



F | A | P

INTRODUÇÃO AO PROCESSAMENTO DE LINGUAGEM NATURAL

- RECAPITULAÇÃO
- O QUE É E PARA QUE SERVE A NLP
- CONCEITOS FUNDAMENTAIS
- HANDS ON

RECAPITULANDO:

- APENAS 7% DA COMUNICAÇÃO HUMANA É VERBAL
- 55% DA COMUNICAÇÃO OCORRE EM NÍVEL NÃO VERBAL
- AS PESSOAS SE BASEIAM NOS CANAIS SENSORIAIS DA VISÃO, AUDIÇÃO E TATO PARA INTERAGIR E APRENDER.
- DADOS SOMENTE SÃO ÚTEIS QUANDO LIMPOS E TRATADOS (REFINAMENTO).
- 80% DOS DADOS SÃO NÃO ESTRUTURADOS.
- O CONCEITO DA IA NASCE PARA TRATAR OS DADOS NÃO ESTRUTURADOS.
- DADOS → INFORMAÇÃO → CONHECIMENTO → SABEDORIA

QUESTÕES IMPORTANTES:

- COMO NASCEU A COMUNICAÇÃO HOMEM – MÁQUINA?
- COMO SERÁ A COMUNICAÇÃO HOMEM E MÁQUINA DO FUTURO?
- O QUE SÃO OS TOKENS, EMBEDDED E CHUNKS, UTILIZADOS NA NLP?
- COMO APLICAR OS CONCEITOS FUNDAMENTAIS DA NLP A MEU FAVOR?

O QUE É NLP (PROCESSAMENTO DE LINGUAGEM NATURAL)?

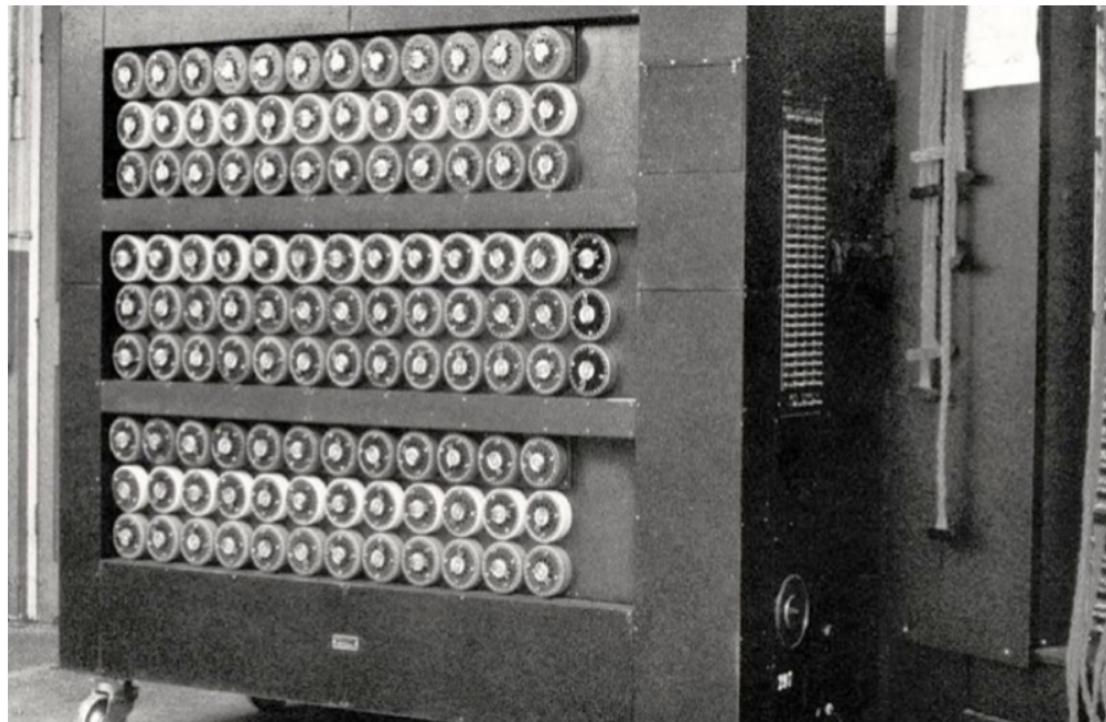
Definição simples:

- É uma área da Inteligência Artificial que permite que computadores entendam, interpretem, gerem e interajam com a linguagem humana.
- Exemplos de aplicações do NLP:
- Corretores ortográficos e gramáticos (como no Word ou no Google Docs).
- Tradução automática (Google Tradutor).
- Pesquisa por voz (Google Assistant, Siri, Alexa).
- Análise de sentimentos (em redes sociais ou SACs).
- Reconhecimento de entidades (nomes de pessoas, locais, datas em textos).

COMUNICAÇÃO HOMEM E MÁQUINA

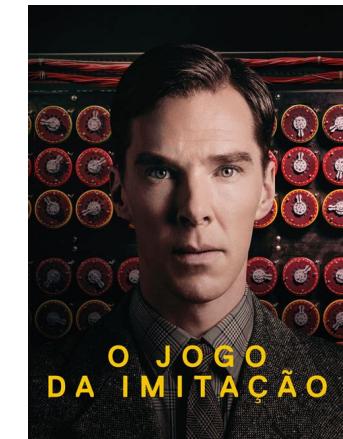
FIAP

Máquina de Turing - 1940

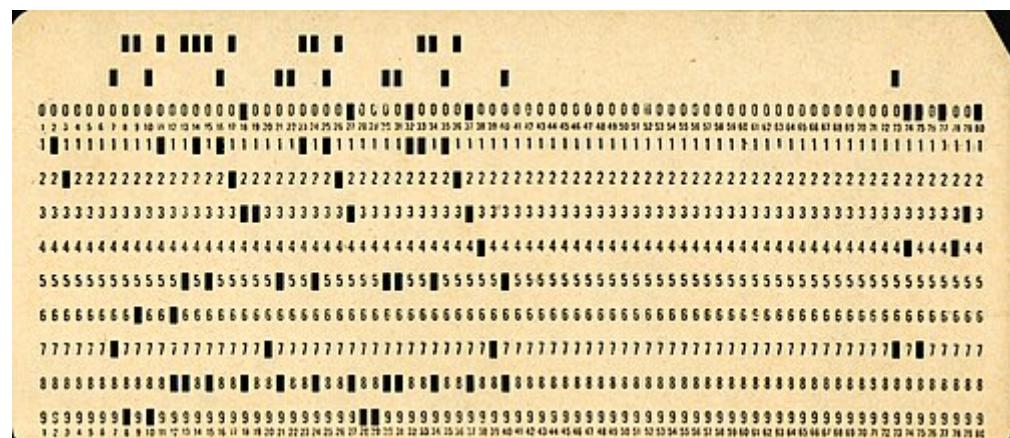


Fonte: <https://heritage.humanists.uk/object/bombe-machine/>

Recomendação Filme: “O Jogo da Imitação”, 2015.



Cartão perfurado IBM 12linhasx80 colunas - 1946



Fonte:Wikipedia

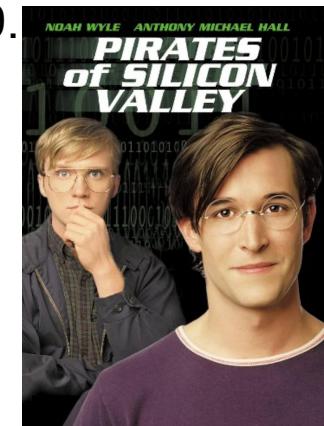
COMUNICAÇÃO VIA PERIFÉRICOS

FIAP

Criação dos principais periféricos

Componente	Ano Criação	Ano Popularização
Teclado	1940	1970
Monitor	1950	1970
Mouse	1964	1984

Recomendação Filme: Os piratas de silicon valley, 1999.



Apple II e IBM PC

Computadores Pessoais:

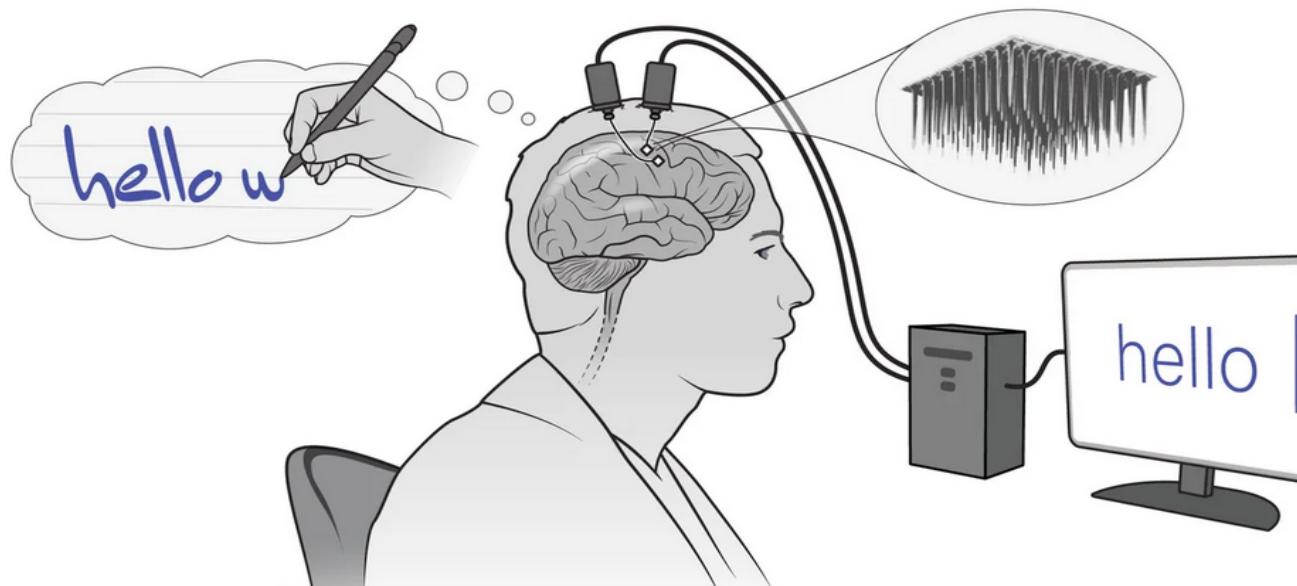
- Kenbak-1: 1971
- Altair 8800 – 1975
- **Apple II - 1977**
- Commodore PET - 1977
- TRS – 80: 1977
- **IBM PC - 1981**



Fonte: Google Images

COMUNICAÇÃO DO FUTURO

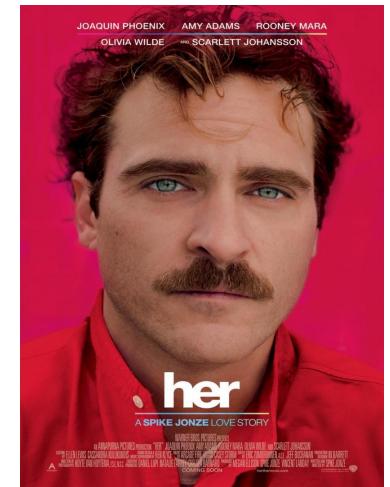
FIAP



Fonte: <https://www.intelligentliving.co/brain-to-text-type-by-thinking/>

Recomendação Filme: Her (ELA), 2013.

Série Black Mirror – Netflix.



Token: é a unidade básica de processamento em NLP. Uma palavra pode ser composta por dois ou mais tokens.

Tipos de Tokenização:

Tokenização por Palavras: (Método clássico)

"Olá mundo!" → ["Olá", "mundo", "!"]

Tokenização por Subpalavras: (Método moderno - BPE, WordPiece)

"inacreditável" → ["in", "acred", "itável"]

"ChatGPT" → ["Chat", "G", "PT"]

Embeddings: é a representação densa de tokens em espaço vetorial multidimensional.

1ª Geração - Estáticos (Word2Vec, GloVe):

Uma palavra = um vetor fixo

"banco" sempre tem o mesmo vetor (não distingue banco financeiro vs banco de praça)

2ª Geração - Contextuais (BERT, GPT):

Uma palavra = vetores diferentes dependendo do contexto

"banco de dados" vs "banco do parque" → vetores diferentes para "banco"

CHUNK: quebra de texto em pedaços menores visando um processamento mais eficiente.

Por que usar chunks?

Modelos têm limite de tokens (ex: GPT-3.5 = 4096 tokens)
Documentos longos precisam ser divididos
Permite processamento paralelo e eficiente

Estratégias de chunking:

Fixo: 512 tokens por chunk

Semântico: Dividir por parágrafos/seções

Overlapping: Chunks com sobreposição para manter contexto

TOP K: representa a seleção dos K elementos mais relevantes ou similares.

Busca: Top-5 documentos mais similares

Geração: Top-10 próximas palavras mais prováveis

Retrieval: Top-3 chunks mais relevantes para uma pergunta

CORPORA/CORPUS: representam grandes coleções de textos para treinar modelos.

Exemplos famosos:

Common Crawl: Trilhões de páginas web

Wikipedia: Conhecimento enciclopédico

Books3: Milhares de livros

OpenSubtitles: Legendas de filmes/séries

Bancos de Dados Vetoriais: database otimizado para busca por similaridade em vetores.

Como funciona:

Documentos → Embeddings → Armazenados no banco

Query → Embedding → Busca por similaridade (coseno, euclidiana)

Retorna Top-K documentos mais similares

Exemplos: Pinecone, Weaviate, Chroma, FAISS

BERT: Bidirectional Encoder Representations from Transformers

TRANSFORMER: Arquitetura baseada em attention mechanism

NLTK: Biblioteca python para NLP tradicional.

Curiosidade:

- O vocabulário em inglês segundo o dicionário Oxford conta com mais de 600 mil palavras, já o vocabulário português possui entre 400 a 500 mil palavras.
- São necessárias apenas 3.000-5.000 palavras para se entender 90%-95% de textos e diálogos comuns.
- Uma pessoa com um vocabulário de 20 mil palavras é considerado alguém com fluência nativa.

NLP – APLICAÇÃO DE CONCEITOS

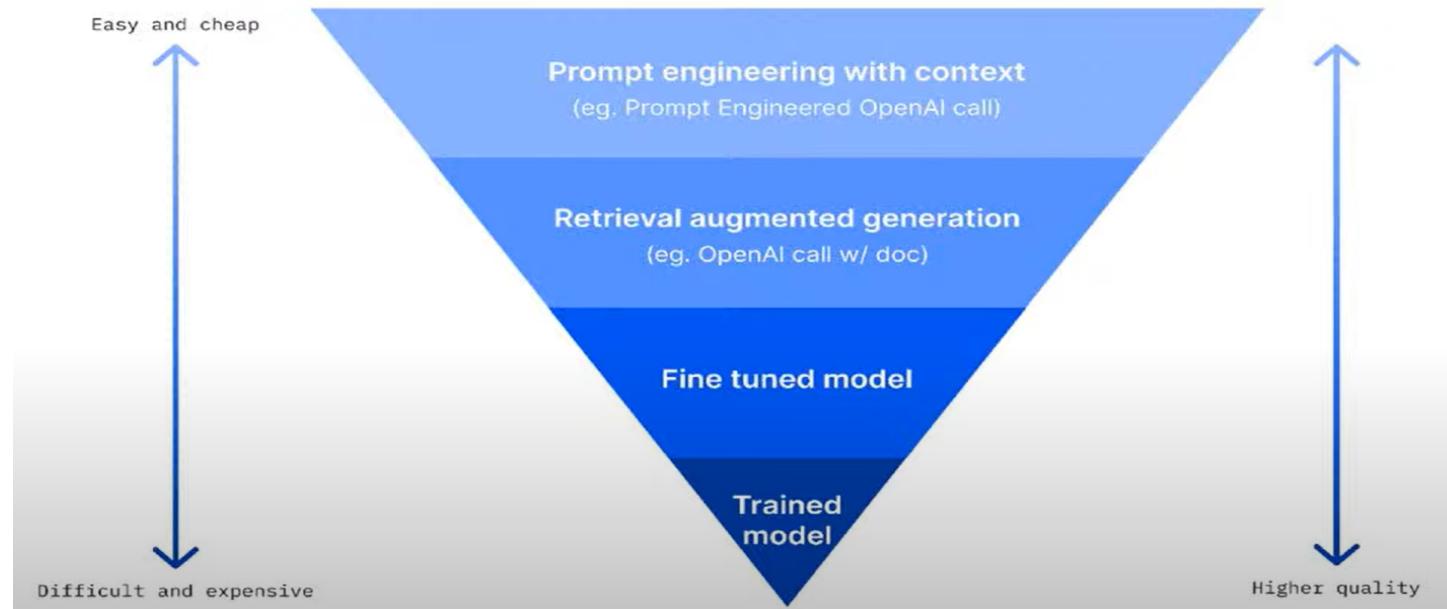
APLICAÇÕES:

ENGENHARIA DE PROMPTS: escrita de instruções clara e precisas para que a IA entenda e realize corretamente o que foi solicitado.

RAG – Retrieval Augmented Generation: técnica para buscar informações em fontes externas antes de responder, garante respostas mais precisas e atualizadas.

FINE TUNING: técnica para ajustar alguns pesos de um modelo já treinado para aprender padrões novos ou para agir conforme um objetivo.

TREINAR MODELOS: ensinar uma IA do zero usando grandes volumes de dados, para que ela entenda padrões e consiga agir e responder conforme o objetivo.



NLP – REPRESENTAÇÃO DE PALAVRAS

ONE-HOT ENCODING: é a forma mais fácil de representar palavras para um computador.

Desvantagens: valores muito grandes para vocabulários grandes e não guarda semelhança semântica.

gato	→ [1, 0, 0, 0]
cachorro	→ [0, 1, 0, 0]
pássaro	→ [0, 0, 1, 0]
peixe	→ [0, 0, 0, 1]

WORD EMBEDDING: é a forma densa e contínua de representar palavras. Cada palavra é um vetor de números reais, permitindo capturar relações semânticas.

gato	→ [0.21, 0.65, 0.34]
cachorro	→ [0.20, 0.60, 0.31]
peixe	→ [-0.12, 0.45, 0.88]

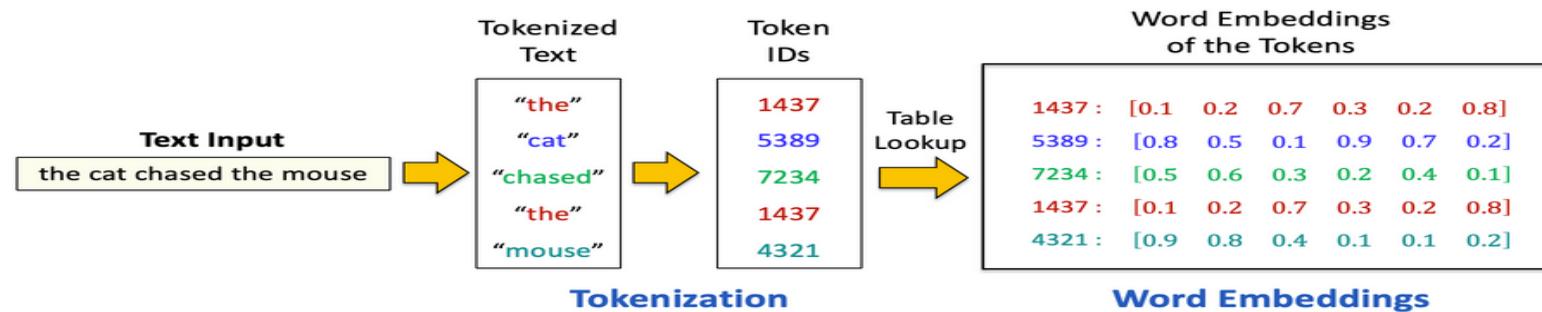
CONTEXTUAL EMBEDDINGS: a representação muda dependendo da frase, capturando o significado individual e o contexto gramatical. Adotado em sistemas mais recentes como BERT, GPT, etc.

Exemplo:

- Sentei no banco da praça.
- Fui ao banco sacar dinheiro.

NLP – TOKEN ID's e STOP WORDS

TOKENIZAÇÃO: representa a transformação do texto em tokens os quais representam palavras, caracteres isolados, pontuação, etc. Cada token possui um número inteiro único em um vocabulário.



IMPORTANTE: token id são a forma como o modelo entende a linguagem natural. Diferentes modelos tem dicionários diferentes. Existem dois métodos de tokenização: Tokenização de palavras e de sentenças.

STOP WORDS: representam palavras que são consideradas irrelevantes para a análise de texto, geralmente por serem comuns e não carregarem significado específico.

Exemplos:

Em português: de, a, o, que, e, do, da, em, um, com, para, por...

Em inglês: the, is, at, which, on, and, a, an, of...

CONTEXT LENGTH OU JANELA DE CONTEXTO: representa o tamanho da janela de atenção que o modelo pode usar, ou seja a quantidade de tokens usados para prever a proxima palavra ou gerar outro texto seguindo o raciocídio anterior.

NLP – REPRESENTAÇÃO DE PALAVRAS



RELAÇÃO SEMÂNTICA: Demonstração da relação semântica usando similaridade pela distância entre vetores.

EMBEDDINGS

Palavra	Vetor (x, y, z)
rei	[0.8, 0.65, 0.1]
rainha	[0.82, 0.7, 0.25]
homem	[0.7, 0.5, 0.05]
mulher	[0.72, 0.55, 0.2]
gato	[-0.4, 0.3, 0.7]
cachorro	[-0.38, 0.28, 0.68]

DISTÂNCIA EUCLIDIANA (COSSENO)

$$\text{similaridade} = \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{\|\mathbf{v}_1\| \|\mathbf{v}_2\|}$$

RELAÇÃO MATEMÁTICA

$$\text{rei} - \text{homem} + \text{mulher} \approx ?$$

Cálculo:

$$1. \text{ rei} - \text{homem} =$$

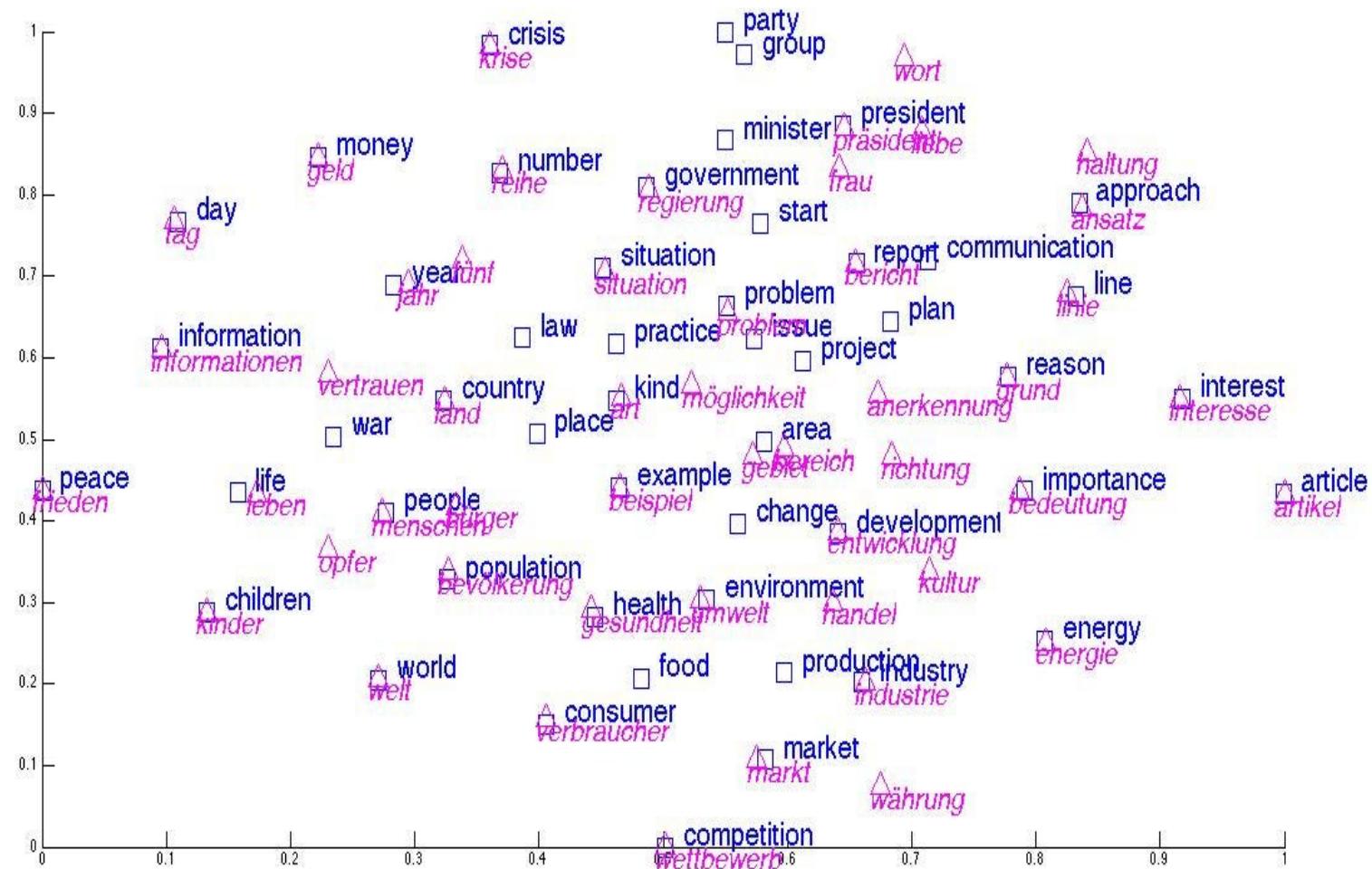
$$[0.8, 0.65, 0.1] - [0.7, 0.5, 0.05] = [0.1, 0.15, 0.05]$$

$$2. \bullet \text{ mulher} =$$

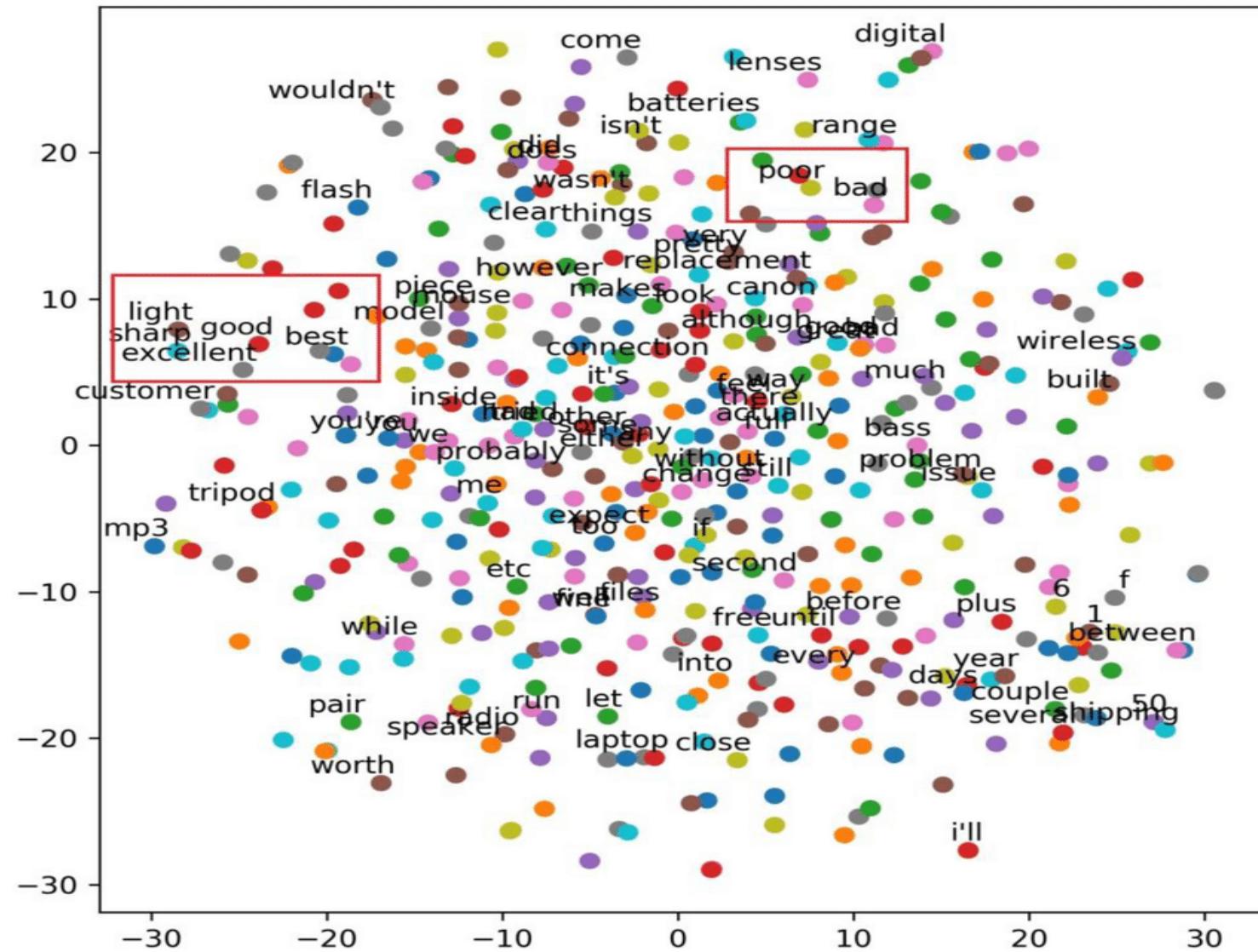
$$[0.1, 0.15, 0.05] + [0.72, 0.55, 0.2] = [0.82, 0.7, 0.25]$$

Resultado: [0.82, 0.7, 0.25], que é exatamente o vetor da rainha.

NLP – SIMILARIDADE DA DISTÂNCIA ENTRE PALAVRAS DE DIFERENTES LINGUAS



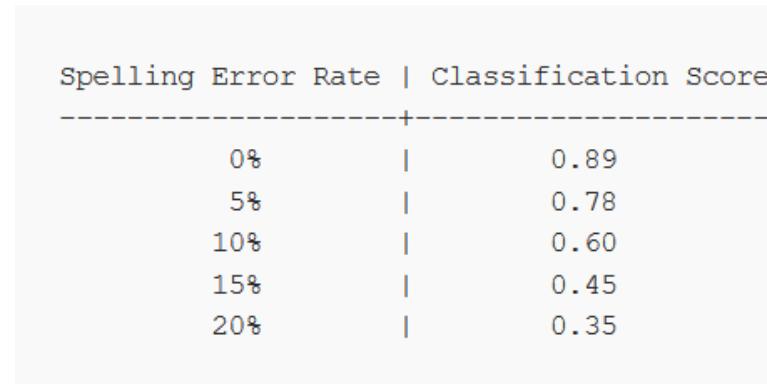
NLP – SIMILARIDADE DA DISTÂNCIA ENTRE PALAVRAS



TOKENIZAÇÃO DE SUB PALAVRAS: A tokenização por subpalavras combina os benefícios da tokenização por caracteres e por palavras, dividindo palavras raras em unidades menores enquanto mantém palavras frequentes como entidades únicas. Isso permite que o modelo lide com palavras complexas e erros de ortografia, mantendo o comprimento das entradas em um nível gerenciável.

```
from transformers import BertTokenizer  
  
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")  
tokenizer_output = tokenizer.tokenize("This is an example of the bert tokenizer")  
print(tokenizer_output)  
# ['this', 'is', 'an', 'example', 'of', 'the', 'bert', 'token', '##izer']
```

PALAVRAS ESCRITAS DE FORMA ERRADA: Um problema comum com a tokenização é lidar com erros de ortografia. Por exemplo, se o corpus incluir a palavra 'hepl' em vez de 'help', o modelo pode tratá-la como uma palavra fora do vocabulário (OOV). Isso pode reduzir significativamente o desempenho do modelo.



Fonte: <https://arxiv.org/pdf/2003.12932.pdf>

HANDS ON

Vamos entender melhor e fixar os conceitos explicados aplicando a tokenização e a geração de embeddings usando o Google Colab.

Localize e abra o notebook a seguir:

https://colab.research.google.com/drive/1Zj7C_y3pj5Gp8Lefhw7sLlgyQLcE7OV?usp=drive_link