Avaliação de *parsers* na detecção de relações essenciais do modelo *Universal Dependencies* para o português

Luana Balador Belisário, Thiago Alexandre Salgueiro Pardo

Núcleo Interinstitucional de Linguística Computacional (NILC)
Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo
São Carlos, Brasil

Resumo. *Este artigo descreve o estudo do desempenho de dois parsers conhecidos para o português com base nas diretrizes do modelo internacional "Universal Dependencies". Visando mapear o estado da arte na área, os parsers foram avaliados com relação à detecção de algumas relações essenciais do modelo que indicam os argumentos principais dos verbos. Mostramos que o parser UDPipe se destaca entre os parsers avaliados, mas que ainda há muito a avançar na área.*

1. Introdução

O modelo *Universal Dependencies*[1] (UD) (Nivre, 2015; Nivre et al., 2020) é uma proposta internacional para anotação "universal" morfossintática e sintática (incluindo características morfológicas, classes gramaticais e dependências sintáticas) de sentenças em diferentes idiomas. A iniciativa já conta com mais de 300 contribuidores, produzindo quase 200 *treebanks* anotados com as diretivas definidas do modelo para mais de 100 idiomas. A Figura 1 ilustra a anotação de uma sentença em português (reproduzida de Rademaker et al., 2017, p. 200). Pode-se ver a sentença original com suas palavras conectadas por relações de dependência sintática acima, assim como os lemas e as etiquetas morfossintáticas abaixo.

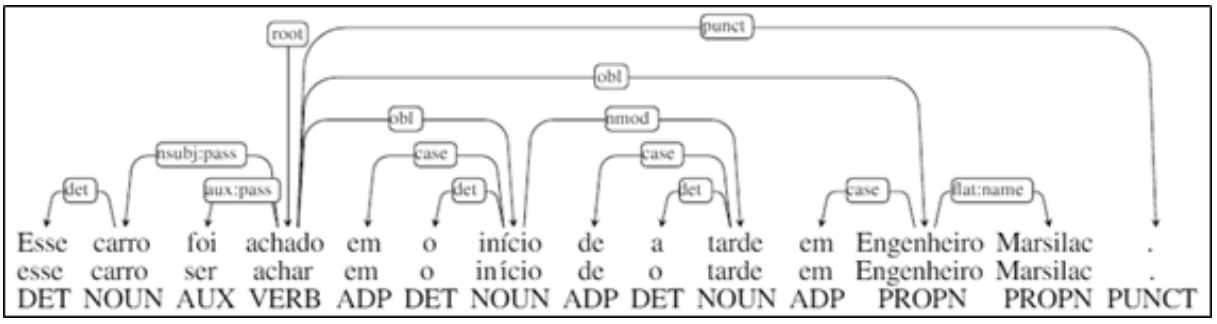

Figura 1. Exemplo de sentença em português anotada segundo o modelo UD.

Em função da grande adesão ao modelo UD e sua utilidade para o desenvolvimento de aplicações de Processamento de Linguagem Natural (PLN), *taggers* e *parsers* com base em UD têm sido criados para diversas línguas. Para o português, há alguns *parsers* que se destacam pelo uso na comunidade de pesquisa e que são objeto de análise neste trabalho, em

[1] https://universaldependencies.org/

especial, o UDPipe (Straka, 2018) e o PassPort (Zilio et al., 2018). Em seus trabalhos originais, os autores relatam valores de desempenho geral na ordem de 85 a 87% para anotação de relações sintáticas.

O objetivo deste trabalho é verificar o desempenho dos dois *parsers* citados de forma mais pontual, calculando precisão, cobertura e medida-f para algumas relações ditas essenciais (*core*) que denotam os argumentos centrais dos verbos, a saber: 'nsubj' (*nominal subject*), 'obj' (*object*) e 'iobj' (*indirect object*). Inspirada por outras iniciativas (por exemplo, Collovini et al., 2018, e Gonçalves et al., 2020), essa proposta faz parte de um esforço de mensurar de forma mais concreta os pontos fortes e fracos dos sistemas, visando fornecer subsídios para futuras pesquisas na área.

A seguir, descrevemos brevemente a metodologia adotada, sendo que os resultados obtidos são sintetizados na Seção 3. A Seção 4 apresenta algumas considerações finais.

2. Metodologia

2.1. O córpus de teste

O córpus Bosque foi anotado sintaticamente segundo as diretrizes da UD (como relatam Rademaker et al., 2017) por um grupo de pesquisadores da área e é utilizado nesse artigo como córpus de referência (*gold standard*), para avaliar a acurácia dos *parsers* testados. O córpus é composto por 9.364 sentenças e 210.957 tokens.

2.2. Um novo tokenizador

Além da versão padrão de tokenização disponibilizada com cada *parser*, também se utilizou um novo tokenizador desenvolvido no âmbito deste trabalho, mais alinhado com as diretrizes da UD, visando-se avaliar seu impacto na tarefa. O novo tokenizador, chamado LBTokenizer[2], pode ser utilizado sozinho ou integrado ao UDPipe.

2.3. A ferramenta de avaliação

O Conllu-File-Comparator[3] é um software desenvolvido na linguagem Python para essa pesquisa que compara as ocorrências das relações 'nsubj', 'obj' e 'iobj' de um arquivo com sentenças anotadas automaticamente pelos *parsers* com suas versões de referência, sendo que as anotações devem estar no formato CoNLL-U, amplamente adotado na área. Esse formato consiste em um conjunto de informações tabeladas, em que as palavras de uma sentença estão nas linhas e cada coluna armazena um tipo diferente de informação sobre as palavras.

Para cada sentença do arquivo de referência, o software contabiliza o número de cada relação essencial presente na sentença e calcula as medidas de precisão, cobertura e medida-f (assim como seus desvios padrões) para cada uma delas. Por exemplo, vamos calcular a precisão, cobertura e medida-f para uma sentença de teste com cinco tokens cujas relações essenciais estão representadas na Figura 1 e a sentença de referência com sua estrutura de relações representada na Figura 2 (note que, para simplificar a demonstração, omitimos as relações não essenciais e trocamos os tokens por números).

[2] Disponível em https://github.com/Lubelisa/LBTokenizer-UDPipe
[3] Disponível em https://github.com/Lubelisa/Conllu-File-Comparator

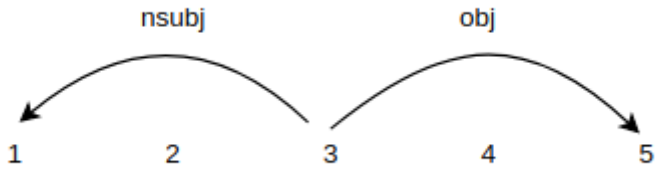

Figura 1. Sentença de teste, podendo ser anotada pelo UDPipe ou pelo PassPort

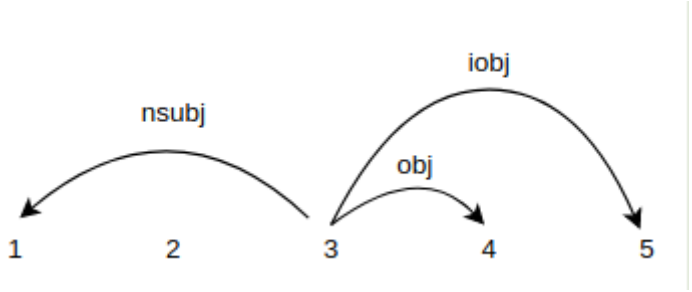

Figura 2. Sentença de referência, com anotação feita/revisada por um especialista

Na sentença de referência, há um caso de cada relação essencial e, na sentença de teste, um caso de relação 'nsubj' e um caso de 'obj'. Para calcular a precisão, precisa-se contabilizar a porcentagem de relações da sentença de teste que estão de acordo com o previsto na sentença de referência. Por exemplo, para calcular a precisão da relação 'nsubj' na sentença de teste, é necessário utilizar a fórmula a seguir.

$$Precisão_{nsubj} = \frac{número\ de\ relações\ 'nsubj'\ na\ sentença\ de\ teste\ que\ estão\ de\ acordo\ com\ a\ referência}{número\ de\ relações\ 'nsubj'\ na\ sentença\ de\ teste}$$

Assim, para a sentença de teste da Figura 1, o valor da precisão para a relação 'nsubj' é dada por $Precisão_{nsubj} = \frac{1}{1} = 1\ ou\ 100\%$. A precisão para a relação 'obj' é zero, pois, apesar de haver na sentença de teste um caso de 'obj', esse caso não é entre os mesmos tokens da sentença de referência e na mesma ordem (do token 3 para o 5).

No cálculo da cobertura, o denominador da fórmula verifica o número de relações da sentença de referência, ou seja, divide-se o número de cada relação essencial da sentença de teste que esteja de acordo com a sentença de referência pelo número de relações essenciais na sentença de referência. Logo, para a relação 'nsubj', a fórmula é:

$$Cobertura_{nsubj} = \frac{número\ de\ relações\ 'nsubj'\ na\ sentença\ de\ teste\ que\ estão\ de\ acordo\ com\ a\ referência}{número\ de\ relações\ 'nsubj'\ na\ sentença\ de\ referência}$$

Para a sentença de teste da Figura 1, o valor da cobertura para a relação 'nsubj' se dá por $Cobertura_{nsubj} = \frac{1}{1} = 1\ ou\ 100\%$. Caso houvesse dois casos de relação 'nsubj' na sentença de referência, o valor da cobertura para a relação seria de 0,5 ou 50%, portanto.

Por fim, a medida-f é uma combinação da precisão e da cobertura de cada relação, com o objetivo de se ter uma métrica única de avaliação. Foi utilizada a fórmula a seguir para calcular a medida-f de cada relação essencial:

$$Medida - f_{relação} = \frac{2*Precisão_{relação}*Cobertura_{relação}}{Precisão_{relação}+Cobertura_{relação}}$$

A seguir, relatamos os resultados obtidos para os dois *parsers* avaliados.

3. Resultados

O procedimento para a realização dos testes com os *parsers* foi passar por eles as sentenças de teste do córpus Bosque e comparar o arquivo de saída - que contém as sentenças anotadas automaticamente - com o arquivo de referência do Bosque que foi revisado por especialistas. Essa comparação e o cálculo dos resultados foi feito pelo software Conllu-File-Comparator e foram realizados três testes com os *parsers*: (1) utilizando o UDPipe para tokenizar e anotar as sentenças, (2) utilizando o LBTokenizer para tokenizar e o UDPipe apenas para anotar e (3) utilizando o PassPort para tokenizar e anotar as sentenças (pois a ferramenta foi desenvolvida para realizar as etapas em conjunto e não foi possível isolá-las). Os resultados médios dos testes (medidas e desvios padrões - DP) são mostrados na Tabela 1.

Tabela 1. Resultados dos *parsers*

	UDPipe			***LBTokenizer + UDPipe***			***PassPort***		
	nsubj	**obj**	**iobj**	**nsubj**	**obj**	**iobj**	**nsubj**	**obj**	**iobj**
Precisão	0,82	0,85	0,73	0,48	0,54	0,33	0,35	0,41	0,37
Cobertura	0,80	0,76	0,29	0,46	0,48	0,10	0,66	0,47	0,33
Medida-f	0,81	0,80	0,41	0,47	0,50	0,15	0,46	0,44	0,35
DP Precisão	0,33	0,30	0,42	0,46	0,45	0,47	0,13	0,37	0,17
DP Cobertura	0,36	0,37	0,44	0,46	0,45	0,28	0,21	0,41	0,35

Analisando-se as tabelas, conclui-se que o UDPipe sozinho obteve o melhor resultado. Para as relações ‘nsubj’ e ‘obj’, o UDPipe atingiu 80% de medida-f, mas ainda abaixo dos 87% relatados no artigo original (que engloba a avaliação de todas as relações). Há, portanto, uma grande margem para melhoria do sistema, o que se torna muito importante quando se considera que a análise produzida pelo *parser* é a entrada para outros processos em aplicações de PLN. Um erro nessa etapa pode ser propagado em outras, podendo impactar significativamente os resultados almejados. Também chama a atenção o baixo desempenho do PassPort para as três relações essenciais, apesar de, na maioria dos casos, esse sistema apresentar os menores desvios padrões para as medidas. É interessante notar que todos os sistemas avaliados têm desempenho menor para a relação ‘iobj’, que é sabidamente um dos casos mais desafiadores na anotação linguística com o modelo UD.

Nota-se que o uso do LBTokenizer degradou os resultados do UDPipe. Isso ocorreu porque o Bosque não segue algumas diretrizes mais atuais da UD, e usar o LBTokenizer gerou sequências de palavras diferentes das utilizadas para o treinamento do UDPipe para o português.

4. Considerações finais

Esse artigo apresenta um esforço em detalhar o desempenho de alguns *parsers* baseados no modelo UD para o português. Os resultados mostram que ainda é necessário avançar nessa frente. Trabalhos futuros incluem a avaliação de outras relações de dependência sintática e também o teste de outros sistemas, como o UDify (Kondratyuk and Straka, 2019).

Esse trabalho faz parte do projeto maior POeTiSA (*POrtuguese processing - Towards Syntactic Analysis and parsing*). Mais detalhes podem ser encontrados no portal web do projeto, em https://sites.google.com/icmc.usp.br/poetisa.

Agradecimentos

Os autores agradecem ao Centro de Inteligência Artificial da USP (C4AI - https://c4ai.inova.usp.br/), que conta com o apoio da IBM e da FAPESP (#2019/07665-4), e à Universidade de São Paulo pelo suporte financeiro.

Referências

Collovini, S.; Santos, H.D.P.; Lima, T.; Fonseca, E.; Pereira, B.; Souza, M.; Moraes, S.; Vieira, R. (2018). Cross-Framework Evaluation for Portuguese POS Taggers and Parsers. 19th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing).

Gonçalves, M.; Coheur, L.; Baptista, J.; Mineiro, A. (2020). Avaliação de recursos computacionais para o português. LinguaMÁTICA, Vol. 12, N. 2, pp. 51-68.

Kondratyuk, D. and Straka, M. (2019). 75 Languages, 1 Model: Parsing Universal Dependencies Universally. In the Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing, pp. 2779-2795.

Nivre, J. (2015). Towards a Universal Grammar for Natural Language Processing. In the Proceedings of the 16th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing), pp. 3-16.

Nivre, J.; Marneffe, M-C.; Ginter, F.; Hajič, J.; Manning, C.D.; Pyysalo, S.; Schuster, S.; Tyers, F.; Zeman, D. (2020). Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In the Proceedings of the 12nd International Conference on Language Resources and Evaluation (LREC), pp. 4034-4043.

Rademaker, A.; Chalub, F.; Real, L.; Freitas, C.; Bick, E.; Paiva, V. (2017). Universal Dependencies for Portuguese. In the Proceedings of the 4th International Conference on Dependency Linguistics (Depling), pp. 197-206.

Straka, M. (2018). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In the Proceedings of the CoNLL Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pp. 197-207.

Zilio, L.; Wilkens, R.; Fairon, C. (2018). PassPort: A Dependency Parsing Model for Portuguese. In the Proceedings of the 13rd International Conference on the Computational Processing of Portuguese (PROPOR), pp. 479-489.