Deep Learning e Text-as-Data

Minicurso

Felipe Lamarca

10º Seminário Discente do IESP-UERJ

2025-11-12

Itinerário do minicurso

1. Primeiro vamos falar de Deep Learning. Com isso, quero dizer que vamos mergulhar nas seguintes perguntas:

- o que é uma rede neural e como ela funciona?
- como se treina uma rede neural?
- qual é a relação entre redes neurais com modelos que já conhecemos (por exemplo, a regressão linear)?

2. Depois vamos falar de Text-as-Data **com** Deep Learning. Isto é:

- o que é text-as-data **sem** Deep Learning?
- o que é text-as-data **com** Deep Learning?
- como funciona uma rede neural com dados textuais?
- e como podemos usar isso nas nossas pesquisas?!

É ambicioso, mas tomara que dê certo.

Parte 1: Deep Learning

Pelo começo: o que é *Deep Learning*?

O deep learning é um subconjunto do aprendizado de máquina que usa redes neurais de várias camadas, chamadas de redes neurais profundas, para simular o complexo poder da tomada de decisão do cérebro humano. Alguma forma de deep learning alimenta a maioria das aplicações de inteligência artificial (IA) em nossas vidas atualmente. (Holdsworth and Scapicchio 2025)

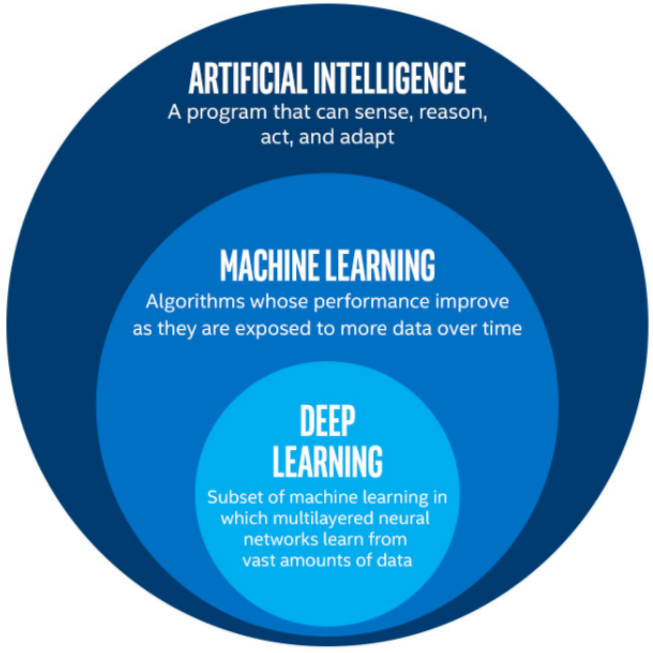


Figure 1: Wowza

O que é uma rede neural?

Uma rede neural artificial é composta por neurônios artificiais (nós) organizados em camadas:

- **Camada de entrada:** recebe os dados (ex.: palavras, números, pixels);
- **Camadas ocultas:** combinam e transformam informações, aprendendo representações;
- **Camada de saída:** produz uma previsão ou classificação.

Cada conexão tem um **peso**, ajustado durante o treinamento para minimizar o erro do modelo.

O que é uma rede neural?

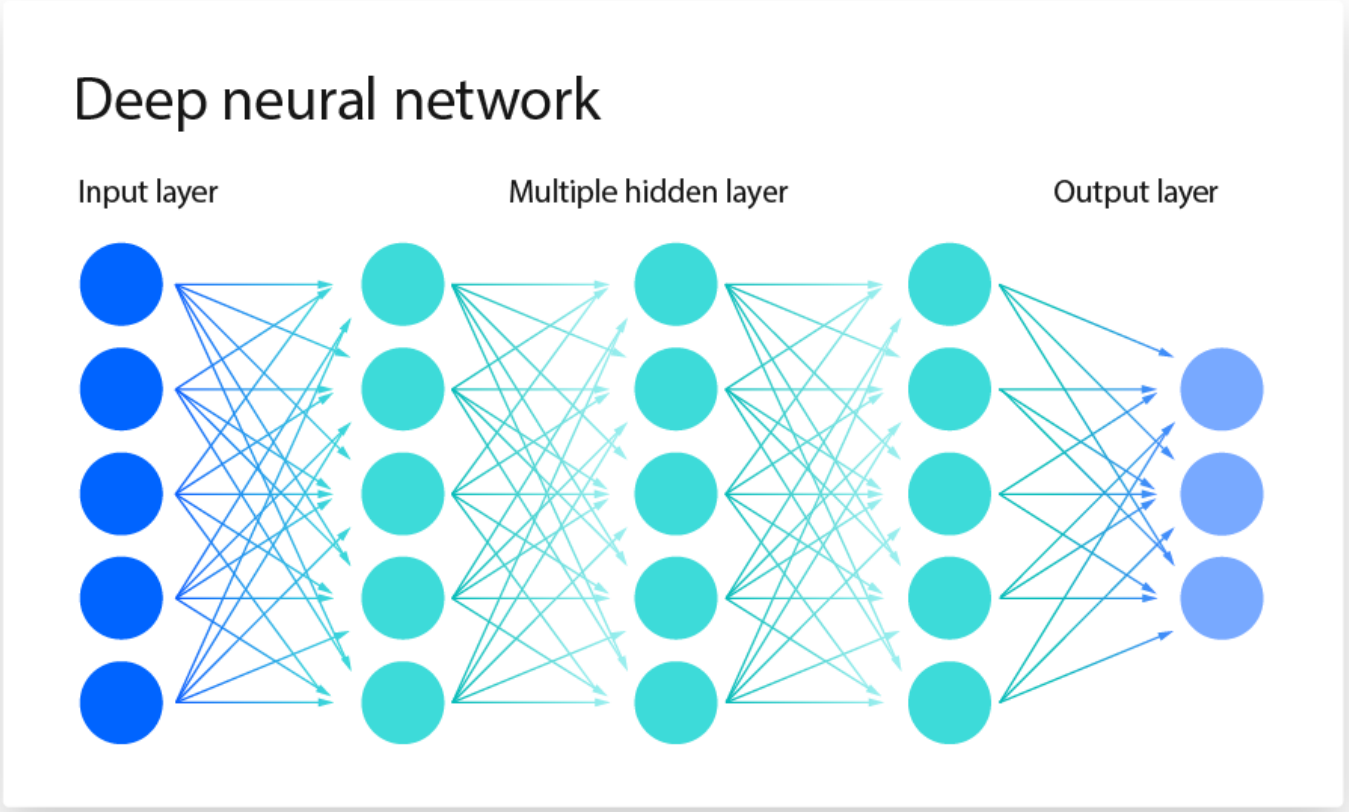


Figure 2: Uma rede neural (IBM 2025)

Como cada neurônio “pensa”?

Uma rede neural combina as entradas $\mathbf{x}$ com **pesos** $\mathbf{w}$ e aplica uma **função de ativação** $\sigma(\cdot)$:

$$A(\mathbf{X}) = \sigma(w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p) = \sigma\left(w_0 + \sum_{p=1}^{P} w_p x_p\right)$$

Essa função de ativação define **como o neurônio reage** a cada combinação de entradas.

Como se treina uma rede neural?

Treinar uma rede neural significa aprender os **pesos** que minimizam alguma função de erro.

Como se treina uma rede neural?

Treinar uma rede neural significa aprender os **pesos** que minimizam alguma função de erro.

Dúvida!

Como se faz isso? E que função de erro é essa?

Como se faz isso?

Para cada observação (ou *batches* de observações) do banco de dados, faça:

1. *Forward pass*: os dados entram na rede e passam camada por camada até gerar uma saída;
2. Cálculo do erro:

- A saída é comparada ao valor verdadeiro;
- A diferença é medida por uma função de erro (ou custo, ou perda, ou *loss*).

3. *Backpropagation*: o erro é propagado de volta pela rede, e o modelo ajusta os pesos proporcionalmente à contribuição de cada um para o erro;
4. Otimização: um algoritmo de gradiente descendente atualiza os pesos um pouquinho a cada iteração;

$$w \leftarrow w - \eta \frac{\partial \text{ Loss}}{\partial w}$$

Que função de erro é essa?

O objetivo do treinamento é minimizar uma função de perda – ou seja, ajustar os pesos para que as previsões fiquem mais próximas dos valores verdadeiros.

- Em problemas de regressão, usa-se o **erro quadrático médio (MSE)**:

$$\text{MSE} = \frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2$$

- Em problemas de classificação binária, é comum usar a **entropia cruzada binária**:

$$\text{Loss} = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]$$

- Em classificações com várias categorias, usa-se a **entropia cruzada categórica**, uma extensão da anterior que compara distribuições inteiras de probabilidades.

Um lembrete sobre derivada

Para ajustar os pesos com base no resultado da função de perda, precisamos saber em qual direção a perda cresce (i.e. o erro aumenta) e em qual direção a perda cai (i.e. o erro diminui).

A derivada nos informa a taxa de variação de uma função univariada; o gradiente nos diz a direção de crescimento de funções multivariadas.

Se formos na direção contrária do gradiente, diminuímos o valor da função! ✋ 😮 ✋

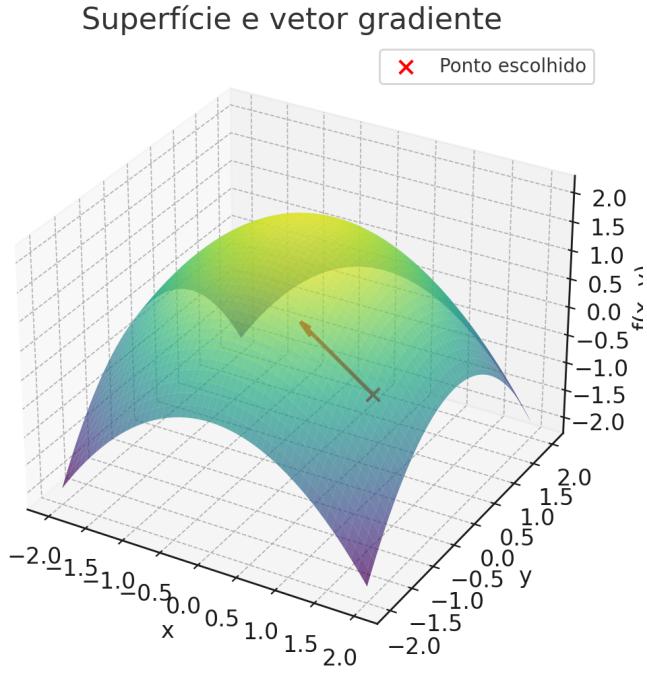


Figure 3: Uma função em $\mathbb{R}^3$ e o vetor gradiente

Uma nota sobre backpropagation

A partir da saída, o algoritmo calcula o quanto cada peso w contribuiu para o erro – isto é, o gradiente $\frac{\partial L}{\partial w}$ (Rumelhart, Hinton, and Williams 1986).

Esse cálculo é feito aplicando a regra da cadeia:

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}$$

onde a é a ativação e z é a combinação linear dos pesos e entradas.

! Eu sei, eu sei...

O importante aqui é entender que a estrutura multicamadas da rede neural torna o cálculo dos pesos algo mais complexo, e o algoritmo de *backpropagation* nos diz quanto cada peso deve ser ajustado.

Prática 1

Notebook 1: Treinando uma rede neural

Vamos voltar a um ponto anterior...

Treinar uma rede neural significa aprender os **pesos** que minimizam alguma função de erro.

Vamos voltar a um ponto anterior...

Treinar uma rede neural significa aprender os **pesos** que minimizam alguma função de erro.

Dúvida!

Isso te lembra alguma coisa?

Vamos voltar a um ponto anterior...

Treinar uma rede neural significa aprender os **pesos** que minimizam alguma função de erro.

Dúvida!

Isso te lembra alguma coisa? Tipo... uma regressão linear?

A rede neural e a regressão linear

Em princípio, o objetivo da regressão linear é encontrar a reta (ou hiperplano) que minimiza o erro quadrático médio:

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon,$$

onde β representa os pesos que ligam as variáveis de entrada $\mathbf{X}$ à saída $\mathbf{Y}$.

A rede neural e a regressão linear

Uma rede neural **sem camadas ocultas** faz exatamente a mesma coisa:

$$\hat{y} = f(\mathbf{w}^\top \mathbf{x} + b).$$

- Se $f(x) = x$, temos uma **regressão linear**.
- Se $f(x) = \sigma(x)$ (função sigmoide), temos uma **regressão logística**.

A rede neural e a regressão linear

Uma rede neural **sem camadas ocultas** faz exatamente a mesma coisa:

$$\hat{y} = f(\mathbf{w}^\top \mathbf{x} + b)$$

- Se $f(x) = x$, temos uma **regressão linear**.
- Se $f(x) = \sigma(x)$ (função sigmoide), temos uma **regressão logística**.

Mas note uma coisa...

Há uma diferença fundamental, no entanto. Na regressão linear, há forma fechada para o β que minimiza o erro – isto é, $\beta = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top y$. No caso da rede neural, o vetor de pesos é encontrado durante o treinamento por uma abordagem iterativa.

Prática 2

Notebook 2: Rede neural e regressão linear

Importante!

Eu falei de mil coisas, mas não falei de outras dez mil. Temas elementares em rede neural não cobertos aqui incluem:

- Funções de ativação, responsáveis por permitir que a rede neural aprende relações não-lineares;
- Algoritmos de otimização (SGD, RMSProp, Adam etc);
- Técnicas de seleção de hiperparâmetros;
- Técnicas para lidar com *overfitting* (*early-stopping*, *dropout* etc)
- Fine-tuning;
- Arquiteturas de rede neural para objetivos específicos (por exemplo, reconhecimento de imagens);
- e por aí vai!

De fato, é um mundo à parte e que evolui todos os dias.

Parte 2: Text-as-Data

Motivação

Nas ciências sociais, em particular, e em várias outras ciências, em geral, textos são fontes de **evidência empírica**:

- Discursos de políticos
- Publicações em redes sociais
- Respostas a perguntas abertas em surveys
- Documentos históricos
- ...

Intuitivamente, a ideia do "texto como dado" é natural, para não dizer óbvia.

Text-as-Data

Podemos analisar textos de várias formas. Nas ciências sociais, a análise de texto por muito tempo esteve restrita a métodos qualitativos: análise de conteúdo, de discurso e assim por diante.

A análise quantitativa de texto é uma oportunidade de analisar corpus textuais **maiores** e de forma mais sistemática.

A evolução do text-as-data

Até recentemente, a análise quantitativa de texto era estritamente baseada em métodos como:

- Bag of Words
- Contagem de palavras
- Métodos baseados em dicionário
- Modelagem de tópico
- N-gramas

A evolução do text-as-data

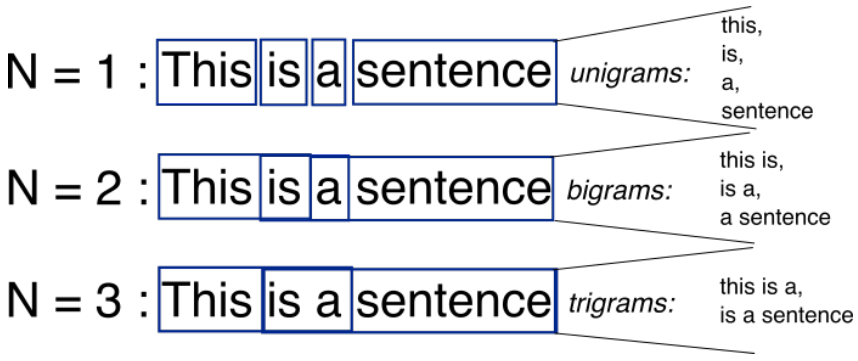


Figure 4: N-grams

	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

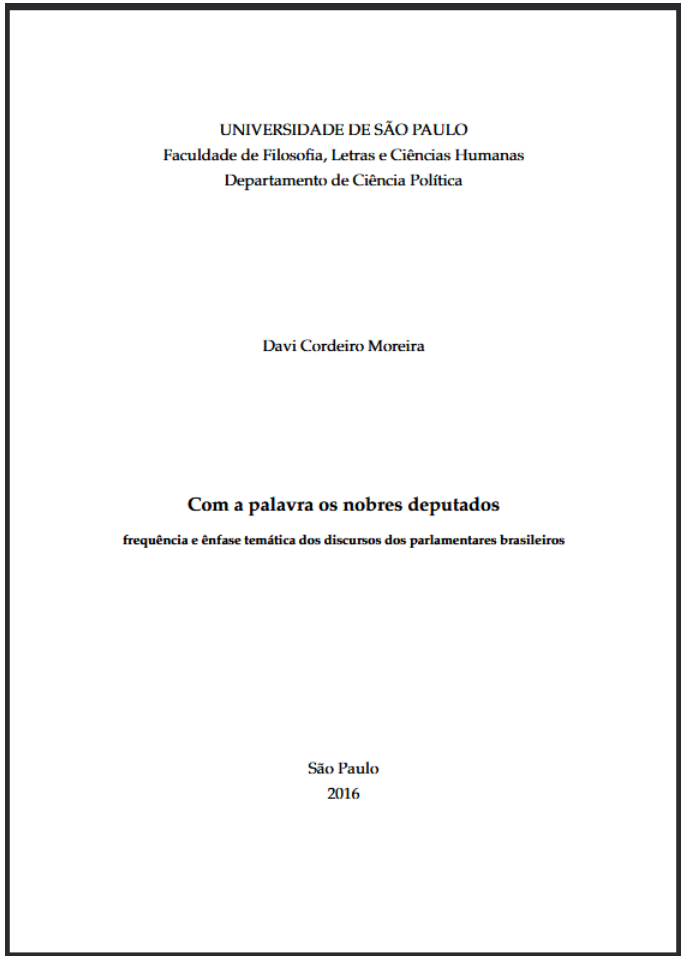
Figure 5: Bag of Words

São, é claro, métodos muito úteis. Mas são incapazes de incorporar **contexto** e **significado** de frases e palavras.

Com a Palavra os Nobres Deputados (Moreira 2020)

Utiliza o **Expressed Agenda Model** (Grimmer 2010), baseado em *topic modeling probabilístico*, para identificar temas e variações na ênfase temática nos discursos de deputados federais.

É um dos primeiros exemplos brasileiros (em ciência política, pelo menos) de como o **texto como dado** amplia a capacidade analítica da ciência política.


UNIVERSIDADE DE SÃO PAULO
Faculdade de Filosofia, Letras e Ciências Humanas
Departamento de Ciência Política

Davi Cordeiro Moreira

Com a palavra os nobres deputados

frequência e ênfase temática dos discursos dos parlamentares brasileiros

São Paulo
2016

Figure 6: Tese de doutorado do Davi Moreira

Embeddings: textos $\rightarrow$ vetores semânticos

O que diferencia um banco de um banco?

Figure 7: Banco

Figure 8: Banco

Embeddings: textos $\longrightarrow$ vetores semânticos

A ideia é que, ao treinarmos um modelo de linguagem, a representação vetorial de palavras e frases passa a capturar algum significado semântico.

Embedding Projector

Transformers

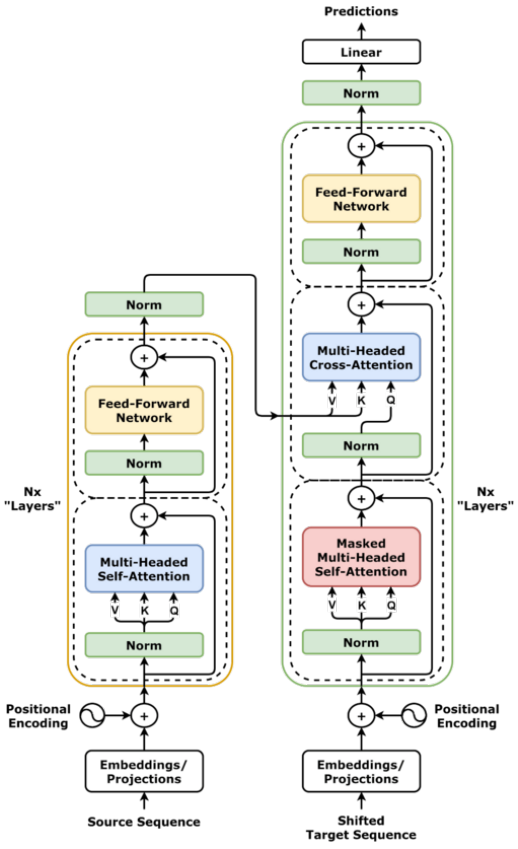


Figure 9: Arquitetura de um Transformer

Transformers são uma arquitetura de rede neural que revolucionou o campo de NLP ao permitir que os modelos aprendam relações contextuais entre palavras em uma frase a partir do mecanismo de *attention*. (Vaswani et al. 2017)

Transformers

Na prática, os *transformers* são capazes de prever a próxima palavra em uma frase, levando em consideração o contexto de todas as palavras anteriores.

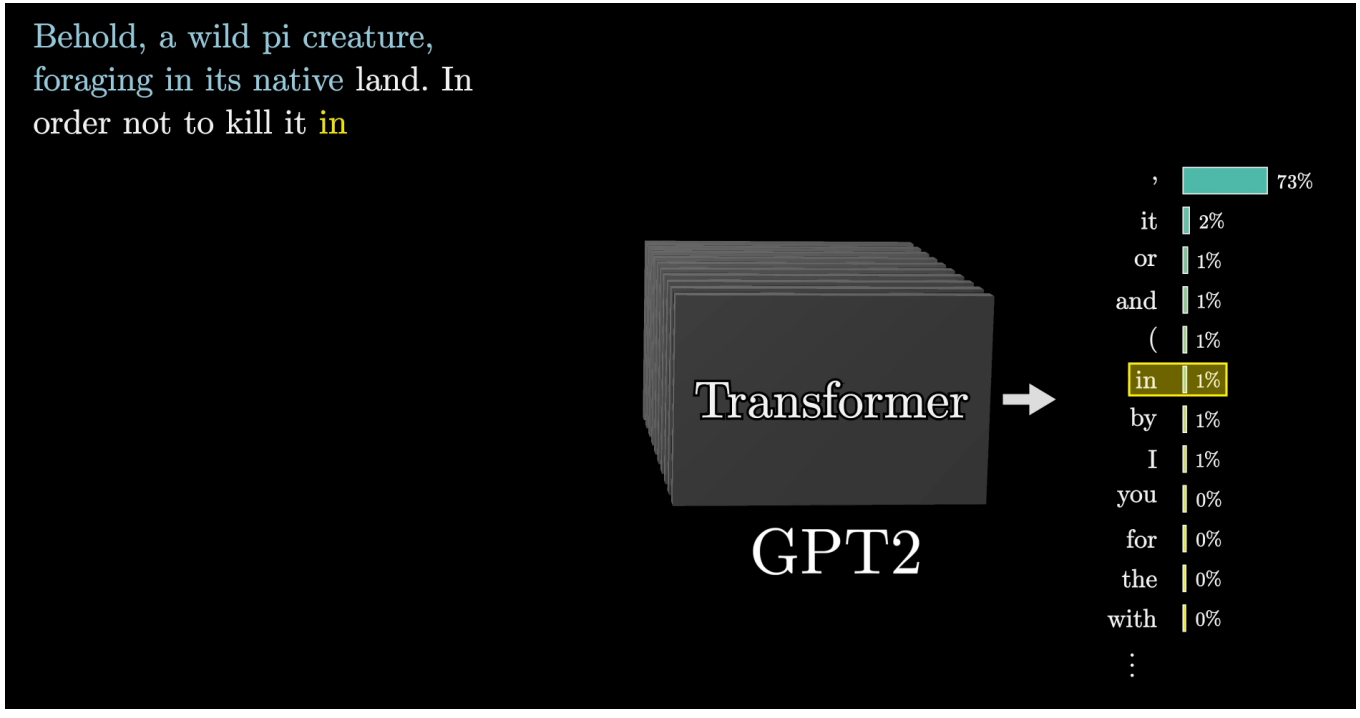


Figure 10: Transformer (3b1b)

Modelos “grandes” e modelos “pequenos”

Em geral, modelos de linguagem podem ser “grandes” ou “pequenos”:

- **Modelos grandes:** treinados com grandes quantidades de dados, geralmente em várias línguas, e com muitos parâmetros. Exemplo: `llama3.1:405b`.
- **Modelos pequenos:** treinados com menos dados e com menos parâmetros. Menor capacidade de generalização. Exemplo: `llama3.2:1b`.

Em geral, quanto maior o modelo, melhor o desempenho; no entanto, menores são as chances de que você consiga rodar o modelo no seu próprio computador.

No que usar?

O céu é o limite! Alguns exemplos:

- *Tradução* de textos
- *Transcrição* de textos
- *Classificação* de textos
- *Cálculo de distância* (i.e., similaridade) entre textos
- Reconhecimento de entidades nomeadas
- Sistemas de recomendação
- E por aí vai...

Como usar?

1. **Opções open-source**

- HuggingFace
- Ollama
- Meta

2. **Opções pagas**

- Modelos open-source via Groq (com limite gratuito diário)
- ChatGPT, da OpenAI
- Grok, da xAI
- Claude, da Anthropic
- Sabiá, da Maritaca AI
- ...

Prática 3

Notebook 3: Usando modelos de linguagem

Algumas referências I

Grimmer, Justin. 2010. "A Bayesian Hierarchical Topic Model for Political Texts: Measuring Expressed Agendas in Senate Press Releases." *Political Analysis* 18 (1): 1–35. https://doi.org/10.1093/pan/mpp034.

Holdsworth, Jim, and Mark Scapicchio. 2025. "O Que é Deep Learning?" IBM; Web page. https://www.ibm.com/br-pt/think/topics/deep-learning.

IBM. 2025. "O Que é Uma Rede Neural?" Web page. https://www.ibm.com/br-pt/think/topics/neural-networks.

Moreira, Davi. 2020. "Com a Palavra Os Nobres Deputados: Ênfase Temática Dos Discursos Dos Parlamentares Brasileiros." *DADOS – Revista de Ciências Sociais* 63 (1): e20180176. https://doi.org/10.1590/001152582020204.

Algumas referências II

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. "Learning Representations by Back-Propagating Errors." *Nature* 323 (6088): 533–36. https://doi.org/10.1038/323533a0.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention Is All You Need." *CoRR* abs/1706.03762. https://arxiv.org/abs/1706.03762.