

# Language Models are Few-Shot Learners

GPT-3

# 1. Questões

1. **Explicação de conceitos importantes do artigo**
2. **A contribuição do artigo**
3. **Resultados interessantes/inesperados**
4. ~~Uma dúvida "básica" que você ou os colegas possam ter~~
5. ~~Um tópico "avançado" para discutirmos~~

## 2. Explicação de conceitos importantes do artigo

1. **Language models:** um tipo de modelo de aprendizado de máquina que é treinado em grandes quantidades de dados de texto e pode gerar texto coerente e fluente.
2. **Zero/One/Few-shot learning:** um tipo de aprendizado de máquina em que um modelo é treinado para aprender a partir de zero, um ou poucos exemplos (tipicamente menos de 100).
  1. **zero-shot:** descrição da tarefa + 0 exemplos + prompt
  2. **one-shot:** descrição da tarefa + 1 exemplo + prompt
  3. **few-shot:** descrição da tarefa + 1+ exemplos + prompt
3. **Transfer learning:** uma técnica de aprendizado de máquina em que um modelo é treinado em um grande conjunto de dados para aprender padrões gerais e, em seguida, ajustado em um conjunto de dados menor para aprender padrões específicos.

### 3. Contribuição

1. Até aquele momento, era o maior LLM já criado com 175B de parâmetros (size matters)
2. Foi introduzido o modelo de linguagem GPT-3 (Generative Pre-trained Transformer 3), que demonstrou impressionantes capacidades de aprendizado com poucos exemplos. O GPT-3 foi treinado em um conjunto de dados massivo de texto e conseguia executar diversas tarefas de processamento de linguagem natural com apenas alguns exemplos de cada tarefa.

### 3. Resultados interessantes/inesperados

1. Os autores descobriram que o GPT-3 era capaz de se sair bem em uma ampla variedade de tarefas linguísticas sem nenhum treinamento específico da tarefa, incluindo resposta a perguntas, tradução de idiomas e até mesmo programação.
2. O GPT e o GPT-2 exigiram mais dados de treinamento específicos da tarefa para alcançar um bom desempenho no aprendizado com poucos exemplos (again size matters).

## 4. Uma dúvida "básica" que você ou os colegas possam ter

1. O que é um modelo de linguagem e como ele funciona?
2. Como funciona a arquitetura Transformer?

## 4.1 O que é um modelo de linguagem e como ele funciona?

“ Um modelo de linguagem é uma distribuição de probabilidade sobre sequências de palavras em um idioma. É um modelo estatístico que atribui uma probabilidade a todas as possíveis sequências de palavras em um determinado idioma.

O objetivo de um modelo de linguagem é prever a probabilidade de uma sequência de palavras dada as palavras anteriores na sequência.

Isso é feito aprendendo a distribuição de probabilidade condicional de cada palavra na sequência, dadas as palavras anteriores.

”

Podemos criar modelos de linguagem com RNN, CNN, LSTM, BERT...

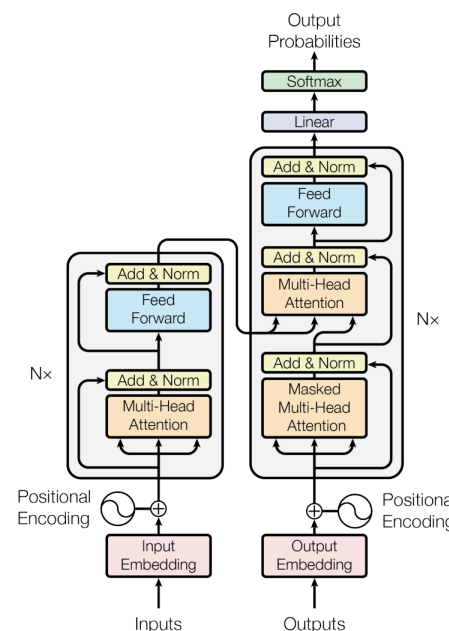
## 4.2 Como funciona a arquitetura Transformer?

O Transformer é uma arquitetura de aprendizado profundo que foi introduzida em um artigo de 2017 chamado "*Attention Is All You Need*". Ele tem como objetivo resolver tarefas de *seq2seq* para sequência, lidando com dependências de longo alcance com facilidade.

A arquitetura Transformer consiste em dois componentes principais: **Encoder** and **Decoder**

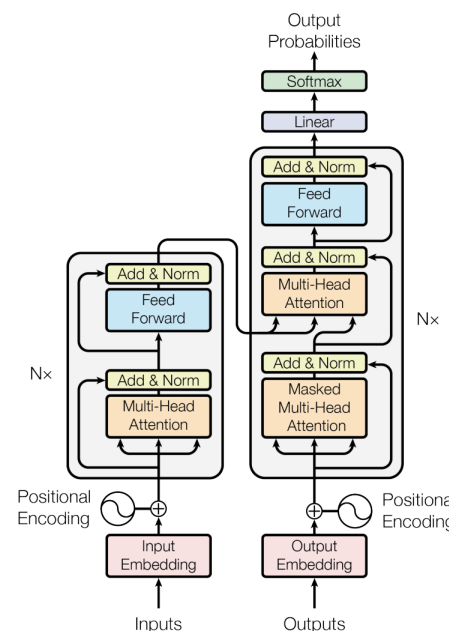
BERT

Encoder



GPT

Decoder





## 4.2.1 Encoder (BERT)

“ O encoder recebe a sequência de entrada de palavras e gera uma sequência de representações ocultas que capturam o significado da entrada. Ele consiste em várias camadas idênticas, cada uma das quais realiza duas operações:

- **Self-attention:** Essa operação permite que o modelo pondere a importância de cada palavra na sequência de entrada ao gerar a representação oculta para cada palavra. Cada palavra recebe um peso com base em sua semelhança com as outras palavras na sequência. Isso permite que o modelo se concentre nas palavras mais relevantes para cada tarefa.
- **Feedforward:** Essa operação aplica uma transformação não linear a cada representação oculta para capturar ainda mais o significado da sequência de entrada.

”

## 4.2.2 Decoder (GPT)

“ O decoder recebe as representações ocultas geradas pelo codificador e gera a sequência de saída de palavras. Assim como o codificador, ele consiste em várias camadas idênticas, cada uma das quais realiza duas operações:

- **Masked self-attention:** Essa operação é semelhante à autoatenção no codificador, mas é aplicada de maneira mascarada para impedir que o modelo olhe adiante e trapaceie usando palavras futuras para gerar a saída.
- **Cross-attention:** Essa operação permite que o modelo pondere a importância de cada representação oculta gerada pelo codificador ao gerar a sequência de saída. Ajuda o modelo a alinhar as sequências de entrada e saída e gerar traduções ou resumos precisos. ”