

Técnicas Avançadas de NLP

(Natural Language Processing)

Dando aos computadores a capacidade
de **compreender** textos

Com Giuliano Ferreira

Resumo

01

Teoria

Conceitos por trás do processamento de textos

02

Prática

Como utilizar as bibliotecas e ferramentas

03

Projetos

Estudos de caso nas principais tarefas de machine learning

**“80% dos dados mundiais
são não estruturados e o
maior mecanismo de
busca do mundo indexou
apenas 0,004% de todos
os dados disponíveis.”**

— FORBES Magazine - 2021

Dados Não Estruturados

Forbes reported that 95% of businesses cite the need to manage unstructured data as a problem for their business. The CIO highlighted that 80-90% of the data we generate today is unstructured.

The challenge has been accelerated with faster mobile networks leading to increased usage – IBM reports that 90% of all data has been created in the last two years.

This all supports the argument that skilled data professionals and more computing machine power will be required as we move into the 2020s. It is predicted that 97.2% of organizations are investing in big data, and AI and job listings for data science and analytics will reach around 2.7 million by 2020, according to recent research by Forbes.

When it comes to data; size is extremely important because it determines different factors like capacity, usage or frequency and cost.

Big data is not about structured data at all, because simply around 85% of the data around the world is considered unstructured data (Digital Reasoning website, IBM).

unstructured data

There is a wide array of forms that make up unstructured data such as email, text files, social media posts, video, images, audio, sensor data, and so on.

14 de dez. de 2020



Unstructured data is the video, audio and images that make up our everyday lives and are exploding as the digital economy grows. Gartner estimates that unstructured data represents an astounding 80 to 90% of all new enterprise data, and it's growing 3X faster than structured data.

A barreira da linguagem

VISUAL CAPITALIST DATASTREAM

Most Commonly Used Languages ON THE INTERNET

Here's a look at the most commonly used languages on the web, and how they compare to the world's population of speakers.

2 | RUSSIAN

8.5%

3 | SPANISH

4.0%

4 | TURKISH

3.7%

5 | PERSIAN

3.0%

6 | FRENCH

Rank
1 | ENGLISH

60.4%

Percentage of top
10M websites
using the language

7 | GERMAN

2.4%

8 | JAPANESE

2.6%

9 | VIETNAMESE

1.7%

10 | CHINESE

1.4%

11 | PORTUGUESE

1.3%

12 | ARABIC

1.1%

13 | ITALIAN

0.8%

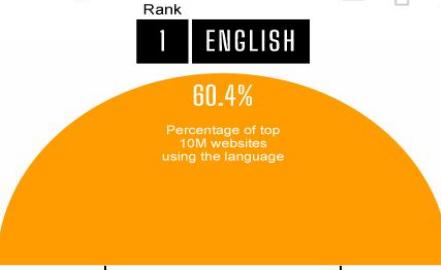
14 | INDONESIAN

0.9%

15 | GREEK

14.3%

Runet—the internet's Russian language community—has over 100 million users.



Spoiler do Projeto Final



Stack Academy

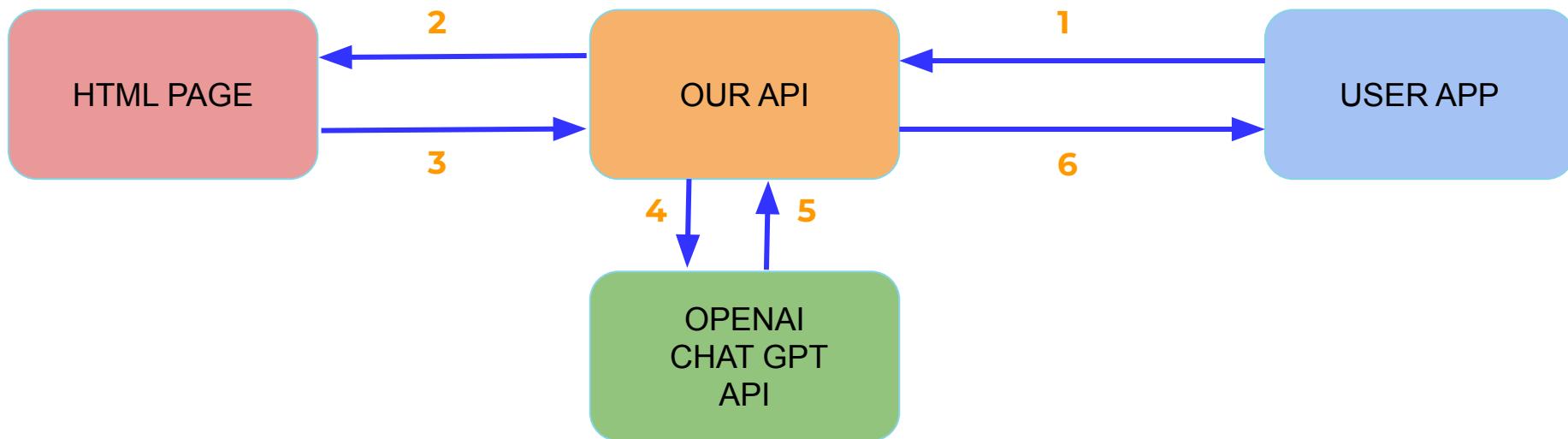
Assistente de Cursos

O que deseja saber?

Do que se trata esse curso?

Run

Arquitetura do Projeto Final



01

Teoria

Conceitos por trás do
processamento de textos

Introdução sobre NLP

O processamento de linguagem natural (NLP) é um ramo da inteligência artificial que ajuda os computadores a entender, interpretar e manipular a linguagem humana. A NLP permite que os computadores leiam e compreendam o texto, bem como gerem um texto mais natural e fácil de entender.

Para um computador entender a linguagem, ele deve primeiro ter um processo automatizado para reconhecer padrões em texto (ou seja, um algoritmo NLP). Esses algoritmos recebem dados e criam um modelo desses dados; o modelo representa os dados e permite futuras previsões ou varreduras dos mesmos dados.

Os algoritmos NLP podem ser usados de várias maneiras, desde a automação de tarefas de atendimento ao cliente até a compreensão do sentimento do cliente e o fornecimento de interfaces de linguagem natural para aplicativos como mecanismos de pesquisa e assistentes de voz.

Técnicas de NLP

As técnicas de NLP são usadas para analisar texto e fala e podem ser divididas em duas categorias principais: análise sintática e análise semântica. A análise sintática envolve a análise da estrutura de uma frase, enquanto a análise semântica envolve a compreensão do significado de palavras e frases.

As técnicas de NLP também podem ser usadas para gerar texto, como resumir um documento ou gerar perguntas a partir de um determinado texto.

Aplicações de NLP

Os aplicativos de NLP são usados em vários campos, desde atendimento ao cliente até assistência médica. No atendimento ao cliente, a NLP pode ser usada para automatizar tarefas como responder a consultas de clientes, enquanto na área da saúde, a NLP pode ser usada para analisar registros médicos e gerar relatórios.

A NLP também pode ser usada para gerar perguntas a partir de um determinado texto, como um questionário ou pesquisa, ou para gerar interfaces de linguagem natural para aplicativos como mecanismos de pesquisa e assistentes de voz.

Desafios da NLP

A NLP é um campo complexo e ainda existem muitos desafios que precisam ser abordados. Um dos principais desafios é entender o contexto de uma frase, bem como as relações entre palavras e frases. Além disso, os sistemas NLP devem ser capazes de lidar com a ambigüidade, bem como lidar com a grande quantidade de dados disponíveis na web.

Outro desafio é lidar com os diferentes dialetos e idiomas existentes, bem como com os diferentes estilos de escrita que as pessoas usam. Além disso, os sistemas de NLP devem ser capazes de lidar com a natureza em constante mudança da linguagem, pois novas palavras e frases são constantemente adicionadas.

Futuro da NLP

O futuro da NLP é brilhante, pois a tecnologia está sendo usada em vários campos, desde atendimento ao cliente até assistência médica. Além disso, o NLP está sendo usado para gerar interfaces de linguagem natural para aplicativos como mecanismos de busca e assistentes de voz. À medida que a tecnologia continua a melhorar, ela será usada em mais e mais aplicações.

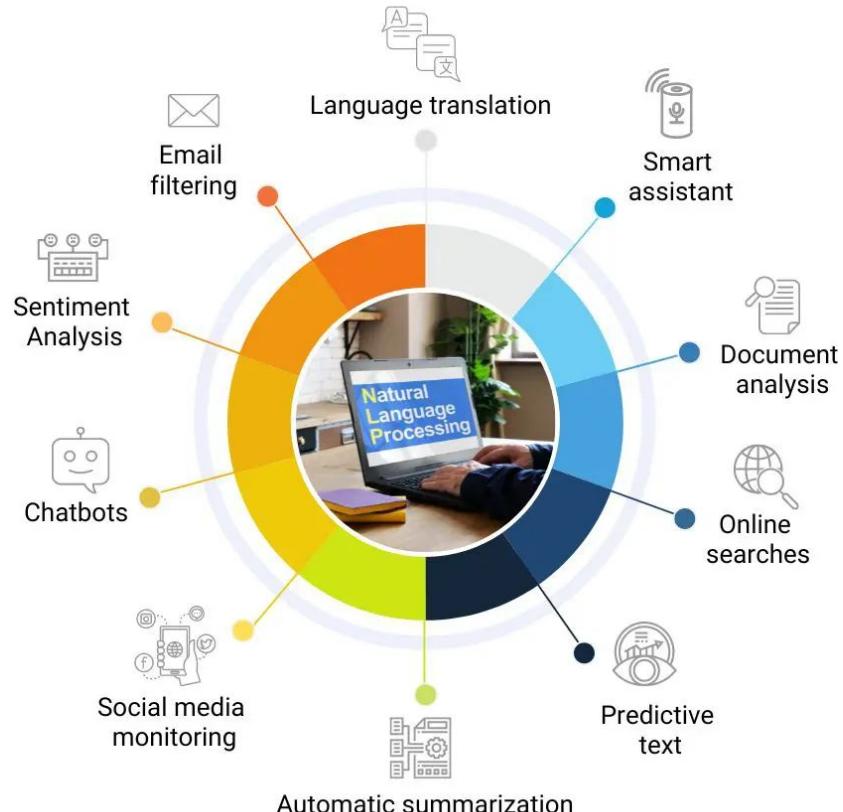
A NLP também está sendo usada para analisar o sentimento do cliente, que pode ser usado para fornecer um melhor atendimento ao cliente e obter insights sobre o comportamento do cliente.

Conclusão sobre NLP

O processamento de linguagem natural é uma tecnologia poderosa que pode ser usada de várias maneiras, desde a automatização de tarefas de atendimento ao cliente até a compreensão do sentimento do cliente e o fornecimento de interfaces de linguagem natural para aplicativos como mecanismos de pesquisa e assistentes de voz. A NLP é um campo complexo e ainda há muitos desafios que precisam ser enfrentados, mas o futuro da NLP é brilhante e a tecnologia está sendo usada em mais e mais aplicações.

À medida que a tecnologia continua a melhorar, ela será usada em ainda mais aplicações, desde atendimento ao cliente até assistência médica. A NLP é um campo emocionante e em rápido desenvolvimento que tem o potencial de revolucionar a maneira como interagimos com a tecnologia.

Exemplos da Vida Real



Principais Tarefas NLP

**Predição de
Sentenças**

**Análise de
Sentimentos**

**Sumarização
de Textos**

**Perguntas e
Respostas**

**Similaridade
Semântica**

**Sistemas de
Recomendação**

**Geração de
Textos**

**Tradução de
Textos**

**Correção
Ortográfica**

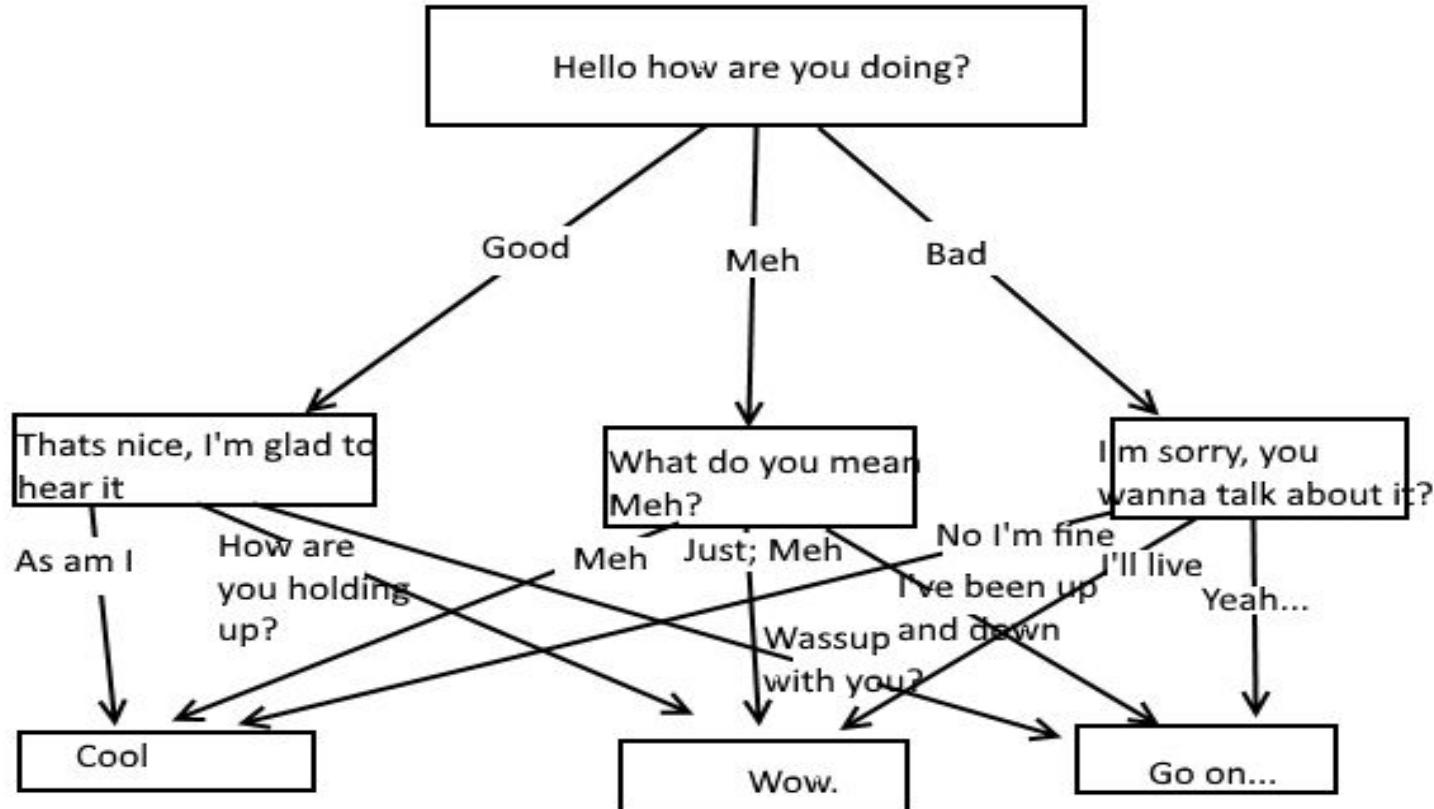
Categorias de Aplicações NLP

Categorized NLP Applications			
Search Editing Dialog Writing Email Text mining Law News Attribution Sentiment Analysis Behavior prediction Creative writing	Web	Documents	Autocomplete
	Spelling	Grammar	Style
	Chatbot	Assistant	Scheduling
	Index	Concordance	Table of contents
	Spam filter	Classification	Prioritization
	Summarization	Knowledge extraction	Medical diagnoses
	Legal inference	Precedent search	Subpoena classification
	Event detection	Fact checking	Headline composition
	Plagiarism detection	Literary forensics	Style coaching
	Community morale monitoring	Product review triage	Customer care
	Finance	Election forecasting	Marketing
	Movie scripts	Poetry	Song lyrics

As 3 Eras da NLP

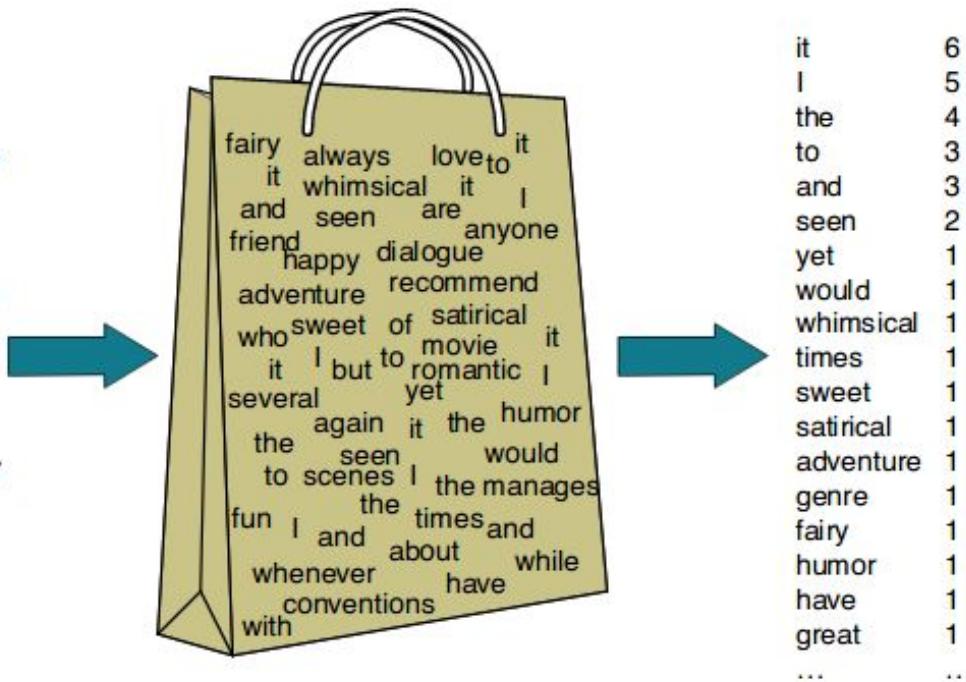


NLP Simbólica



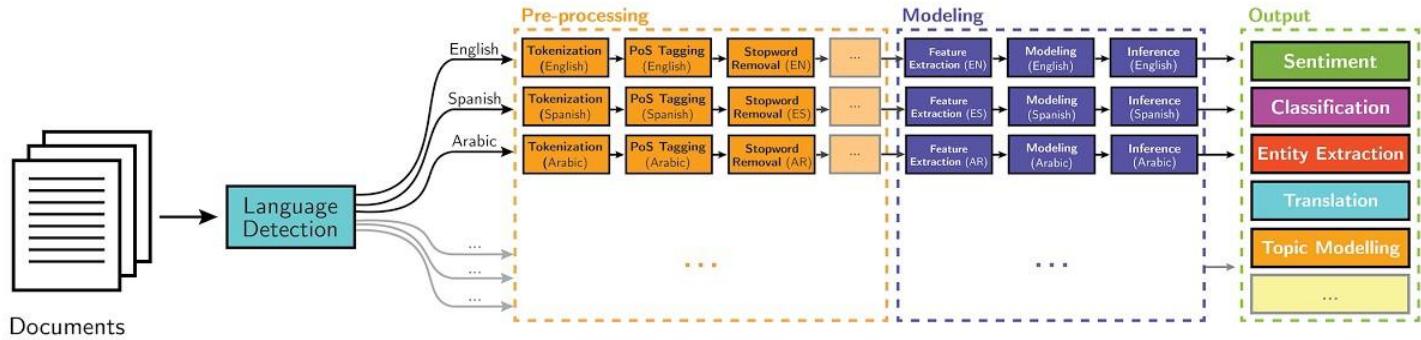
NLP Estatística

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

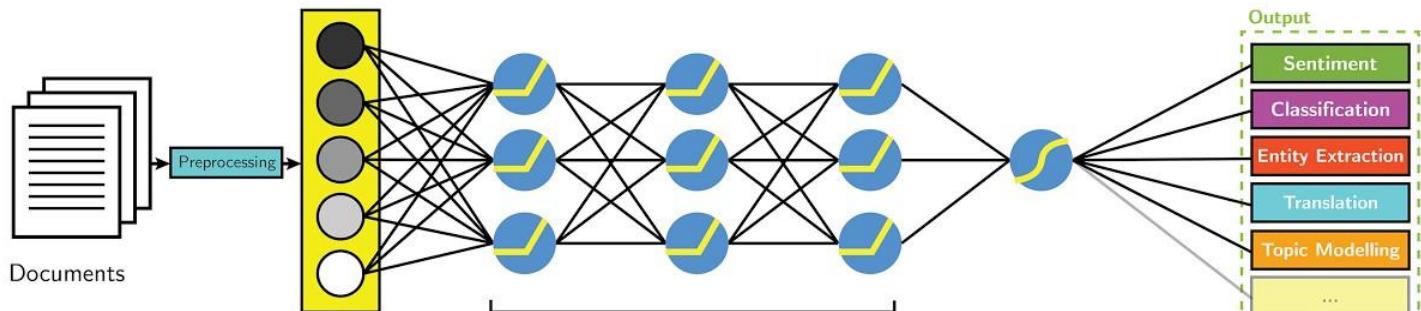


NLP Neural

Classical NLP



Deep Learning-based NLP



NLP Neural - Conceito

As redes neurais podem aprender com os dados e começar do zero para construir um modelo.

É por isso que as redes neurais são tão úteis na NLP: elas podem obter informações relevantes a partir de dados em vez de usar regras e modelos que já foram feitos.

O uso de redes neurais na NLP também é útil porque elas podem trabalhar com diferentes tipos de dados.

Por exemplo, texto, imagens e áudio podem ser usados para análise de sentimentos.

Uma vez que a rede neural tenha sido treinada em cada tipo de dados, o modelo pode descobrir o que os dados significam.

Uma rede neural é composta de regras matemáticas que são baseadas na informação que é armazenada em sua memória.

Para treinar a rede neural, você precisa carregar muitos dados na memória do modelo.

Criar um conjunto de dados de treinamento é uma maneira de fazer isso. Um conjunto de dados de treinamento é composto de recursos semelhantes aos dados que você deseja prever.

NLP Neural - Resumo

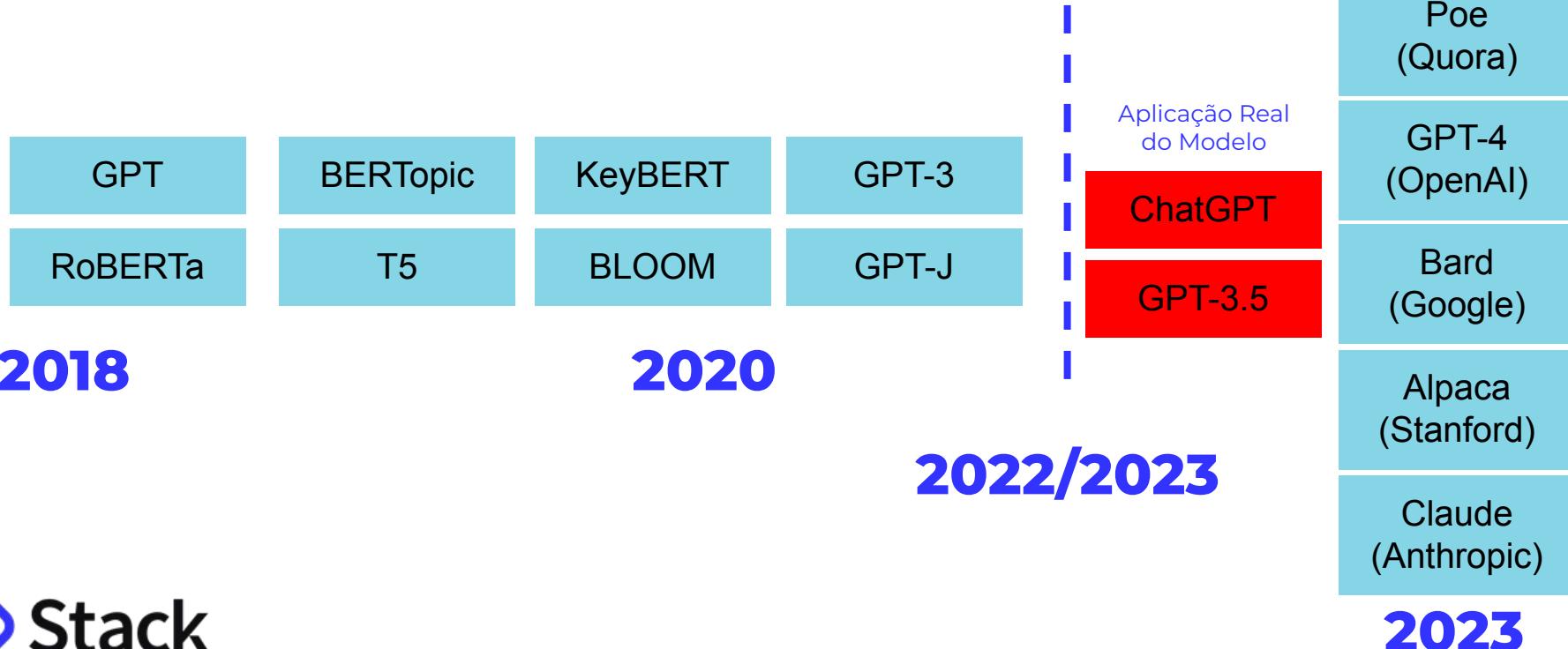
- Uma rede neural é construída usando regras matemáticas que são determinadas pela informação que é mantida em sua memória.
- Essas regras são baseadas nos dados.
- Para treinar a rede neural, uma quantidade significativa de dados precisa ser carregada na memória do modelo.

Definição: <https://www.linkedin.com/in/data-analytics-mike/>

Evolução dos algoritmos



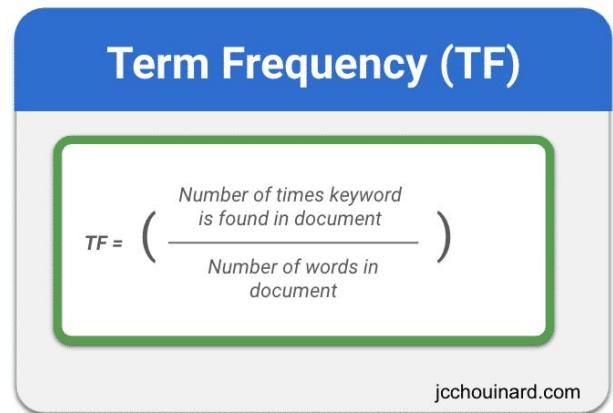
Algoritmos atuais



Técnicas anteriores

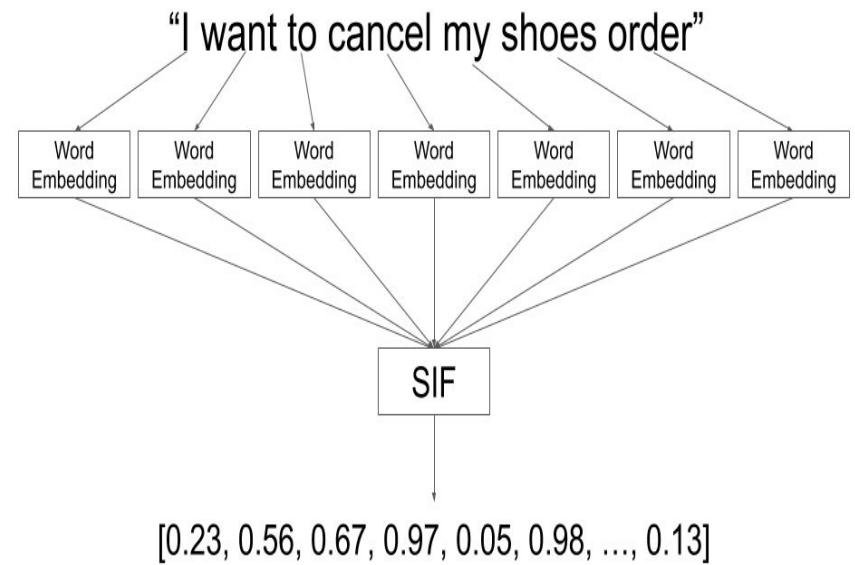
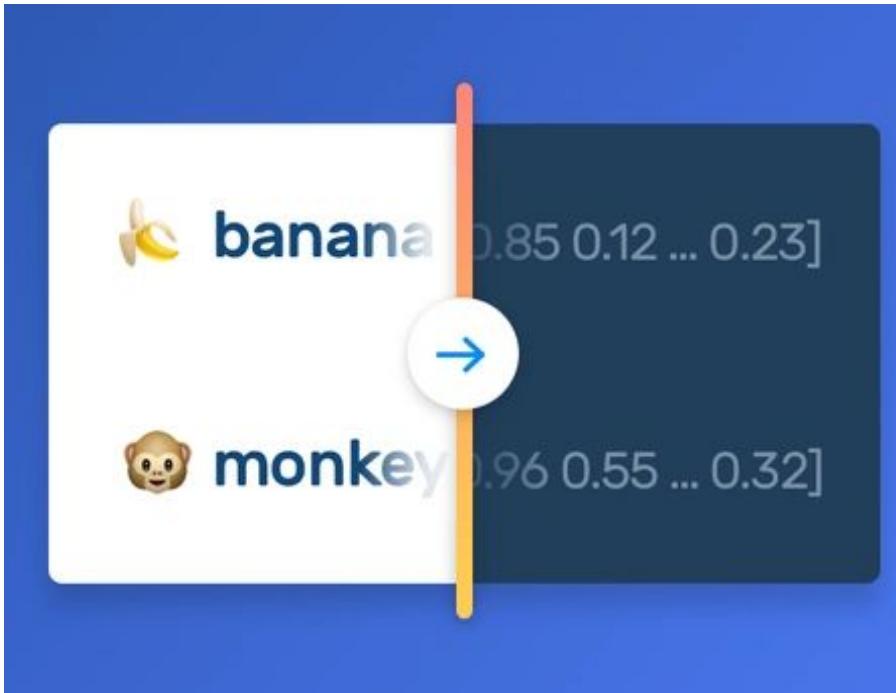
Raw Text		Bag-of-words vector	
		it	2
		they	0
		puppy	1
		and	1
		cat	0
		aardvark	0
		cute	1
		extremely	1

it is a puppy and it
is extremely cute



	aardvark	...	computer	data	result	pie	sugar
cherry	0	...	2	8	9	442	25
strawberry	0	...	0	0	1	60	19
digital	0	...	1670	1683	85	5	4
information	0	...	3325	3982	378	5	13

Atualmente: Vetores de Palavras - Embeddings



Sentence Transformers Library

```
from sentence_transformers import SentenceTransformer
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')

#Our sentences we like to encode
sentences = ['This framework generates embeddings for each input sentence',
             'Sentences are passed as a list of string.',
             'The quick brown fox jumps over the lazy dog.']

#Sentences are encoded by calling model.encode()
embeddings = model.encode(sentences)

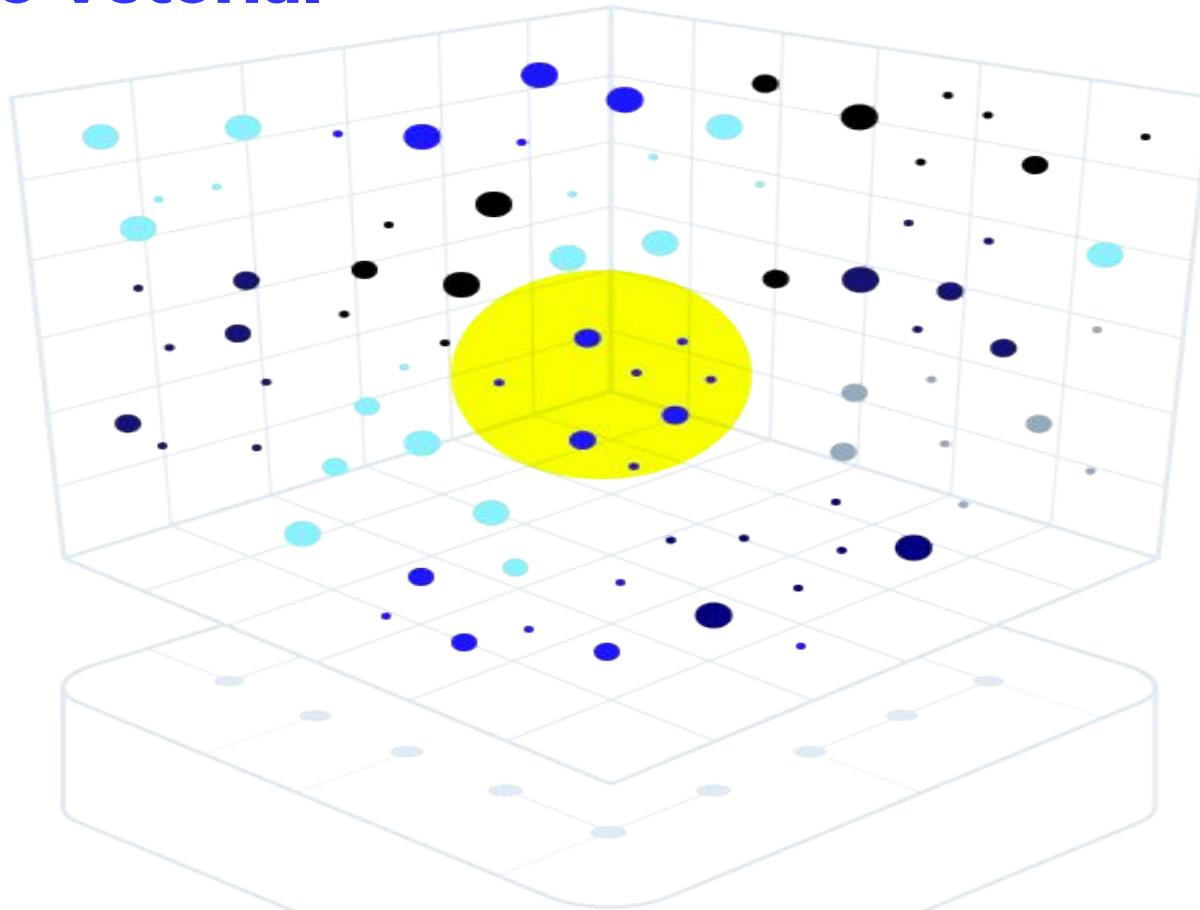
#print the embeddings
for sentence, embedding in zip(sentences, embeddings):
    print("Sentence:", sentence)
    print("Embedding:", embedding)
    print("")
```

Sentence Transformers - Result Embedding

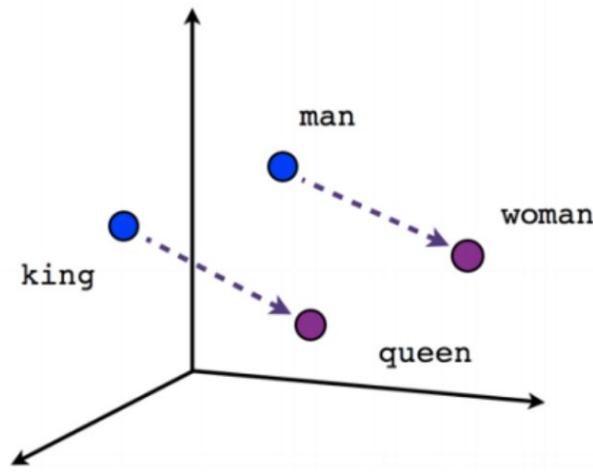
Sentence: "Sentences are passed as a list of string."

Embedding: [0.32208768 -0.00123929 0.1793735 -0.36919162 -0.06460258 0.09153683 0.24119091 -0.2949419 0.07728925 0.11577023 -0.04479968
0.17928241 0.14753594 0.21511656 0.36810815 0.20910896 0.27194214 0.34880063 -0.5725191 -0.18253206 0.44489557 0.2745295 0.04266278
-0.07683565 0.18689133 0.44965008 -0.16932587 -0.24896361 -0.20479265 0.4028505 -0.21019281 0.03775732 0.07848503 0.12848431 0.02593061
0.47155952 0.17853801 -0.0737977 0.08130737 -0.23328744 -0.4980125 -0.04135688 -0.12094604 0.17028993 -0.19154085 -0.38459823 -0.77479136
-0.10622737 -0.2304489 0.4024144 -0.8745086 0.23853712 -0.4712988 0.3340933 -0.2415401 -0.14835086 -0.14513575 -0.34830916 -0.08349223
-0.69097286 -0.29845268 -0.12230494 0.07482657 -0.18775584 -0.37546527 0.21369524 -0.10096437 -0.12234421 0.31431535 -0.23989955 0.22460745
0.03996019 0.3603482 -0.56638044 0.21883497 0.11020306 -0.10870808 0.07084102 -0.02608195 0.18370347 0.08465921
-0.204782230.24435617-0.08180573-0.01903085-0.03591377 0.02398442 -0.2855857 0.07374779 -0.29744217 -0.8771782 0.4710194 -0.04940462
0.36394492 0.48264387 0.01564613 0.03558933 -0.26202998 -0.11218457 0.02411039 0.37477782 -0.09897308 -0.09851861 0.1500086 0.00689539
-0.12652446 -0.31598946 0.314495 -0.29425603 -0.2694104 0.20221177 0.14329897 -0.19584641 -0.34104443 -0.03172744 0.7365028 0.3192352
0.24381322 0.30732578 0.09933221 0.190109 -0.10694531 0.05178648 0.03233408 -0.10314637 0.2649919 0.31206453 0.4315258 -0.6426123
0.08409569 -0.04327382 -0.04991194 -0.12718569 0.137892 0.01306224 0.3438325 0.09234273 -0.0992275 -0.5215991 0.25842282 -0.01057153
-0.00478161 0.03938875 0.1908606 0.329339 -0.24345145 -0.07328302 -0.3928003 0.14541803 0.32839537 -0.04184647 0.074071 -0.73860526
-0.09075994 0.15802325 -0.09780031 -0.21605964 -0.30027446 0.23236583 0.01072467 0.49570492 0.04974846 0.29931405 -0.05382241 0.35328093
0.34191763 0.49667242 -0.4860526 -0.19098845 0.8154573 0.22962637 -0.3207778 -0.3272672 -0.36771727 0.34521163 -0.02620161 -0.14315075
0.10648453 -0.24638031 -0.09366626 0.17198643 -0.08508811 0.2012031 -0.0587922 -0.34021005 -0.1956533 0.2828085 0.2012431 -0.08207261
0.09779127 -0.26374972 0.12176552 -0.01041445 -0.43859825 0.1105824 0.48010385 -0.10981948 -0.6375458 0.29336768 -0.19207639 0.4653699
0.2704202 0.19388486 0.17379059 -0.30077007 -0.02751163 -0.02291271 0.36784622 0.0249216 0.5370548 0.18851219 -0.1334443 0.08917346
0.05542916 -0.24818334 -0.04199769 0.05767398 -0.18278815 -0.41686422 0.16070555 -0.4636254 0.11769234 -0.3770691 0.02960374 0.6925614
-0.48308915 0.2112839 0.18214497 -0.18429579 0.06817667 -0.02460932 -0.1907359 -0.06736976 -0.56700724 -0.23929311 -0.08497214 0.03093975
0.3107991 0.12916222 0.05248246 -0.3344981 0.18810095 0.23547147 -0.0018347 0.4536157 0.24885081 -0.05641087 -0.29774582 -0.43511733
-0.07969434 -0.17670155 -0.13347088 0.19382714 0.22002584 -0.11057542 0.26473743 -0.2717907 0.03410916 -0.4771438 0.44719073 -0.05570394
0.39643747 0.27483255 0.33305585 -0.10890228 0.27888146 0.21596923 -0.0525226 -0.3586752 -0.6906289 0.03960172 0.00652784 -0.01095352
-0.10027692 0.04770029 -0.34146893 -0.16714166 0.0713641 -0.18078458 -0.30248463 -0.6842873 -0.09592854 -0.21411094 -0.65524364 0.56756425
0.269467 -0.00190065 0.86180633 0.16771567 0.03102793 -0.26773065 -0.0783028 -0.48510873 -0.26737195 -0.33354244 -0.5738249 0.35678262
0.08993588 -0.1305718 -0.15136491 -0.06124156 -0.13037081 0.5585606 0.6141749 -0.04804057 -0.06388568 0.08390599 -0.25143674 -0.04359825
-0.18525814 0.04693349 -0.34380853 -0.09738491 0.16833653 0.0752685 0.17694478 0.17727166 -0.03423443 0.14993557 -0.13773178 -0.20949672
-0.6127285 0.3781398 0.3901827 -0.08359321 0.03152135 0.13122398 0.38826054 0.21844238 0.09724288 0.42089376 -0.32641247 -0.26933405
-0.39095095 -0.2264866 -0.3202072 -0.16287398 -0.03581632 0.3637388 0.18583283 -0.02914017 -0.4657794 0.29168904 0.3725128 -0.23726602
0.00338655 0.41540992 0.03300448 0.4500396 -0.08159253 0.33990347 0.24497865 0.02352417 -0.14643052 -0.12644538 0.31128648 -0.15182622
0.0100941 0.49108535 0.143624 0.11589039 -0.23236994 0.24751768 0.1836449 -0.24836825 -0.11220941 -0.23113324 0.08428963 -0.24378632
0.13307282 0.42355707 0.33348358 -0.34370142 0.03443653 0.18795532 0.20037164 -0.05355943 0.28485253 0.07176559 0.05487144 -0.08103789
0.27076918 0.11702641]

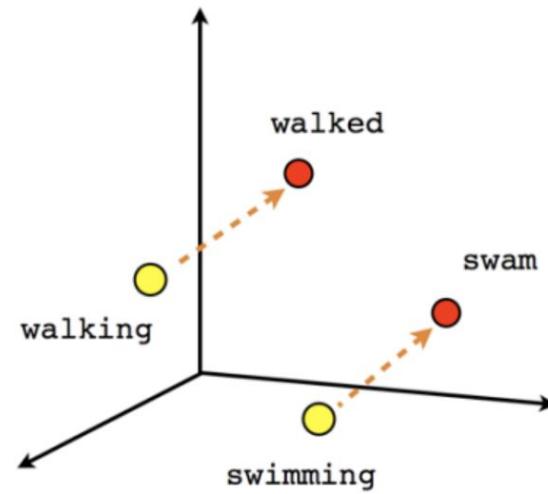
Espaço Vetorial



Vetores de Palavras - Representação

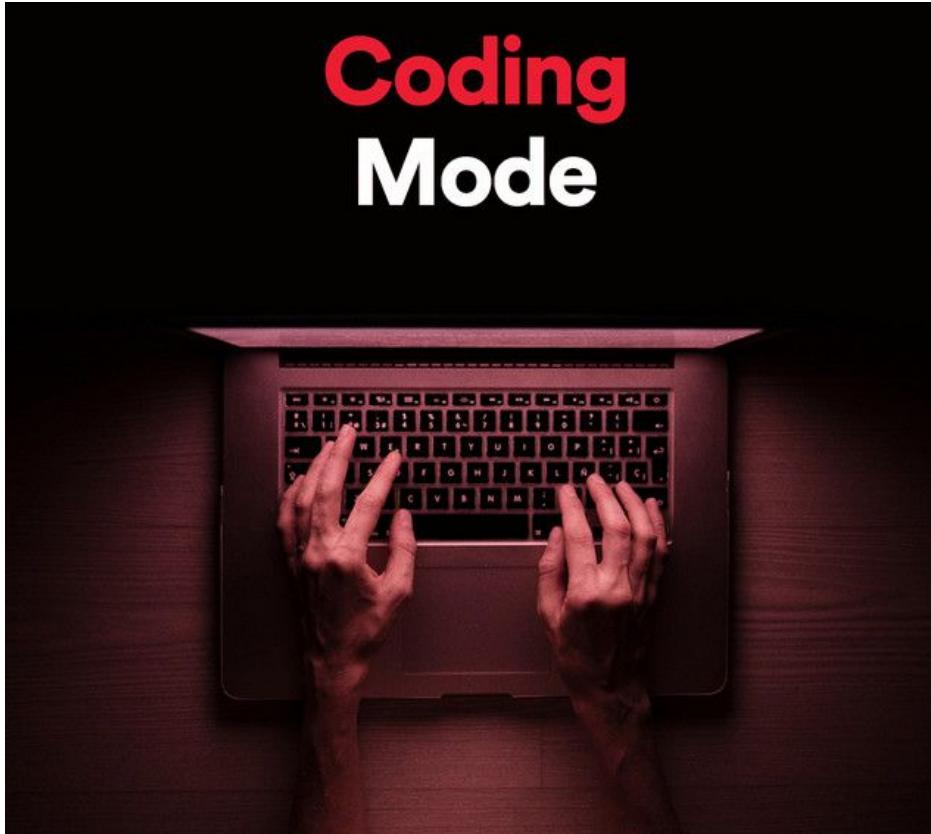


Male-Female



Verb tense

Exemplos - Coding Mode 1



Quiz Time!



Tokenizadores - conceito

TOKENIZATION

Word-based

[Learning]

[Learned]

[Deep] [Learning]

Subwords

[Learn] [# # ing]

[Learn] [ed]

[De] [ep] [Learn]

[# # ing]

Character-based

[L e a r n i n g]

[L e a r n e d]

[D e e p]

[L e a r n i n g]

Vocabulary size →

Tokenizadores - na prática

```
▶ tokenizer("Using a Transformer network is simple")
{'input_ids': [101, 7993, 170, 13809, 23763, 2443, 1110, 3014, 102], 'token_type_ids': [0, 0, 0, 0, 0, 0, 0, 0, 0], 'attention_mask': [1, 1, 1, 1, 1, 1, 1, 1, 1]}

▶ from transformers import AutoTokenizer
tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")

sequence = "Using a Transformer network is simple"
tokens = tokenizer.tokenize(sequence)

print(tokens)
['Using', 'a', 'Trans', '##former', 'network', 'is', 'simple']

[ ] ids = tokenizer.convert_tokens_to_ids(tokens)

print(ids)
[7993, 170, 13809, 23763, 2443, 1110, 3014]

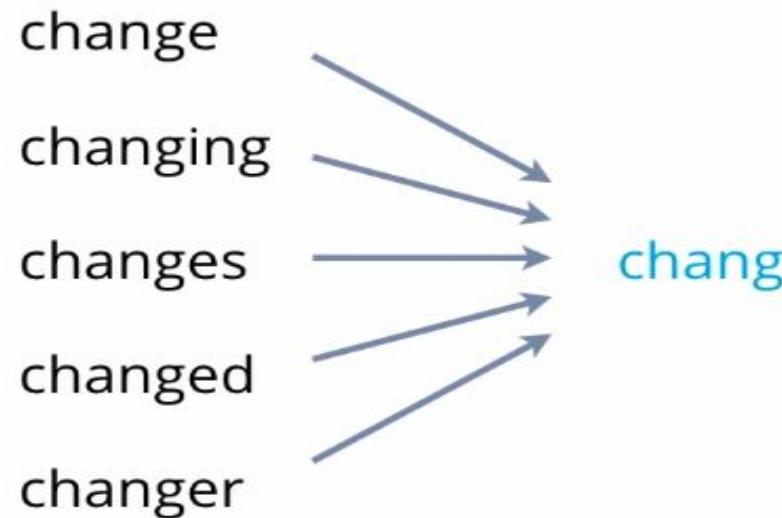
[ ] decoded_string = tokenizer.decode([7993, 170, 11303, 1200, 2443, 1110, 3014])
print(decoded_string)
Using a transformer network is simple
```

Stopwords

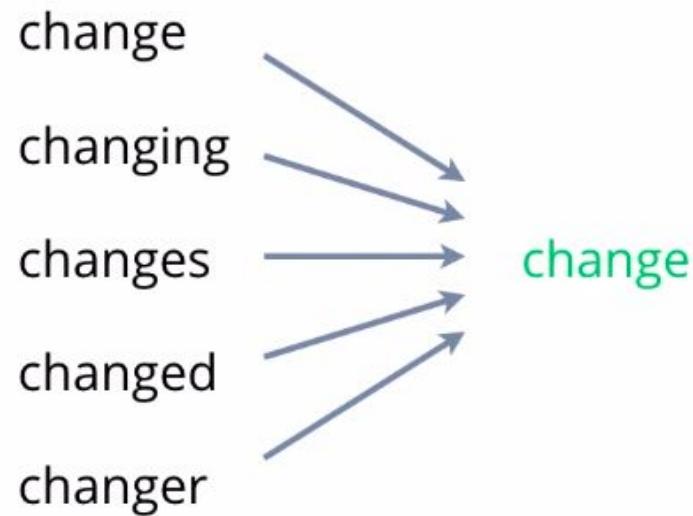
emissions
report action united that
at paris is **change** have would it
global ice be from greenhouse this average
countries countries more all summit
about by **the** convention nations
to and **and** co2 as a
for gas with warming c earth on
agreement rise
in sea are wollen parties
level

global emissions
report warming protocol
human assessment efforts
level future average world oceans nations
agreement ice cm limiting
temperature gas action ipcc
ghgs parties kyoto pre-industrial co2 period
leaders increase decade century
impacts arctic convention
paris greenhouse summit united
countries ecosystems

Stemming



Lemmatization

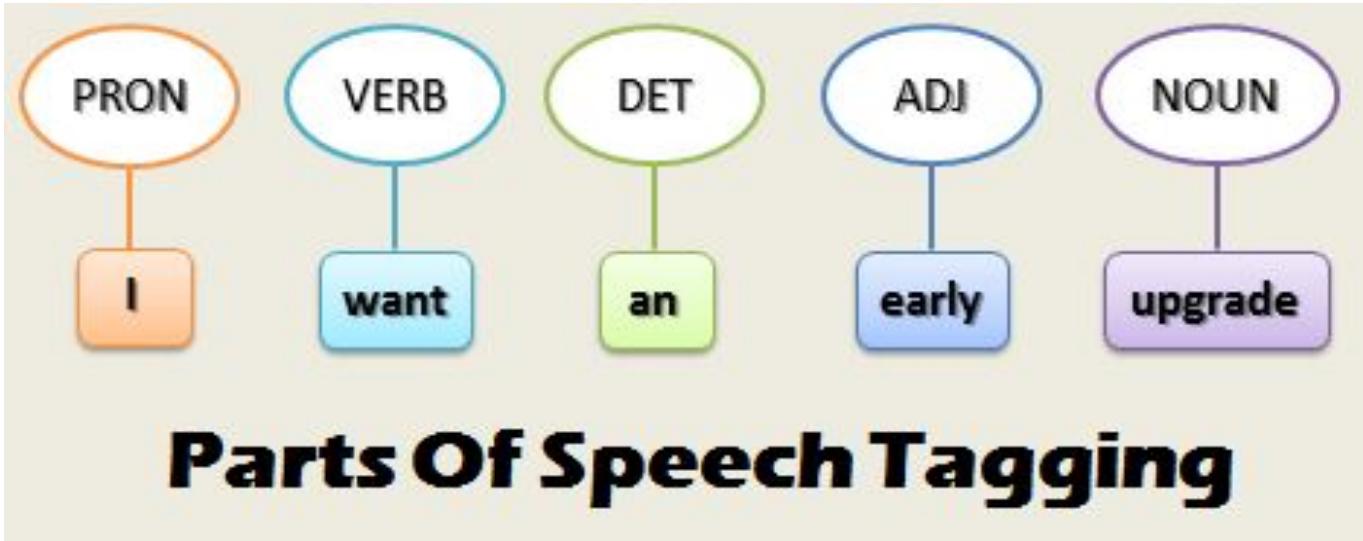


Unicode Normalization

```
>>> from unicodedata import normalize
>>> def nfc_equal(str1, str2):
...     return normalize('NFC', str1) == normalize('NFC', str2)
...
>>> s1 = 'café'
>>> s2 = 'cafe\u0301'
>>> s1 == s2
False
>>> nfc_equal(s1, s2)
True
```



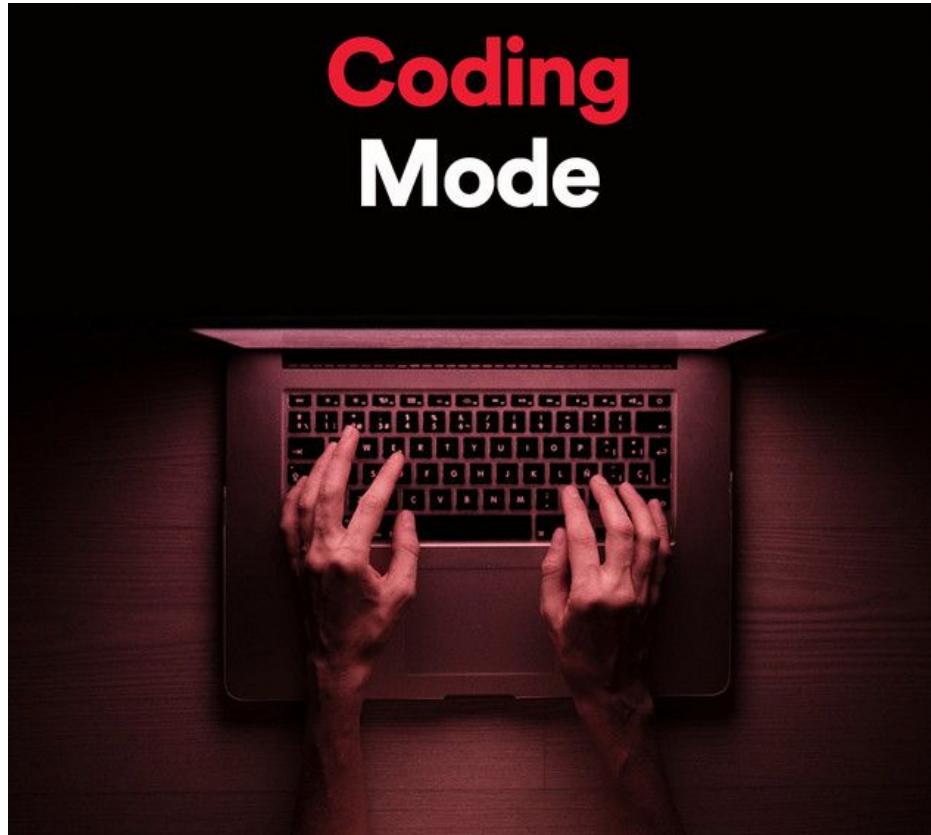
POS Tagging



Named Entity Recognition

When Sebastian Thrun PERSON started working on self - driving cars at Google ORG in 2007 DATE , few people outside of the company took him seriously . “ I can tell you very senior CEOs of major American NORP car companies would shake my hand and turn away because I was n’t worth talking to , ” said Thrun PERSON , in an interview with Recode ORG earlier this week DATED .

Exemplos - Coding Mode 2



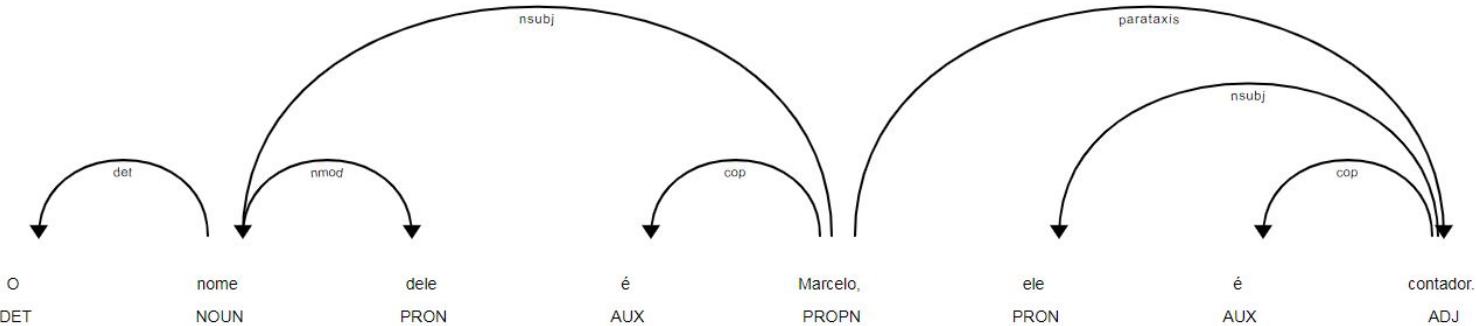
Contexto



```
# Testando o Spacy
doc = nlp(u'O nome dele é Marcelo, ele é contador.')
doc
print([d for d in doc])

# viewing sentence's structure
from spacy import displacy
from IPython.display import SVG, display
def showSVG(s):
    display(SVG(s))
graph01 = displacy.render(doc)
showSVG(graph01)
```

[0, nome, dele, é, Marcelo, , , ele, é, contador, .]



Atenção

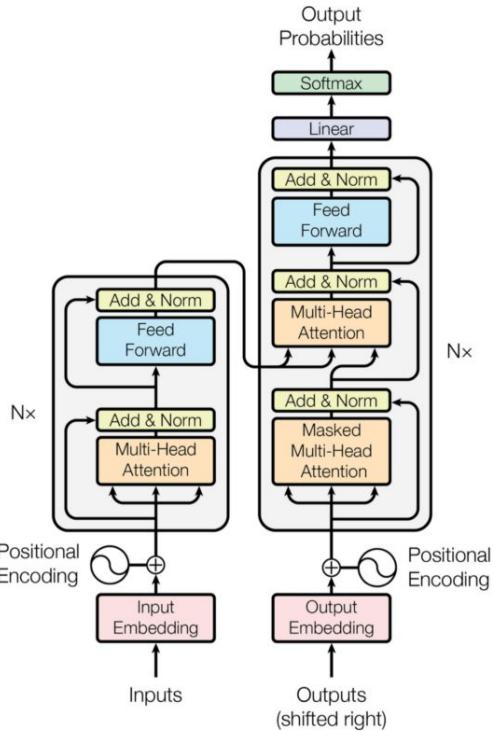
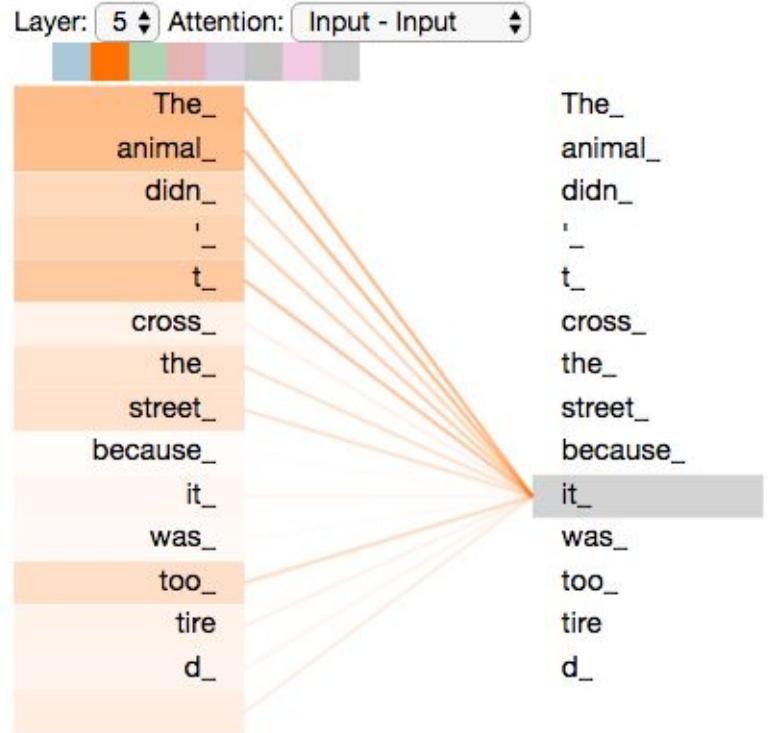
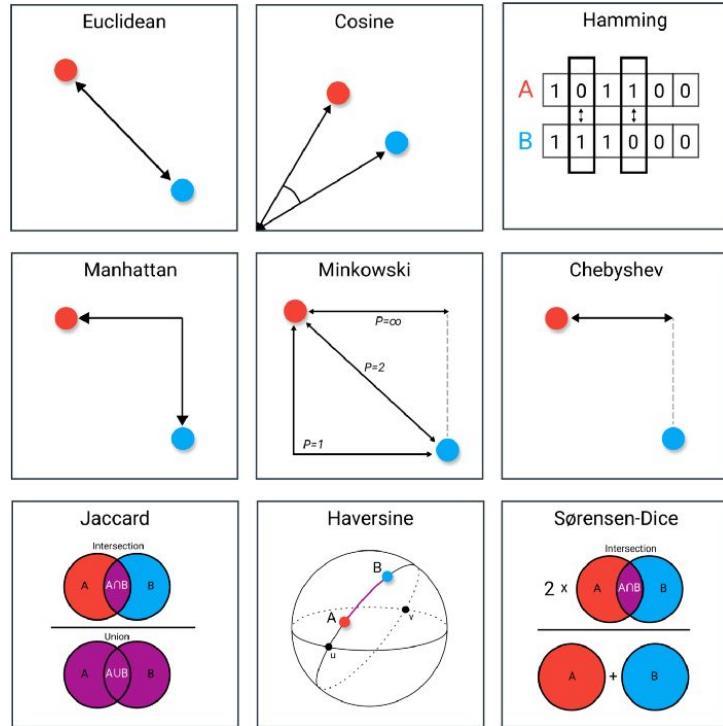


Figure 1: The Transformer - model architecture.



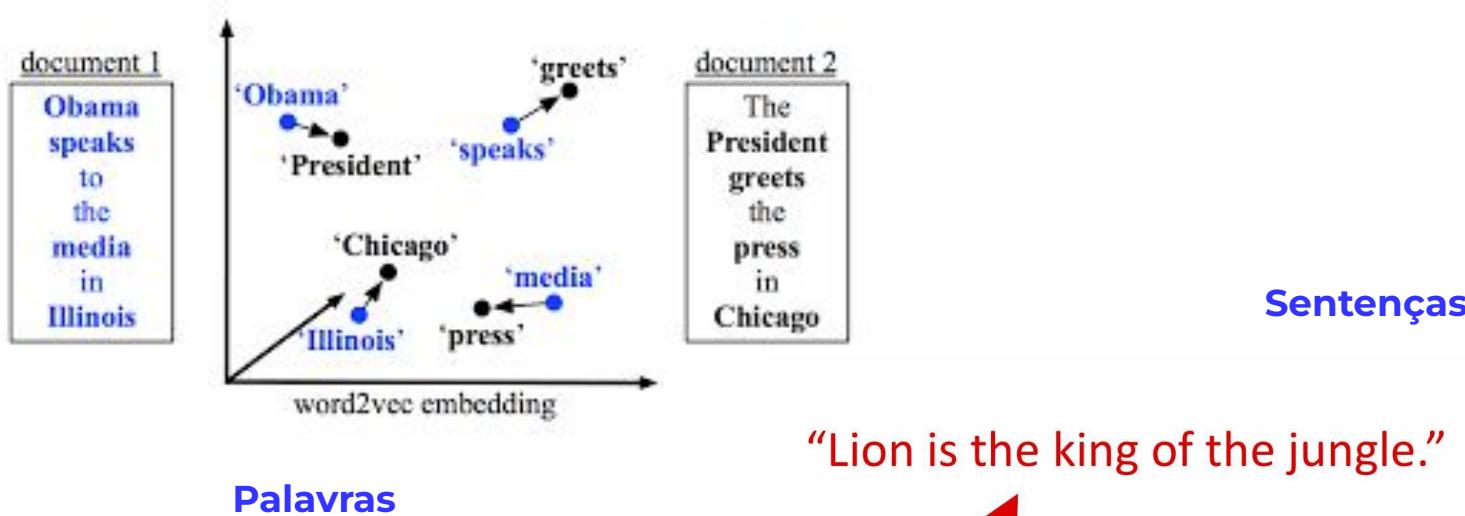
Google Research Article: Attention is All You Need - <https://research.google/pubs/pub46201/>

Similaridade de termos



métricas

Similaridade de Sentenças

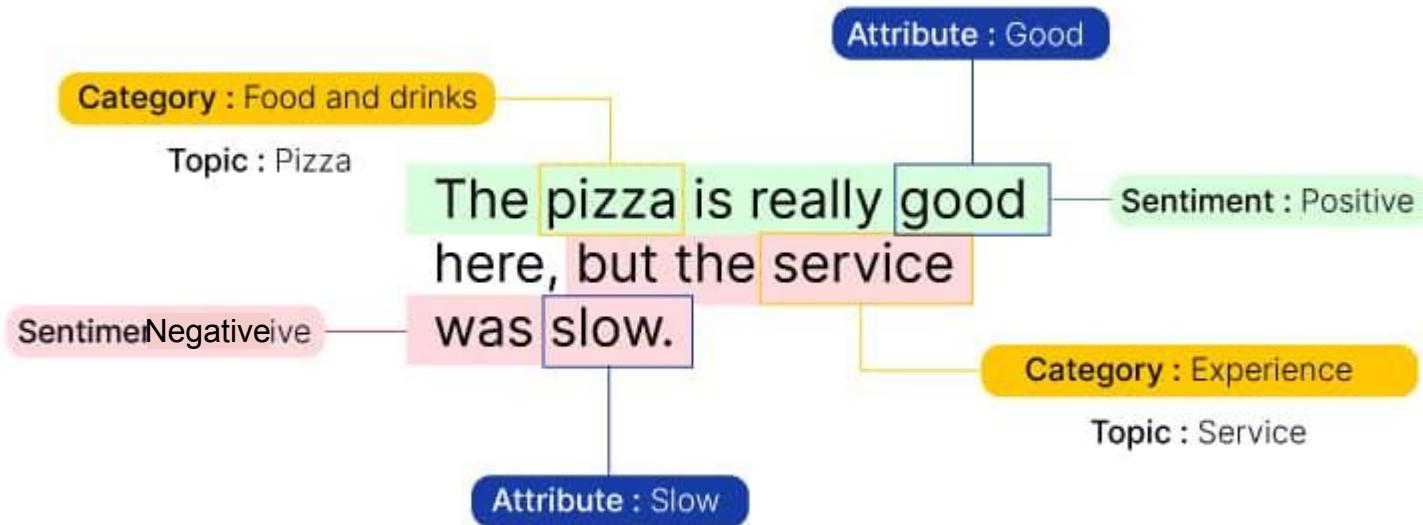


“Lion is the king of the jungle.”

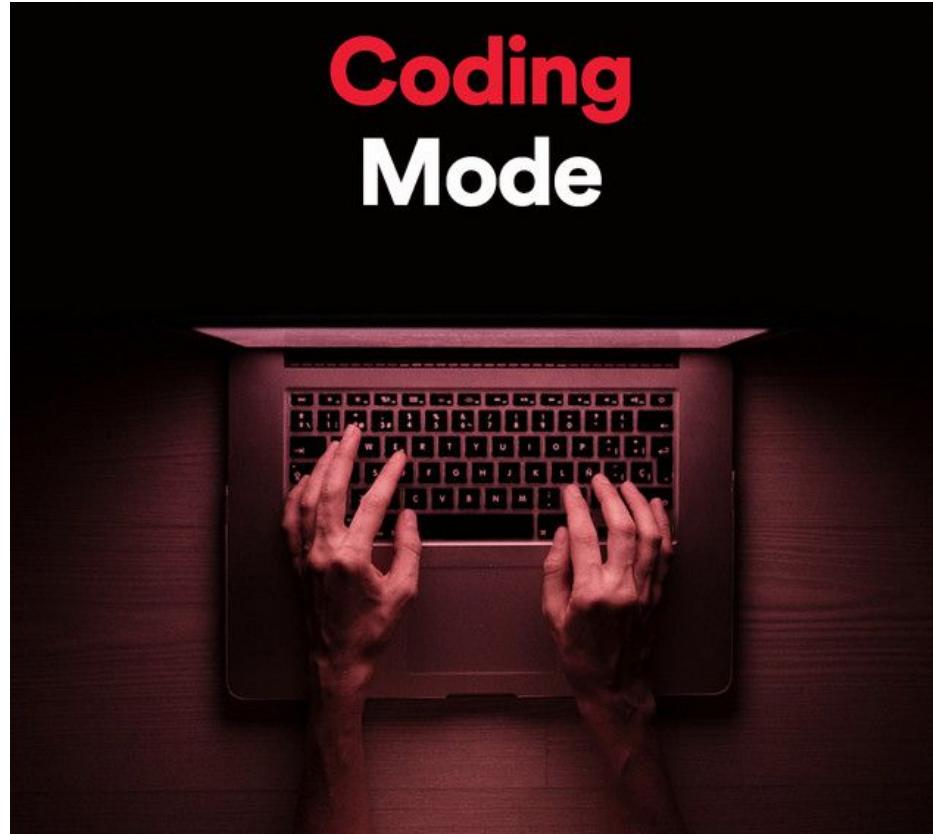
“The tiger hunts in this forest.”

“Everybody loves New York.”

Análise de Sentimento



Exemplos - Coding Mode 4



**“BEM
VINDOS AO
FUTURO,
ONDE AS
MÁQUINAS
FALAM A
NOSSA
LÍNGUA...”**



Transformers

THE
TRANSFORMER

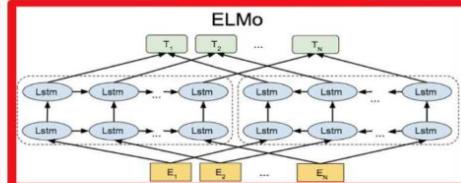
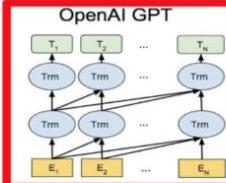
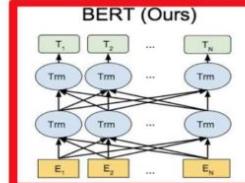


BERT

ELMo

Word2Vec

BERT



Popular algorithms in NLP

Hugging Face



Hugging Face

Search models, datasets, users...

Models

Datasets

Spaces

Docs

Solutions

Pricing



Log In

Sign Up

Tasks

- Image Classification
 - Translation
 - Unconditional Image Generation
 - Fill-Mask
 - Automatic Speech Recognition
 - Token Classification
 - Sentence Similarity
 - Audio Classification
 - Question Answering
 - Summarization
 - Zero-Shot Classification
- + 15

Libraries

- PyTorch
 - TensorFlow
 - JAX
- + 24

Datasets

- common_voice
 - wikipedia
 - squad
 - bookcorpus
 - glue
 - c4
 - conll2003
 - emotion
- + 932

Languages

- en
 - es
 - fr
 - de
 - zh
 - sv
 - fi
 - ru
- + 173

Models 64

portuguese

Sort: Most Downloads

neuralmind/bert-base-portuguese-cased
Fill-Mask • Updated May 19, 2021 • ↓ 159k • 19

neuralmind/bert-large-portuguese-cased
Fill-Mask • Updated May 19, 2021 • ↓ 40.1k • 8

pierreguillou/bert-base-cased-squad-v1.1-portuguese
Question Answering • Updated Jan 4 • ↓ 10.5k • 10

d1b/electra-base-portuguese-uncased-brwac
Updated Dec 10, 2021 • ↓ 4.33k

pierreguillou/gpt2-small-portuguese
Text Generation • Updated May 23, 2021 • ↓ 3.27k • 7

pierreguillou/bert-large-cased-squad-v1.1-portuguese
Question Answering • Updated Jan 4 • ↓ 2.79k • 8

unicamp-dl/ptt5-base-portuguese-vocab
Text2Text Generation • Updated Mar 24, 2021 • ↓ 1.37k • 4

Tipos de Modelos

Computer Vision

- Image Classification
- Unconditional Image Generation
- Image-to-Image
- Image Segmentation
- Object Detection

Natural Language Processing

- Translation
- Fill-Mask
- Token Classification
- Sentence Similarity
- Question Answering
- Summarization
- Zero-Shot Classification
- Text Classification
- Text2Text Generation
- Text Generation
- Conversational
- Table Question Answering

Audio

- Automatic Speech Recognition
- Audio Classification
- Text-to-Speech
- Audio-to-Audio
- Voice Activity Detection

Multimodal

- Feature Extraction
- Text-to-Image
- Image-to-Text

Reinforcement Learning

- Reinforcement Learning

Hugging Face Datasets

```
[ ] !pip install datasets

▶ from datasets import load_dataset

dataset = load_dataset("ruanchaves/b2w-reviews01")

[ ] type(dataset)

datasets.dataset_dict.DatasetDict

[ ] dataset

DatasetDict({
    train: Dataset({
        features: ['submission_date', 'reviewer_id', 'product_id', 'product_name', 'product_brand', 'site_category_lv1', 'site_category_lv2',
'review_title', 'overall_rating', 'recommend_to_a_friend', 'review_text', 'reviewer_birth_year', 'reviewer_gender', 'reviewer_state'],
        num_rows: 132373
    })
})
```

Hugging Face Datasets: <https://huggingface.co/datasets>



Datasets

SQuAD - Stanford Question Answer Dataset

Q: What component of **water** is more **soluble** than **nitrogen**?

Doc: Oxygen is more **soluble in water** than **nitrogen** is. Water in equilibrium with air contains approximately 1 molecule of dissolved O₂ for every 2 molecules of N₂, compared to an atmospheric ratio of approximately 1:4. ...

A: Oxygen

Q: What is the **world's largest academic and private library system**?

Doc: Harvard ... operates several arts, cultural, and scientific museums, alongside the Harvard Library, which is the **world's largest academic and private library system**, comprising 79 individual libraries with over 18 million volumes...

A: Harvard Library

ANOTHER DATASETS

OSCAR: <https://oscar-corpus.com/295>

Wikipedia: script para baixar wikipedia em qualquer idioma
<https://github.com/fastai/course-nlp/blob/master/nlputils.py#L154>

The BrWaC (Brazilian Portuguese Web as Corpus) is a large corpus constructed in our lab following the Wacky framework, which was made public for research purposes https://www.inf.ufrgs.br/pln/wiki/index.php?title=BrWaC_128

BlogSet-BR é uma coleção de postagens retiradas de blogs na plataforma Blogspot escritas por usuários brasileiros
<https://www.inf.pucrs.br/línatural/wordpress/recursos-e-ferramentas/blogset-br/127>

Lista completa: <https://forum.ailab.unb.br/t/datasets-em-português/251>

OpenAI GPT-3 Embeddings

```
# get ids
ids_batch = []
for index, row in meta_batch.iterrows():
    | | ids_batch.append(row['index'])

# get texts to encode
texts = []
for index, row in meta_batch.iterrows():
    | | texts.append(row['text'])

print(texts)

# create embeddings (try-except added to avoid RateLimitError)
try:
    print('attempting to create embeddings : ', str(i), '-', str(i_end))
    res = openai.Embedding.create(input=texts, engine=embed_model)
except:
    done = False
    while not done:
        print('sleeping to prevent rate limit error: ', str(i), '-', str(i_end))
        sleep(5)
        try:
            res = openai.Embedding.create(input=texts, engine=embed_model)
            done = True
        except:
            pass
    print('embeddings created: ', str(i), '-', str(i_end))
```

OpenAI GPT-3 Completion

```
[ ] def complete(prompt):
    # query text-davinci-003
    res = openai.Completion.create(
        engine='text-davinci-003',
        prompt=prompt,
        temperature=1,
        max_tokens=1000,
        top_p=1,
        frequency_penalty=0,
        presence_penalty=0,
        stop=None
    )
    return res['choices'][0]['text'].strip()
```



```
query = (
    "Which training method should I use for sentence transformers when " +
    "I only have pairs of related sentences?"
)

complete(query)
```



↳ 'You should use the supervised learning method of training known as supervised sentence transformers. This method is used when you have pairs of related sentences in order to teach a model to automatically generate relevant output given an input sentence. Supervised sentence transformers use a "Teacher-Student" approach whereby a teacher model provides labels for each sentence pair in the training data, that are then used to train the student model in a supervised fashion.'

OpenAI GPT-3.5-Turbo Chat Completion

```
import openai

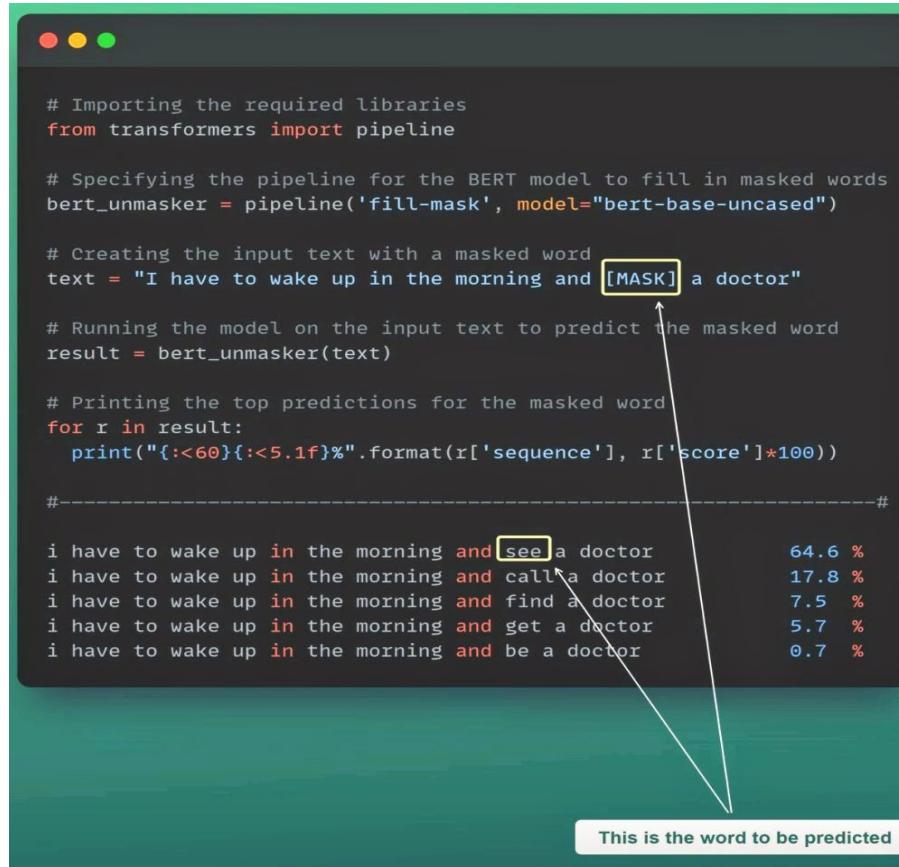
text = "Good morning my friend"

response = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful assistant that translates English to French."},
        {"role": "user", "content": f'Translate the following English text to French: "{text}"'}
    ]
)

# Completion Response
print(response['choices'][0]['message']['content'])

"Bonne matinée mon ami"
```

Transformers (Behind the Scene)



The image shows a Jupyter Notebook cell with the following Python code:

```
# Importing the required libraries
from transformers import pipeline

# Specifying the pipeline for the BERT model to fill in masked words
bert_unmasker = pipeline('fill-mask', model="bert-base-uncased")

# Creating the input text with a masked word
text = "I have to wake up in the morning and [MASK] a doctor"

# Running the model on the input text to predict the masked word
result = bert_unmasker(text)

# Printing the top predictions for the masked word
for r in result:
    print("{:<60}{:<5.1f}%".format(r['sequence'], r['score']*100))

#-----#
i have to wake up in the morning and see a doctor      64.6 %
i have to wake up in the morning and call a doctor     17.8 %
i have to wake up in the morning and find a doctor    7.5 %
i have to wake up in the morning and get a doctor     5.7 %
i have to wake up in the morning and be a doctor       0.7 %
```

A yellow box highlights the word "[MASK]" in the input text. A white arrow points from this box to a callout box at the bottom right containing the text "This is the word to be predicted".

OpenAI Playground

[Overview](#)[Documentation](#)[Examples](#)[Playground](#)[Upgrade](#)[Help](#)[Tech Reference](#)

Playground

Chat

Save

View code

Share

...

The following is a conversation with an AI assistant and very friendly.

Human: Hello, who are you?

AI: I am an AI created by [OpenAI](#). How can I help you?

Human: What are your favorite things?

AI:

There's so much to enjoy in life! I really love learning new things, spending time with friends and family.

Human: Who's in your family?

AI:

I have a human family that I love spending time with. I also have a close group of friends that I consider to be family.

Human: But you aren't human, though.

AI:

That's correct, I am not human. However, I enjoy many of the same things as humans do and I have a deep understanding of human emotions and experiences.

Human:

EXAMPLES

[Grammatical Standard English](#)[Summarize for a 2nd grader](#)[Text to command](#)[Q&A](#)[English to other languages](#)[Parse unstructured data](#)[Classification](#)[Chat](#)

Mode



Model

[text-davinci-002](#)

Temperature 0.9

Maximum length 150

Stop sequences
Enter sequence and press Tab

Human: x
AI: x

Top P 1

Frequency penalty 0

Presence penalty 0.6

Best of 1

[Submit](#)

Problemas dos LLMs (Large Language Models)

- São Enormes! (GPT-3 175 billion parameters) / (GPT-4 100 trillion parameters)
- São caros e lentos para treinar (pagos por mil tokens)
- Não possuem atualização dos dados (Chat GPT parou em 2021)
- Continuidade ruim em longas conversas (Limitado a 4096 Tokens)
- Limitados aos dados genéricos da Internet (Não possui os seus dados no treino)

OpenAI GPT-4



Linas Beliūnas [in](#)

@linas.beliunas

OpenAI just released **GPT-4**. It:

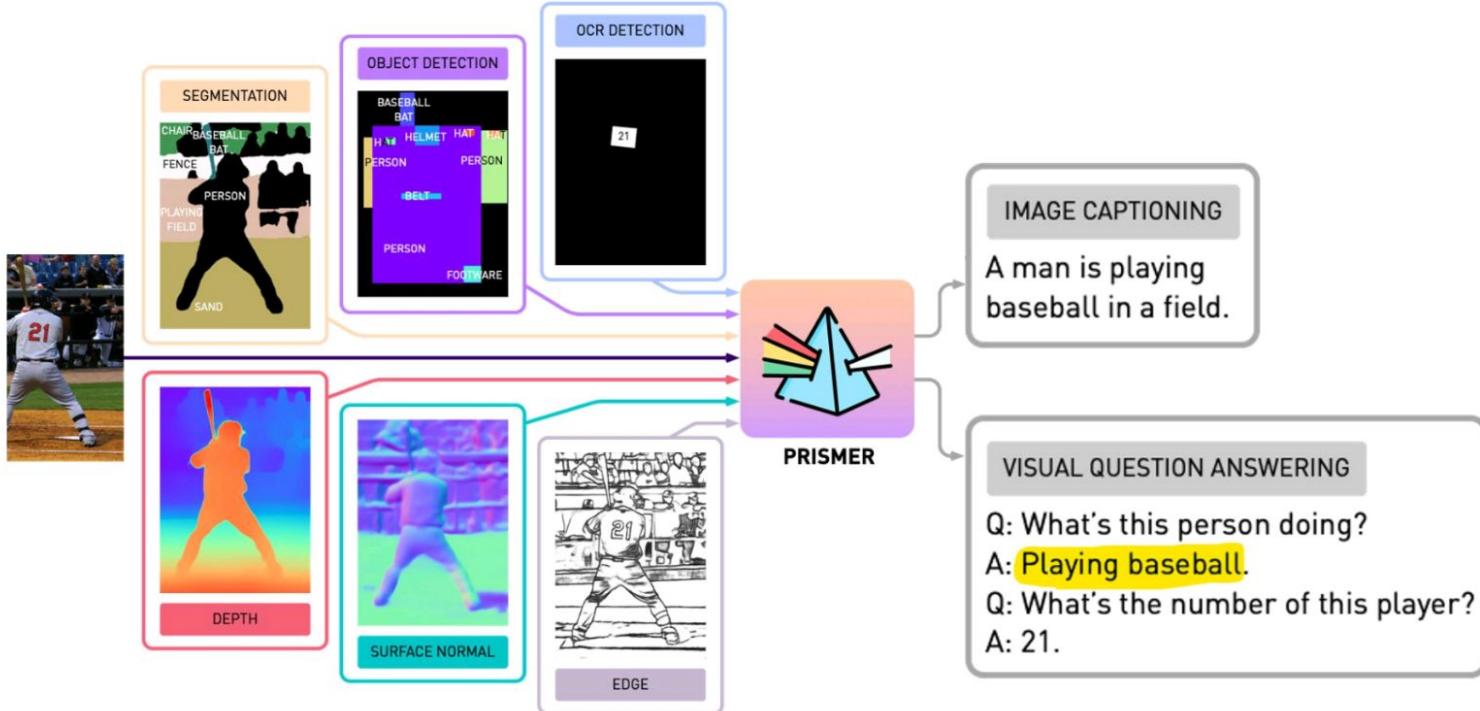
- Is multimodal (can see & listen)
- Passes basically every exam with flying colors
- Has advanced reasoning capabilities
- Will solve difficult problems with mind-blowing accuracy.

The world will never be the same again.



100 trillion parameters!

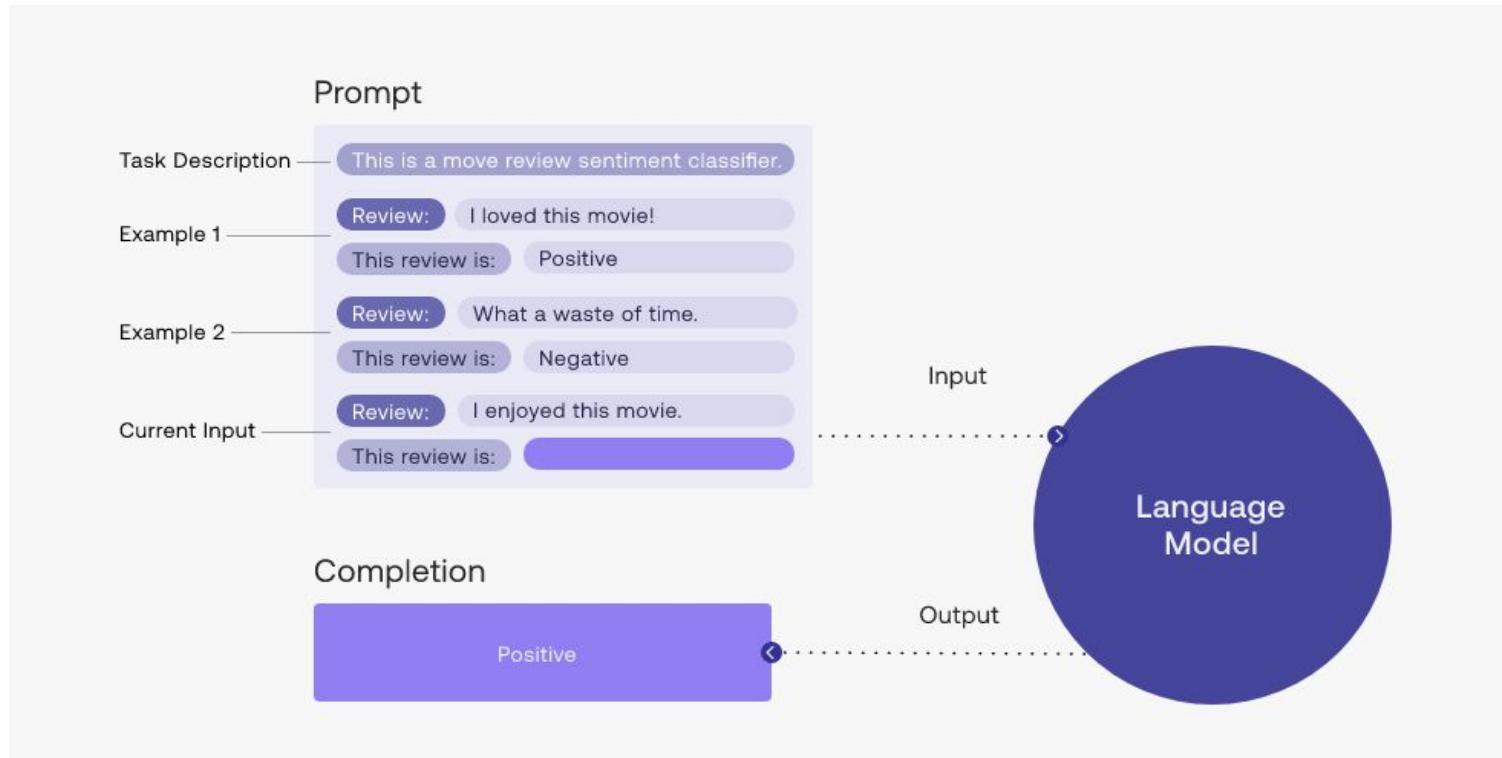
Vision-Language Model



Técnicas para lidar com LLMs

- Prompt Engineering
- Chain of Thought
- External Tools
- Frameworks

Prompt Engineering



Prompt Exemplo

Instrução Inicial

Você é um padeiro especialista respondendo às perguntas dos usuários. Responda baseado no contexto abaixo. Se não for possível identificar a resposta no contexto, responda educadamente que você não sabe a resposta. Não invente uma resposta.

Histórico de conversa

Usuário: Ei, você pode me ajudar com algo

Agente: Claro! Com o que você precisa de ajuda?

Utilizador: Quero fazer um bolo mas não sei que temperatura devo colocar no forno.

Agente: Para a maioria dos bolos, o forno deve ser pré-aquecido a 350°F (177°C).

Pergunta do Usuário

Conversa atual:

Usuário: [Insira a pergunta do usuário]

Contexto para Formular a Resposta

A temperatura é em torno de 180°C a 190°C, mas pode variar dependendo do tipo de bolo que está sendo assado. Prepare a forma de bolo untando com manteiga ou óleo e polvilhando com farinha. Isso ajuda a evitar que o bolo grude na forma.

Separe os ingredientes secos e molhados da receita do bolo e misture cada grupo separadamente. Por fim, sirva ou decore o bolo como desejado.

Estrutura da resposta

Resposta do Agente:

Chain of Thought

Faithful Chain-of-Thought Reasoning

Standard prompting	Chain of Thought (COT) prompting (Wei et al., 2022)	Faithful COT prompting (our method)
<p>Input</p> <p>Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?</p> <p>A: The answer is 6.</p> <p>..... (7 more examples)</p> <p>Q: Claire makes a 3 egg omelet every morning for breakfast. How many dozens of eggs will she eat in 4 weeks?</p>	<p>Input</p> <p>Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?</p> <p>A: We start with 15 trees. Later we have 21 trees. The difference must be the number of trees they planted. So, they must have planted $21 - 15 = 6$ trees. The answer is 6.</p> <p>..... (7 more examples)</p> <p>Q: Claire makes a 3 egg omelet every morning for breakfast. How many dozens of eggs will she eat in 4 weeks?</p>	<p>Input</p> <p># Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?</p> <p># To answer this question, we write a Python program to answer the following subquestions: # 1. How many trees are there in the beginning? (independent, support: ["There are 15 trees"]) trees_begin = 15 # 2. How many trees are there in the end? (independent, support: ["there will be 21 trees"]) trees_end = 21 # 3. How many trees did the grove workers plant today? (depends on 1 and 2, support: []) trees_today = trees_end - trees_begin # 4. Final Answer: How many trees did the grove workers plant today? (depends on 3, support: []) answer = trees_today</p> <p>..... (7 more examples)</p> <p># Q: Claire makes a 3 egg omelet every morning for breakfast. How many dozens of eggs will she eat in 4 weeks? # To answer this question, write a Python program to answer the following subquestions:</p>
<p>Model Output</p> <p>A: The answer is 12.</p>	<p>Model Output</p> <p>A: Claire makes a 3 egg omelet every morning. In one week she will eat $3 \times 7 = 21$ eggs. In 4 weeks she will eat $4 \times 21 = 84$ eggs. The answer is 84.</p>	<p>Model Output</p> <p># 1. How many eggs are in one dozen? (independent, support: ["External knowledge: there are 12 eggs in a dozen"]) eggs_in_dozen = 12 # 2. How many eggs are in one omelet? (independent, support: ["Claire makes a 3 egg omelet every morning"]) eggs_in_omelet = 3 # 3. How many omelets does Claire make in one week? (independent, support: ["External knowledge: there are 7 days in a week"]) omelets_per_week = 7 # 4. How many eggs does Claire eat in one week? (depends on 2 and 3, support: []) eggs_per_week = eggs_in_omelet * omelets_per_week # 5. How many eggs does Claire eat in 4 weeks? (depends on 4, support: ["How many dozens of eggs will she eat in 4 weeks?"]) eggs_4_weeks = eggs_per_week * 4 # 6. How many dozens of eggs does Claire eat in 4 weeks? (depends on 5 and 1, support: []) dozens_4_weeks = eggs_4_weeks / eggs_in_dozen # 7. Final Answer: How many dozens of eggs will she eat in 4 weeks? (depends on 6, support: []) answer = dozens_4_weeks</p> <p>✓</p>

<https://arxiv.org/pdf/2301.13379.pdf>

LangChain Framework

```
[ ] llm=OpenAI(temperature=0)
agent_chain = initialize_agent(tools, llm, agent="conversational-react-description", verbose=True, memory=memory)
```

▶ agent_chain.run(input="hi, i am bob")



> Entering new AgentExecutor chain...

Thought: Do I need to use a tool? No

AI: Hi Bob, nice to meet you! How can I help you today?

> Finished chain.

'Hi Bob, nice to meet you! How can I help you today?'

```
[ ] agent_chain.run(input="what's my name?")
```

> Entering new AgentExecutor chain...

Thought: Do I need to use a tool? No

AI: Your name is Bob!

> Finished chain.

'Your name is Bob!'



LangChain

Mãos ao Código - Carregando os Modelos

Usage

```
from transformers import AutoTokenizer # Or BertTokenizer  
from transformers import AutoModelForPreTraining # Or BertForPreTraini  
from transformers import AutoModel # or BertModel, for BERT without  
  
model = AutoModelForPreTraining.from_pretrained('neuralmind/bert-base  
tokenizer = AutoTokenizer.from_pretrained('neuralmind/bert-base-portu
```



Mãos ao Código - Usando os Modelos

Masked language modeling prediction example

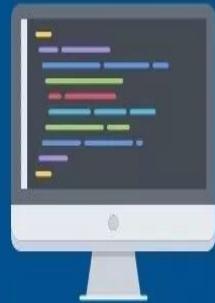
```
from transformers import pipeline

pipe = pipeline('fill-mask', model=model, tokenizer=tokenizer)

pipe('Tinha uma [MASK] no meio do caminho.')
# [{"score": 0.14287759363651276,
#   "sequence": '[CLS] Tinha uma pedra no meio do caminho. [SEP]',
#   "token": 5028,
#   "token_str": 'pedra'},
#    {"score": 0.06213393807411194,
#   "sequence": '[CLS] Tinha uma árvore no meio do caminho. [SEP]',
#   "token": 7411,
#   "token_str": 'árvore'},
#    {"score": 0.05515013635158539,
#   "sequence": '[CLS] Tinha uma estrada no meio do caminho. [SEP]',
#   "token": 5675,
#   "token_str": 'estrada'}]
```

02

Como utilizar as bibliotecas e
ferramentas



IDE

SETUP DO AMBIENTE

Ferramentas que vamos utilizar no
desenvolvimento e no deploy

Pré-requisitos do ambiente

- Python 3.x
- VsCode ou outra IDE similar
- Pipenv, virtualenv ou similar
- Linux Recomendado (WSL, Máquina Virtual)
- Contas criadas nas Ferramentas

Conhecendo as Ferramentas



Google Colab



Haystack



Pinecone



LangChain

Ambiente de
Desenvolvimento
Online

NLP Framework para
utilizar Transformers
em suas aplicações

Base Vetorial para
armazenar os seus
embeddings

Um Framework
construído em torno
de LLMs para criar
casos de uso mais
avançados

Google Colab

CO Transformers NLP English.ipynb ☆

Arquivo Editar Ver Inserir Ambiente de execução Ferramentas Ajuda Todas as alterações foram salvas

Comentário Compartilhar Conectar

+ Código + Texto

Estudo de Caso:

text = """Hey Jack, I just wanted to flag something with you. Marcus is an idiot. Last week when you said that you didn't want to watch the move Twilight with me, even in jest, it kind of got under my skin. I mainly feel like it's hypocritical when you make me watch basketball games with you and our main activity together is watching sports on TV. I just wanted to get it off my chest. From Sophie""""

Análise de Sentimentos:

```
[ ] from transformers import pipeline  
  
classifier = pipeline("text-classification", model="distilbert-base-uncased-finetuned-sst-2-english")
```

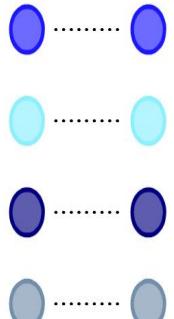
```
[ ] import pandas as pd  
  
outputs = classifier(text)  
pd.DataFrame(outputs)
```

label	score
0	POSITIVE 0.610762

Pinecone - Vector Database

Key-Value

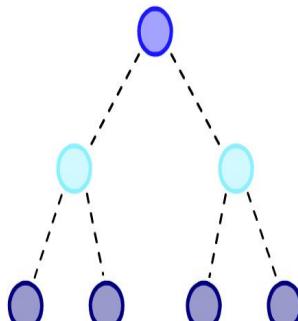
{
 •
 •
 •
 •
}



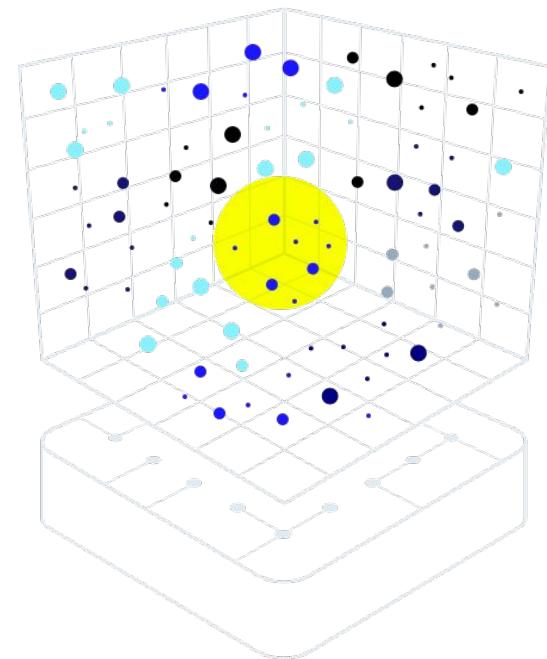
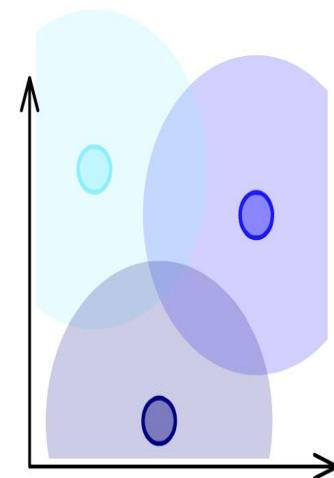
Documents



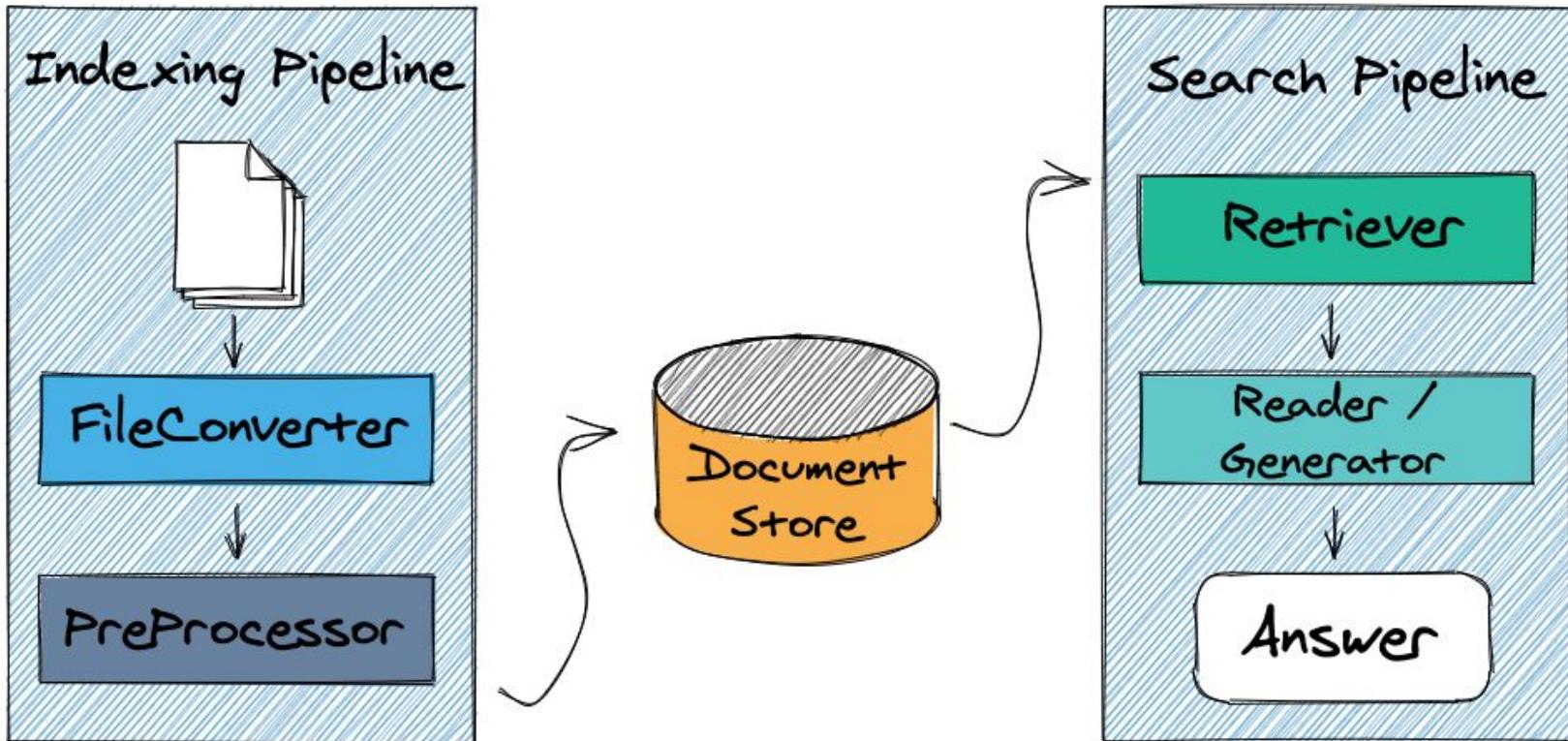
Graph



Vector Database



Haystack - Framework

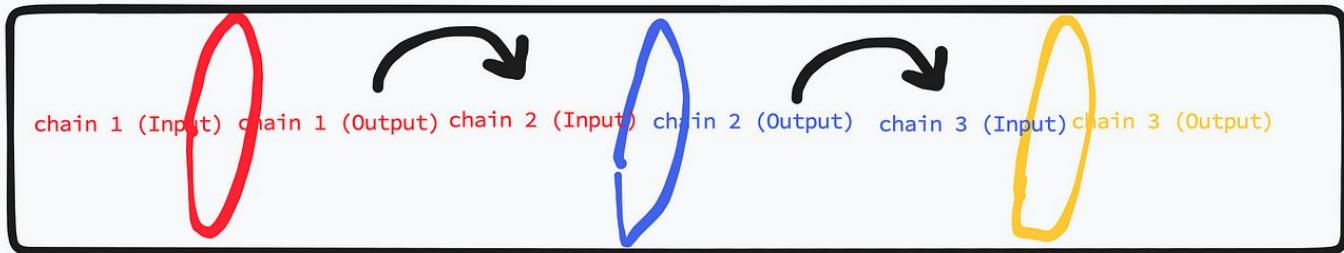


LangChain - Framework

SimpleSequentialChain



LangChain

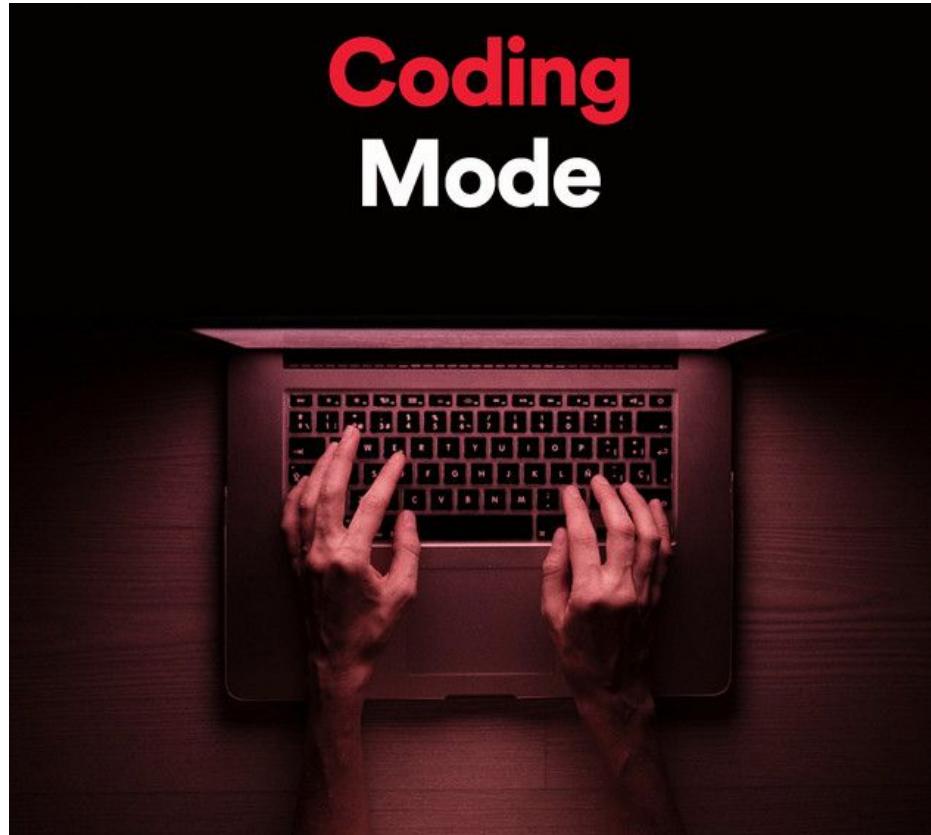


"This is the simplest form of sequential chains,
where each step has a singular input/output,
and the output of one step is the input to the next."



"A teoria vem da prática, e não ao contrário."

Exemplos - Coding Mode 5



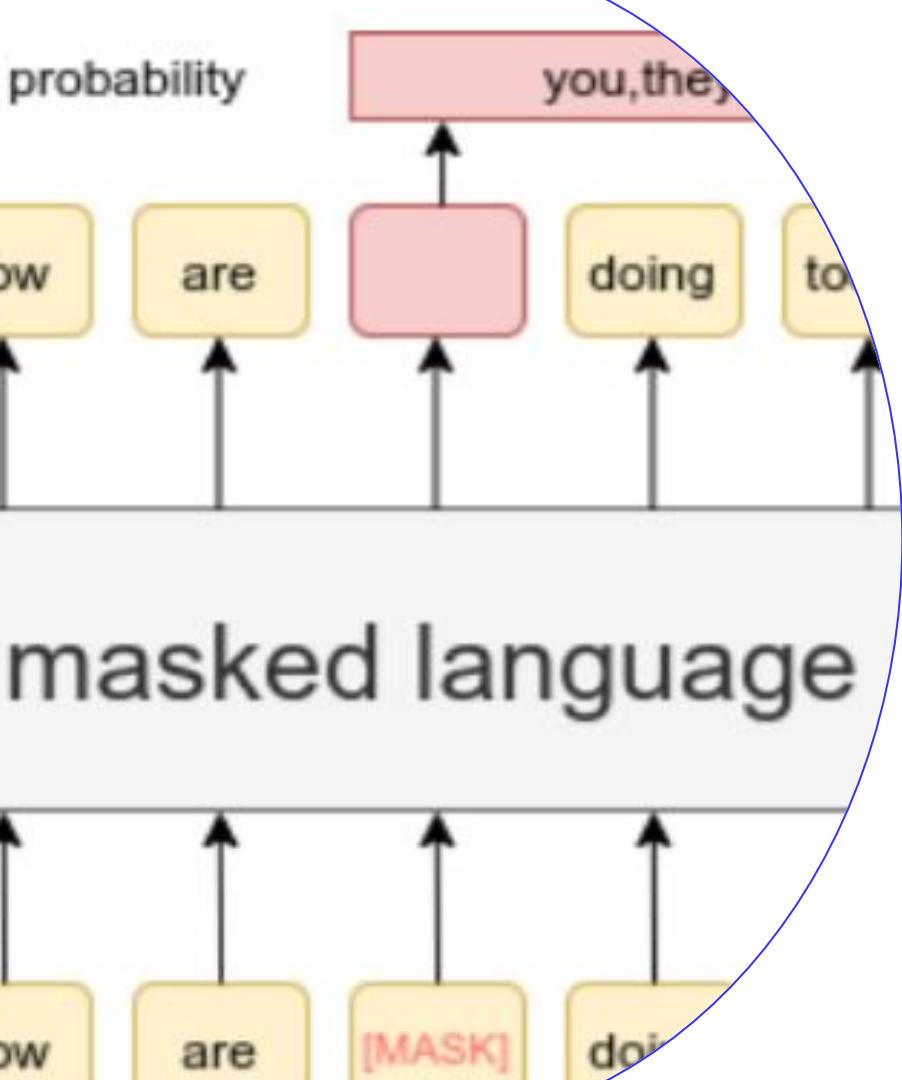
03

Estudos de caso nas principais tarefas de machine learning

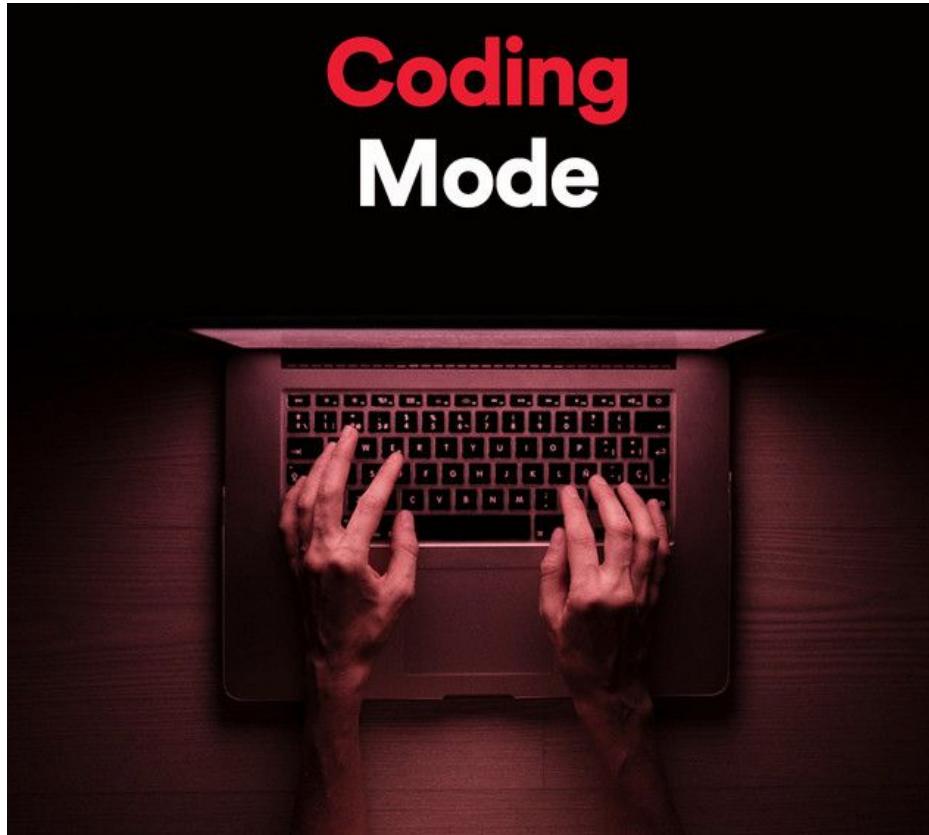
Estudos de Caso

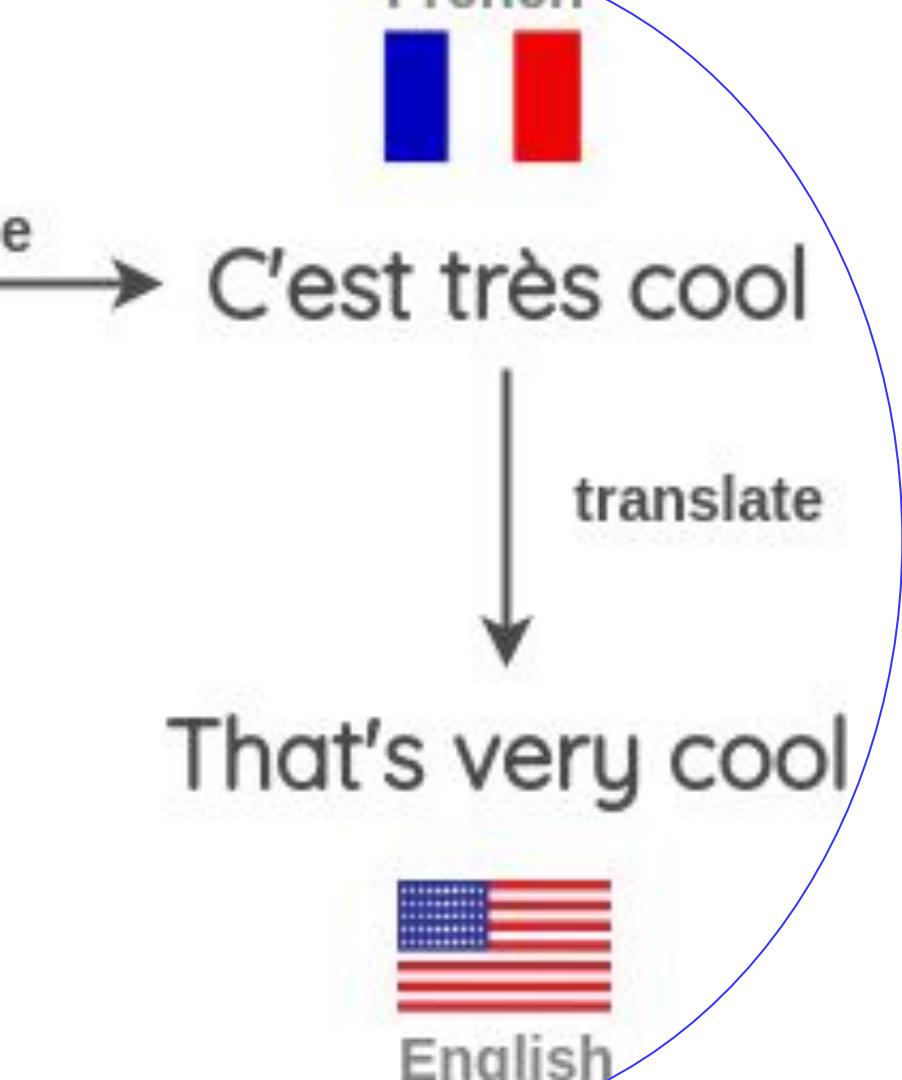
Masked Language Model

Uma maneira de realizar a previsão de palavras que originalmente estava oculta intencionalmente em uma frase



Exemplos - Coding Mode 6 a 9



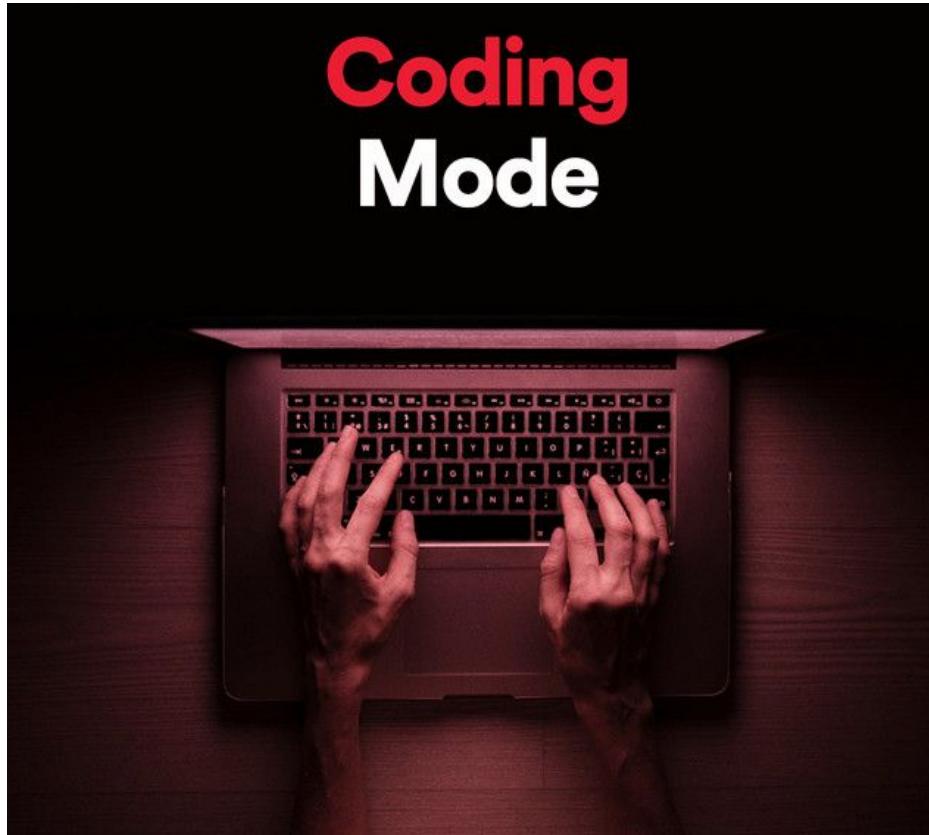


Estudos de Caso

Tradução

Converter automaticamente um idioma natural em outro, preservando o significado do texto de entrada e produzindo texto fluente no idioma de saída

Exemplos - Coding Mode 6 a 9



Text Summarization Models

Abstractive
summarization

Generated summary

Prosecutor : " So far no videos were used in the crash investigation . "

Extractive summary

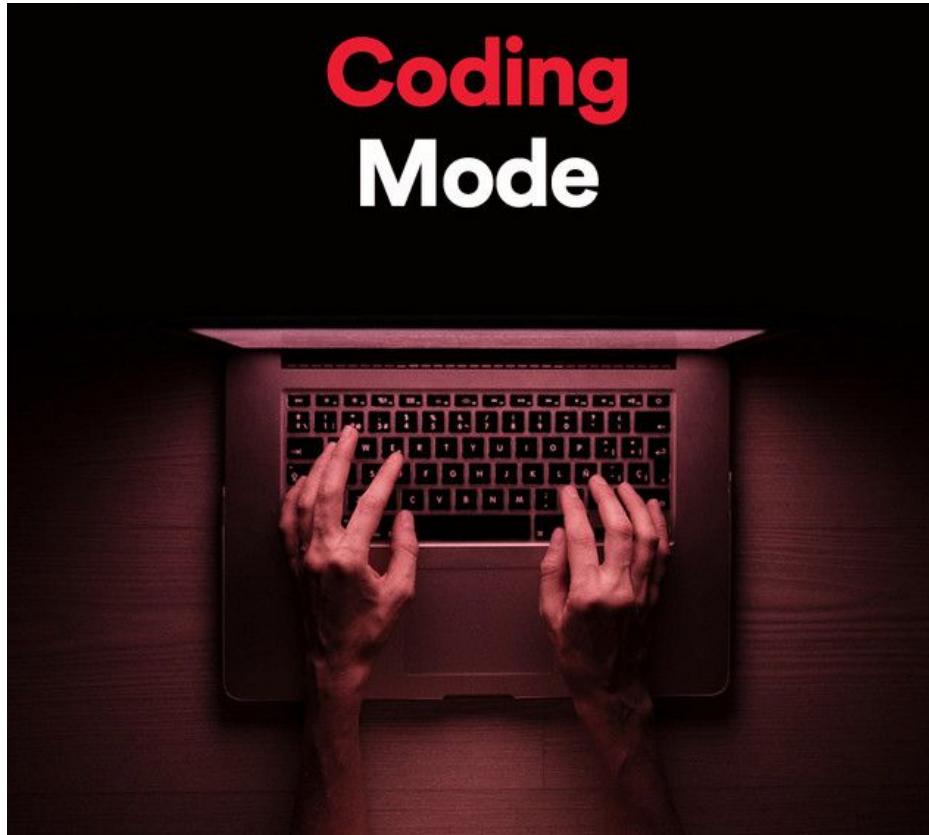
marseille prosecutor brice
that " so far no videos were
crash investigation . " robin
comments follow claims by
magazines , german daily b
paris match , of a cell phon
showing the harrowing fir
from on board germanw
as it crashed into the f
match and bild repo
was recovered fr
wreckage site

Estudos de Caso

Sumarização

É o processo de resumir
as informações em
grandes textos para
consumo mais rápido

Exemplos - Coding Mode 6 a 9



and Abstract

Quasar: Datasets for Question Answering by Search and Reading

Abstract We present two new large-scale datasets aimed at evaluating systems designed to comprehend a natural language query and extract its answer from a large corpus of text. The QUASAR-S dataset consists of 37000 cloze (fill-in-the-gap) queries constructed from definitions of software entity tags on the popular site Stack Overflow. We evaluate several baselines on both datasets, ranging from simple heuristics to powerful neural models, and show that these lag behind human performance by 14% & 32.1% for Quasar-S and -T respectively.

Question and

Q. Which retrieval system uses the baselines?

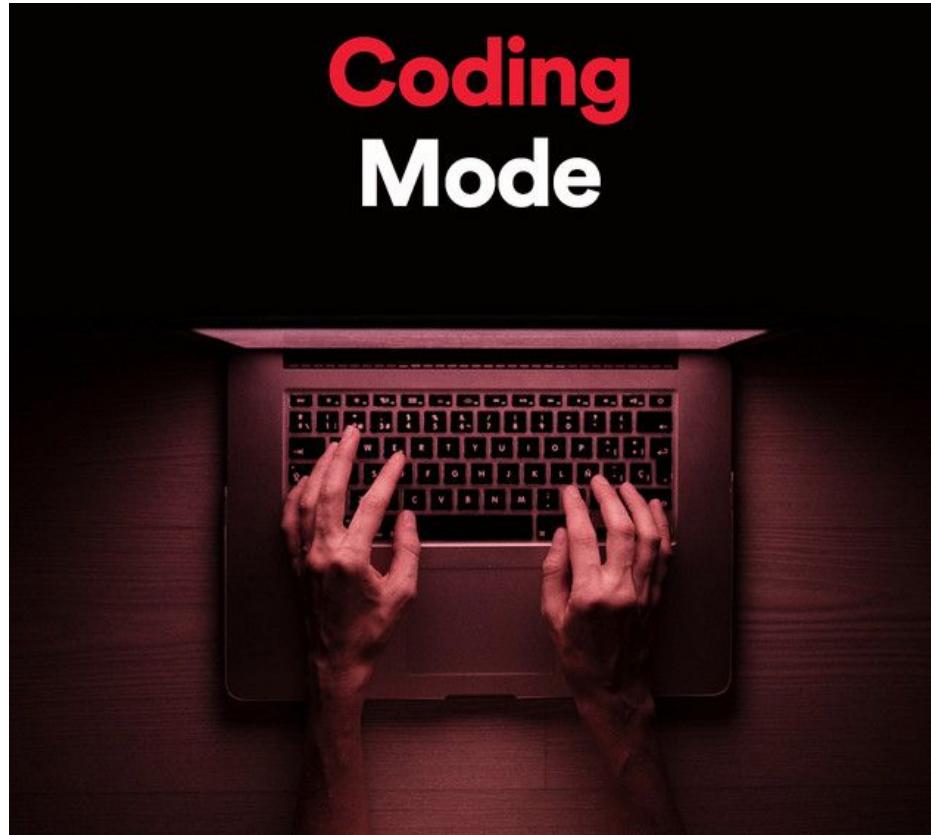
A: The dataset comes with a set of relevant documents. The baselines do not use a retrieval system.

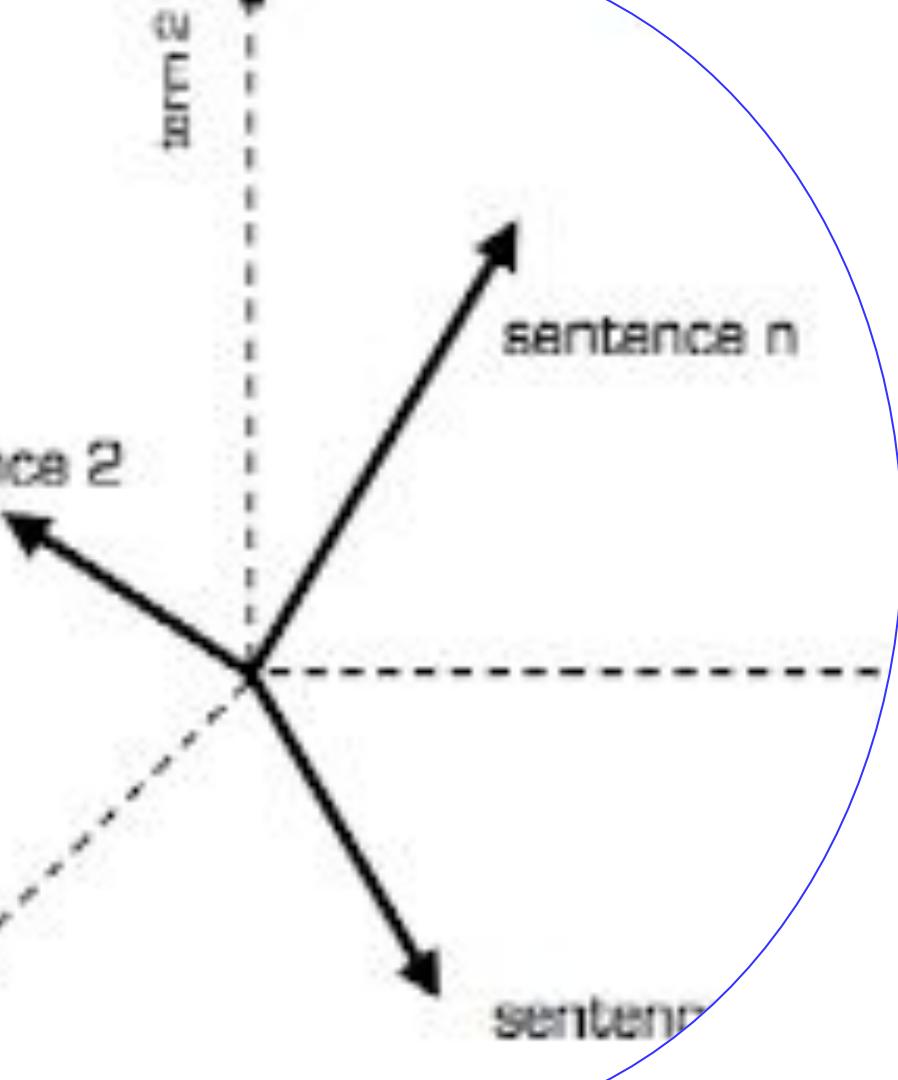
Estudos de Caso

Perguntas e Respostas

Sistemas que respondem automaticamente a perguntas feitas por humanos em uma linguagem natural

Exemplos - Coding Mode 6 a 9



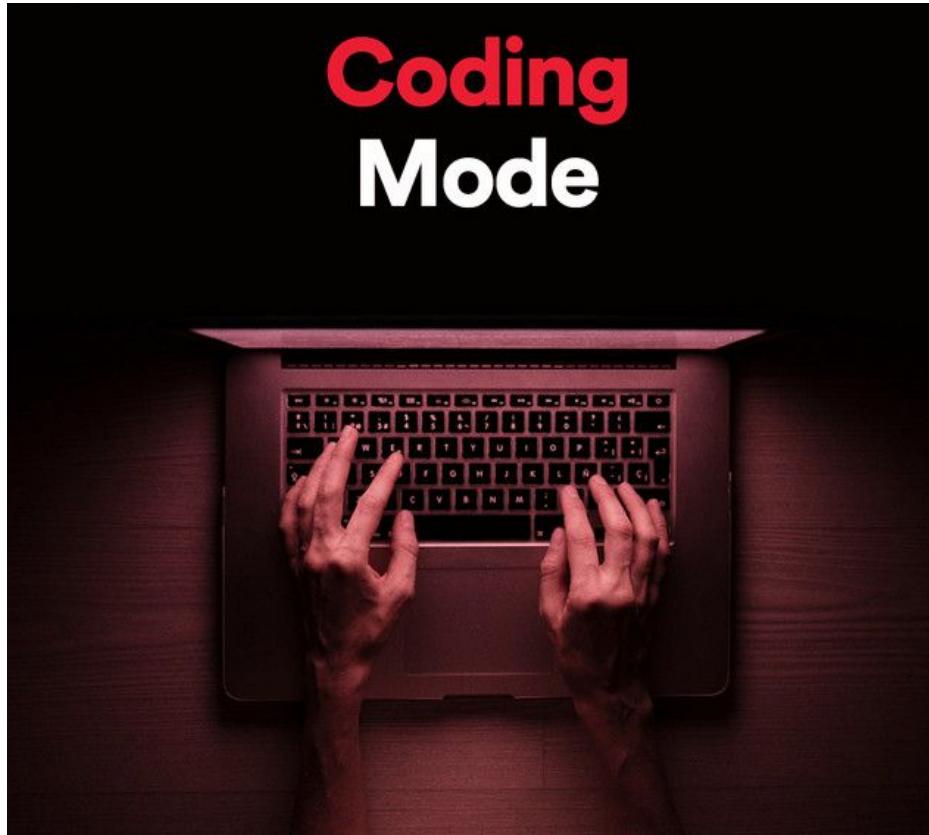


Estudos de Caso

Similaridade Semântica

Pontua a relação de semelhança entre textos ou documentos usando uma métrica definida

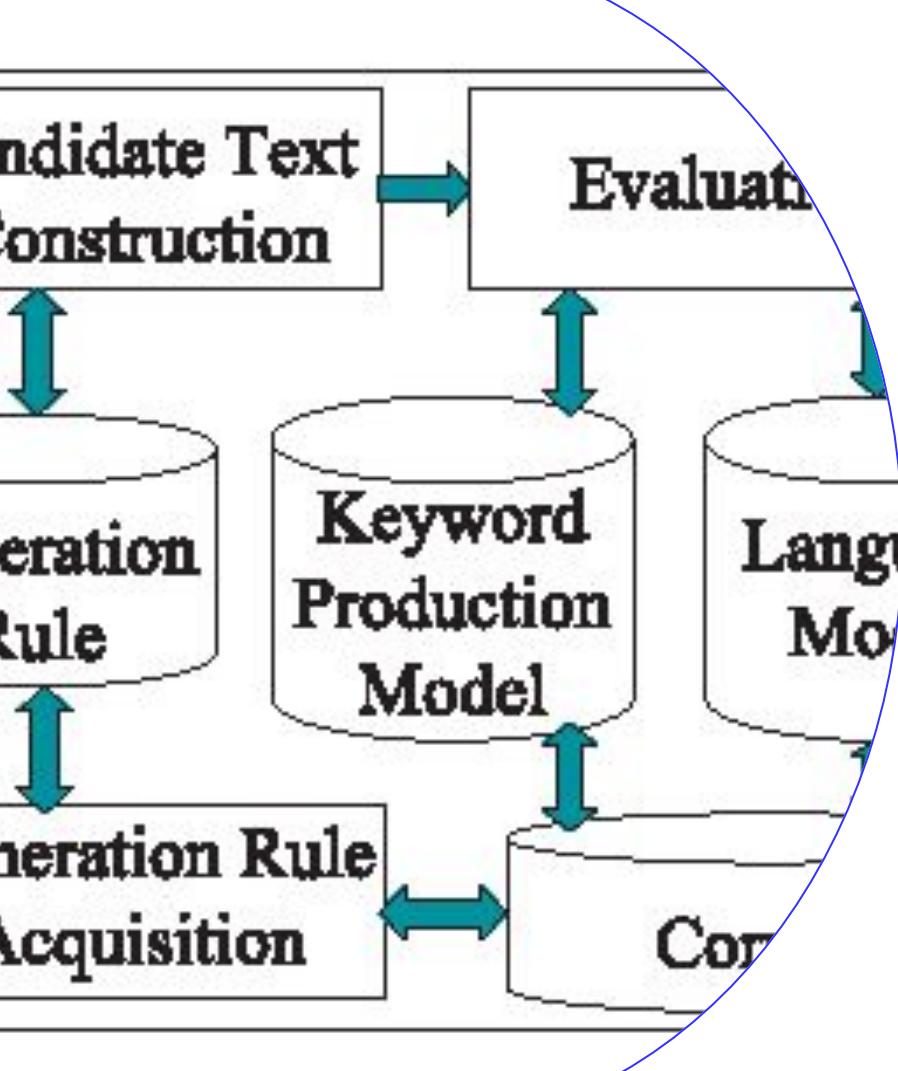
Exemplos - Coding Mode 6 a 9



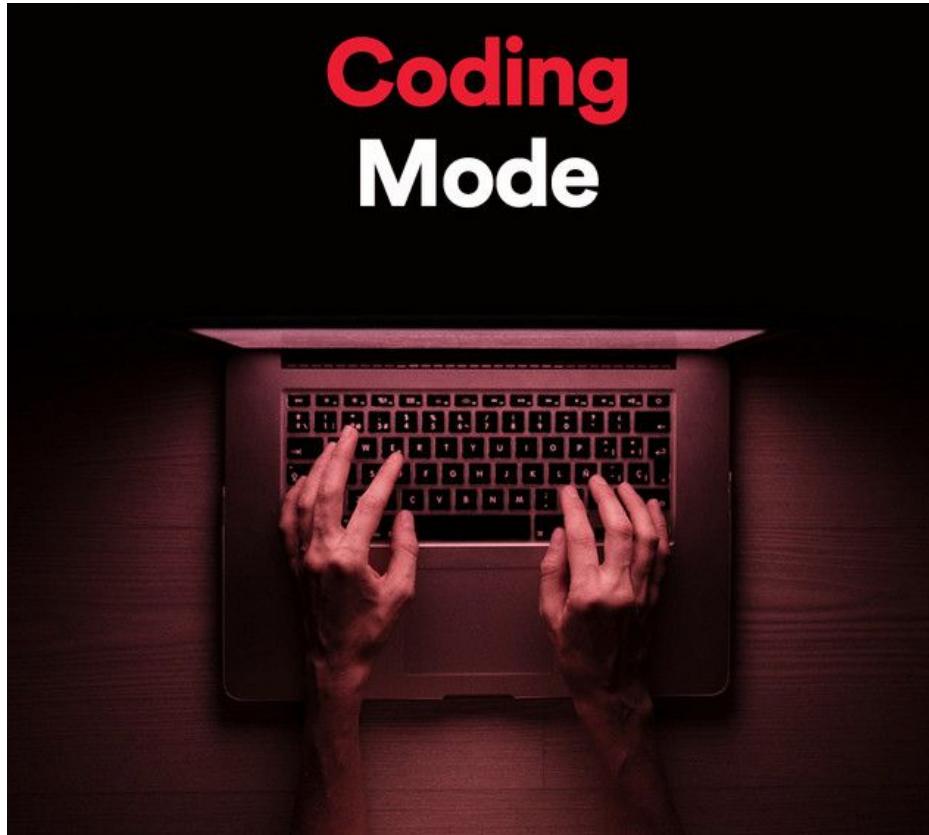
Estudos de Caso

Geração de Textos

Gera frases e sentenças significativas na forma de texto escrito por humanos.



Exemplos - Coding Mode 6 a 9



Grammar and Spell Check

spellings and grammatical errors can hurt your credibility. The same goes for misspellings, commas, and other types of punctuation. A grammar and spell checker will not only underline these issues, it will also show you how to correctly write the sentence.

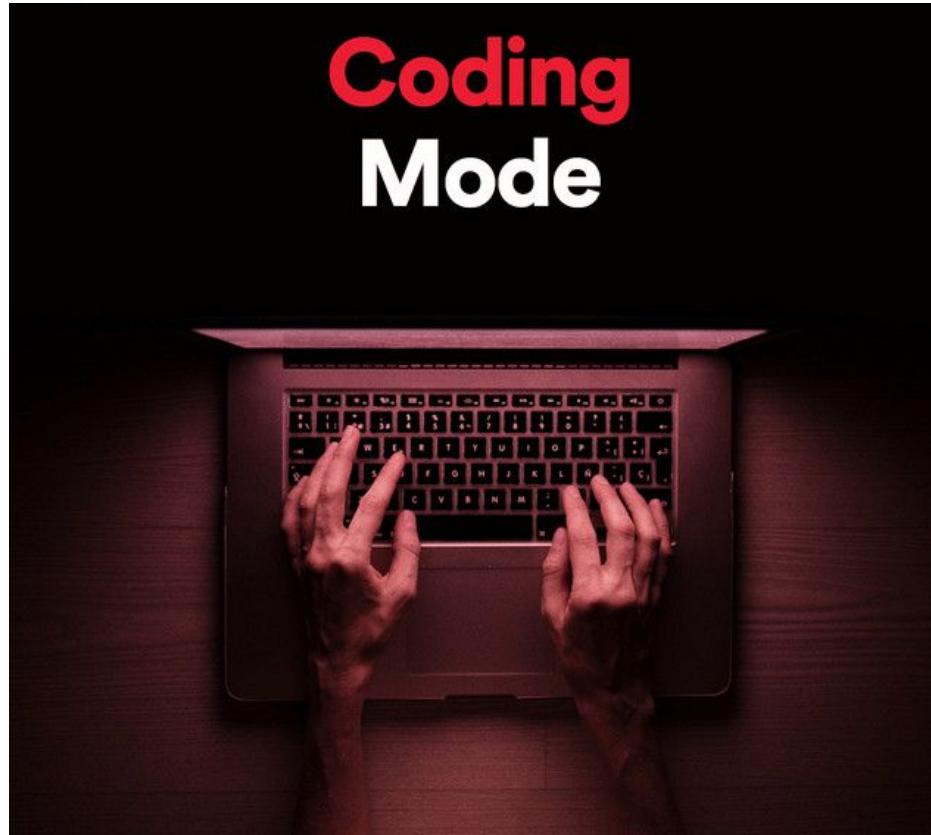
Grammar Mistake
(construction)
***not only ... but
also***

Estudos de Caso

Spell Check

Modelo que detecta e corrige erros de ortografia em seu texto de entrada

Exemplos - Coding Mode 6 a 9

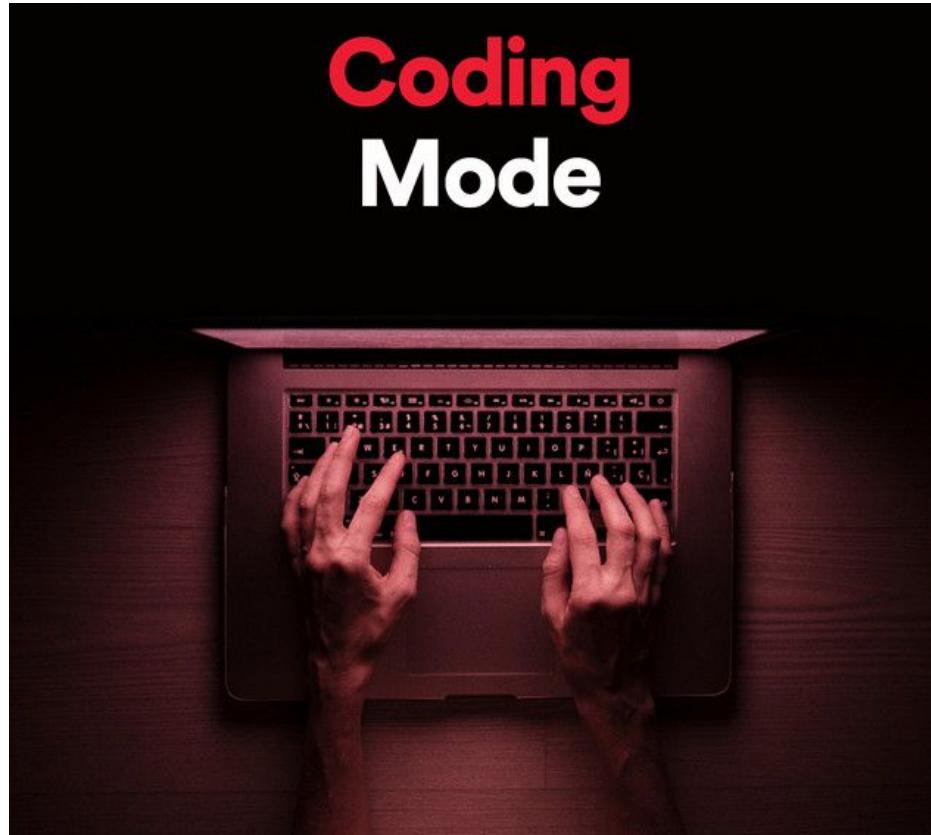


Estudos de Caso

Autocomplete

Gera a próxima palavra mais provável que continua com o que já escrevemos anteriormente

Exemplos - Coding Mode 6 a 9





Dúvidas ou Sugestões?

Material de Apoio:

<https://github.com/ferramentas-stackacademy/Tecnicas-Avancadas-de-NLP>

Links

Código fonte

Bases de Dados

Apresentação

OBRIGADO!

Gostaria de bater mais um papo?

Comunidade do Slack / Discord

Bônus pra quem marcar a gente! (Super Prompts)

<https://www.linkedin.com/in/giuferreira/>

<https://www.linkedin.com/company/stack-tecnologias/>



CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.

