Superando o estado da arte na etiquetagem morfossintática por meio de regras de pós-etiquetagem

Cid Ivan da Costa Carvalho[1]
Davis Macedo Vasconcelos[2]
Leonel Figueiredo de Alencar[3]

RESUMO: Segundo Kveton e Oliva (2002), os erros de etiquetagem, quando se utiliza um *corpus* para treinar estatisticamente algum sistema de processamento automático da linguagem natural, constituem desvios das regularidades que se espera que o sistema aprenda, resultando num modelo falso da língua. Esses autores propõem um método de correção desses erros num processo de pós-etiquetagem, levando em conta a detecção de *n*-gramas impossíveis na língua a ser modelada. Neste trabalho, procura-se sistematizar os erros cometidos pelo etiquetador morfossintático Aelius, construído por meio do NLTK (BIRD, KLEIN e LOPER, 2009). Como ponto de partida deste trabalho, compilamos um *corpus* de textos de comunicação mediada por computador, gênero que se caracteriza pela abundância de abreviaturas, grafias não padrão e desvios da norma culta. Após a anotação feita com o Aelius, levantamos os erros cometidos pela ferramenta e os classificamos. Com base nisso, elaboramos um primeiro conjunto de regras para correção automática da etiquetagem. Numa segunda etapa, tendo como meta chegar a 99% de acurácia, acima, portanto, do estado da arte de 97%

[1] Professor Assistente da Universidade Federal Rural do Semi-Árido e doutorando pela Programa de Pós-Graduação em Linguística pela UFC. cidivanc@gmail.com.

[2] Professor do Instituto Federal do Ceará e doutorando pela Programa de Pós-Graduação em Linguística pela UFC,. davis.ifce@gmail.com.

[3] Professor e orientador do Programa de Pós-Graduação em Linguística e do Departamento de Letras Estrangeiras da Universidade Federal do Ceará. leonel_de_alencar@yahoo.com.br.

(GÜNGÖR, 2010, p. 207), implementaremos regras em Python para correção dos erros cometidos por esse sistema de etiquetagem.
PALAVRAS-CHAVE: Aelius, erros de etiquetagem, *n*-grama, *corpus*.

ABSTRACT: According to Kveton e Oliva (2002) tagging errors, when a corpus is used in order to statistically train a natural language processing system, constitute deviations from regularities which the system is expected to learn, leading to a false model of the analyzed language. These authors propose a correction method for these errors in a post-tagging process, taking into account the identification of impossible n-grams in the language to be modeled. In this paper we try to systematize the errors made by the Brazilian Portuguese morphosyntactic tagger Aelius, built through the NLTK (BIRD, KLEIN and LOPER, 2009). The departing point was the compilation of computer mediated communication corpus. This genre is characterized by an abundance of abbreviations, spelling variation and standard language deviations. After the Aelius annotation was done, we listed the errors made and classified them. This was the basis for a set of rules for automatic tagging correction. Our goal was to reach a 99% accuracy, above the state of the art 97 (GÜNGÖR, 2010, p. 207), and in order to reach that we implement a system of Python rules to correct the errors created by the tagging system.
KEYWORDS: Aelius, tagging errors, n-grams, *corpus*.

1 Introdução

Antes de desenvolver os programas computacionais no universo linguístico é preciso "procurar as forças que estão em jogo, de modo permanente e universal. Em todas as línguas e deduzir as leis gerais as quais se possam referir todos os fenômenos peculiares" (SAUSSURE, 2006, p. 24) – ou seja, identificar "uma estrutura linguística imutável que sustenta a língua e subjaz a quaisquer outras realizações que dela se façam" (SAUTCHUK, 2010, p.3).

Esse ponto é fundamental para reiteração do caráter científico da linguística. Mioto (2007, p.13) enuncia que "Também na linguística esperamos ser capazes de fazer observações atentas e acuradas de maneira tão objetiva e imparcial quanto possível". É uma busca pelos "princípios que estejam na base de todo fenômeno sintático existente.

A esse conjunto de postulações básicas e de afirmações consequentes chamamos um modelo teórico.

É através dessa colaboração entre ciência da computação, através da subdisciplina da Inteligência Artificial - IA, e a linguística consolidar-se-ia mais tarde numa nova disciplina, a linguística computacional." (ALENCAR, 2006, p. 12-13) – ciência, hoje, subdividida em "linguística de corpus e o processamento de linguagem natural (PLN)". (OTHERO; MENUZZI, 2005, p. 22).

Nesse contexto, a linguística computacional caracteriza-se pela utilização de "computadores para o armazenamento e acesso a textos escritos ou falados" que "pode ser rapidamente pesquisado para informações a respeito da regularidade da língua, tais como frequência de palavras, de formas ou de construções". (VIEIRA; LIMA, on-line). Sua relevância é verificada na "elaboração de teorias gramaticais formalmente mais consistentes e psicolinguisticamente mais realistas […] e, assim, testar, com um grau de sofisticação que dificilmente poderia ser atingido por seres humanos, a adequação dos modelos postulados". (ALENCAR; OTHERO, 2011, p. 9-10).

Uma moderna definição para corpus linguístico é:

> uma coleção classificada de objectos linguísticos para o uso em Processamento de Linguagem Natural / Linguística Computacional /Linguística em que uso pode ser estudo, medição, teste, ou avaliação enquanto variados *objectos linguísticos* são textos, frases, palavras, entrevistas, erros ortográficos, entradas de dicionário, citações, pareceres jurídicos, filmes, imagens com legendas, traduções, correções (de textos de alunos de língua ou de tradução), telefonemas, simulações do tipo Wizard of Oz, programas (SANTOS, 2008, p. 45-46).

Esses dois conceitos acima expressos estão relacionados com a ferramenta de uso para o processamento da linguagem natural. A ferramenta, *Natural Language ToolKit* (doravante NLTK), aqui utilizada. Ela facilita a manipulação ou tratamento textual no que se refere ao processo de etiquetagem sintática – ou seja, categorização de palavras em classes gramaticais: substantivos, verbos, adjetivos, advérbios (no idioma Inglês). Tal ferramenta é utilizada pelo etiquetador Aelius.

Esse artigo se divide em quatro partes: no primeiro, apresenta-se a arquitetura do sistema Aelius, as funções para o pré-processamento

de corpora, o processo de anotação sintática e o tagset do sistema; no segundo tópico, expõe alguns erros de etiquetagem mais frequentes e relevantes cometidos por esse pacote em Python, feita com textos mediados por computador, bem como, um exemplo de implementação feita em Python com o intuito de aumentar da acurácia.

2 Etiquetador Aelius

A etiquetagem morfossintática das partes do discurso é uma das tarefas mais estudadas no processamento de linguagem natural. Ela se constitui como uma área muito importante, uma vez que auxilia em algumas ferramentas e em modelos de implementação que leva a aplicação em determinadas tarefas. Além disso, é uma tarefa aparentemente simples para o processamento da linguagem, no entanto, o desempenho de outras ferramentas depende diretamente desse processo, como mencionados acima.

Além disso, a etiquetagem morfossintática é uma tarefa intermediária que tem como objetivo principal analisar e entender a língua natural. Porém, a maior dificuldade nesse sistema reside na ambiguidade das palavras, em que cada palavra pode pertencer a várias categorias discerníveis geradas pelo contexto linguístico e/ou nos módulos desenvolvidos pelo sistema.

O processo de anotação automática é feito pelos etiquetadores morfossintáticos. Para a etiquetagem dos textos mediados por computador, utilizou-se o sistema Aelius.

Para compor o *corpus* de texto, selecionou-se comentários de blogs com temáticas sobre a educação. Esse gênero possui uma característica muito peculiar: a espontaneidade no uso linguagem, sendo assim, mais informal, o que contribui para observação do desempenho de na etiquetagem morfossintática.

O etiquetador Aelius[4] é uma ferramenta para anotação automática de *corpora* que possui uma arquitetura híbrida, ou seja, "recorre às regras, formuladas manualmente em expressões regulares, para etiquetar as palavras inexistentes no *language model*", (ALENCAR, 2010, p. 3), e ao sistema estatístico estocástico baseado

[4] O sistema Aelius é livremente disponível no endereço http://sourceforge.net/projects/aelius/files/.

em *n*-grama. Ele desempenha o pré-processamento de *corpus* de treino e anota o texto, com o *tagset* do Tycho Brahe, verificando o uso de nomes próprios no contexto linguístico .

As regras formuladas contribuem para a resolução de ambiguidade na etiquetagem. Segundo Voutilainen (2009), essas regras podem ser baseados sobre duas fontes de informações, ambas codificada no etiquetador na forma de uma linguagem modelo: a informação sobre a palavra em si, ou seja, em que contexto efetivo a palavra é mais usado, por exemplo, a palavra como: verbo ou advérbio; e as informações sobre a sequência palavra/etiqueta (ou contexto informacional): isto é o modelo pode preferir analisá-lo como um verbo a uma conjunção, se o termo precedente for uma advérbio ou um determinante.

O componente estatístico contribui durante o treinamento do sistema, produzindo regras simbólicas que, quando utilizada no corpus de teste, podem ser modificadas. Nas palavras de Voutilainen (2009), o sistema estatístico é, muitas vezes, baseada no que é conhecido sobre o léxico. Se, por exemplo, é sabido que o léxico contém todas "as classes fechadas de palavras" assim como os pronomes e artigos, o "adivinhador" pode com segurança propor apenas a análise de classes abertas (isto é, a análise de nomes e verbos).

Além de sua arquitetura, o Aelius possui uma série de funções em Python que fazem o pré-processamento do *corpus* de treino, conforme parâmetros especificados pelo usuário; nele, o usuário pode, também, especificar um conjunto de etiquetas a serem ignorados na construção do modelo, bem como, dividir o *corpus* de base em um número específico de blocos e embaralhá-los de forma aleatória, o que permite obter um *corpus* de treino e um corpus de teste mais balanceados. (ALENCAR, 2010).

Outro módulo fundamental desse sistema é a anotação morfossintática. Antes da anotação, no *input*, o Aelius realiza o processo de tokenização, ou seja, o interior do texto é dividido em *tokens* para outra análise como: marcas de pontuação, palavras compostas unidas por hífens, etc.

Depois a anotação de *corpus* nessa ferramenta ocorre da seguinte forma: no *input*, entra-se um texto não etiquetado. Enquanto etiqueta o texto, o sistema verifica se a palavra pertence a classe dos nomes próprios, nos casos afirmativos, o sistema etiqueta a palavra com a *tag* NPR. Depois, no *output*, o Aelius retorna um texto

etiquetado com o *tagset* do Corpus Histórico do Português Tycho Brahe.[5] Como mostra a figura abaixo:

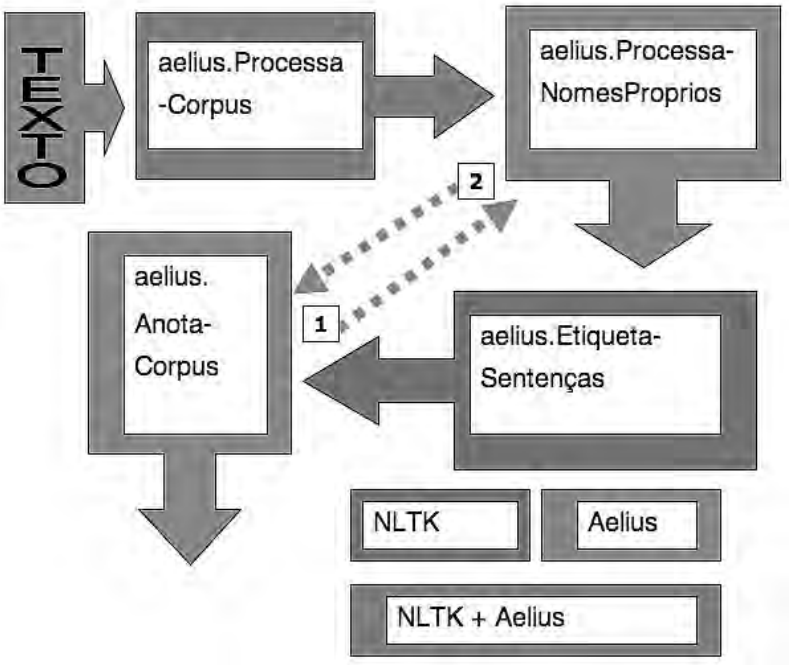

FIGURA 1
Anotação morfossintática pelo Aelius
Fonte: Alencar (2010)

Alencar (2010, p. 4), referindo ao módulo de nome próprio, diz que:

> A grande quantidade de erros relacionadas à etiqueta NPR (nome próprio) levou-nos a desenvolver e implementar em Python um algoritmo para distinguir nomes próprios de outras palavras com inicial maiúscula. Esse algoritmo supera grave deficiência do etiquetador de expressões regulares do NLTK, que não leva em conta o contexto onde um *token* ocorre.

[5] Para maiores informações sobre o *Corpus* Histórico do Português Tycho Brahe consulte o sítio do projeto: http://www.tycho.iel.unicamp.br/~tycho/corpus/index.html.

Esse módulo desenvolvido em Python contribui para o aumento da acurácia desse sistema, mesmo utilizando o *tagset* extenso como o do Tycho Brahe, o Aelius é ferramenta dispõe de alta acurácia na anotação morfossintática para textos literários do século XIX, com desempenho de 95% de acertos.

Corpus Histórico do Português Tycho Brahe foi desenvolvido junto ao projeto Padrões Rítmicos, fixação de parâmetros e mudanças linguísticas; é um *corpus* eletrônico anotado, composto de textos em português escritos por autores nascidos entre 1380 e 1845.

Atualmente, possui 53 textos (2.464.191 palavras) que estão disponíveis para pesquisa livre, com um sistema de anotação linguística em duas etapas: anotação morfológica (aplicada em 31 textos); e anotação sintática (aplicada em 14 textos).

Da organização desse projeto, foi elaborado um conjunto de etiquetas adequadas à representação e à discriminação das categorias necessárias à descrição dos enunciados da língua portuguesa para aquele período. Para tanto, dividiram o sistema de etiquetas morfológicas em dois grupos que, ao todo, perfazem um total de 383 *tags*, segundo Milidíu, Santos e Duarte (2008): as etiquetas categorias e as etiquetas flexionais. As primeiras classificam o item lexical em uma classe gramatical e as segundas indicam traços como gênero, número, pessoa, tempo e modo (CORPUS, 2010). Exemplo, desses grupos, podemos ver nos verbos plenos que recebem a etiqueta VB, quando está no infinitivo e; quando flexionais, acrescenta- se a essa, a etiqueta que determina a flexão do verbo. Dessa maneira, o tagset do corpus Tycho Brahe procura representar os termos das sentenças, tendo em conta a classificação das palavras e as discrimina conforme a flexão verbal e/ou nominal das unidades do discurso.

Como visto acima, o Aelius é um sistema com grande desempenho para a etiquetagem do português brasileiro, especialmente, para textos literários do século XIX. Porém, com o intuito de melhorá-lo no que se refere à precisão para os textos atuais, apresentam-se alguns erros mais frequentes cometidos por esse sistema quando testado em corpus de textos mediados por computador.

3 Erros e correção de pós-etiquetagem

A correção manual de textos automaticamente etiquetados levou a perceber certos erros frequentes na etiquetagem. Estes erros,

muitas vezes, eram claramente dependentes do contexto linguístico em que as palavras ocorriam. Em português, os nomes próprios e nomes comuns podem vir depois de artigos definidos, porém o etiquetador cometeu claramente um erro nos exemplos a seguir:

(1) … o/D professor/NPR que/C não/NEG adotar/VB o/D livro/N didático/ADJ...

(2) Espera/VB-I -/+ se/SE que/C o/D professor/NPR tire/VB-SP...

pois o termo "professor" foi etiquetado como nome próprio no (1) e no (2), enquanto deveria ser etiquetado como nome, ou seja, receber a etiqueta N. Esse termo está precedido de um determinante. Diferentemente do que acontece com os exemplos acima, o contexto (3) apresenta a etiquetagem, com a mesma palavra, se deu de forma correta.

(3) … muitos/Q-P professores/N-P fazem/VB-P um/D-UM

Outro caso de erro em que se dever observar com muita atenção foi a etiquetagem feita com os termos "Escola e escolar" que receberam, inadequadamente, as etiquetas NPR e VB, respectivamente, porém deveriam ser etiquetados como sendo nome/N e adjetivo/ADJ-G. Veja o contexto desse termos nas expressões em que foram utilizadas no *corpus*:

(4) … a/D-F realidade/N escolar/VB que/C dirigem/VB-P .

(5) … aconteceu/VB-D na/P+D-F Escola/NPR que/WPRO trabalhava/VB-D

No contexto (4), o sistema reconheceu a palavra "escolar" como sendo um verbo no infinitivo, logo, recebendo a etiqueta VB. Esse erro foi cometido devido à palavra terminar com AR, que é uma desinência dos verbos no infinitivo e, além disso, tanto a classe dos verbos quanto a classe dos adjetivos podem vir depois de um Nome. Esse problema não pode ser corrigido com um algorítmico que leva em consideração o final do termo AR antecedido de um nome/N deveria ser etiquetado com ADJ-G, pois os verbos também podem ser precedidos de nome.

O erro no contexto (5) não constitui inconsistência no processador de nomes próprios implementados (ver ALENCAR, 2010, p.5) ocorrido nessa situação, pois outros termos que inicia com letra maiúsculas não são etiquetados com NPR. Outros exemplos, no mesmo *corpus*, apresentam que o módulo implementado é bastante consistente. Veja os exemplos:

(6) … este/D assunto/N nas/P+D-F-P escola/N municipais/ ADJ-F-P …

(7) Nós/PRO em/P São/NPR Caetano/NPR do/P+D Sul/ NPR queremos/VB-P …

No contexto (6), a mesma palavra foi etiquetada corretamente. Observe ainda que o termo está antecedido pela etiqueta P+D, ou seja, pronome e determinante. Da mesma forma no (7), o termo "São" poderia ser etiquetado como sendo verbo ser, porém foi classificado como sendo um nome próprio. O erro no exemplo (5) foi cometido devido a erros no corpus de teste.

Além desse casos, percebem-se outras inconsistências na etiquetagem de termos em início de sentenças. Veja os exemplos (8) e (9).

(8) Envio/VB-P de/P verbas/N-P para/P este/D fim/N …

(9) Concordo/N totalmente/ADV contigo/P+PRO ./.

A palavra "Envio", que é um nome, foi etiquetada como sendo VB-P. De modo inverso acontece com o termo "Concordo" que é VB-P e está etiquetado como sendo nome. Esse caso é necessário observar que as duas palavras apareceram no início da sentença com as letras em maiúsculas. A etiquetagem da primeira palavra não levou em consideração a preposição que com ela forma um bigrama.

Para título de exemplo, foi feito um módulo Python de para correção desses erros dentro do contexto apresentado.

4 Implementação em Python

Abaixo, apresentam-se os processos pelos quais foram realizados o estudo: o programa 1 realiza etiquetagem com o sistema

Aelius, o programa 2, dada uma sentença, faz a correção automática dos erros em bigrama e o programa 3 verifica a correção das sentenças.

Em Python o símbolo sustenido (#) indica que o resto da linha é um comentário. O extensão do arquivo de ser em texto puro, ou seja, em txt para que o sistema possa processar os dados.

Programa 1: Etiquetagem pelo Aelius

```
# define o arquivo de entrada
arq = "ENTRADA.txt"

# importa o projeto Aelius
import Aelius

# realiza etiquetagem, gerando o arquivo anotado
from Aelius import AnotaCorpus
AnotaCorpus.anota_texto(arq,"AeliusBRUBT.pkl")

# renomeia arquivo temporário
import os
os.rename("ENTRADA.nltk.txt","SAIDA.etiquetado.txt")
```

Programa 2: Correção automática de sentenças

```
# processa as sentenças
tam = len(linhas)
# processa as sentenças (sents)
for i in range(0,tam):
sent = linhas[i]
sents = sent.decode("utf-8").split()
for cad in sents:
y = cad.find("@") # verifica se houve correção, ou seja, se existe o simbolo "@"
if (y > -1):
```

```
x = cad.find("/") # se houve, procura o simbolo "/"
palavra = cad[0:x] # copia para "palavra" o substring antes do "/"
nova_etiq = cad[y+1:] # copia para "nova_etiq" o substring depois
do "@"
out = palavra + "/" + nova_etiq # ajusta "nova_etiq" para "out"
else:
out = cad # se não houve correção
lista.append(out) # "out" é inserida na lista
```

Programa 3: Verificação da correção

```
sent='que/C o/D professor/NPR tire/VB-SP'
sents=sent.decode('utf-8').split()
from nltk import bigrams
cad=bigrams(sents)
for(w1,w2)in cad:
        x=w1.find('/')
        palavra1=w1[0:x]
        tag1=w1[x+1:]
        x=w2.find('/')
        palavra2=w2[0:x]
        tag2=w2[x+1]
        if ((tag1=='D')and(palavra2=='professor')):
                tag2='N'
                print palavra2,'/',tag2
```

A correção das outras sentenças segue o mesmo exemplo da primeira, pois a implementação segue uma relação de bigramas e não uma relação entre etiquetas, ou seja, o contexto em que os termos estão dispostos na sentença. Isso produz um ganho na acurácia do etiquetador, uma vez que trata o problema local, mas contextos com outras palavras.

5 Considerações

Alencar (2010), no desenvolvimento do etiquetador, procedeu incrementalmente, repetindo várias vezes o ciclo editar – compilar – testar – depurar o sistema de etiquetagem. Esse processo envolveu contribui para otimização do etiquetador baseado em expressões regulares e, também, a eliminação de tokens com etiquetas que não se referem à análise morfossintática bem como correções de inconsistências do próprio *corpus* de treino.

Ele acrescenta que a grande quantidade de erros relacionadas à etiqueta NPR (nome próprio) levou a desenvolver e implementar em Python um algoritmo para distinguir nomes próprios de outras palavras com inicial maiúscula. Esse algoritmo supera grave deficiência do etiquetador de expressões regulares do NLTK, que não leva em conta o contexto onde um *token* ocorre. Todavia, é necessário depurar e rever o módulo para aumentar a qualidade na etiquetagem, quando o *corpus* é constituído por textos atuais.

Além disso, outra possibilidade de aumentar a acurácia do etiquetador é diminuir erros que envolvem contextos sintáticos claramente identificáveis, por meio da inclusão de regras de reetiquetagem elaboradas manualmente.

Referências

ALENCAR, Leonel Figueiredo de. Aelius: uma ferramenta para anotação automática de corpora usando o NLTK. IX ENCONTRO DE LINGUÍSTICA DE CORPUS. *Anais*... Porto Alegre, PUCRS, 8 e 9 de outubro de 2010.

ALENCAR, Leonel Figueiredo de. *Linguagem e Inteligência Artificial*: na coletânea Linguagens. As expressões do Múltiplo. Fortaleza: Premius, 2006.

ALENCAR, L. F.; OTHERO, G. A. (Org.). *Abordagens computacionais da teoria da gramática.* Campinas, SP: Mercado das Letras, 2011.

BIRD, S.; KLEIN, E.; LOPER, E. *Natural Language Processing with Python*: Analyzing Text with the Natural Language Toolkit. Creative Commons Attribution-NonCommercial-No Derivative Works 3.0. New York, 2007.

CARNIE, A. *Syntax*: a generative introduction. Blackwell Publishing, 2007.

CORPUS Histórico do Português Tycho Brahe. Campinas: Instituto de Estudos da Linguagem/Universidade Estadual de Campinas, 2010. Disponível em: <http://www. tycho.iel.unicamp.br/~tycho/corpus/ > Acesso em: 30. set. 2010.

KVETON, P.; OLIVA, K. (Semi-)Automatic Detection of Errors in PoS-Tagged Corpora. INTERNATIONAL CONFERENCE ON COMPUTATIONAL LINGUISTICS. *Proceedings*... n. 19, 2002, Taipei. Stroudsburg: Association for Computational Linguistics, 2002.

MILIDIU, Ruy Luiz; SANTOS, Cícero Nogueira dos; DUARTE, Julio Cesar. *Portuguese corpus-based learning using ETL*. J. Braz. Comp. Soc. [online], v. 14, n. 4, p. 17-27. ISSN 0104-6500, 2008.

MIOTO, C.; SILVA, M. C. F.; VASCONCELLOS, R. E. *Novo manual de sintaxe*. Florianópolis: Insular, 2007.

OTHERO, G. A; MENUZZI, S. M. *Linguística computacional*: teoria & prática. São Paulo: Parábola Editorial, 2005.

SANTOS, D. Corporizando algumas questões. In: TAGNIN, Stella E. O.; VALE , Oto Araújo. (Org.). Na coletânea *Avanços da linguística de corpus no Brasil*. São Paulo: Humanitas, 2008.

SARDINHA, T. B.; ALMEIDA, G. M. B. A Linguística de Corpus no Brasil. In: TAGNIN, Stella E. O.; VALE , Oto Araújo. (Org.). Na coletânea *Avanços da linguística de corpus no Brasil*. São Paulo: Humanitas, 2008.

SAUTCHUK, I. *Prática de morfossintaxe*: como e por que aprender aprender análise (morfo)sintática. 2. ed. Barueri, SP: Manole, 2010.

SAUSSURE, F. *Curso de Linguística Geral.* 27. ed. São Paulo: Cultrix, 2006.

VIEIRA, R.; LIMA, V. L. *Linguística computacional*: princípios e aplicações. São Leopoldo: Unisinos. Disponível em: www.inf.unioeste.br/~jorge/.../linguística%20computacional.pdf.

VOUTILAINEN, Atro. Part-of-speech tagging. In: MITKOV, Rusland (Org.). *The Oxford Handbook of computational linguistics.* Oxford: Oxford university Press, 2009.