Comparação entre algoritmos de Stemmer: desempenho em aplicação prática e método de Paice

Douglas Castro
Programa de Engenharia de Sistemas e Computação Universidade Federal do Rio de Janeiro
Email: douglascastro@cos.ufrj.br

João Paulo Monteiro
Programa de Engenharia de Sistemas e Computação Universidade Federal do Rio de Janeiro
Email:jmonteiro@cos.ufrj.br

Vitor Brandão Sabbagh
Programa de Engenharia de Sistemas e Computação Universidade Federal do Rio de Janeiro
Email: vitorsab@gmail.com

***Resumo*—Algoritmos de Stemmer são amplamente utilizados em processamento de linguagem natural, com o objetivo de conflacionar variantes da mesma palavra. Uma forma de medir o desempenho desses algoritmos foi proposta por Paice, por meio do método de mesmo nome. Neste artigo é apresentada a execução do método em diferentes algoritmos de Stemmer e obtidas medidas de desempenho dos mesmos: *overstemming-index* e *understemming-index*. Em um segundo momento, os mesmos algoritmos foram aplicados em um problema de classificação e tem sua eficácia medida em uma aplicação prática. Os algoritmos avaliados apresentam acurácias similares no problema de classificação enquanto, no método de Paice, o RSLP apresenta ERRT menor, sugerindo um melhor desempenho. Conclui-se que, para os dois algoritmos analisados, não há uma relação explícita entre o indicador de Paice e o desempenho em uma aplicação prática dos diferentes algoritmos.**

1. Introdução

Algoritmos de *Stemming*, também referidos como algoritmos de Stemmer ou, simplesmente, *Stemmers*, são umas das principais ferramentas utilizadas em recuperação da informação, com o objetivo de evitar que variações das palavras buscadas sejam erroneamente descartadas na varredura dos documentos. Stemmers executam uma conflação nas variantes da mesma palavra como (secreto, secretamente, secreta) e (pular, pulo, pulando).

Nas últimas décadas, foram desenvolvidos diversos algoritmos de stemming em diferentes idiomas. Para os idiomas ocidentais, os algoritmos são fortemente baseados em remoção de sufixos, seguindo regras variadas. Alguns deste algoritmos apresentam melhor desempenho de forma geral, outros desempenham melhor em contextos específicos.

1.1. Problemas

De acordo com Flores [2], uma forma de medir a qualidade de um algoritmo de *stemming* é avaliando a eficácia do algoritmo em mapear diferente formas das palavras em um mesmo *stem*. Outra forma é medindo o desempenho do algoritmo em um aplicação prática. Flores [2] avalia, por meio do método de Paice, a eficácia de diversos algoritmos de *stemming* em Português, bem como o desempenho dos mesmos algoritmos em um problema de Recuperação de Informações. Seu trabalho mostra que existe uma relação entre ambas as medidas, porém não tão forte quanto se poderia esperar.

Neste trabalho, foi escolhida uma abordagem parecida com o que foi proposto por Flores [2], foi desenvolvido um comparativo de alguns algoritmos de stemmer para a língua portuguesa e a aplicação destes em um problema de classificação real.

O objetivo principal, é averiguar se os resultados obtidos na avaliação do método proposto por Paice [5] implicarão na indicação do melhor stemmer para o problema de classificação proposto, assim, aumentando a sua taxa de acerto. O artigo é dividido em duas partes, a primeira parte faz um comparativo entre os métodos Snowball e RSLP utilizando a abordagem proposta por Paice. Estes algoritmos foram escolhidos pois, no trabalho proposto por Flores [2] foram alcançados ótimos resultados com os algoritmos baseados em RSLP, e foi decidido então a comparação deste com um outro algoritmo muito utilizado na literatura chamado Snowball. A segunda parte visa resolver um problema de classificação amplamente difundido chamado ”análise de sentimentos”, que visa utilizar de aprendizado supervisionado e técnicas de PLN (Processamento de Linguagem Natural) para definir se um dado texto pode ser classificado como Positivo,Negativo ou Netro. Os algoritmos de stemmer (Snowball e RSLP) serão utilizados como etapa de pré-processameto, assim, será avaliada a taxa de acerto e revelando assim qual configuração de classificação é o melhor.

2. Revisão da Literatura

2.1. *RSLP*

O algoritmo de *stemming RSLP* é um algoritmo de remoção de sufixos para o portugues baseado em regras

onde cada passo contém um conjunto de regras que são examinadas em sequencia onde apenas uma regra pode ser aplicada em um dado passo. Um ponto importante é que o sufixo mais longo é sempre removido primeiro devido à ordem das regras dentro de uma etapa. A figura 1 ilustra como estas regras são compostas. [1]

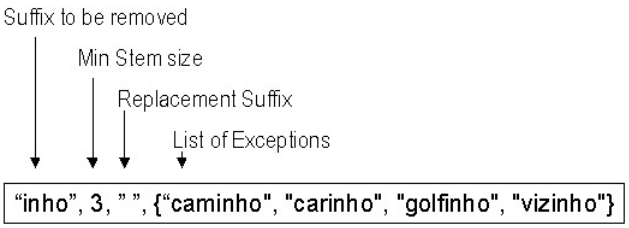

Figure 1. Regras do *RSLP*

2.2. *Porter*

Flores [2] descreve em seu trabalho o algoritmo de Porter.

O algoritmo de Porter se baseia em uma série de regras condicionais que são aplicadas em sequência. Uma regra é definida da forma mostrada na figura 2.

(condição) sufixo1 -> sufixo2

Figure 2. Regras do *PORTER*

Na regra acima, condição é uma condição a ser testada para a aplicação da regra, sufixo1 é o sufixo a ser procurado e removida da palavra e sufixo2 é o sufixo a ser adicionado na palavra após a remoção do sufixo1. Desta forma, a regra é aplicada se o sufixo1 for encontrado na palavra e as condições de condição forem satisfeitas, caso em que o sufixo1 será removido da palavra e substituído por sufixo2, se este estiver definido na regra.

Foram definidos 5 passos de regras no algoritmo, quais sejam:

- *Passo 1: remoção de sufixos comuns (por exemplo: eza, ismos, ável, ível, oso, aço es, mente, ância...)*
- *Passo 2: remoção de sufixos verbais, caso a palavra não tenha sido alterada pelo passo 1 (por exemplo: ada, ida, aria, ará, ava, isse, iriam, aram, endo, indo, arão, íreis, êssemos...)*
- *Passo 3: remoção do sufixo “i”, se precedido de “c” e se a palavra foi alterada pelos passos 1 ou 2.*
- *Passo 4: remoção de sufixos residuais (os, a, i, o, á, í, ó) se nenhum dos passos anteriores alterou a palavra;*
- *Passo 5: remoção dos sufixos “e”, “é” e “ê”, tratamento do cedilha e das sílabas “gue”, “gué” e “guê”.*

1. http://www.inf.ufrgs.br/ viviane/rslp/index.htm

Além disso, antes das aplicações das regras, as palavras que possuem “ã” e “õ” são substituídas por “a ” e “o ”. Ao final da aplicação das regras, elas retornam para “ã” e “õ” para formar o stem resultante.

2.3. Método de Paice

Paice [5] aponta dois problemas principais no uso de *stemmers*: ocorrência de pares de palavras etimologicamente relacionadas porém com significado diferente, por exemplo ”autor” e ”autoritário”. O segundo problema apontado por Paice [5] diz respeito à ocorrência de sufixos irregulares, que fogem aos padrões de afixos do idioma.

Paice aponta, neste contexto, dois tipos de erros principais nos Stemmers: *overstemming* e *understemming*.

2.4. TF - IDF

Uma dos métodos mais populares para se obter uma medida de similaridade entre uma consulta e uma coleção de documentos é o TF - IDF, conforme descreve Lee em [6]. Este método envolve a utilização de termos-chaves, que são extraídos dos documentos, em seguida é aplicado um algoritmo para ranqueamento baseado em modelo vetorial.

O peso de um termo em um documento pode ser determinado de diversas formas. Uma forma usual é o método *tf x idf*, onde o peso é determinado por dois fatores: frequência em que ocorre em um documento e a frequência em que ocorre na coleção de documentos.

2.5. Análise de sentimentos

”Análise de sentimentos” refere-se a uma serie de metodos, técnicas e ferramentas que se dedicam a extrair informações subjetivas, como opinião ou atitude inseridos na linguagem. Tradicionalmente análise de sentimentos é sobre opinião polarizada (positiva, negativa ou neutra) isto é, quando um determinado indivíduo tem opinião positiva, negativa ou neutra sobre um determinado assunto. O objeto da análise de sentimento normalmente é um produto ou serviço cuja revisão foi tornada pública na Internet. Isso pode explicar porque a análise de sentimento e a mineração de opinião são frequentemente usadas como sinônimos, embora pensemos que é mais correto ver os sentimentos como opiniões carregadas de emoção.

3. Metodologia

Para resolver a tarefa de comparação entre algoritmos de *stemming*, utilizamos duas abordagens distintas. Na primeira abordagem decidimos testar o uso desses *Stemmers* em um problema de classificação, já na segunda abordagem utilizamos o método de Paice [5] que foi detalhado na seção 2.3.

3.1. Problema de classificação

Nesta seção explicaremos em detalhes a abordagem de comparação de *Stemmers* através de um problema de classificação de texto.

3.1.1. *Dataset*. O dataset utilizado nos experimentos foi o *Portuguese Tweets for Sentiment Analysis* [4] o qual contém tweets que foram coletados entre as datas 01/08/2018 e 20/10/2018 e contém tweets divididos em duas classes que representam os sentimentos positivos e negativos.

3.1.2. Modelo utilizado. O algoritmo de aprendizado de máquina utilizado nos experimentos foi o *Naive Bayes* que é um classificador probabilístico que se fundamenta na aplicação Teorema de Bayes, o qual foi criado por Thomas Bayes (1701 - 1761) 3.1.2.

$$\boxed{\text{Teorema de Bayes}}\ P(A/B) = \frac{P(A)*P(B/A)}{P(B)}$$

3.1.3. Experimento. Para realização da comparação dos algoritmos de *stemming*, utilizamos a vetorização *TF-IDF* em conjunto com a remoção de stopwords dos tweets. Além disso, os experimentos foram realizados utilizando a validação cruzada *k-fold* que consiste no particionamento do conjunto de dados em k subconjuntos mutuamente exclusivos do mesmo tamanho e, a partir daí, um subconjunto é utilizado para teste e os $k-1$ restantes são utilizados para o treinamento do modelo. Para este experimento foi utilizado $k = 10$ que implica em que 10% do dataset era utilizado para teste em cada etapa da validação cruzada. Seguindo esta metodologia foram obtidos os resultados apresentados na tabela 1.

TABLE 1. ACURÁCIA DOS ALGORITMOS DE STEMMER

·	Acurácia	Desvio Padrão
RSLP	0.6855	0.0188
Snowball	0.690	0.0176

Como podemos perceber com base na tabela 1 o desempenho do algoritmo de *stemming Snowball* se saiu ligeiramente superior *RSLP* obtendo uma acurácia maior com um desvio padrão menor. Como a única variação de técnicas de pré-processamento de texto aplicada sobre a base de dados foram os algoritmos de *stemming*, esse desempenho superior se dá apenas pelo uso destes algorimtmos e podemos concluir que o algoritmo *Snowball* foi superior ao *RSLP* na tarefa de análise de sentimentos para esta base de dados.

3.2. Método de Paice

Para a aplicação do Método de Paice, foi feita uma implementação em Python 3.0. Os métodos RSLP e Snowball para a lingua portuguesa foram adquiridos utilizando a biblioteca NLTK também feita em python. O experimento utilizou também uma base de palavras, separadas em grupos dado seu sentido funcional. A mesma está disponível em Tartarus [3]

3.2.1. Resultados. Após rodar o experimento, é fácil notar que o RSLP tem o menor valor de ERRT, e o mesmo ganha nos índices de understemming(UI) e overstemming(OI). Portanto, nesta fase do experimento, dadas as métricas definidas por Paice [5] podemos afirmar que o RSLP é o melhor Stemmer.

TABLE 2. MÉTRICAS DE DESEMPENHO DOS ALGORITMOS DE STEMMER

·	Snowball	RSLP
GUMT	215,5	48,5
GDMT	6976	6976
GWMT	223,5	54
GDNT	2091200	2091200
UI	0,03089	0,00695
OI	0,00010	0,00025
ERRT	0,11828	0,02740

4. Conclusão

A avaliação individual das partes deste experimento, nos leva a concluir que uma Stemização eficiente pouco influencia em tarefas de classificação, como foi demonstrado na seção[3]. Apesar de o Stemmer RSLP ter alcançado níveis de ERRT bem inferiores ao do seu concorrente Snowball, o resultado geral aplicado ao problema de análise de sentimentos teve pouco ganho dado o setup utilizando estes Stemmers. Concluímos então que os problemas não são estritamente relacionados, e que se o objetivo do analista é melhorar a taxa de acertos para um problema de classificação deve-se explorar outras vertentes como o próprio algoritmo de classificação a ser utilizado e outras técnicas de pré-processamento de texto.

Referências

[1] Julie Beth Lovins, *Development of a Stemming Algorithm* Cambridge, Massachusetts: Mechanical Translation and Computational Linguistics, 1968.

[2] Felipe Nunes Flores, *Avaliando o Impacto da Qualidade de um Algoritmo de Stemming na Recuperação de Informações* Rio Grande do Sul, Brasil: Universidade Federal do Rio Grande do Sul, 2009.

[3] Knuth: Computers and Typesetting, `http://snowball.tartarus.org/algorithms/portuguese/voc.txt`

[4] Portuguese Tweets for Sentiment Analysis `https://www.kaggle.com/augustop/portuguese-tweets-for-senti`

[5] Chris D. Paice, *An Evaluation Method for Stemming Algorithms*. Bailrigg, Lancaster: Department of Computing, Lancaster University, 1994.

[6] Lee, D. L., Chuang, H., and Seamons, K. *Document Ranking and the Vector-Space Model* IEEE Software, 1997.