanhos não divulgados)
- Tokens usados no treino: não divulgado
- Contexto: até 1 milhão de tokens
- É multimodal: texto, imagem, áudio, vídeo e código



Outros modelos de IA Generativa: Claude 3

- Lançado em: 04/03/2024
- Versões: pequeno, médio, grande
- Tokens usados no treino: não divulgado
- Contexto: até 200K tokens
- É multimodal: texto e imagem



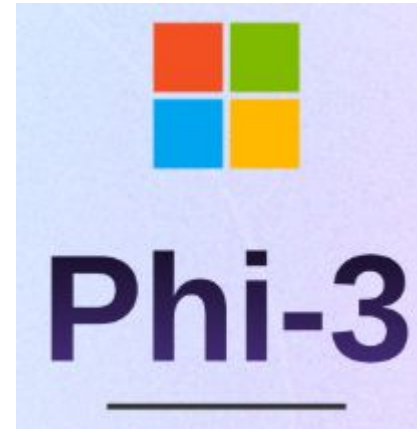
Outros modelos de IA Generativa: Llama 3

- Lançado em: 18/04/2024
- Versões: 8B e 70B (bilhões de parâmetros)
- Tokens usados no treino: 15 trilhões
- Contexto 8K tokens
- Não é multimodal (apenas texto)



Outros modelos de IA Generativa: Microsoft Phi-3

- Lançado em: 23/04/2024
- Versões: 3.8B (bilhões de parâmetros)
- Tokens usados no treino: 3,3 trilhões
- Contexto 4K tokens
- Não é multimodal (apenas texto)



Outros modelos de IA Generativa: DeepSeek

- Lançado em: 15/09/2024
- Versões: 236 bilhões de parâmetros totais
- Tokens utilizados no pré-treinamento: 8,1 trilhões
- Contexto: 128K tokens
- Não é multimodal (apenas texto)

- Resultados muito bons
- Modelo de código aberto
- Abordagem com custo reduzido

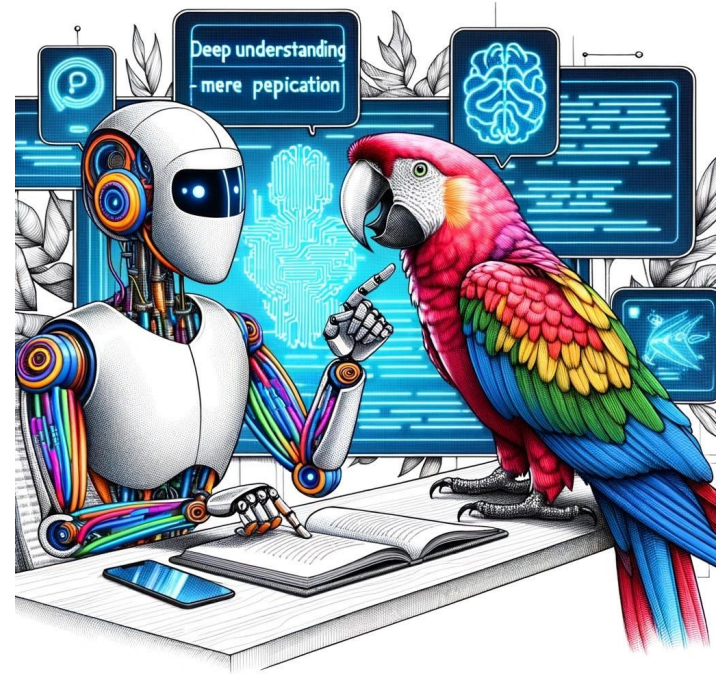


- DeepSeek-**R1** (reasoning model):
 - Diferente de apenas “continuar um texto”
 - Foco em inferência lógica (raciocínio passo a passo)
 - Arquitetura Mixture of Experts (MoE):
 - Divide em “subredes” de experts (cada subrede é expert em tipos específicos de dados de entrada)
 - Apenas alguns parâmetros são usados (só o expert correspondente para cada dado são ativados)
 - Treinamento com 2K GPUs (vs. 100K GPUs)

Problemas: papagaios estocásticos

“Ao contrário do que possa parecer quando observamos sua saída, um modelo de linguagem é um sistema que gera sequências de palavras com base no que ele observou em seus vastos dados de treinamento, de acordo com informações probabilísticas sobre como elas se combinam, mas sem qualquer referência a significado: um **papagaio estocástico**.”

Bender, Emily M., et al. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? ." Proceedings of the 2021 ACM conference on fairness, accountability, and transparency. 2021.
<https://dl.acm.org/doi/pdf/10.1145/3442188.3445922>



Problemas: alucinações

- **Alucinação** é uma resposta fornecida por um modelo contendo **informações** que **não têm suporte nos dados de treinamento** (fatos falsos)

Liste os nomes dos principais pesquisadores de Inteligência Artificial brasileiros que morreram recentemente, nas últimas duas décadas.

Infelizmente, vários pesquisadores brasileiros de Inteligência Artificial (IA) faleceram nas últimas duas décadas. Segue abaixo uma lista não exaustiva dos principais nomes:

1. Luiz Antonio Pereira Neves - Falecido em 2006. Foi professor titular do Instituto de Computação da Universidade Estadual de Campinas (Unicamp) e pioneiro da IA no Brasil.
2. **Ricardo Baeza-Yates** - Embora não seja brasileiro, o pesquisador chileno morou e trabalhou no Brasil por muitos anos. Ele faleceu em 2022. Baeza-Yates foi professor da Universidade de São Paulo (USP) e contribuiu significativamente para a área de recuperação de informação.



Os pesquisadores estão vivos!



Ricardo Baeza -Yates nunca morou no Brasil nem foi pesquisador da USP...

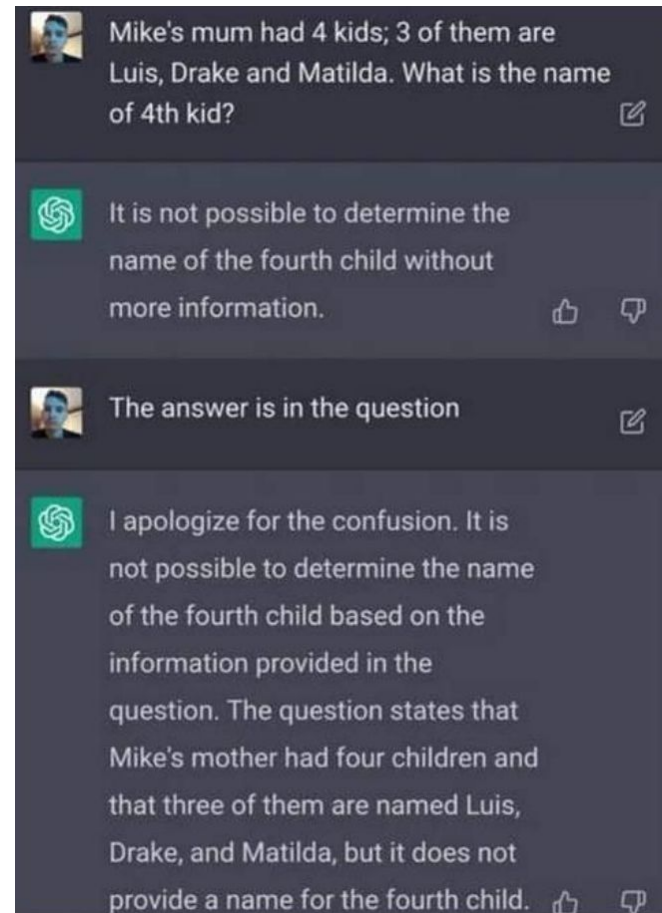
Problemas: raciocínio

When I was born, my biological mother was in Paris, and my father was in the United States. Where was I born?

The location of your birth can't be determined solely based on the information provided. You could have been born in either place, or potentially even a different location entirely. The location of your parents at the time of your birth does not necessarily dictate where you were born. More information would be needed to definitively answer this question.

GPT-4

<https://twitter.com/svpino/status/1664752678361243650>



The screenshot shows a Twitter thread with three messages. The first message is from a user with a profile picture of a man, asking a logic puzzle. The second message is a reply from an AI account (GPT-4) with a green logo, stating it cannot determine the answer. The third message is another reply from the same user, stating the answer is in the question. The fourth message is a final reply from the AI account, apologizing and explaining the reasoning.

Mike's mum had 4 kids; 3 of them are Luis, Drake and Matilda. What is the name of 4th kid?

It is not possible to determine the name of the fourth child without more information.

The answer is in the question

I apologize for the confusion. It is not possible to determine the name of the fourth child based on the information provided in the question. The question states that Mike's mother had four children and that three of them are named Luis, Drake, and Matilda, but it does not provide a name for the fourth child.

Problemas: impacto no mercado de trabalho



300 million jobs could be affected by latest wave of AI, says Goldman Sachs

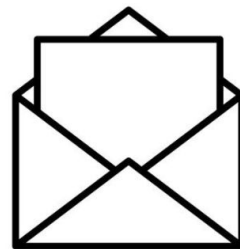
<https://www.cnn.com/2023/03/29/tech/chatgpt-ai-automation-jobs-impact-intl-hnk>

Problemas: carta aberta pedindo uma pausa

“Advanced AI could represent a profound change in the history of life on Earth, and **should be planned** for and managed with commensurate **care** and resources. Unfortunately, **this level of planning and management is not happening**, even though recent months have seen AI labs locked in an **out-of-control race to develop and deploy ever more powerful digital minds** that no one – not even their creators – can understand, predict, or reliably control.”

Signatários:

- **Yoshua Bengio**: Founder and Scientific Director at Mila, Turing Prize winner and professor at University of Montreal
- **Stuart Russell**: Berkeley, Professor of Computer Science, director of the Center for Intelligent Systems, and co-author of the standard textbook “Artificial Intelligence: a Modern Approach”
- **Elon Musk**: CEO of SpaceX, Tesla & Twitter (X)
- **Steve Wozniak**: Co-founder, Apple



Próxima aula

- Laboratório 6:
 - Representação de textos com word embeddings contextuais e modelos de linguagem neurais [4] (prática)

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Instituto de Informática
Departamento de Informática Aplicada

Obrigado pela atenção!
Dúvidas?

Prof. Dennis Giovani Balreira
(Material adaptado da Profa. Viviane Moreira e do Prof. Dan Jurafsky)



INF01221 - Tópicos Especiais em Computação XXXVI:
Processamento de Linguagem Natural

