

Pré-processamento de Texto



Pré-processamento de Texto

Objetivo: limpar e padronizar textos para que possam ser usados em modelos de PLN de forma eficiente e coerente.

O pré-processamento é uma das etapas mais importantes do PLN, pois os modelos só aprendem bem quando os dados textuais estão limpos, normalizados e estruturados.



Limpeza de Dados

Etapas principais:

- Remover **pontuação, números e caracteres especiais**
- Remover **stopwords** (palavras muito comuns como “de”, “a”, “o”)
- Converter o texto para **minúsculas**
- Remover **acentos e espaços extras**

Exemplo prático (NLTK + regex):

```
import nltk
from nltk.corpus import stopwords
import re, unicodedata

# Baixar stopwords do NLTK (somente na primeira vez)
nltk.download('stopwords')

texto = "Olá! Este é um exemplo de pré-processamento de texto, em Português."

# Coloca tudo em minúsculas
texto = texto.lower()

# Remove pontuação e caracteres especiais
texto = re.sub(r'[^\w\s]', '', texto)

# Remove acentos
texto = unicodedata.normalize('NFKD', texto).encode('ascii', 'ignore').decode('utf-8')
```

```
# Remove stopwords
stop_words = set(stopwords.words('portuguese'))
palavras = [w for w in texto.split() if w not in stop_words]

print(palavras)
# Saída: ['ola', 'exemplo', 'preprocessamento', 'texto', 'portugues']
```

Tokenização

Definição: processo de dividir o texto em unidades menores chamadas *tokens* (palavras, subpalavras ou frases).

Tipos principais:

- **Palavra:** divide por espaços e pontuação
- **Subpalavra:** divide partes menores (WordPiece, BPE)
- **Frase:** quebra por sentenças completas

Exemplo com SpaCy:

```
import spacy
nlp = spacy.load("pt_core_news_sm")

doc = nlp("Modelos de linguagem são incríveis!")
tokens = [token.text for token in doc]

print(tokens)
# Saída: ['Modelos', 'de', 'linguagem', 'são', 'incríveis', '!']
```

Lematização e Stemming

Técnica	Descrição	Exemplo
Stemming	Reduz palavras ao radical (forma truncada, sem contexto).	<i>correr, correndo → corr</i>
Lematização	Reduz à forma canônica (usa regras linguísticas).	<i>melhores → bom</i>



Dica: use lematização quando a **semântica** for importante; use stemming quando quiser **desempenho rápido**.

Exemplo com NLTK (stemming):

```
from nltk.stem import RSLPStemmer

stemmer = RSLPStemmer()
palavras = ["correndo", "correu", "corrida", "correria"]

for p in palavras:
    print(p, "→", stemmer.stem(p))
# Saída aproximada:
# correndo → corr
# correu → corr
# corrida → corr
# correria → corr
```

Exemplo com SpaCy (lematização):

```
import spacy
nlp = spacy.load("pt_core_news_sm")

doc = nlp("Os melhores estudantes estão estudando muito.")
lemmas = [token.lemma_ for token in doc]

print(lemmas)
# Saída: ['o', 'bom', 'estudante', 'estar', 'estudar', 'muito', '.']
```



Normalização

Objetivo: padronizar o formato do texto (maiúsculas, acentos, números etc.)

Passos comuns:

- Converter para **minúsculas**
- Remover **acentos**
- Substituir **números por tokens** (2025 → <NUM>)
- Expandir **abreviações** e **contrações**

Exemplo simples:


```
texto = "Eu MORO em São Paulo, desde 2020!"
texto_norm = texto.lower()
# → "eu moro em são paulo, desde 2020!"

import unicodedata
texto_norm = unicodedata.normalize('NFKD', texto_norm).encode('ascii',
'ignore').decode('utf-8')
# → "eu moro em sao paulo, desde 2020!"
```



Observações Importantes

- O pré-processamento ideal depende da **tarefa**:
 - Análise de sentimentos → manter palavras emocionais.
 - Tradução automática → preservar estrutura gramatical.
 - LLMs → otimizar desempenho e tempo de resposta
- Evite remover palavras que **carreguem significado** (ex: “não”).
- Para textos grandes, use **pipelines automatizadas**:
 - `SpaCy (nlp.pipe)`
 - `HuggingFace Tokenizers`

 **Lembre-se:** o pré-processamento é o alicerce da qualidade dos embeddings e do desempenho de qualquer modelo de PLN.