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

Aula 13: Representação de textos com word embeddings contextuais e modelos de linguagem neurais [1]: Modelos de Compreensão (ELMo, BERT)

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 [1]:
  - Introdução e motivação
  - Pré-treino e fine-tuning
  - Transfer learning
  - Grandes modelos de linguagem encoder:
    - ELMO
    - BERT e suas variantes

Onde estamos em PLN?

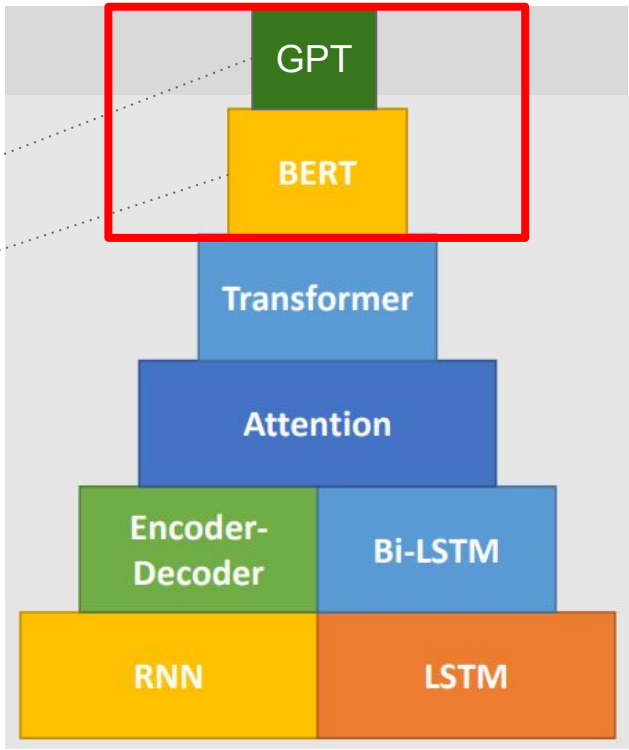
- Algoritmos tradicionais
  - Predominantes entre o final dos anos 1990 até ~2016
  - BoW features + Aprendizado de Máquina

- Embeddings fixas + Deep Learning
  - Predominates 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!

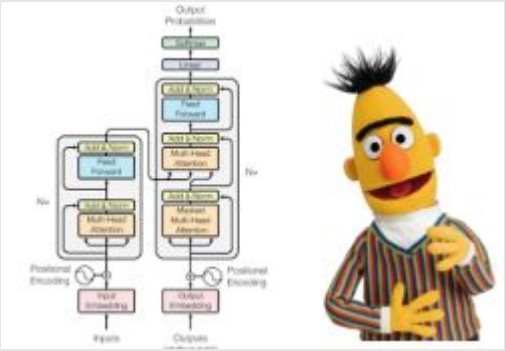
2023 – ?
Modelos Generativos GPT e Prompting

2018 – 2022
Modelos baseados no Transformer - NLU

2013 – 2017
Deep Learning & Word Embeddings

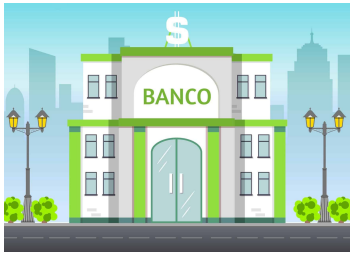
1985 – 2012
Modelos Estatísticos

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

Motivação



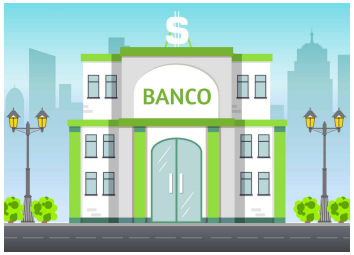
banco

0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7

- Lembrando que nosso último “estado-da-arte” visto na disciplina foram as embeddings fixas (Word2Vec, GloVe, FastText)
  - Primeiro é gerado um vocabulário fixo com as palavras independentemente do contexto

- Ex:
  1.Havia uma fila enorme no banco por causa do dia do pagamento dos trabalhadores.

Motivação



banco

0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7

- Lembrando que nosso último “estado-da-arte” visto na disciplina foram as embeddings fixas (Word2Vec, GloVe, FastText)
  - Primeiro é gerado um vocabulário fixo com as palavras independentemente do contexto

- Ex:
  1.Havia uma fila enorme no banco por causa do dia do pagamento dos trabalhadores.

  2.Joana sentou no banco da praça para terminar de ler seu livro.

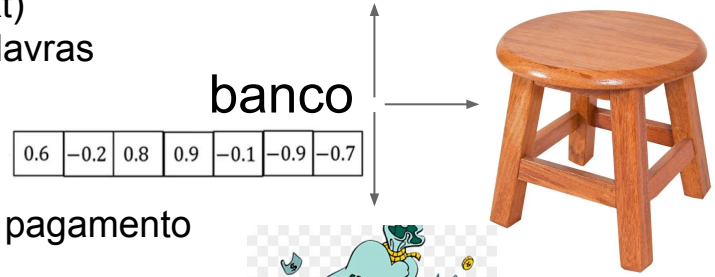
Motivação

- Lembrando que nosso último “estado-da-arte” visto na disciplina foram as embeddings fixas (Word2Vec, GloVe, FastText)
  - Primeiro é gerado um vocabulário fixo com as palavras independentemente do contexto

- Ex:
  1.Havia uma fila enorme no banco por causa do dia do pagamento dos trabalhadores.

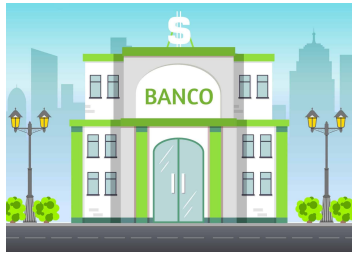
  2.Joana sentou no banco da praça para terminar de ler seu livro.

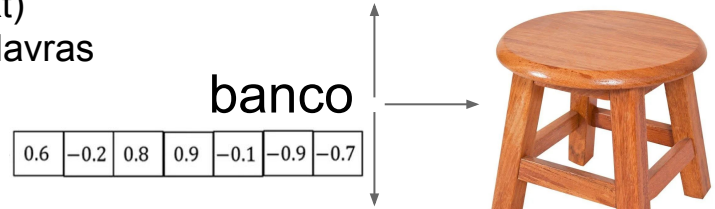
  3.Se você não tiver dinheiro, eu banco nossa viagem ao exterior.



Motivação

- Lembrando que nosso último “estado-da-arte” visto na disciplina foram as embeddings fixas (Word2Vec, GloVe, FastText)
  - Primeiro é gerado um vocabulário fixo com as palavras independentemente do contexto





- Ex:
  1.Havia uma fila enorme no banco por causa do dia do pagamento dos trabalhadores.

  2.Joana sentou no banco da praça para terminar de ler seu livro.

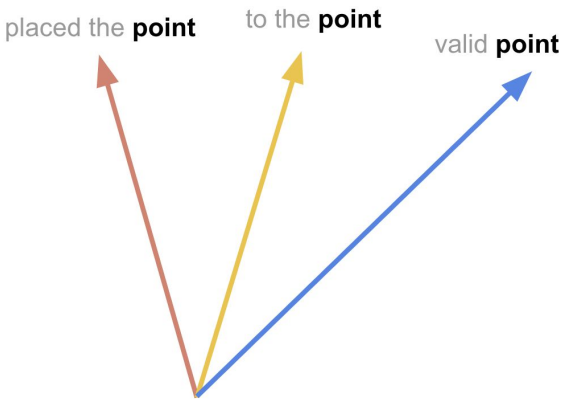
  3.Se você não tiver dinheiro, eu banco nossa viagem ao exterior.

- Todos os diferentes significados de “banco” compartilham a mesma representação!

Word embeddings contextuais

- Levam em consideração o contexto na representação
  - São gerados vetores diferentes para cada ocorrência, considerando o contexto da sentença

- Revolucionaram PLN de 2018 para cá!
  - Conseguem capturar propriedades sintáticas e semânticas
  - Melhoraram o desempenho de várias tarefas
  - Inúmeros modelos disponíveis na Hugging Face



https://www.linkedin.com/pulse/static-vs-contextual-embeddings-understanding-word-nlp-chaudhari-qk4kf/

Large Language Models

- São redes neurais profundas treinadas em quantidades massivas de dados que implementam embeddings contextuais
- Utilizam arquiteturas neurais avançadas
  - Ex: BiLSTM, 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

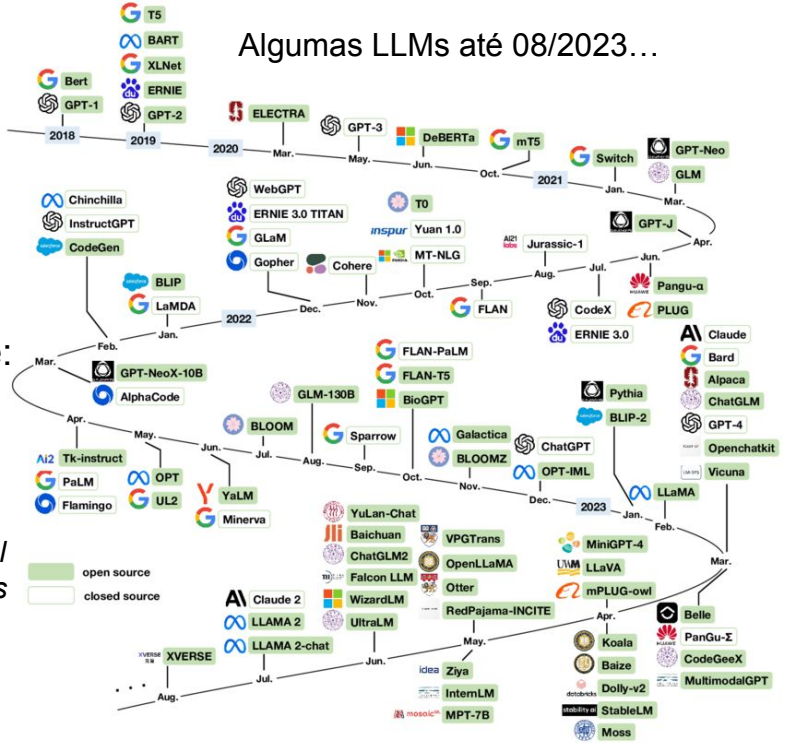
Large Language Models

- 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)

ELMo (AI2, University of Washington): *Embeddings from Language Model*
BERT (Google): *Bidirectional Encoder Representations from Transformers*

Algumas LLMs até 08/2023…



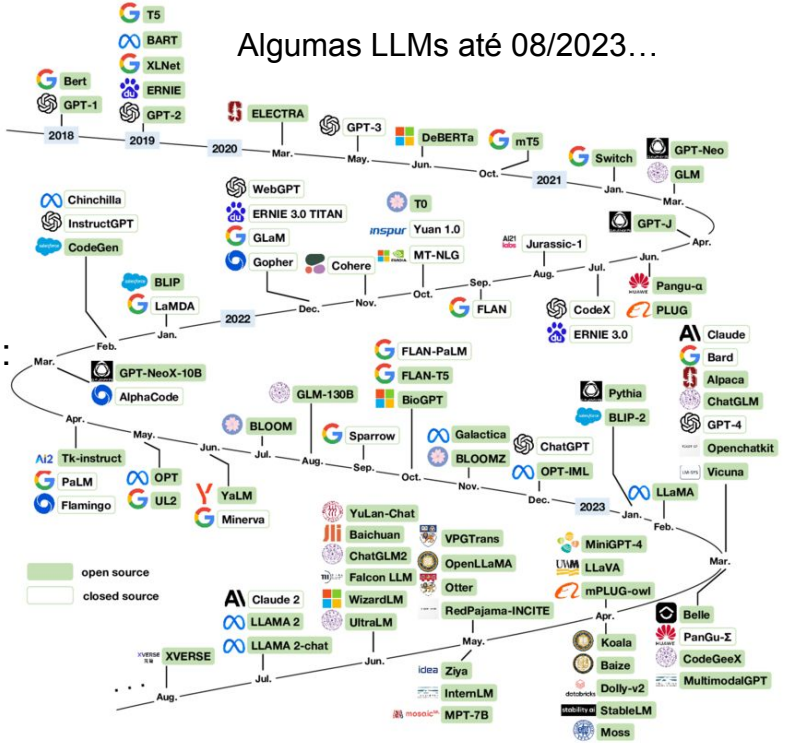
https://www.researchgate.net/figure/A-chronological-overview-of-large-language-models-LLMs-multimodal-and-scientific_fig2_373451304

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)

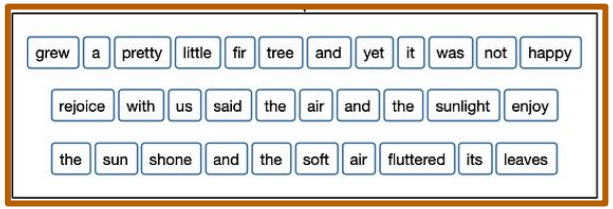
Algumas LLMs até 08/2023…

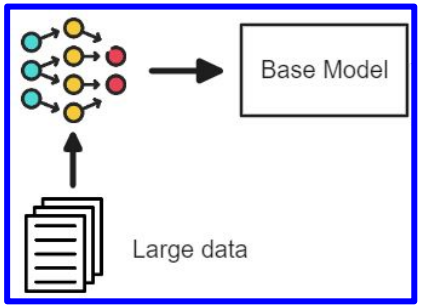


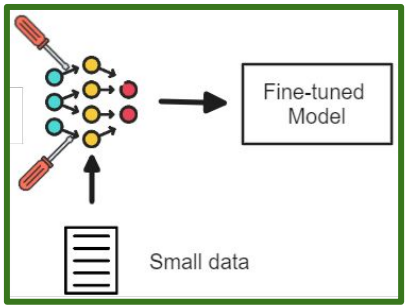
https://www.researchgate.net/figure/A-chronological-overview-of-large-language-models-LLMs-multimodal-and-scientific_fig2_373451304

Large Language Models (NLU - encoder-only)

- Processo geral de treinamento:
  1. Coleta de dados
  2. Pré-processamento (tokenização)
  3. Pré-treino
  4. *Fine-tuning*







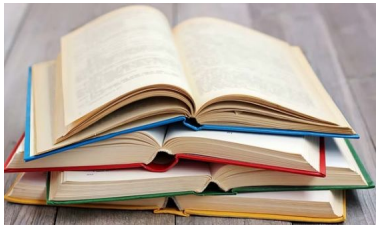
Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

1. Coleta de dados

- Coleta de uma grande quantidade de texto de diferentes fontes:
- Artigos da Wikipedia, livros, websites, notícias
- Exemplos:
  - ELMo:
    - Corpus 1 Billion Word Benchmark: dataset público com aproximadamente 800 milhões de palavras, contendo texto em inglês extraído de fontes como notícias e artigos. Corpus conhecido pela diversidade de tópicos e estilo de escrita

WIKIPEDIA
The Free Encyclopedia



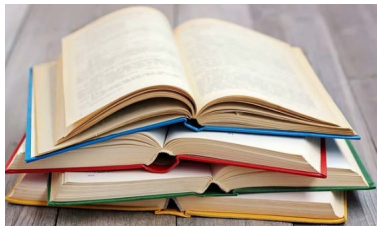
Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

1. Coleta de dados

- Coleta de uma grande quantidade de texto de diferentes fontes:
- Artigos da Wikipedia, livros, websites, notícias
- Exemplos:
  - ELMo
  - BERT:
    - Wikipedia (EN): corpus extenso e de alta qualidade, contendo artigos sobre uma ampla variedade de tópicos - cerca de 2,5 bilhões de palavras foram extraídas da Wikipedia
    - BooksCorpus: corpus de texto que inclui mais de 11 mil livros publicados online, totalizando cerca de 800 milhões de palavras. Inclui diálogos, narrativas e textos expositivos, adicionando diversidade ao treinamento

WIKIPEDIA
The Free Encyclopedia



Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

1. Coleta de dados

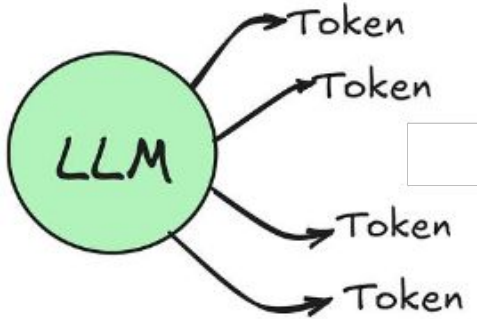
Aspecto	ELMo	BERT
Fonte de Dados	1 Billion Word Benchmark	Wikipedia + BooksCorpus
Tamanho do Corpus	~800 milhões de palavras	~3,3 bilhões de palavras
Diversidade	Textos de notícias	Textos enciclopédicos e narrativos
Tokenização	Baseada em palavras (Word)	Baseada em subpalavras (WordPiece)
Pré-processamento	Limpeza básica e tokenização	Tokenização de subpalavras, normalização, remoção de HTML

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

2. Pré-processamento (tokenização)

- A entrada de texto é convertida em tokens
- LLMs modernas usam técnicas de tokenização de subpalavras
  - Técnica de pré-processamento para lidar com palavras desconhecidas (out-of-vocabulary - OOV)
- Ideia: em vez de representar o texto como palavras inteiras, a tokenização divide em unidades menores (subpalavras, prefixos, sufixos, caracteres)
  - Reduz o problema de palavras OOV e permite representação mais eficiente da linguagem
  - Até mesmo palavras raras ou novas podem ser subdivididas em subpartes conhecidas
- O ELMo usa tokenização baseada em palavras (problema OOV)
  - Uma solução é utilizar caracteres como tokens

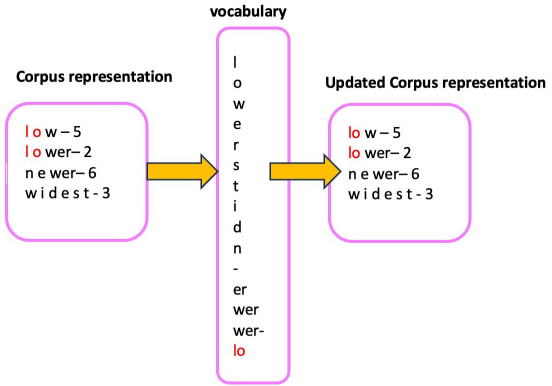


Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

2. Pré-processamento (tokenização)

- *Byte Pair Encoding* (BPE):
  - Começa com vocabulário de caracteres e vai combinando pares de caracteres ou subpalavras frequentes iterativamente
  - Dividido em:
    - *Token learner*: aprende conjunto de tokens a partir do corpus de treinamento (separado em palavras)
    - *Token segmenter*: pega uma frase de teste e segmenta nos tokens do vocabulário
  - Usado na arquitetura Transformers (original)



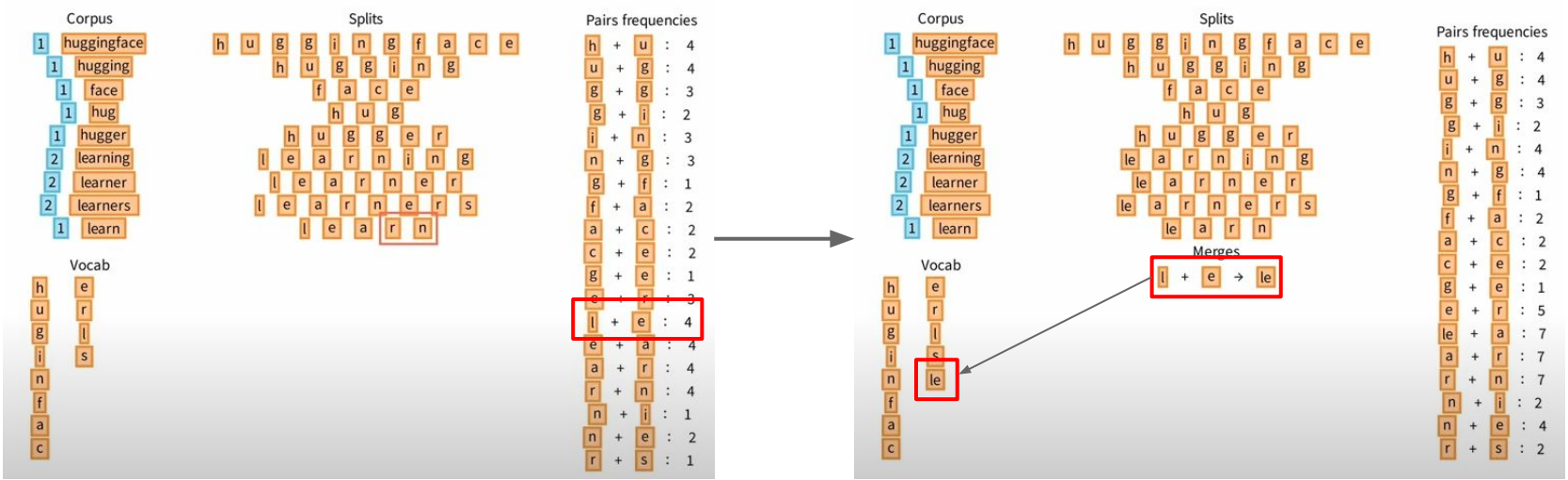
Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

2. Pré-processamento (tokenização)

- *Byte Pair Encoding* (BPE):
  - Exemplo:

https://www.youtube.com/watch?v=HEikzVL-IZU



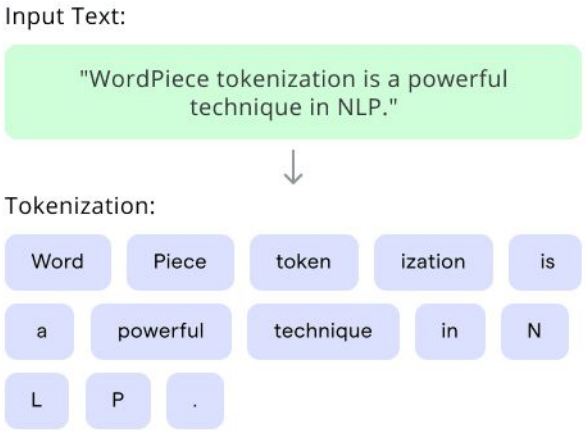
Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

2. Pré-processamento (tokenização)

[CLS] Token1, Token2, ..., TokenN, [SEP]

- *WordPiece*:
  - Começa com um vocabulário de caracteres e constrói subpalavras com base em uma métrica hierárquica, escolhendo as subpalavras que maximizam a probabilidade de ocorrência
  - Avalia o que o modelo perde ao juntar dois símbolos para decidir se vale a pena ou não
  - Usado no BERT!
  - É mais sofisticado, capturando melhor o contexto e a semântica das subpalavras

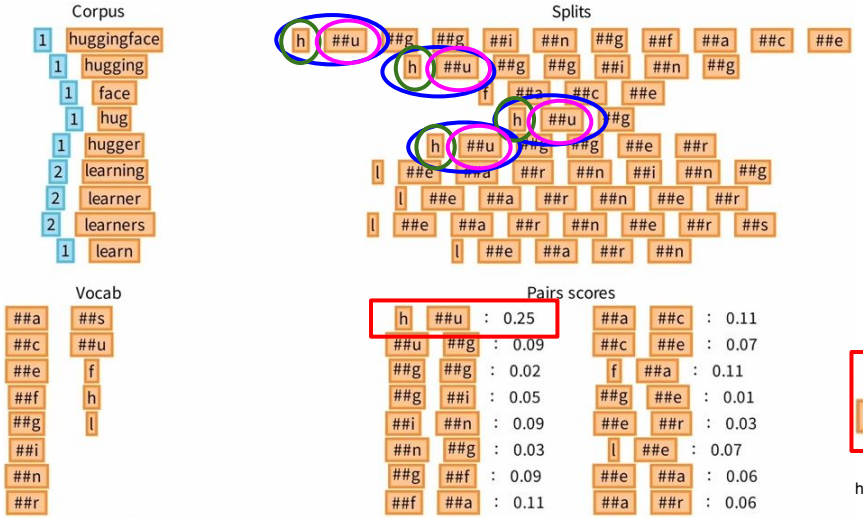


Large Language Models (NLU - encoder-only)

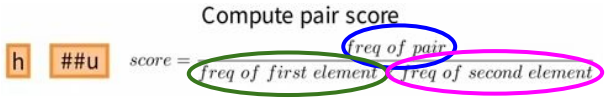
1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

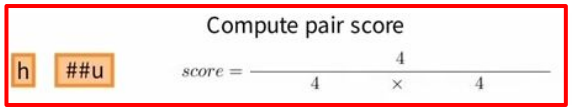
2. Pré-processamento (tokenização)

- *WordPiece:*
  - Exemplo:



indica que a subpalavra é sequência de alguma anterior





https://www.youtube.com/watch?v=qpv6ms_t_1A

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

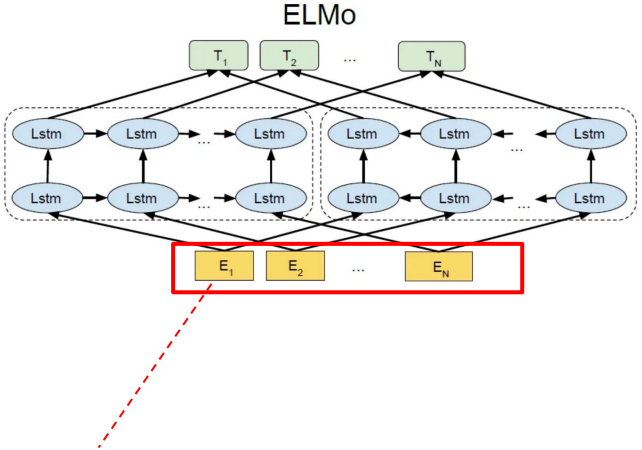
- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo:
  - Treina dois modelos separadamente (BiLSTM)



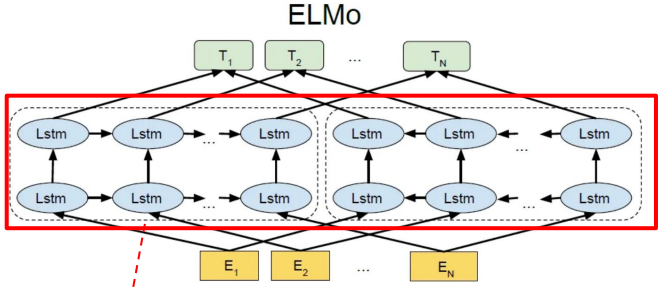
Tokens (vetores) representando as palavras inteiras

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo:
  - Treina dois modelos separadamente (BiLSTM)



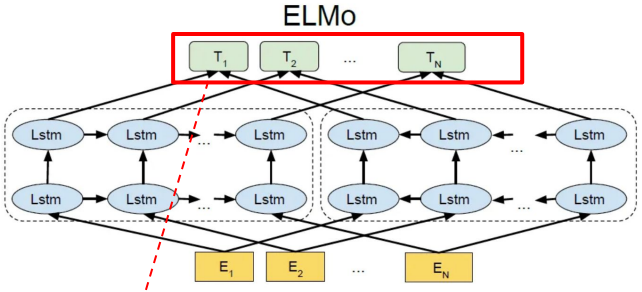
Duas camadas BiLSTM, que são combinadas para gerar uma representação contextual para cada palavra

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo:
  - Treina dois modelos separadamente (BiLSTM)



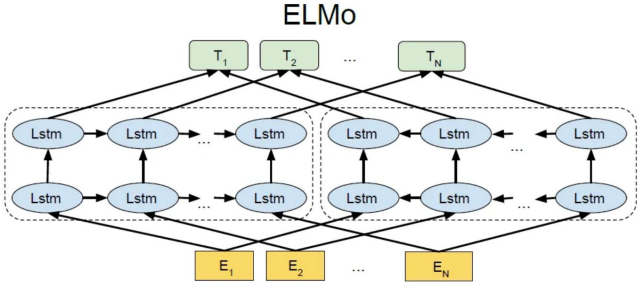
Representações finais para cada token de entrada levando em conta o contexto completo da sentença

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo:
  - Treina dois modelos separadamente (BiLSTM):
    - *Forward Language Model* (prevê próxima palavra)
    - *Backward Language Model* (prevê palavra anterior)
  - Os dois conjuntos de embeddings são combinados

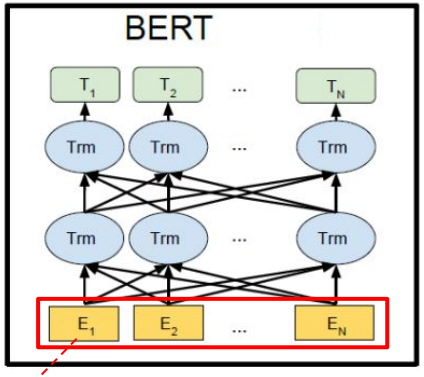


Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo
- BERT:
  - Utiliza arquitetura baseada em Transformer "*encoder-only*"



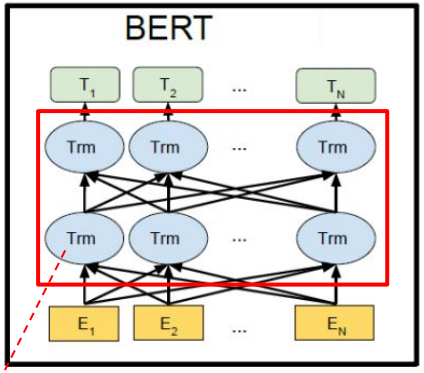
Embeddings de entrada para cada token na sequência:
- Token embedding: representação vetorial do token
- Segment embedding: indica qual sentença o token pertence
- Position embedding: captura posição do token na sequência

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

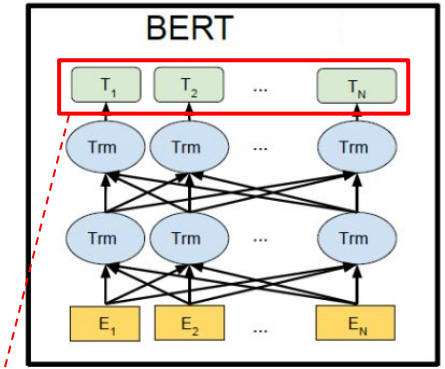
- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo
- BERT:
  - Utiliza arquitetura baseada em Transformer "*encoder-only*"



Camada Transformer Encoder:
- Self-attention: cada token presta atenção a todos os outros tokens na sequência para capturar o contexto bidirecional
- Feed-forward neural network: rede neural densa que processa a saída da camada de autoatenção, permitindo que o modelo aprenda representações mais complexas

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*



3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo
- BERT:
  - Utiliza arquitetura baseada em Transformer "*encoder-only*"

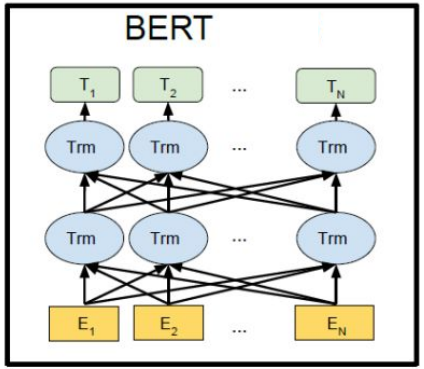
Representações de saída:
- Cada token possui seu embedding contextual equivalente, capturando tanto o significado da palavra individualmente quanto a relação com todas as outras palavras da sentença

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- O modelo é treinado em um grande corpus de texto de forma auto-supervisionada
- Objetiva aprender representações gerais da linguagem
  - Muito custoso!
- ELMo
- BERT:
  - Utiliza arquitetura baseada em Transformer "*encoder-only*"
  - O treino é feito de forma bidirecional
  - É pré-treinado em duas tarefas principais:
    - 1. *Masked Language Modeling* (MLM)
    - 2. *Next Sentence Prediction* (NSP)



Large Language Models (NLU - encoder-only)

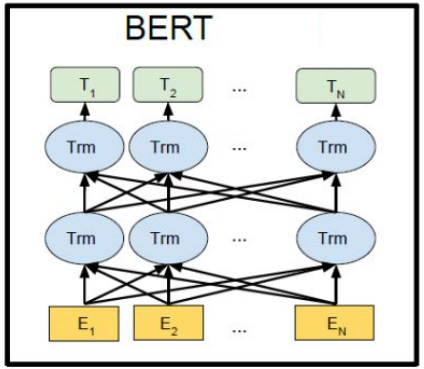
1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

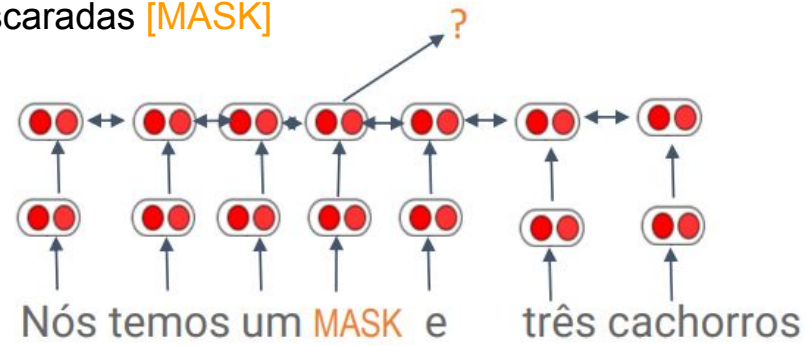
3. Pré-treino

- BERT:
  - 1. *Masked Language Modeling* (MLM):
    - Técnica de pré-treino onde o BERT aprende a prever palavras mascaradas em uma sentença
    - Busca prever palavras mascaradas usando o contexto ao redor
    - Cerca de 15% das palavras são mascaradas [MASK]
    - Ex:
      "O [MASK] está dormindo no sofá".

      > BERT usa contexto (resto da frase) para prever [MASK] como "gato"

      Modelo é penalizado se errar a previsão



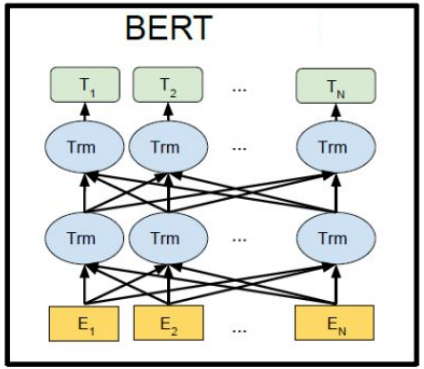


Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

- BERT:
  - 2. *Next Sentence Prediction* (NSP):
    - Introduzida para ajudar o BERT a compreender relações entre as sentenças
    - Tarefa de classificação binária:
      - A partir de um par de sentenças, descobrir se a segunda segue a primeira ou não (se é aleatória)
    - Estratégia de criação de sentenças:
      - 50% dos pares são extraídos consecutivamente do texto
      - 50% dos pares são formados por sentenças aleatórias

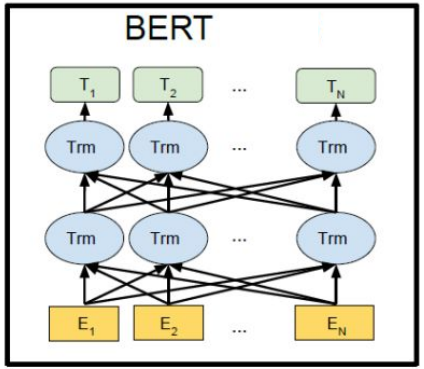


Input = [CLS] the man went to [MASK] store [SEP]
he bought a gallon [MASK] milk [SEP]
Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
penguin [MASK] are flight ##less birds [SEP]
Label = NotNext

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

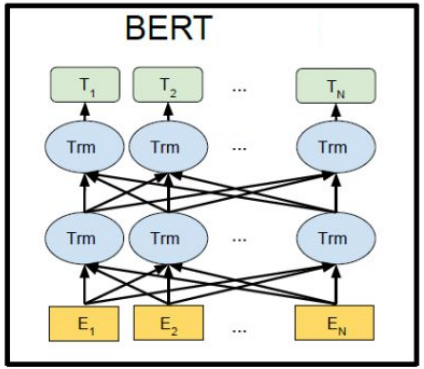


3. Pré-treino

- BERT:
  - Visão geral:
    - 1. Grandes corpora de texto são selecionados (Wikipedia + BooksCorpus)
    - 2. Texto é tokenizado via WordPiece
    - 3. Tokens são passados em diversas camadas “Encoder” do Transformer de forma bidirecional
    - 4. O modelo é treinado simultaneamente para:
      - Prever palavras mascaradas (MLM)
      - Prever se uma sentença segue a outra (NSP)
    - 5. A loss function é minimizada pela combinação de ambas tarefas em (4)

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

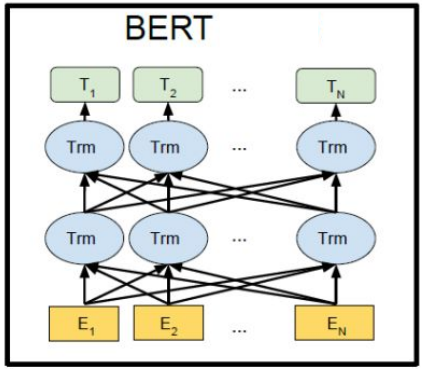


3. Pré-treino

- BERT:
  - Saída da camada final:
    - Matriz de tamanho (N, d_{hidden}), onde:
      - N é o número de tokens da sequência de entrada
      - d_{hidden} é a dimensão do embedding resultante
    - No modelo BERT-base, d_{hidden} possui dimensão 768
    - No modelo BERT-large, d_{hidden} possui dimensão 1024
  - Assim:
    - Cada linha da matriz é um embedding contextual do seu respectivo token de entrada (considerando o contexto completo da sentença)

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*



3. Pré-treino

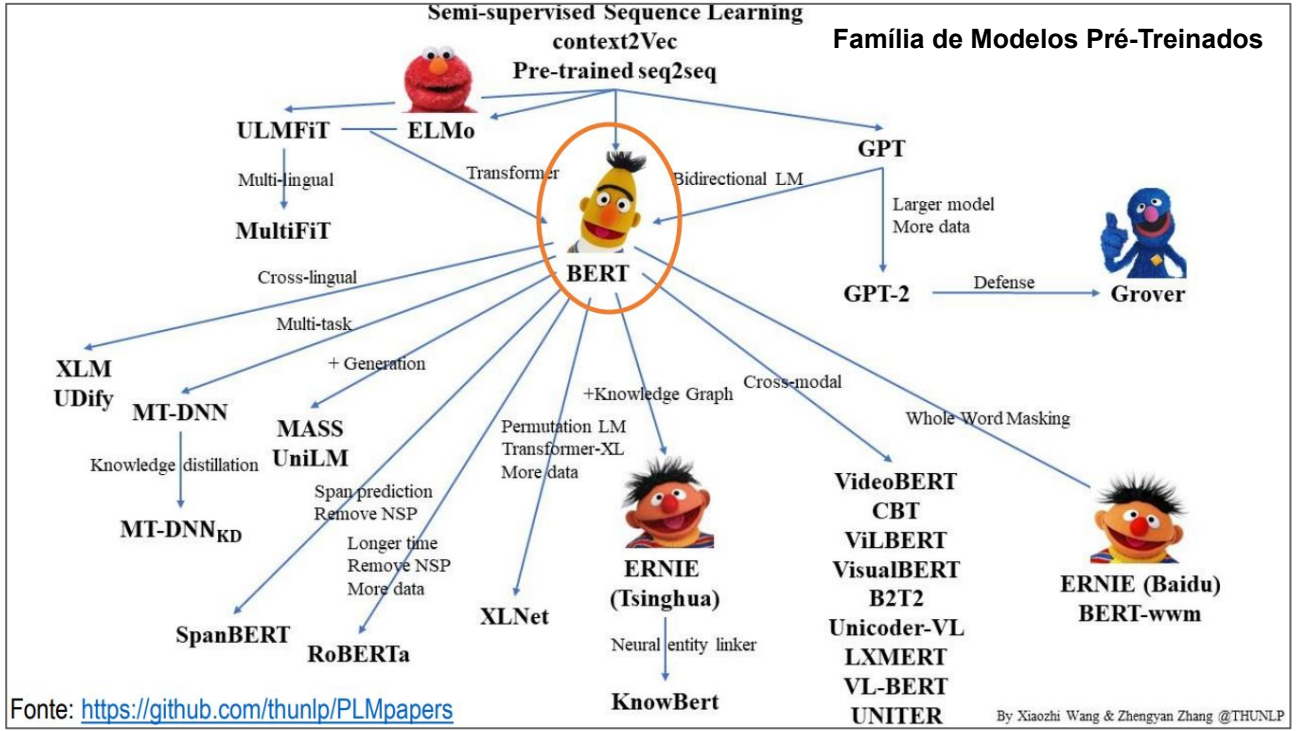
- BERT:
  - Tokens especiais na saída:
    - [CLS] (Classification): é adicionado ao início da sequência e sua saída é usada como uma representação agregada da sentença inteira
    - [SEP] (Separator): é usado para indicar o final de uma sentença ou para separar duas sentenças

[CLS] Token1, Token2, ..., TokenN, [SEP]

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino



Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

Method	Architecture	Encoder	Decoder	Objective	Dataset
ELMo	LSTM	✗	✓	LM	1B Word Benchmark
GPT	Transformer	✗	✓	LM	BookCorpus
GPT2	Transformer	✗	✓	LM	Web pages starting from Reddit
BERT	Transformer	✓	✗	MLM & NSP	BookCorpus & Wiki
RoBERTa	Transformer	✓	✗	MLM	BookCorpus, Wiki, CC-News, OpenWebText, Stories
ALBERT	Transformer	✓	✗	MLM & SOP	Same as RoBERTa and XLNet
UniLM	Transformer	✓	✗	LM, MLM, seq2seq LM	Same as BERT
ELECTRA	Transformer	✓	✗	Discriminator (o/r)	Same as XLNet
XLNet	Transformer	✗	✓	PLM	BookCorpus, Wiki, Giga5, ClueWeb, Common Crawl
XLM	Transformer	✓	✓	CLM, MLM, TLM	Wiki, parellel corpora (e.g. MultiUN)
MASS	Transformer	✓	✓	Span Mask	WMT News Crawl
T5	Transformer	✓	✓	Text Infilling	Colossal Clean Crawled Corpus
BART	Transformer	✓	✓	Text Infilling & Sent Shuffling	Same as RoBERTa

Fonte: Liu, Qi, Matt J. Kusner, and Phil Blunsom. "A survey on contextual embeddings." arXiv preprint arXiv:2003.07278 (2020).

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

Modelo	Camadas (L)	Unidades ocultas (H)	Cabeças de Atenção (A)	Parâmetros
BERT-base-uncased	12	768	12	110 Milhões
BERT-large-uncased	24	1024	16	336 Milhões
BERT-base-multilingual-cased	12	768	12	179 Milhões
ALBERT-base	12	768	12	11 Milhões
GPT-2	12	768	12	117 Milhões
GPT-2-Large	26	1280	20	774 Milhões
T5-base	12	768	12	220 Milhões
T5-large	24	1024	16	770 Milhões
T5-11B	24	1024	128	11 Bilhões
BART-base	12	768	16	139 Milhões
BART-large	24	1024	16	406 Milhões
GPT-3	96	2048	128	175 Bilhões
GPT-4	?	?	?	?

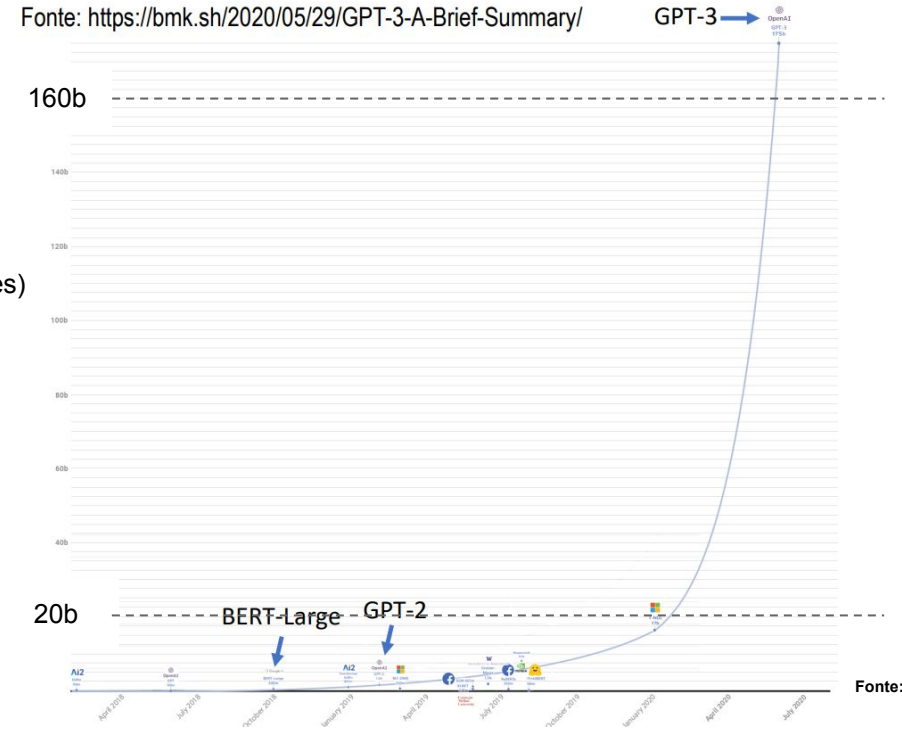
Fonte: https://huggingface.co/transformers/pretrained_models.html

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

3. Pré-treino

Número de parâmetros (bilhões)



Fonte: https://bmk.sh/2020/05/29/GPT-3-A-Brief-Summary/

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

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:
  - Classificação de texto
  - Reconhecimento de entidades nomeadas
  - Análise de sentimentos
  - …

- *Fine tuning* permite especializar o modelo em uma tarefa específica

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- É baseado na técnica de *transfer learning*:
  - Modelo de aprendizado profundo treinado em um grande conjunto de dados é usado para realizar tarefas semelhantes em outro conjunto de dados

- O modelo “grande” é o pré-treinado (Etapa 3)

- Ideia geral: é melhor usar um modelo pré-treinado como ponto de partida para resolver um problema em vez de construir um modelo do zero (“reuso de conhecimento geral”)

- É um processo menos custoso do que o pré-treino

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

+ Desempenho
+ Custo

- Formas:
  - 1. Treinar toda a arquitetura:
    - O modelo pré-treinado é totalmente ajustado durante o treinamento supervisionado da nova tarefa
    - Todos os pesos são atualizados com base no novo conjunto de dados
  - 2. Treinar algumas camadas e congelar outras:
    - Mantém o peso das camadas iniciais do modelo congelados, retreinando apenas as camadas superiores
  - 3. Congelar toda a arquitetura (treinar apenas camadas novas):
    - Acrescentar novas camadas, treinando somente elas

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- BERT:
  - Exemplos:
    - 1. Classificação de sentimento:
      Entrada: “O filme foi incrível!”
      Saída: embedding do token [CLS] (1,768) é usado para prever se a classe é positiva ou negativa via camada de classificação
      - Adiciona-se uma *fully connected layer* sobre o embedding do [CLS]
      - Adiciona-se uma função de ativação sigmóide (binária) ou softmax (multiclasse)
      - Treinamento apenas da camada de classificação (o BERT em si não é ajustado)

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- BERT:
  - Exemplos:
    - 2. Reconhecimento de Entidades Nomeadas:
      Entrada: “Barack Obama nasceu no Havaí”
      Tokens: `["[CLS]", "Barack", "Obama", "nasceu", "no", "Ha", "##vai", "[SEP]"]`
      Saída: embedding de todos os tokens (pois cada token precisa ser classificado)
      - Adiciona-se uma *fully connected layer* sobre os embeddings de cada token para prever o rótulo correspondente (PERSON, LOCATION, O, etc.)
      - Adiciona-se uma função de ativação softmax (multiclasse) aplicada a cada token individualmente

Large Language Models (NLU - encoder-only)

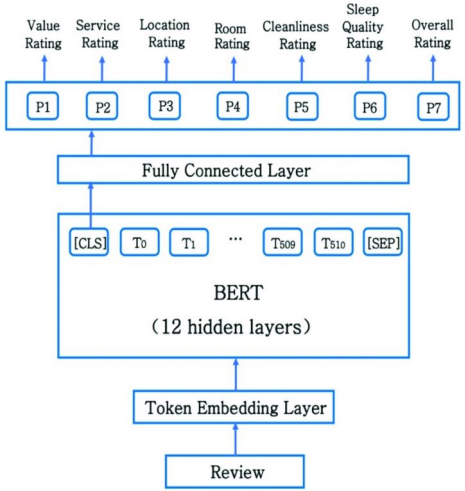
1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- BERT:
  - Exemplos:
    - 3. Sistema de recomendação de hotel

      Tarefa de regressão para 7 pesos

      "In order to apply the BERT model to the rating prediction task, fine tuning is performed by introducing a fully connected layer in the final hidden state corresponding to the [CLS] input token."



Exemplo de fine-tuning do BERT para regressão

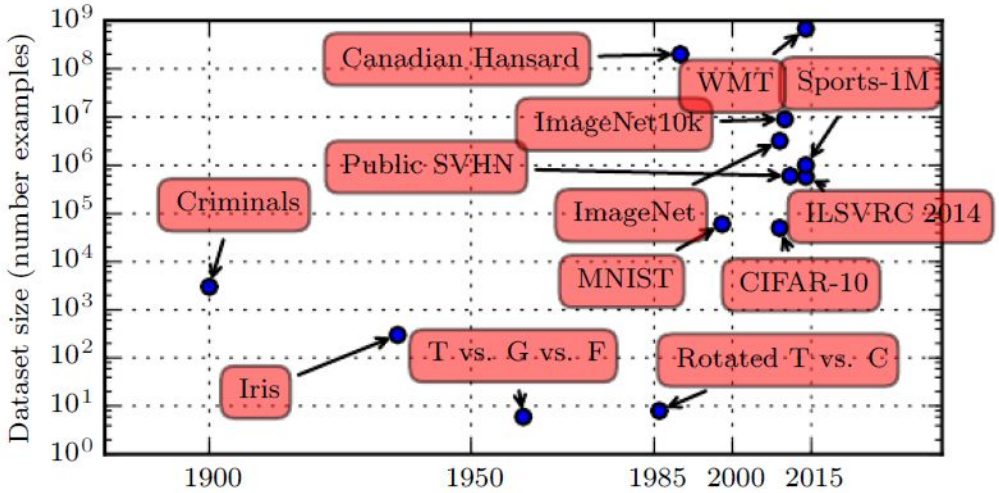
https://www.researchgate.net/publication/353363291_A_BERT-Based_Multi-Criteria_Recommender_System_for_Hotel_Promotion_Management

Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- A escolha da forma do *fine-tuning* depende de:
  - Tamanho do dataset

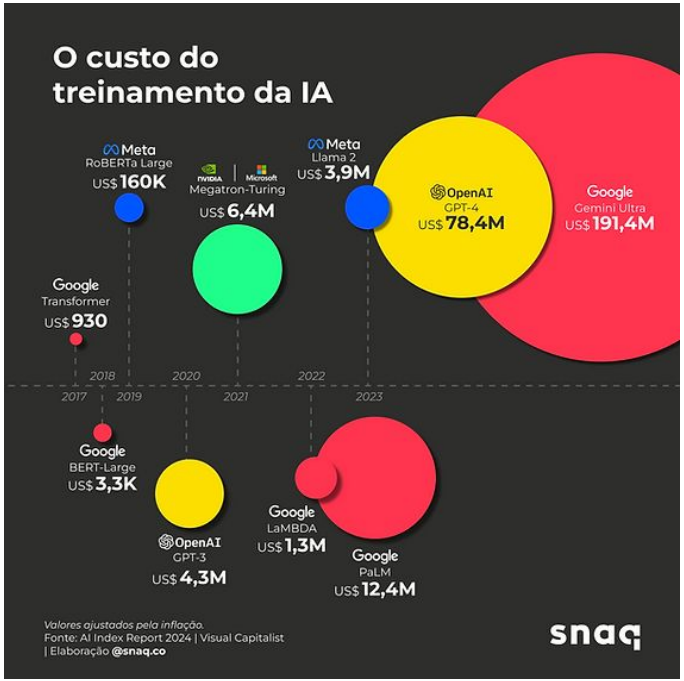


Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- A escolha da forma do *fine-tuning* depende de:
  - Tamanho do dataset
  - Custo computacional

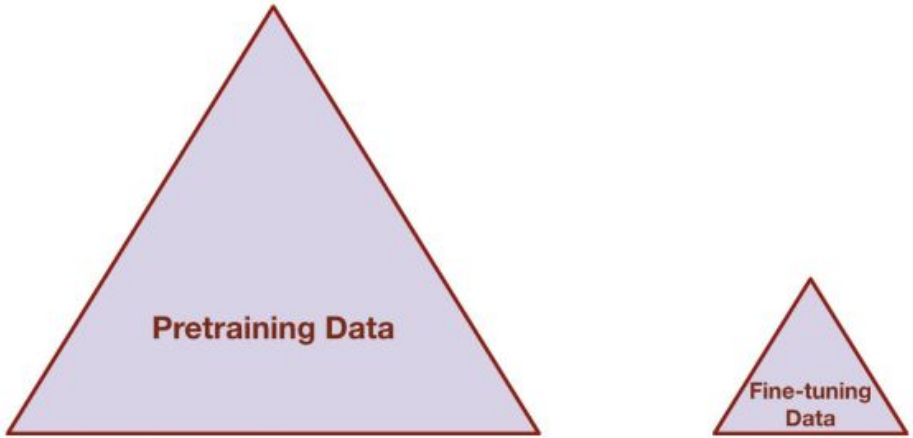


Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

- A escolha da forma do *fine-tuning* depende de:
  - Tamanho do dataset
  - Custo computacional
  - Semelhança entre o pré-treino e a nova tarefa



Large Language Models (NLU - encoder-only)

1. Coleta de dados
2. Pré-processamento
3. Pré-treino
4. *Fine-tuning*

4. *Fine tuning* (ajuste fino)

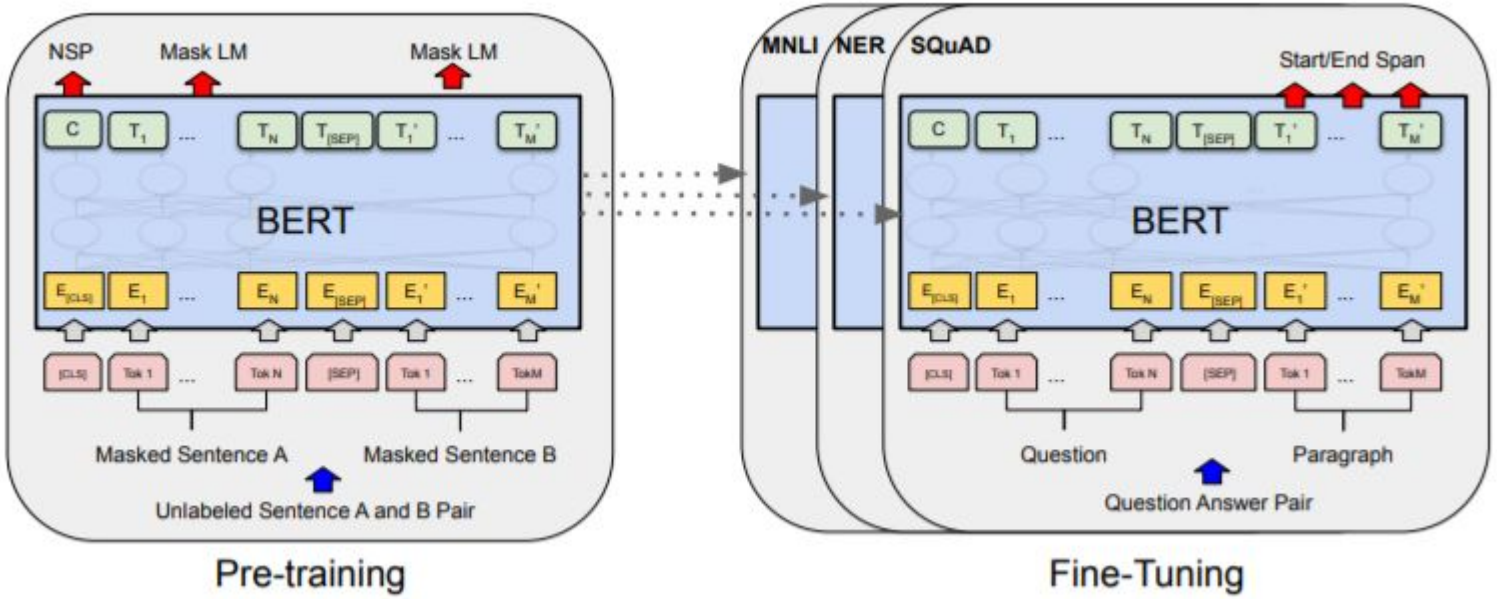
- A escolha da forma do *fine-tuning* depende de:
  - Tamanho do dataset
  - Custo computacional
  - Semelhança entre o pré-treino e a nova tarefa

- Todas as abordagens são válidas (dependem do cenário)

Large Language Models (NLU - encoder-only)

- Pré-treino vs. Continuação de pré-treino vs. *Fine-tuning*
  - Pré-treino: treinamento inicial em grandes corpora genéricos para aprender representações gerais da linguagem
  - Continuação do Pré-treino: treinamento adicional em um corpus especializado para adaptar o modelo a um novo domínio ou idioma, ainda sem foco em uma tarefa específica
  - Fine-tuning: ajuste supervisionado para uma tarefa específica (e.g., classificação de texto), usando dados rotulados para maximizar a performance nessa tarefa

Large Language Models (NLU - encoder-only)



Large Language Models (NLU - encoder-only)

- Onde conseguir modelos pré-treinados?
  - Diretamente na página dos criadores
  - Na Hugging Face 
    - Centenas de milhares de modelos disponíveis
    - https://huggingface.co/transformers/pretrained_models.html

Model id	Details of the model
bert-base-uncased	12-layer, 768-hidden, 12-heads, 110M parameters. Trained on lower-cased English text.
bert-large-uncased	24-layer, 1024-hidden, 16-heads, 336M parameters. Trained on lower-cased English text.
bert-base-cased	12-layer, 768-hidden, 12-heads, 109M parameters. Trained on cased English text.
bert-large-cased	24-layer, 1024-hidden, 16-heads, 335M parameters. Trained on cased English text.
bert-base-multilingual-uncased	(Original, not recommended) 12-layer, 768-hidden, 12-heads, 168M parameters. Trained on lower-cased text in the top 102 languages with the largest Wikipedias

LLMs para o Português

- BERT multilíngue:
  - Original do Google - Treinado em 104 idiomas (Wikipedia)
  - https://github.com/google-research/bert/blob/master/multilingual.md

- Versão criada apenas com dados em Português:
  - BertPT e AlbertPT (Diego Feijó e Viviane Moreira - UFRGS)
  - https://github.com/diego-feijo/bertpt

- Versão que parte do BERT multilíngue e continua o pré-treinamento com dados em Português:
  - BERTimbau (Fábio Souza, Rodrigo Nogueira e Roberto Lotufo - Unicamp)
  - https://github.com/neuralmind-ai/portuguese-bert

Variações do BERT

- Variações para diferentes domínios, idiomas e aplicações específicas:
  - Modelos para o português:
    - BERTimbau
    - Portuguese BERT (PT-BERT)
  - Modelos multilíngues:
    - mBERT (Multilingual BERT)
    - XLM-R (XLM-RoBERTa)
  - Modelos adaptados para domínios específicos:
    - BioBERT (textos biomédicos)
    - LegalBERT (textos jurídicos)
    - BERTikal (textos legais)

Variações do BERT

- Variações em termos de eficiência:
  - DistilBERT:
    - Versão reduzida e mais eficiente do modelo BERT, desenvolvida pela Hugging Face
    - Utiliza uma técnica chamada Distillation (destilação de conhecimento), que permite treinar um modelo menor (o aluno) para imitar o comportamento de um modelo maior e mais complexo (o professor)
    - Possui cerca de 60% dos parâmetros do BERT-base, mas mantém 97% da performance
    - É 40% mais rápido durante o treinamento e inferência
    - Muito interessante de ser usado “previamente”!

Variações do BERT

- Variações na quantidade de parâmetros do treinamento:

Modelo	Camadas (Layers)	Dimensão Oculta (Hidden Size)	Cabeças de Atenção	Parâmetros Totais
DistilBERT	6	768	12	**66 milhões**
BERT-base	12	768	12	**110 milhões**
BERT-large	24	1024	16	**340 milhões**
TinyBERT	4	312	12	**14,5 milhões**
MiniBERT	4	256	4	**11 milhões**
RoBERTa-base	12	768	12	**125 milhões**
RoBERTa-large	24	1024	16	**355 milhões**
ALBERT-base	12	768	12	**12 milhões** (compartilhamento de parâmetros)
ALBERT-large	24	1024	16	**18 milhões** (compartilhamento de parâmetros)

Como o BERT funciona?

- Bertologia!
  - Sabemos que o BERT funciona muito bem, mas ainda não sabemos direito porque
- Constatações importantes*
  - BERT consegue capturar concordância verbal
  - BERT tem dificuldades com representações de números
  - Conhecimento do mundo:
    - Em tarefas de completar sentenças, assemelha-se a sistemas que usam bases de conhecimento
    - Tem algum conhecimento de mundo – mas não consegue raciocinar a partir dele
    - Consegue lidar com polissemia (sentidos distintos formam clusters distintos)

* Rogers, Anna, Olga Kovaleva, and Anna Rumshisky. "A primer in bertology: What we know about how BERT works." Transactions of the Association for Computational Linguistics 8 (2020): 842-866.

Limitações (BERT)

- Tamanho da entrada: máximo 512 tokens (subpalavras)
  - Limitar o tamanho ajuda a equilibrar a eficiência computacional e a capacidade de capturar contexto suficiente
  - Estudos em corpora de texto, como Wikipedia e livros, mostraram que a maioria das sentenças e parágrafos pode ser representada com menos de 512 tokens
  - Os exemplos do pré-treino são gerados a partir de parágrafos e sentenças de tamanho variável, mas truncados para um máximo de 512 tokens

- Se o texto de entrada tiver mais que 512 tokens:
  - Truncamento
  - Divisão em partes menores com sobreposição (*sliding window*)

Limitações (BERT)

- Tamanho da entrada: máximo 512 tokens (subpalavras)
- Não lida bem com negação
- Não tem senso comum

prateleira madeira

Context	$\text{BERT}_{\text{LARGE}}$ predictions
Pablo wanted to cut the lumber he had bought to make some shelves. He asked his neighbor if he could borrow her ____	*car, house, room, truck, apartment*
The snow had piled up on the drive so high that they couldn't get the car out. When Albert woke up, his father handed him a ____	*note, letter, gun, blanket, newspaper*
At the zoo, my sister asked if they painted the black and white stripes on the animal. I explained to her that they were natural features of a ____	*cat, person, human, bird, species*

Context	$\text{BERT}_{\text{LARGE}}$ predictions
A robin is a ____	*bird, robin, person, hunter, pigeon*
A daisy is a ____	*daisy, rose, flower, berry, tree*
A hammer is a ____	*hammer, tool, weapon, nail, device*
A hammer is an ____	*object, instrument, axe, implement, explosive*
A robin is not a ____	*robin, bird, penguin, man, fly*
A daisy is not a ____	*daisy, rose, flower, lily, cherry*
A hammer is not a ____	*hammer, weapon, tool, gun, rock*
A hammer is not an ____	*object, instrument, axe, animal, artifact*

Ettinger, Allyson. "What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models." Transactions of the Association for Computational Linguistics 8 (2020): 34-48.

Custo?

- GPUs/TPUs estão mais baratas, mas o tamanho dos modelos vem aumentando absurdamente

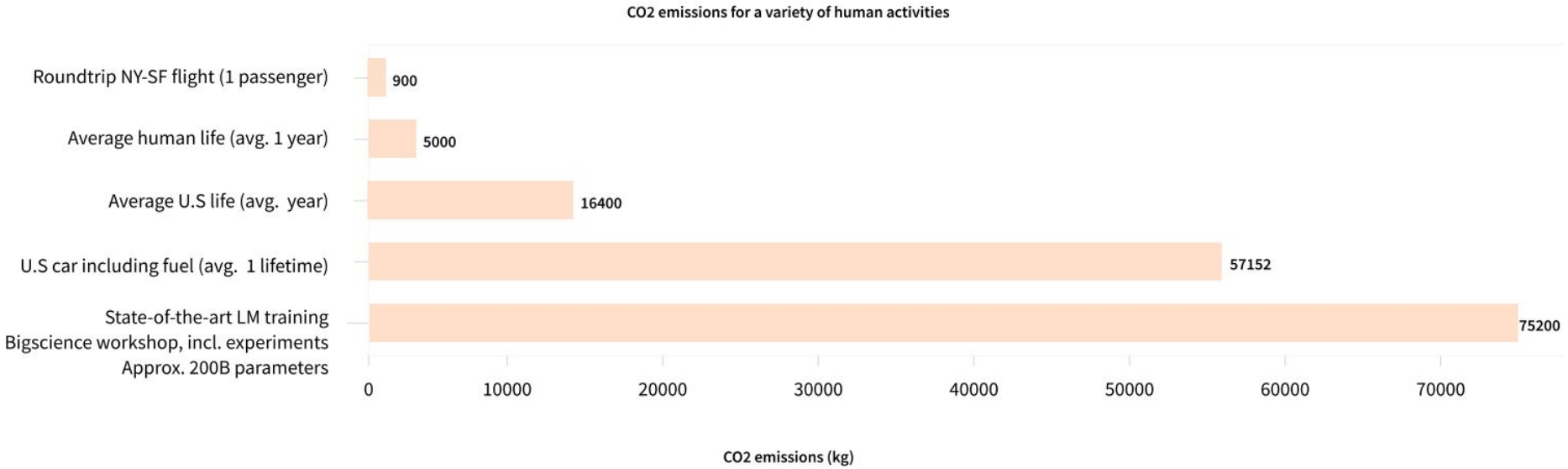
- Depende:
  - Do tamanho
  - Da quantidade de treinamentos:
    - Ajuste de parâmetros
    - Minimizar efeitos aleatórios

- Estimativas:
  - $2.5k - $50k (110 milhões de parâmetros)
  - $10k - $200k (340 milhões de parâmetros)
  - $80k - $1.6m (1.5 bilhões de parâmetros)

CPU GPU TPU

Sharir, Or, Barak Peleg, and Yoav Shoham. "The cost of training NLP models: A concise overview." arXiv preprint arXiv:2004.08900 (2020).

Impacto ambiental?



Fonte: https://huggingface.co/course/chapter1/4

Próxima aula

- Representação de textos com word embeddings contextuais e modelos de linguagem neurais [2]:
  - Grandes modelos de linguagem decoder (GPT, etc.)
  - Grandes modelos de linguagem encoder-decoder (T5, etc.)
  - Aprendizado com poucos dados (zero-shot, one-shot e few-shot learning)

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