Usando Python e Deep Learning para Análise de Sentimentos e Reconhecimento de Contexto

Ribeirão Preto
23 a 28 de outubro

Olá!

Arthur Fortes

Cientista de Dados @ Cellere

Doutorando em Computação @ ICMC - USP

@fortesarthur | arthurfortes.github.io

Agenda

1. Processamento de texto
2. Problema de classificação de sentimentos/ contexto
3. Deep Learning usando Keras
4. Conclusões

1.

Processamento de Texto

O processamento de texto é uma das tarefas mais comuns em muitos aplicativos de Aprendizado de Máquina.


Information Retrieval
Doc A
Doc 1
Doc 2
Doc 3
Sentiment Analysis
Information Extraction
Machine Translation
Text Processing
QuestionAnswering
Human: When was Apollo sent to space?
Machine: First flight - AS-201, February 26, 1966

Pré-processamento de Dados

- Tokenização — converter frases em palavras;
- Remover pontuação desnecessárias e *tags;*
- Remover *stop words [SW]* (palavras comuns que não tem uma semântica específica), como por exemplo “o”, “a”, “tem”, “seu”, etc;
- Radical (Stemming) — as palavras são reduzidas a uma raiz removendo a inflexão através da eliminação de caracteres desnecessários, geralmente um sufixo.

Estou apresentando na Python Brasil!!!

Aplicando técnicas de pré-processamento:

apresent python brasil

Podemos usar python para realizar muitas operações de pré-processamento de texto:

- **NLTK**: https://www.nltk.org/
- **BeautifulSoup**: https://www.crummy.com/software/BeautifulSoup/
- **Spacy** https://spacy.io/

Tokenização

```
import nltk
from nltk.tokenize import word_tokenize
# função para dividir texto em palavras
tokens = word_tokenize("The quick brown fox jumps over the lazy dog")
print(tokens)
>>  [‘The’, ‘quick’, ‘brown’, ‘fox’, ‘jumps’, ‘over’, ‘the’, ‘lazy’, ‘dog’]
```

Remoção de Stop Words

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
stop_words = set(stopwords.words(‘english’))
tokens = [w for w in tokens if not w in stop_words]
print(tokens)
>>  [‘quick’, ‘brown’, ‘fox’, ‘jumps’, ‘lazy’, ‘dog’]
```

Stemming

```
from nltk.stem.porter import PorterStemmer
porter = PorterStemmer()
stems = []
for t in tokens:
    stems.append(porter.stem(t))
print(stems)
>> [ ‘quick’, ‘brown’, ‘fox’, ‘jump’, ‘lazi’, ‘dog’]
```

Extração de Features

- Existem diversos extratores de features para transformar texto em números:
  - Bag of words:
    - Contador de palavras
    - TF/IDF
  - Word2Vec
  - Glove: Global Vectors for Word Representation
- Gerar representações (Word Embedding) baseado nessas features.

Extração de Features

- Dicionário: {0: ‘study’, 1: ‘quick’, 2:‘brown’, 3: ‘fox’, 4: ‘jump’, 5: ‘day’, 6: ‘lazi’, 7: ‘dog’, 8: ‘house’, 9: ‘eat’, 10: ‘food’, 11: ‘hour’}
- Frase: “The dog eat every day in the same hour.”
  - Etapa 1: Tokenização e remover stop words
    - [‘dog’, ‘eat’, ‘day’, ‘hour’]
  - Etapa 2: Word Embedding
    - [8, 9, 5, 11]

2.

Problema de classificação de sentimentos/ contexto

A classificação de sentimentos/contexto é a tarefa de analisar um texto e dizer se alguém gosta ou não do que está falando e sobre o que está falando.

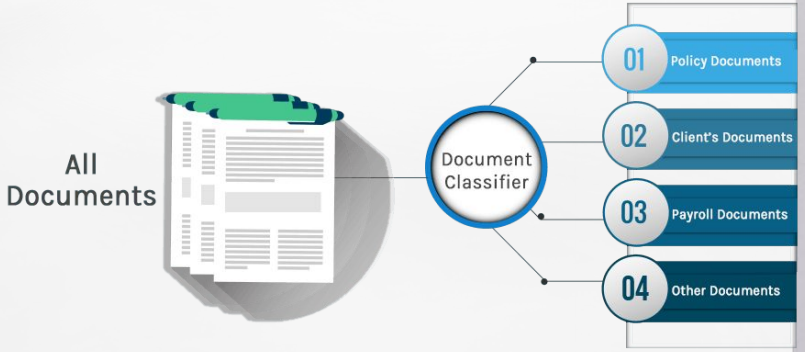
Características do Problema

- A entrada é um pedaço de texto
- A saída é o sentimento/contexto que queremos prever
  - Por exemplo: a classificação por estrela de uma crítica de cinema.

"This movie is fantastic! I really like it because it is so good!" ★★★★☆

"Not to my taste, will skip and watch another movie" ★★☆☆☆

"This movie really sucks! Can I get my money back please?" ★☆☆☆☆

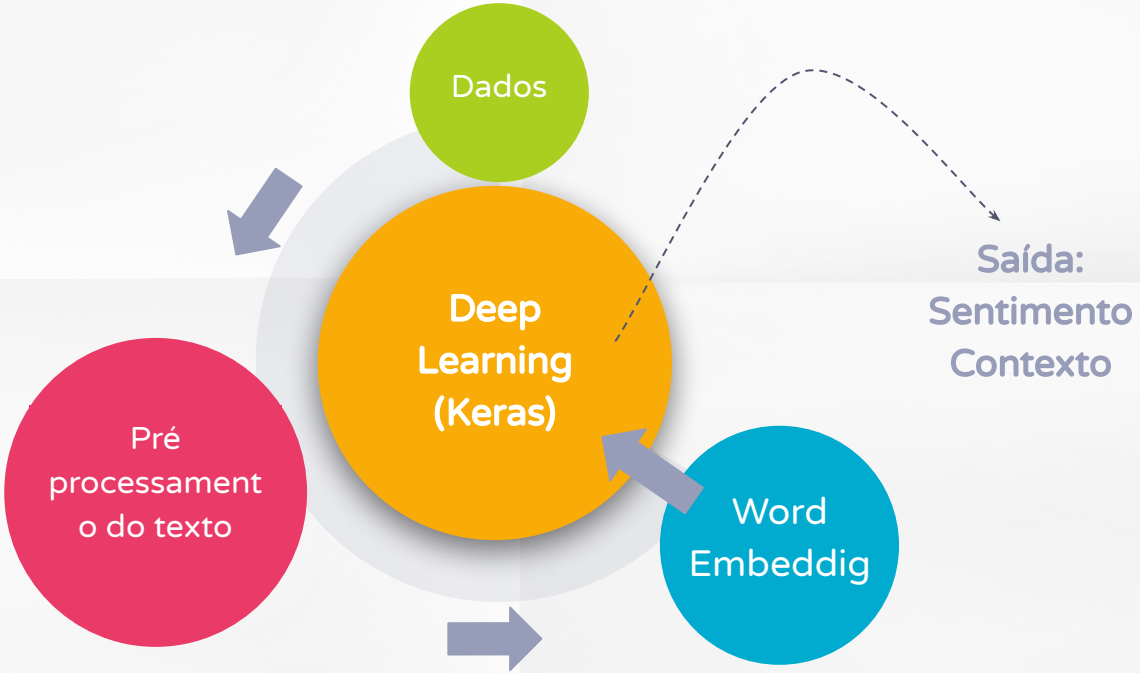


Características do Problema

- Treinar um sistema para mapear a entrada X na saída Y com base em um conjunto de dados rotulado para prever o sentimento e contexto de um texto.

- Vamos nos concentrar nas tarefas de:
  - Construir uma representação de texto para servir como entrada para um modelo de deep learning
  - Criar uma rede neural profunda para classificação de sentimentos e contexto.

Características do Problema



3.

Deep Learning usando Keras


TensorFlow ™


K
Keras


Array(d),k=new
n.ready.promise
&&(n(d).
function K(){
n(l))break;
(c.style.zoom=1)),c.
c?!1:!b||b!==!0

Código

Conclusões

- O pré-processamento é uma fase essencial para a acurácia final do classificador;
- A língua e a quantidade de dados são um impedimento;
- A configuração da rede pode mudar conforme os dados;
- Nem sempre deep learning é a melhor solução.

Obrigado!

@fortesarthur & fortes.arthur@gmail.com

PYTHON BRASIL 2019

Ribeirão Preto
23 a 28 de outubro