Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 8, n. 1, 2021.

Trabalho apresentado no XL CNMAC, Evento Virtual - Co-organizado pela Universidade do Mato Grosso do Sul (UFMS).

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Avaliação de abordagens para classificação automática de documentos jurídicos: um estudo comparativo aplicado a petições do Tribunal de Justiça do Estado de Alagoas

José Augusto Silva[1]
Instituto de Computação, UFAL, Maceió, AL
Valério Nogueira Jr[2]
Instituto de Computação, UFAL, Maceió, AL
Hugo Oliveira[3]
Instituto de Computação, UFAL, Maceió, AL
Adriano Barbosa[4]
Faculdade de Ciências Exatas e Tecnologia, UFGD, Dourados, MS
Thales Vieira[5]
Instituto de Computação, UFAL, Maceió, AL
Krerley Oliveira[6]
Instituto de Matemática, UFAL, Maceió, AL

Resumo. Classificação de documentos é um problema estudado na literatura há vários anos com diversas soluções robustas já disponíveis. Porém, devido a peculiaridades de cada língua e da natureza dos documentos, faz-se necessário investigar a abordagem que melhor se adeque a um problema específico. Neste trabalho, realizamos um estudo comparativo de diversas metodologias usadas para classificação de documentos, com foco no problema de classificar diversos tipos de documentos jurídicos escritos em língua portuguesa. Mais especificamente, analisamos o desempenho de cinco abordagens para realizar a tarefa de reconhecer 11 tipos de petições intermediárias de uma vara de execução fiscal do Tribunal de Justiça do Estado de Alagoas. Em nossos experimentos, a abordagem baseada em representação vetorial TF-IDF com classificador SVM (TFIDF-SVM) destacou-se pela alta acurácia e baixo tempo de treinamento, além de gerar modelos caixa-branca.
Palavras-chave. processamento de linguagem natural, classificação de documentos, justiça.

1 Introdução

Classificação de documentos, ou classificação de texto, é um problema de extrema importância que vem sendo estudado na literatura há muitos anos por pesquisadores das áreas de Banco de Dados, Mineração de Dados, Recuperação de Informação, Processamento de Linguagem Natural, dentre outros [1]. Atualmente, podemos observar aplicações de classificação de documentos em uma grande diversidade de áreas do conhecimento, incluindo: análise de redes sociais, finanças, filtragem de SPAM, filtragem de *fake news*, classificação de textos médicos, e, de nosso interesse, classificação de documentos jurídicos [2].

Apesar de existirem na literatura diversos métodos para classificação de documentos, faz-se necessário avaliar a robustez desses para aplicações específicas. Textos jurídicos, em geral, possuem um vocabulário e uma estrutura peculiar, além de alto grau de estrutura própria, o que dificulta a compreensão mesmo por indivíduos falantes da língua que sejam leigos na área jurídica. Inclusive, esta dificuldade tem motivado pesquisas por metodologias para análise destes tipos de

[1]jass2@ic.ufal.br
[2]vnjr@ic.ufal.br
[3]htmo@ic.ufal.br
[4]adrianobarbosa@ufgd.edu.br
[5]thales@ic.ufal.br
[6]krerley@im.ufal.br

textos [3]. Além disso, características inerentes de cada língua inviabilizam que o conhecimento adquirido através de pesquisas em Processamento de Linguagem Natural para uma determinada língua seja aproveitado em outras pesquisas. Por exemplo, línguas morfologicamente ricas, como italiano e português, podem se aproveitar de informação morfológica das palavras para inferir classes gramaticais [11, 17], e isso não pode ser transferido para outras línguas.

Neste trabalho, investigamos diversas abordagens para classificação multi-classe de documentos, com o objetivo de realizar classificação de petições jurídicas escritas em português. O desenvolvimento de um classificador multi-classe automático de petições tem o potencial de dar celeridade processual a unidades da justiça, substituindo o trabalho repetitivo que em geral é realizado de forma manual por um servidor. Mais especificamente, são investigadas neste trabalho a robustez e performance computacional das abordagens comparadas, quando treinadas e avaliadas em uma base de dados composta por 1.019 petições judiciais, rotuladas de acordo com 11 classes distintas. Esta base foi coletada especialmente para a realização desta pesquisa na 15^a vara de execuções fiscais de Maceió do Tribunal de Justiça do Estado de Alagoas (TJ-AL). Mais especificamente, tratamos de um problema de aprendizado supervisionado, buscando-se treinar bons classificadores multi-classe de documentos jurídicos.

É interessante destacar que, em uma entrevista realizada no início deste trabalho com os servidores desta vara, que contava naquele momento com mais de 60.000 processos em andamento, o trabalho de classificação manual destas petições foi apontado como um dos principais gargalos para aumentar a celeridade dos processos. Consequentemente, o desenvolvimento de ferramentas inteligentes robustas tem o potencial de dar maior celeridade no fluxo dos processos.

2 Abordagens comparadas para classificação de documentos

Nesta seção descrevemos as técnicas usadas em cinco abordagens avaliadas neste trabalho. Três abordagens são baseadas na combinação de vetores de características com classificadores distintos, e duas são baseadas em diferentes arquiteturas de redes neurais profundas *end-to-end*.

Representação vetorial TF-IDF. A *Term Frequency-Inverse Document Frequency* (TF-IDF, [13]) é uma medida estatística baseada em Teoria da Informação usada para representar quão importante uma palavra é em um documento, levando-se em consideração uma coleção de documentos [12]. TF é a abreviação para *Term Frequency* e IDF significa *Inverse Document Frequency*. Uma das formas de obter a frequência TF de uma palavra w é contar todas as ocorrências de w em um documento d, denotada por $f_{w,d}$, e dividir pelo número total de ocorrências de todas as palavras do documento:

$$\mathrm{TF}(w,d) = \frac{f_{w,d}}{\sum_{w' \in d} f_{w',d}}.$$

Por outro lado, a frequência inversa do documento descreve a relação da ocorrência geral de uma palavra w na coleção de documentos D, sendo definida por

$$\mathrm{IDF}(w,D) = \log\left(\frac{|D|}{|\{d \in D\colon w \in d\}|}\right).$$

Na formulação do IDF, a razão faz com que palavras raras sejam mais evidenciadas em relação a palavras comuns, enquanto que a escala logarítmica é aplicada para que a importância dada a cada palavra seja suavizada, evitando discrepâncias muito grandes.

Finalmente, multiplicando o valor TF pelo valor IDF, obtemos o valor TF-IDF de uma palavra w, considerando um determinado documento d e uma coleção de documentos D:

$$\text{TF-IDF}(w,d,D) = \mathrm{TF}(\mathrm{w,d}) \cdot \mathrm{IDF}(w,D).$$

DOI: 10.5540/03.2021.008.01.0376

Calculando-se o valor TF-IDF para cada palavra de um dicionário contendo m palavras, é possível gerar um vetor $v_D(d) \in \mathbb{R}^m$ para representar cada documento d da coleção D, onde cada entrada de $v_D(d)$ representa o valor TF-IDF para uma determinada palavra do dicionário.

Representação vetorial Doc2Vec. A representação Doc2Vec [8] é obtida através de um treinamento não supervisionado usando redes neurais. Assim como a TF-IDF, ela tem o objetivo de representar qualquer documento através de um vetor de dimensão fixa. O treinamento desta representação é realizado de forma similar à bem conhecida representação vetorial de palavras *Word2Vec*. Para mais detalhes, ver [8].

Classificadores SVM. Máquinas de Vetores de Suporte (SVM, [14]) são uma família de classificadores desenvolvidos para realizar aprendizagem supervisionada. Suas variações permitem a solução de problemas de classificação e regressão. Para problemas de classificação, é possível adotá-lo em dados linearmente ou não-linearmente separáveis, e também para problemas multi-classe. Através de um processo de otimização, esses métodos calculam um hiperplano, ou conjunto de hiperplanos, cujos parâmetros serão utilizados no modelo de classificação (ou regressão). De modo geral, um classificador SVM binário (usado para classificação binária) não-linear tem a forma

$$\hat{f}(x) = \sum_{x_i \in \mathrm{SV}} y_i \alpha_i K(x_i, x), \tag{1}$$

onde K é uma função kernel; y_i é o rótulo do i-ésimo exemplo; SV é um conjunto de vetores de suporte; e os coeficientes α_i são aprendidos num processo de otimização. Adotaremos neste estudo o SVM linear cujo kernel é o produto escalar convencional, devido a seu conhecido bom desempenho em problemas envolvendo dados textuais [6]. Além disso, o uso do SVM linear possibilita a criação de modelos de classificação *caixa-branca*, que são capazes de explicitar suas regras de decisão, conforme mostraremos em nossos experimentos. Finalmente, para tratar o problema de classificação multi-classe, adotaremos a abordagem um-contra-todos [9].

Classificadores Naïve Bayes. Os classificadores Naïve Bayes são adequados para problemas nos quais é válida a hipótese de independência das variáveis entre si. Consequentemente, um modelo desta família de classificadores, baseado na regra de Bayes, pode ser representado como

$$\hat{y} = \underset{k \in \{1,\ldots,m\}}{\arg\max}\ p(C_k) \prod_{i=1}^{n} p(x_i | C_k),$$

onde cada C_k é o rótulo de uma das classes; cada x_i é uma das m variáveis (ou características); $\hat{y}$ é o rótulo da classe atribuída pelo classificador; e p é uma distribuição de probabilidades. Em problemas de classificação de documentos, bons resultados geralmente são alcançados com distribuições de probabilidade do tipo multinomial [4], as quais adotaremos neste estudo.

Redes Neurais Profundas. Duas arquiteturas de redes neurais recorrentes (Emb-BiLSTM e Emb-CNN-BiLSTM) foram avaliadas para classificação de documentos, baseadas em trabalhos anteriores da literatura [7]. Em ambos os casos, o documento é representado como uma sequência de *tokens* (ou palavras), que alimenta a rede uma palavra por vez, e a classificação do documento é retornada após o processamento da última palavra. As arquiteturas experimentadas são exibidas na Figura 1. Em ambas, cada token é inicialmente codificado em um vetor *one-hot* de dimensão m. Na primeira camada, este vetor é projetado em uma dimensão mais baixa através de uma camada de *embedding*. Na primeira arquitetura (Emb-BiLSTM), o vetor resultante segue para uma camada bidirecional de células recorrentes do tipo *Long Short-Term Memory* (BiLSTM, [15]). Na segunda

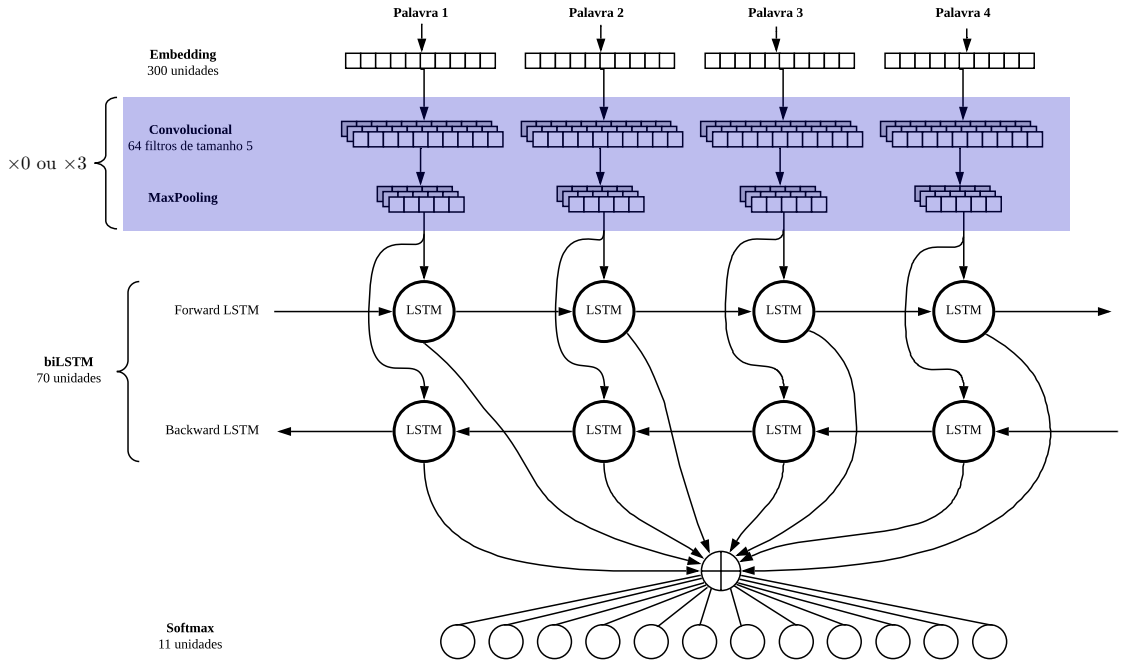


Figura 1: Arquiteturas de redes neurais experimentadas: na Emb-BiLSTM, as camadas na região azul não são usadas; na Emb-CNN-BiLSTM, estas são aplicadas três vezes antes das camadas BiLSTM.

arquitetura (Emb-CNN-BiLSTM), o vetor é enviado antes para 3 grupos em sequência de camadas convolucionais unidimensionais seguidas por camadas de agrupamento máximo (*MaxPooling*), antes de seguir para a camada BiLSTM. Finalmente, em ambas as arquiteturas, a saída da camada recorrente segue para classificação em uma camada com função de ativação *softmax*.

3 Experimentos

Cinco abordagens foram avaliadas nos experimentos realizados: representação TF-IDF com classificador SVM linear (TFIDF-SVM); representação TF-IDF com classificador Naïve Bayes (TFIDF-NB); representação Doc2Vec com classificador SVM linear (Doc2Vec-SVM); e as duas arquiteturas de redes neurais (Emb-BiLSTM e Emb-CNN-BiLSTM).

Base de dados. Com o auxílio de analistas da 15^a vara do TJ-AL, foram selecionadas as 11 classes, representando os tipos de petições mais frequentes na vara, para os experimentos realizados neste estudo. Em seguida, a coleta de exemplos de documentos de cada classe foi realizada pelos analistas na base de dados do TJ-AL, de forma aleatória. Ao final do processo, notou-se um desbalanceamento entre as classes, que posteriormente não se mostrou capaz de prejudicar o experimento. As classes e a quantidade de exemplos coletados podem ser vistos na Tabela 1.

Pré-processamento. Para tratar documentos escaneados, foi utilizada a ferramenta Tesseract OCR [16] para extração do texto dos documentos. Esta etapa foi avaliada empiricamente, resultando em textos brutos de boa qualidade. De posse do texto bruto, foi aplicado um pré-processamento para remover acentos, pontuações, espaços, linhas em branco, conversão para caixa baixa, palavras vazias (ou *stopwords*, incluindo preposições, artigos, etc), além de stemização para remover flexões das palavras.

Tabela 1: Quantidade de documentos por classe.

Classe	**Notação**	**# de documentos**
Suspensão CPC	c_{01}	143
Pedido de citação	c_{02}	138
Citação por edital por procurador	c_{03}	123
Extinção do feito	c_{04}	100
Suspensão CTM	c_{05}	100
Suspensão LEF	c_{06}	99
Penhora de bens	c_{07}	97
Redirecionamento de sócio	c_{08}	97
Pedido BACENJUD	c_{09}	49
Pedido de desistência	c_{10}	37
Renajud/Infojud	c_{11}	36
Total		**1019**

Visualização dos documentos. Para verificar se as representações vetoriais TF-IDF e Doc2Vec dos documentos formavam grupos (*clusters*) bem definidos de acordo com suas classes, foram realizadas projeções multidimensionais no plano dos conjuntos de vetores obtidos por cada uma destas representações, para possibilitar a visualização destes dados. Estas projeções foram calculadas aplicando-se a técnica de projeção *t-Distributed Stochastic Neighbor Embedding* (t-SNE, [10]). Os resultados visuais se revelaram promissores para ambas as representações, como é possível observar na Figura 2: a maioria dos documentos ficou bem agrupada em um ou mais *clusters*, de acordo com sua classe.

Acurácia e tempo de treinamento. Duas métricas foram usadas para avaliar as abordagens. Para comparar a robustez, foram realizados experimentos de validação cruzada de forma estratificada, onde 70% das amostras de cada classe foram aleatoriamente selecionadas para treino, e os 30% restantes para teste. As representações vetoriais foram treinadas a partir dos 70% dos dados de treino, e em seguida as representações do conjunto de teste foram calculadas pelo modelo obtido. As redes neurais foram treinadas com o otimizador Adam [5] da biblioteca Keras [7] usando somente uma CPU. Repetimos esse procedimento 100 vezes e calculamos a acurácia média para avaliar a robustez das abordagens, conforme exibido na Tabela 2, onde é possível também observar o tempo de treinamento.

[7]https://keras.io

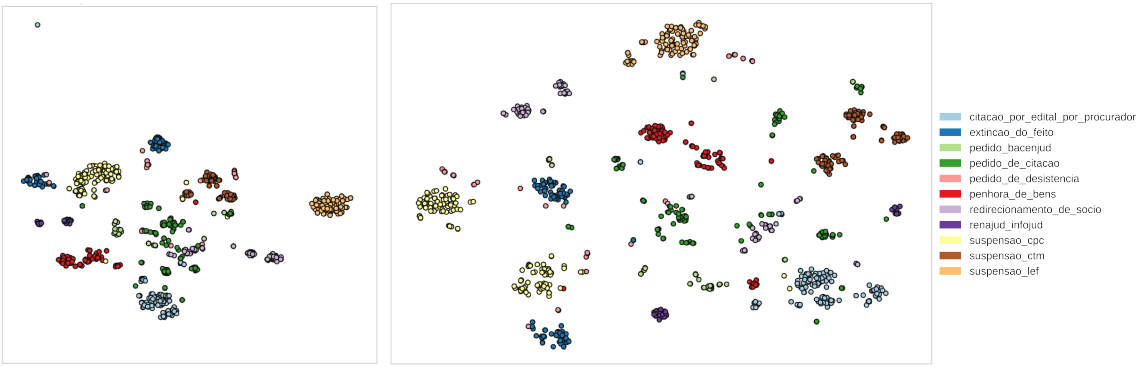


Figura 2: Visualização dos documentos: projeção usando t-SNE sobre as representações vetoriais TD-IDF (esquerda) e Doc2Vec (direita).

DOI: 10.5540/03.2021.008.01.0376

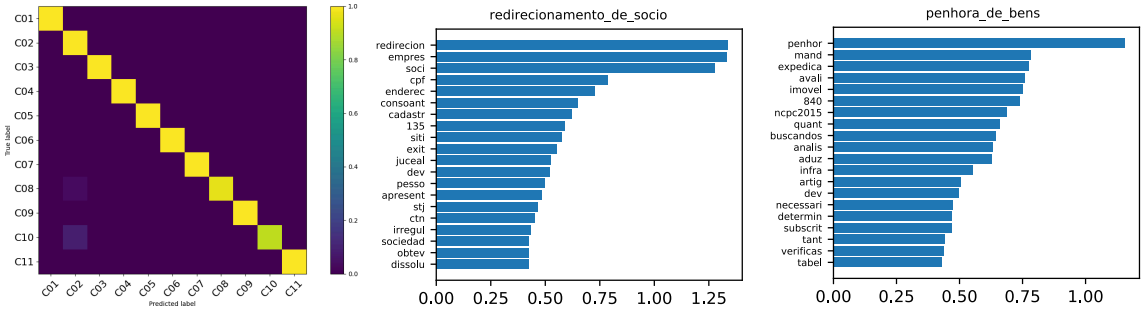
Tabela 2: Comparação entre as abordagens avaliadas.

Abordagem	**Tempo de treinamento (s)**	**Acurácia**	**Caixa-branca**
TFIDF-SVM	0,17814	99,35%	sim
Doc2Vec-SVM	10,58	99,02%	não
TFIDF-NB	0,11828	98,04%	não
Emb-CNN-BiLSTM	40,1	97,71%	não
Emb-BiLSTM	3,423	84,64%	não

Discussão. Considerando que a maioria das abordagens atingiu alta acurácia, adotamos como critério de desempate o tempo de treinamento e a geração de modelos caixa-branca, ou seja, modelos que permitam uma análise das regras geradas automaticamente pelos mesmos. Apesar da arquitetura de rede neural Emb-CNN-BiLSTM atingir resultados promissores, seu tempo de treinamento é ordens de magnitude maior do que abordagens vetoriais. Consequentemente, a abordagem preferencial foi a TFIDF-SVM, que é capaz de gerar modelos caixa-branca. Isto significa que, usando classificadores SVM linear, é possível analisar as palavras mais relevantes para que um determinado documento seja atribuído a uma determinada classe. Substituindo K na Equação (1) pelo produto escalar, é fácil mostrar que a importância da k-ésima característica (ou palavra do dicionário), é dada por $\sum_{x_i \in \mathrm{SV}} y_i \alpha_i x_{i_k}$, onde x_{i_k} é a k-ésima componente do vetor de suporte x_i. Dessa forma, podemos ordenar as palavras do dicionário por importância para cada classe, e plotar gráficos com as palavras mais importantes, como ilustra as Figuras 3b e 3c. No primeiro caso, o classificador considerou como mais relevantes para a classe de petições de redirecionamento de sócio, as palavras (stemizadas) *redirecion*, *empres* e *soci*, enquanto que para a classe de petições para penhora de bens, a palavra *penhor* teve grande importância. Estes exemplos claramente validam o modelo e mostram a vantagem dos modelos caixa-branca. Vale a pena mencionar também números como *135* e *840* que surgem como palavras relevantes em ambos os casos. Ao analisar alguns documentos, é fácil concluir que tratam-se de números de leis que sempre são mencionadas e que fundamentam as petições. Finalmente, a Figura 3a exibe a matriz de confusão para esta abordagem, que revela apenas problemas localizados, como alguns exemplos das classes c_{08} e c_{10} sendo incorretamente classificados como pertencentes à classe c_{02}.

4 Conclusão e trabalhos futuros

Realizamos um estudo comparativo de diversas metodologias usadas para classificação de documentos, com foco no problema de classificar documentos jurídicos escritos em língua portuguesa. Analisamos o desempenho de cinco metodologias para realizar a tarefa de reconhecer 11 tipos distintos de petições intermediárias de uma vara de execução fiscal do TJ-AL. A abordagem que consideramos mais adequada foi a TFIDF-SVM, baseada em representação vetorial TF-IDF com classificador SVM linear, devido a



(a) Matriz de confusão. (b) Palavras mais importantes de c_{08}. (c) Palavras mais importantes de c_{07}.

Figura 3: Análise da abordagem TFIDF-SVM.

sua alta acurácia combinada com baixo tempo de treinamento e geração de modelos caixa-branca. Como trabalho futuro, pretendemos continuar a análise comparativa utilizando outras ferramentas de OCR como o Textract [8] e aplicar os classificadores a outros problemas do setor jurídico.

Referências

[1] Aggarwal, C. C. and Zhai, C. (2012). A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

[2] Da Silva, N. C., Braz, F., de Campos, T., Gusmao, D., Chaves, F., Mendes, D., Bezerra, D., Ziegler, G., Horinouchi, L., Ferreira, M., et al. (2018). Document type classification for brazil's supreme court using a convolutional neural network. In *The tenth international conference on forensic computer science and cyber law-ICoFCS*, pages 7–11.

[3] Ferreira, A. C. and dos Santos Maculan, B. C. M. (2019). Metodologia para a análise de assunto de acórdãos no contexto do controle externo: proposta de um modelo de leitura técnica. *Em Questão*, 25(3):99–131.

[4] Kibriya, A. M., Frank, E., Pfahringer, B., and Holmes, G. (2004). Multinomial naive bayes for text categorization revisited. In *Australasian Joint Conf on Artificial Intelligence*, pages 488–499. Springer.

[5] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

[6] Kumar, M. A. and Gopal, M. (2010). An investigation on linear svm and its variants for text categorization. In *2010 2nd Int Conf on Machine Learning and Computing*, pages 27–31. IEEE.

[7] Lai, S., Xu, L., Liu, K., and Zhao, J. (2015). Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2267–2273. AAAI Press.

[8] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196.

[9] Liu, Y. and Zheng, Y. F. (2005). One-against-all multi-class svm classification using reliability measures. In *2005 IEEE Int Joint Conference on Neural Networks, 2005.*, volume 2, pages 849–854. IEEE.

[10] Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

[11] Marulli, F., Pota, M., and Esposito, M. (2018). A comparison of character and word embeddings in bidirectional lstms for pos tagging in italian. In *International Conference on Intelligent Interactive Multimedia Systems and Services*, pages 14–23. Springer.

[12] Rajaraman, A. and Ullman, J. D. (2011). *Data Mining*, page 1–17. Cambridge University Press.

[13] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

[14] Schölkopf, B., Smola, A. J., Bach, F., et al. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

[15] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

[16] Smith, R. (2007). An overview of the tesseract OCR engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, pages 629–633, Parana.

[17] Sousa, R. and Lopes, H. (2019). Portuguese pos tagging using blstm without handcrafted features. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 120–130.

[8] https://aws.amazon.com/textract/

