

part-of-speech-tagging

Trabalho da disciplina de processamento de linguagem natural, UFRN 2025.1

Aluno: Davi Diógenes Ferreira de Almeida

Repositório GitHub

1. O objetivo

O objetivo do trabalho é implementar um part of speech tagger, ou seja, um programa que classifique as palavras em uma frase de acordo com sua classe gramatical (substantivo, adjetivo etc.).

Os experimentos foram realizados com mais de uma técnica de treinamento e inferência, como o uso de unigramas, bigramas, trigramas e o condicionamento por tag no lugar de por palavra.

2. Os dados

O corpus utilizado é composto por textos tratados do Penn Treebank, com anotações no formato PALAVRA_TAG.

3. O treinamento

A "fase de treinamento", nesse contexto, é composta pela geração de um dataset que agrega informações específicas sobre cada palavra ou

série de palavras no corpus. Esses datasets são gerados pelos programas na pasta `extractors`, sendo todos baseados nas seções `Secs0-18 - training` do Penn Treebank.

O dataset gerado pelo treinamento do modelo de trigrama, por exemplo, tem o formato

|Penultima Palavra|Última Palavra|Palavra|Tag|.

4. A inferência

A fase de inferência de tags é executada por um módulo de inferência chamado *driver*. O driver utiliza o dataset gerado na fase de treinamento e os conjuntos de desenvolvimento e teste do corpus para prever a qual tag cada palavra pertence. Ao fim da fase de inferência, é gerado um dataset com as colunas |Palavra|Tag Real|Tag Inferida| para facilitar a análise futura dos resultados obtidos.

5. Análise exploratória

Descartamos, para esta e futuras análises, as tags e os tokens relativos aos seguintes elementos de pontuação:

```
[',', ' ', '\', '"', ':', ';', '#', '%', '"', "'", "$", ":", "(", ")"]
```

| Dataset | Tokens | Palavras únicas | Tags únicas |
|---------|--------|-----------------|-------------|
| Train | 912344 | 42036 | 38 |
| Dev | 131768 | 14822 | 38 |
| Test | 129654 | 13710 | 38 |

Three bar charts comparing the frequency of the top 15 tags in Train, Dev, and Test datasets. The y-axis represents Frequency, and the x-axis represents the Tag. The tags are: NH, AK, Ane, DT, JJ, CD, RB, AD, Ab, CC, TO, VCC, VNN, and PHO. The Train dataset has the highest frequencies, followed by Dev, and then Test.

| Tag | Train Frequency | Dev Frequency | Test Frequency |
|-----|-----------------|---------------|----------------|
| NH | 125000 | 18000 | 17500 |
| AK | 95000 | 13500 | 13500 |
| Ane | 88000 | 13200 | 13200 |
| DT | 80000 | 11000 | 11000 |
| JJ | 60000 | 8000 | 8000 |
| CD | 58000 | 8000 | 8000 |
| RB | 35000 | 5200 | 5200 |
| AD | 30000 | 4800 | 4800 |
| Ab | 28000 | 4200 | 4200 |
| CC | 25000 | 3500 | 3500 |
| TO | 22000 | 3200 | 3200 |
| VCC | 21000 | 3000 | 3000 |
| VNN | 20000 | 2800 | 2800 |
| PHO | 18000 | 2500 | 2500 |

Three bar charts comparing the frequency of the top 15 words in Train, Dev, and Test datasets. The y-axis represents Frequency, and the x-axis represents Word. The words are: the, is, a, of, to, it, and, that, on, for, he, she, we, you, and.

| Word | Train Frequency | Dev Frequency | Test Frequency |
|------|-----------------|---------------|----------------|
| the | 39500 | 5800 | 5800 |
| is | 22000 | 3100 | 3100 |
| a | 21500 | 3000 | 2800 |
| of | 18500 | 2700 | 2500 |
| to | 15500 | 2300 | 2200 |
| it | 14500 | 2200 | 2100 |
| and | 8500 | 1300 | 1400 |
| that | 7500 | 1100 | 1100 |
| on | 7500 | 1000 | 1000 |
| for | 6500 | 900 | 900 |
| he | 6500 | 800 | 800 |
| she | 5500 | 700 | 700 |
| we | 4500 | 600 | 600 |
| you | 4500 | 500 | 500 |
| and | 4500 | 400 | 400 |

- Todos os treinamentos foram realizados com o arquivo Secs0-18 - training e os testes de desenvolvimento com o arquivo Secs19-21 - development .
- Todos os experimentos foram realizados com o modelo de unigrama

6.1 Palavra desconhecida

A modelagem de palavra desconhecida é importante para que o programa possa ser mais generalista e não se limite apenas às palavras no corpus de treino.

6.1.1 Removendo palavras com uma aparição

Com esse formato, todas as palavras do dataset de treino com apenas uma aparição foram alteradas para *unk-word*, definindo sua tag como a mais comum entre as palavras selecionadas. A tag definida foi NNP, dentre 22173 palavras selecionadas.

Utilizando este modelo no dataset de desenvolvimento, foram obtidos os seguintes resultados:

- Accuracy: 0.9056
- Precision: 0.8168
- Recall: 0.7792
- F1-Score: 0.7872

6.1.2 Mantendo as palavras com uma aparição

Aplicando este método, as palavras com uma aparição não são alteradas para *unk-word*, mas suas tags são acumuladas para compor uma nova entrada para a *unk-word*.

Esse modelo trouxe os seguintes resultados:

- Accuracy: 0.9143
- Precision: 0.8181
- Recall: 0.7867
- F1-Score: 0.7922

Essa abordagem resultou em menos palavras sendo classificadas incorretamente como NNP e melhorando as métricas.

6.2 Normalização

6.2.1 Modelo 1

Esse modelo mantém o experimento da seção 6.1.2 e trata palavras com e sem letras maiúsculas como palavras diferentes, por exemplo: mesmo que a palavra "factory" esteja no dataset de treino, caso nos testes seja encontrada a palavra "Factory", ela será modelada como "unk-word".

- Accuracy: 0.9143
- Precision: 0.8181
- Recall: 0.7867
- F1-Score: 0.7922

6.2.2 Modelo 2

Esta segunda abordagem, antes de recorrer à palavra desconhecida, busca a palavra equivalente em lowercase e, em seguida, busca a palavra em forma capitalizada. Essa estratégia tem o objetivo de diminuir o número de vezes que recorremos à *unk-word* e melhorar as métricas do programa.

- Accuracy: 0.9136
- Precision: 0.8305
- Recall: 0.7897
- F1-Score: 0.7954

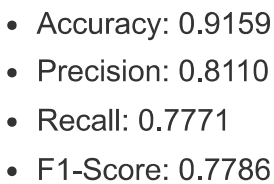
As métricas mostram um melhor equilíbrio entre as classes e uma maior facilidade em acertar classes menos comuns, mesmo que com erro maior no geral.

Teste Final

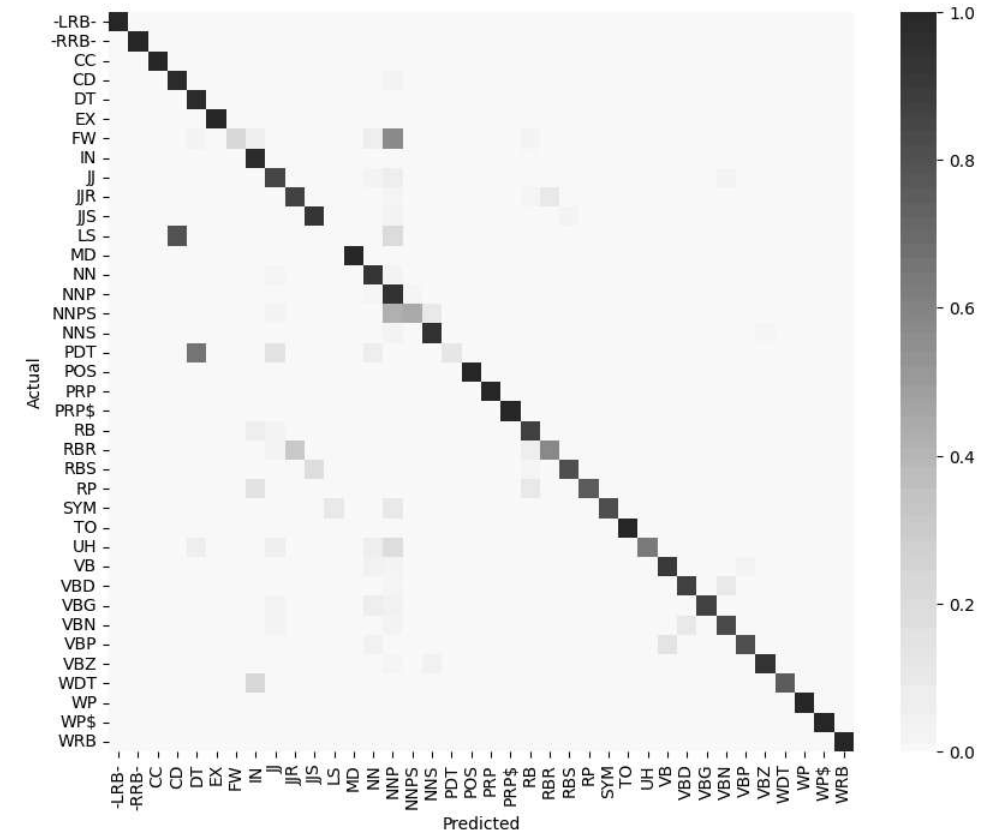
- Os resultados finais reportados são de inferências sobre o conjunto Secs22-24 - testing
- Para o unigrama, foi utilizada a abordagem descrita na seção 6.2.2 para normalização e 6.1.2 para tratamento da palavra desconhecida.
- Para o smoothing de Bigramas e Trigramas, foi utilizada a técnica de Backoff.

Unigrama

Esse modelo leva em conta apenas a tag mais comum para cada palavra no dataset de treino.



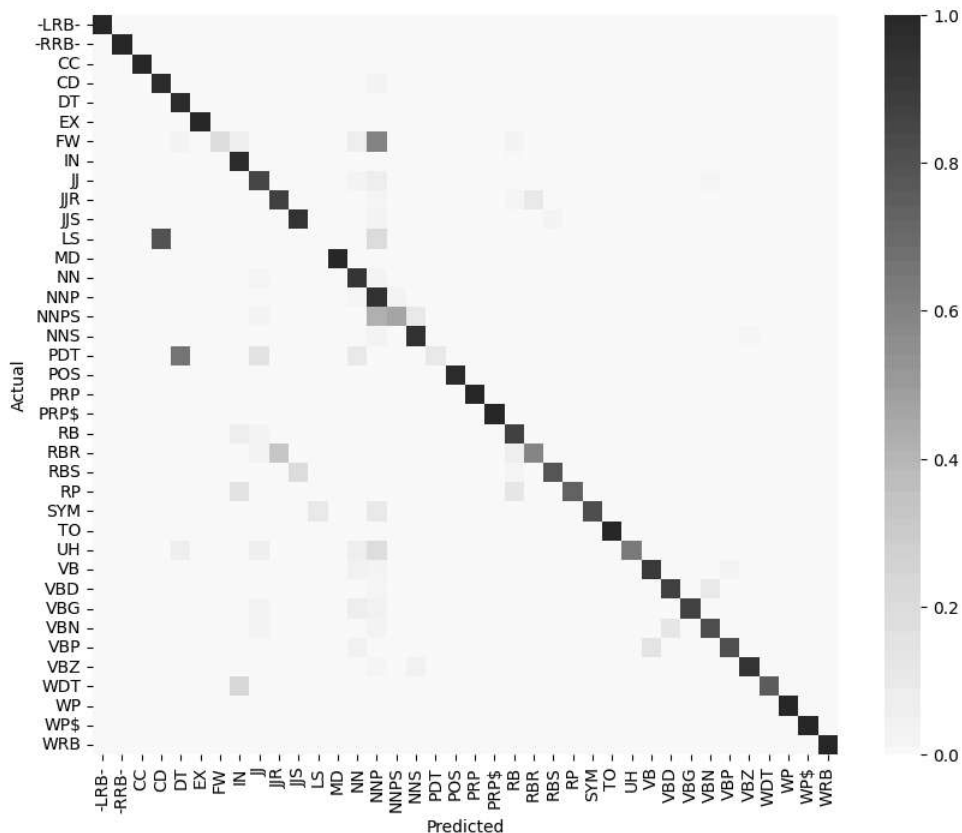
Esse modelo é baseado na tag mais comum para a segunda palavra de cada tupla (palavra1, palavra2)



- Accuracy: 0.9357
- Precision: 0.8572
- Recall: 0.8327
- F1-Score: 0.8408

Trigrama com palavras

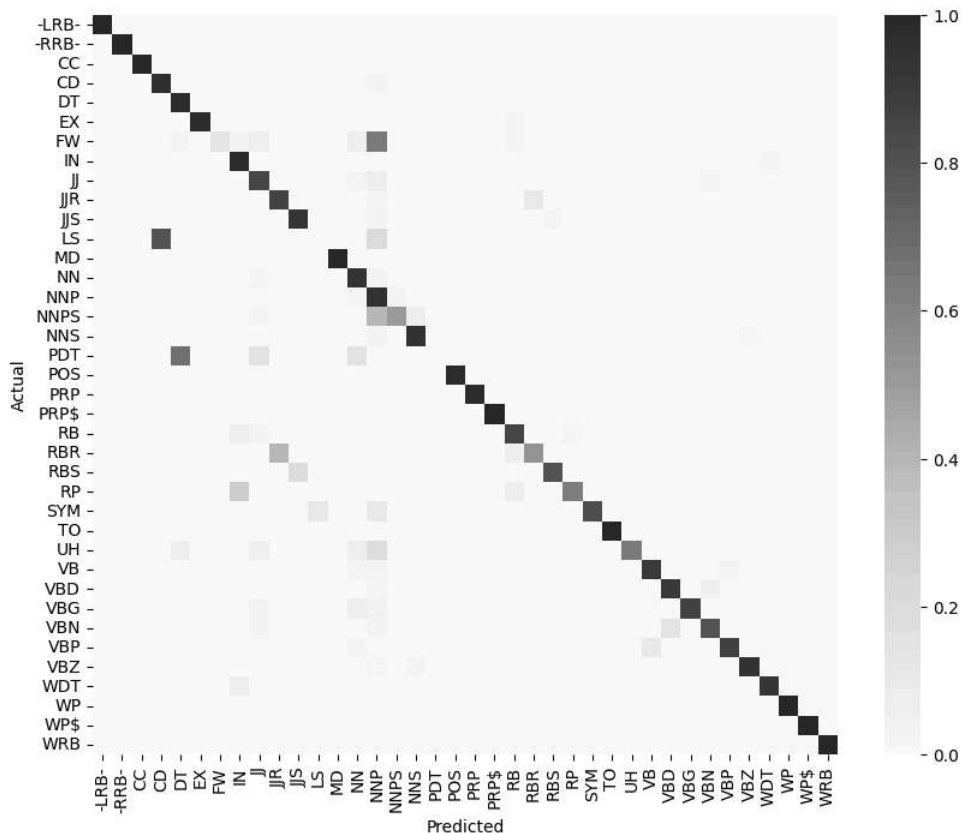
Esse modelo é baseado na tag mais comum para a terceira palavra de cada tupla (palavra1, palavra2, palavra3)



- Accuracy: 0.9354
- Precision: 0.8532
- Recall: 0.8308
- F1-Score: 0.8383

Bigrama com tags

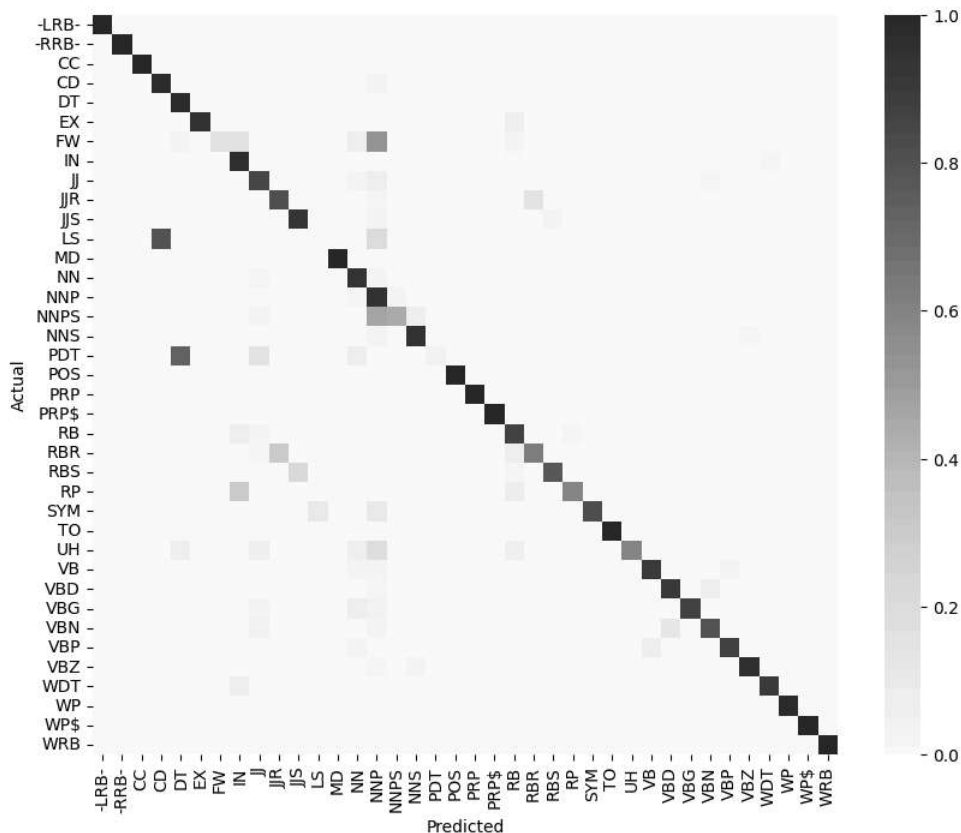
Esse modelo é baseado na tag mais comum para a segunda palavra baseado na tag da palavra anterior e na palavra atual. (tag_anterior, palavra)



- Accuracy: 0.9370
- Precision: 0.8422
- Recall: 0.8291
- F1-Score: 0.8309

Trigrama com tags

Esse modelo é baseado na tag mais comum para a terceira palavra baseado nas tags das palavras anteriores e na palavra atual.
(tag_anterior1, tag_anterior2, palavra)



- Accuracy: 0.9367
- Precision: 0.8435
- Recall: 0.8262
- F1-Score: 0.8308

Conclusão

Após análise dos resultados, foi possível concluir que os métodos de Bigrama e Trigrama baseados em palavras foram os que apresentaram as melhores métricas. Apesar dos bons resultados, foram observados

alguns pontos de erro comuns a todos os métodos, sendo os principais:

- Marcadores de Lista denominados como Números Cardinais (LS ==> CD), uma vez que os dois são representados por valores numéricos.
- Pre Determiners marcados como Determiners (PDT ==> DT)
- Palavras estrangeiras marcadas como nome próprio (FW ==> NNP), já que o marcador de palavra desconhecida ficou definido como nome próprio.
- Nomes próprios no plural marcados como nome próprio no singular (NNPS ==> NNP), já que o marcador de palavra desconhecida ficou definido como nome próprio no singular.