UNIVERSIDADE
FEDERAL RURAL
DE PERNAMBUCO

Viviane Barbosa de Araujo

Utilização de Processamento de Linguagem Natural para Identificação do Domínio da Escrita Formal em Redações da Língua Portuguesa

Recife

2020

Viviane Barbosa de Araujo

Utilização de Processamento de Linguagem Natural para Identificação do Domínio da Escrita Formal em Redações da Língua Portuguesa

Monografia apresentada ao Curso de Bacharelado em Ciências da Computação da Universidade Federal Rural de Pernambuco como requisito parcial para a obtenção do título de Bacharel em Ciências da Computação

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Computação

Curso de Bacharelado em Ciências da Computação

Orientador: Dr.Rafael Ferreira Mello

Recife

2020

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

A663u Araujo, Viviane Barbosa de
Utilização de processamento de linguagem natural para identificação do domínio da escrita formal em redações da língua portuguesa / Viviane Barbosa de Araujo. - 2020.
23 f. : il.

Orientador: Rafael Ferreira Mello.
Inclui referências, apêndice(s) e anexo(s).

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Ciência da Computação, Recife, 2020.

1. Redação. 2. Processamento de Linguagem Natural. 3. Identificação de erros ortográficos e gramaticais. I. Mello, Rafael Ferreira, orient. II. Título

CDD 004

MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO (UFRPE)
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

http://www.bcc.ufrpe.br

FICHA DE APROVAÇÃO DO TRABALHO DE CONCLUSÃO DE CURSO

Trabalho defendido por Viviane Barbosa de Araújo às 10 horas do dia 07 de dezembro de 2020, no link meet.google.com/tnx-tmvu-kpo, como requisito para conclusão do curso de Bacharelado em Ciência da Computação da Universidade Federal Rural de Pernambuco, intitulado **Utilização de Processamento de Linguagem Natural para Identificação do Domínio da Escrita Formal em Redações da Língua Portuguesa**, orientado por Rafael Ferreira Leite de Mello e aprovado pela seguinte banca examinadora:

Rafael Ferreira Leite de Mello
DC/UFRPE

Péricles Barbosa Cunha de Miranda
DC/UFRPE

Resumo

No Brasil, o principal meio de ingressar em uma universidade pública ou privada é através do Exame Nacional do Ensino Médio, o ENEM. Esse exame exige que o candidato possua a habilidade de redigir um bom texto dissertativo-argumentativo de acordo com a norma formal da língua portuguesa, podendo ser eliminado do exame caso não cumpra esse requisito. Com o objetivo de ajudar o candidato a identificar os seus erros e ajudar no processo de escrita de uma boa redação, este artigo propõe a implementação de uma ferramenta capaz de identificar os erros ortográficos e gramaticais de um texto utilizando técnicas de Processamento de Linguagem Natural (PLN). A análise das ferramentas mostrou que os resultados obtidos pela pesquisa são promissores, principalmente em relação à identificação de erros gramaticais.

Palavras-chave: Redação, Processamento de Linguagem Natural, Identificação de erros ortográficos e gramaticais.

Abstract

In Brazil, the main means of entering a public or private university is through the National High School Exam, ENEM. This exam requires that the candidate has the ability to write a good dissertation-argumentative text according to the formal norm of the Portuguese language, and can be eliminated from the exam if he does not fulfill this requirement. In order to help the candidate to identify his mistakes and help in the process of writing a good essay, this article proposes the implementation of a tool capable of identifying the spelling and grammatical errors of a text using techniques of Natural Language Processing (PLN). The analysis of the tools showed that the results obtained by the research are promising, mainly in relation to the identification of grammatical errors.

Keywords: Essay, Natural Language Processing, Identification of Spelling and Grammatical errors.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Escrever uma boa redação é um fator primordial para todos os estudantes que queiram ingressar em uma universidade pública ou privada no Brasil, pois o Exame Nacional do Ensino Médio (ENEM) exige que o candidato tenha um bom domínio da escrita formal da língua portuguesa e que seja capaz de desenvolver um texto dissertativo-argumentativo coerente com o tema proposto. Além disso, a redação do ENEM é a única prova do exame que o candidato poderá tirar uma nota igual a zero, desclassificando o aluno totalmente do exame, independente do bom ou mal resultado que o candidato obteve nas provas das áreas de linguagens, ciências humanas e exatas, ou seja, mesmo que a aluno obtenha uma nota superior a 700 em ciências humanas,exatas e linguagem, mas uma redação com nota igual a Zero, o aluno será desclassificado do exame, precisando assim esperar o próximo ano para realizar mais uma vez o ENEM (INEP, 2017).

Outro fator importante nesse contexto que pode atrapalhar o estudante para redigir uma boa redação é o uso intensivo de aplicativos de mensagem simultânea, pois com a necessidade de se comunicar de forma mais rápida, o aluno tende a fugir das regras gramaticais e ortográficas, se acostumando assim a escrever de forma informal e errada, na maioria das vezes (KOMESU; TENANI, 2015). Os famosos ”memes da internet”também incentivam as pessoas a escreverem errado indiretamente, por exemplo, expressões como ”Esse Bilete”ou ”Tá serto”são populares no meio virtual, mas o contato direto com essas expressões faz com que o cérebro do aluno absorva essa informação de forma tão natural que quando é preciso redigir um texto formal, o aluno acaba escrevendo de maneira errada sem perceber. Segundo Marcuschi (2000) citado por (FALCãO, 2020) *”Este “novo idioma chamado internetês”*, como é chamado a linguagem do ambiente virtual, *vem deixando gramáticos em pânico”*.

Portanto, com o objetivo de ajudar os alunos a escreverem uma boa redação pesquisadores vem desenvolvendo ferramentas de análise e correção automática de redações. No entanto, a maioria dessas ferramentas são limitadas a análise de textos em inglês.Ainda assim, no Brasil existem algumas ferramentas automáticas que auxiliam o aluno nesse processo, podemos citar o Avaliador Automático de Coesão Textual em Redação Dissertativa (AVAC) cujo principal objetivo é a análise a coesão e coerência das Redações, que correspondem a competência 3 e 4 do ENEM (NOBRE; PELLEGRINO, 2010) e o CoGrOO, que é Corretor Gramatical do LibreOffice, capaz de fazer correção ortográfica e gramatical de textos à medida que o aluno escreve frases utilizando o LibreOffice. (COGROO, 2014).

Diante do contexto citado acima, este artigo propõe o desenvolvimento e análise de ferramentas capazes de identificar erros ortográficos em Redações utilizando algoritmos de processamento de linguagem natural, o aperfeiçoamento do algoritmo do CoGrOO de identificação de erros gramaticais e a análise dos algoritmos XGBoost e AdaBoost para classificação da nota da competência 1 do ENEM,competência que avalia o domínio da escrita formal do candidato, com o objetivo de ajudar os alunos a identificar os seus principais erros a fim de ajudá-los a melhorar a sua escrita.

2 Embasamento Teórico

Nesta seção definimos o que é a Competência 1 do ENEM, Processamento de Linguagem Natural, Classificação e o CoGrOO, em seguida serão apresentadas algumas ferramentas de avaliação automática de textos disponíveis na literatura.

2.1 Competência 1 do Enem

A competência 1 do ENEM avalia o participante quanto ao domínio da modalidade formal da língua portuguesa, ou seja, em relação as normas ortográficas e gramaticais. A correção da competência 1 exige dos corretores uma avaliação precisa dos textos, pois os textos diferem entre si quanto a estrutura sintática (deficiente, regular, boa e excelente) e a quantidade de desvios (erros ortográficos). O principal desafio dos corretores é avaliar redações que apresentam uma estrutura sintática boa, mas apresenta muitos desvios e vice-versa. Além disso, desvios gramaticais mais comuns encontrados nas redações são respectivamente: Regência, Concordância , crase e emprego de pronome, sendo os erros de concordância os que necessitam de uma atenção maior por parte dos corretores junto com os erros de grafia que são bem mais comuns do que outros desvios (INEP, 2019).

2.2 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) é uma subárea da Inteligência Artificial que utiliza técnicas computacionais para interpretar a linguagem natural humana com o objetivo de realizar uma determinada tarefa, seja análise ou classificação de dados (LIDDY, 2001). As principais técnicas de PLN envolvem tokenização ou segmentação, detecção de sentenças, identificação das classes gramaticais do texto, a relação entre as entidades e análise de correferência, sendo bastante utilizadas na fase de Pré-Processamento.

O Pré-processamento do texto é uma técnica de PLN que envolve a preparação das sentenças que serão utilizadas como entrada de dados. Essa preparação implica em retirar do texto elementos que são irrelevantes, isso inclui stopwords que são: artigos, pronomes e advérbios que estão presentes no texto. Assim como, a transformação das palavras para seu estado primitivo e seu radical, ou seja, a lematização e a Segmentação do texto.

2.3 Classificação

A classificação de um texto é feita a partir de um conjunto de frases, textos ou documentos que estão previamente rotulados, ou seja, que pertencem a uma determinada classe. Esses dados são processados e treinados por um algoritmo de aprendizagem de máquina supervisionada. No final do treinamento, o algoritmo é capaz de rotular ou classificar qualquer novo elemento que lhe é apresentado como pertencente a uma determinada classe (ROLIM, 2016).

Entre os algoritmos utilizados no processo de aprendizagem de máquina estão o AdaBoost e o XGBoost. AdaBoost é um classificador de aprendizagem de máquina utilizado para problemas de classificação e regressão. O algoritmo tem como objetivo ajudar as amostras consideradas mais ”fracas”a ter um bom desempenho. O algoritmo faz isso através de pesos ponderados que são atribuídos entre cada amostra. Em cada interação do algoritmo, as amostras que acertaram a sua classificação durante o treinamento recebem um peso menor e as que erraram recebem um peso maior, dessa forma os exemplos mais fracos são forçados a aprender e ter um bom desempenho (TH. PARK DC., 2012).

Já o XGBoost é um outro algoritmo de aprendizagem de máquina que utiliza a técnica de Gradient Booster ou Aumento gradativo para resolver problemas de classificação e regressão de forma mais rápida e eficiente. Ele se utiliza de um modelo de árvore de decisão que analisa as possibilidades positivas e negativas de um conjunto de árvores gerando uma função que aproveita o resultado das previsões de cada árvore, dessa forma os resultados se complementam a fim de gerar uma precisão maior. Essa função base é otimizada levando em consideração a função de perdas e a regularização do algoritmo (CHEN; GUESTRIN, 2016).

2.4 CoGrOO

O CoGrOO é um corretor gramatical para documentos do OpenOffice e LibreOffice que verifica e corrige erros de Grafia, Concordância Verbal, Concordância Nominal e crase em textos escritos em português utilizando métodos de processamento de linguagem natural. No entanto, essa correção é feita em ”tempo real”à medida que o aluno redige um texto na plataforma. Para a classificação de erros gramaticais, o CoGrOO utiliza como base de dados uma arquivo XML, chamado de rules, que contém a descrição e o formato das principais regras gramaticais. No total são 129 regras inseridas, no entanto a última atualização das regras foi no ano de 2014 de modo que algumas dessas regras ficaram desatualizadas e estão inativas no sistema (COGROO, 2014).

3 Trabalhos Relacionados

Nessa seção serão mostrados alguns trabalhados relacionados na área de Classificação de Texto utilizando Algoritmos Tradicionais de Aprendizagem Supervisionada e Processamento de Linguagem Natural.

Com foco em melhorar a classificação de Redações, pesquisadores chineses desenvolveram uma nova característica para análise de dados levando em conta as classes gramaticais das palavras. Essa nova característica identifica todas as classes gramaticais e a frequência que elas aparecem no texto selecionando as 3 classes predominantes no texto e estabelecendo uma relação entre elas. A adição desta nova relação de classes aumentou o desempenho do algoritmo em 1.9%. No entanto, a base de dados de redação utilizada na pesquisa foram redações em inglês escritas por chineses aprendizes da língua inglesa. O algoritmo não foi testado em textos em inglês escritos por nativos ou falantes do idioma de outros países (Wang et al., 2010).

Outro exemplo de correção automática de redação foi feito por (Israel et al, 2010). Esse sistema identifica a colocação de vírgulas erroneamente em redações da língua inglesa para usuários nativos e não nativos. A aplicação utiliza um algoritmo que calcula a distância entre as sentenças de modo a inferir em que lugar do texto a vírgula deveria ser adicionada,desse modo o algoritmo conseguiu uma precisão de 89% (ISRAEL; TETREAULT; CHODOROW, 2012). Ao analisar os tradicionais algoritmos de aprendizagem de máquina para classificação de texto, (Mishu et at,2016) chegaram a conclusão que uma boa classificação de um texto *"depende de quão bem estruturada a base de dados está e da escolha de um bom método de classificação"*. Dessa maneira, eles perceberam que a acurácia dos algoritmos foram bem similares, possuindo uma diferença de 22.5% do menor para o maior resultado (Mishu; Rafiuddin, 2016).

Um exemplo de correção automática de redação na língua portuguesa foi a desenvolvida por (NOBRE; PELLEGRINO, 2010). Esses pesquisadores desenvolveram um algoritmo que recebe como entrada as frases presentes no texto e avaliam de modo individual os seus elementos anafóricos criando listas de focos explícitos e implícitos. Após este processo, as frases adjacentes são avaliadas comparando a sua relação com a frase anterior e gerando um valor que será verificado na tabela de coesão. Dessa forma, o algoritmo é capaz de dizer o grau de coesão de cada frase. O desempenho desse algoritmo foi de 70% de acerto na classificação da nota geral similar a nota que o professor daria.

No ano de 2013, os pesquisadores (Thanh et al., 2013), analisaram os algoritmos de aprendizagem de máquina semi supervisionados junto com o aperfeiçoamento

do algoritmo SVM para classificação de texto. O resultado obtido pela pesquisa foi uma acurácia de 95,3%. Os pesquisadores concluíram com essa pesquisa que o bom desempenho desses algoritmos foi proporcional a quantidade de dados usados para treinamento. Já no ano de 2015, pesquisadores tendo como objetivo melhorar o feedback das redações, resolveram juntar as notas das redações com os atributos do estudantes. Dessa forma, eles conseguiram identificar quais áreas os estudantes precisam se esforçar mais e dar um feedback mais assertivo (CROSSLEY et al., 2015).

Em 2017 (RASJID; SETIAWAN, 2017) compararam a performance dos algoritmos de Naive Bayes e K-NN para classificação de Documentos de Texto utilizando-se do software RapidMiner. Após o experimento, os pesquisadores chegaram a conclusão que ambos os algoritmos podem ser usados para classificação de texto, sendo o K-NN melhor que o Naive Bayse quando o valor de K é maior que 1.

Tabela 1 – Diferença entre os trabalhos relacionados

Trabalhos	Análise Ortográfica	Análise Gramatical	Classificação Automática
(NOBRE; PELLEGRINO, 2010)	Não	Não	Sim
(Wang et al., 2010)	Não	Sim	Sim
(ISRAEL; TETREAULT; CHODOROW, 2012)	Não	Sim	Sim
(Mishu; Rafiuddin, 2016)	Não	Não	Sim
(Thanh et al., 2013)	Não	Não	Sim
(CROSSLEY et al., 2015)	Não	Não	Sim
(RASJID; SETIAWAN, 2017)	Não	Não	Sim
Proposta	Sim	Sim	Sim

4 Método de Pesquisa

A primeira fase dessa pesquisa compreende a revisão da literatura. Nesta fase foi feita a busca por trabalhos relacionados na base de dados da IEEE Explorer, ACM, Science Direct e Google Scholar. A string de busca utilizadas foi:(Text Classification OR text classifier) and (Natural Language Processing OR essay OR Supervised learning).

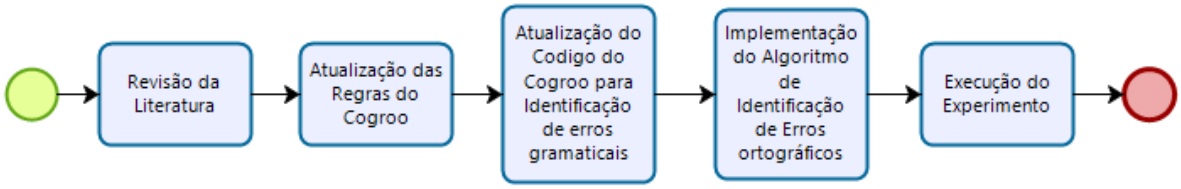

Figura 1 – Etapas do Projeto

A segunda fase da pesquisa correspondeu na atualização das regras gramaticais da base de dados do CoGrOO. Para isso foi necessário revisar as regras gramaticais existentes no banco e adicionar novas regras. Como fonte de dados foi utilizado o livro Novas Palavras: literatura, gramática, redação e leitura (LEITE, 1997). As novas regras foram adicionadas na base de dados XML do algoritmo do CoGrOO que está configurado com os seguintes campos: categoria gramatical, mensagem, descrição da regra, sugestão de escrita e exemplo da regra.

Com as novas regras adicionadas, a próxima etapa da pesquisa foi a adaptação do código do CoGrOO para receber uma redação, identificar e contabilizar os erros gramaticais (concordância verbal, concordância nominal, crase, regência nominal e verbal e emprego de pronomes) encontradas no texto gerando um arquivo CSV(O detalhamento dos atributos gerados pelo CSV pode ser visto na Tabela 2) Nessa etapa a linguagem de programação utilizada foi Java e o Eclipse como o principal ambiente de desenvolvimento. Os dados utilizados como treinamento dos algoritmos foram as redações do banco de dados da UOL que possui 2164 redações,cada redação possui a nota geral, nota em cada competência do ENEM e erros encontrados. Na Tabela 3 vemos as quantidades de Redações, contidas no banco analisado, por cada nota da Competência 1.

A quarta fase foi a implementação de um algoritmo de identificação de erros ortográficos. Para a confecção desse algoritmo foi utilizada a linguagem de programação Python e as bibliotecas de processamento de linguagem natural, como natas, nltk e spacy. O algoritmo recebe como entrada uma redação,faz o pré-processamento dos dados com a tokenização das palavras, a identificação de nome próprios,nome de en-

Tabela 2 – Dados Gerados pelo Algoritmo de Identificação de Erros Gramaticais

Classes	Tipo de Dados
Crase	inteiro
Concordância Verbal	Inteiro
Concordância Nominal	Inteiro
Regência Nominal	Inteiro
Regência Verbal	Inteiro
Emprego de mim e ti	Inteiro
Emprego de mau e mal	Inteiro
Colocação pronomial	Inteiro
Emprego de eu e mim	Inteiro
Vírgula	inteiro
Redundância	inteiro

Tabela 3 – Quantidade de Redações por Nota da Competência 1

Nota	Quantidade
0	93
0.5	428
1	887
1.5	567
2	189

tidades,números e a extração de stopwords.Após isso, um dicionário de português foi criado utilizando as bibliotecas machado, mac_morpho, floresta, genesis contidas no nltk e o dicionario pt-br do spacy.

Com base nesses dicionários, o algoritmo verifica a grafia das palavras encontradas na redação e retorna um CSV com a contabilização dos erros encontrados. As saídas dos algoritmos,isto é, a quantidade de erros gramaticais separados por categorias(crase, concordância verbal,concordância nominal, emprego de mim ti,emprego mau e mal,colocação pronomial, regência verbal,emprego de eu e mim, regência nominal,vírgula e redundância) e a quantidade de erros ortográficos junto com a quantidade de erros identificados por professores do UOL, a nota geral e a nota da competência 1 atribuída por eles foram utilizadas como entrada de dados dos algoritmos AdaBoost e XGBoster com o objetivo de tentar prever a nota da competência 1 de forma automática. Esses algoritmos foram escolhidos porque conseguem ter um bom desempenho na classificação de dados mesmo tendo como entrada um conjunto de dados considerados mais simples, ou seja, com poucas características de entrada (menos de 20), como é o caso das características extraídas nesse projetos e por meio das interações do algoritmos, eles conseguem balancear o treinamentos das característicos de entrada ajudando-os à aprender.

As métricas usadas para avaliar os algoritmos foram a acurácia e Kappa. A acurácia corresponde a formula abaixo:

$$Ac = \frac{VerdadeirosPositivos + VerdadeirosNegativos}{NumeroTotaldeExemplos}$$

O Número Total de Exemplos = VP + VN + FP + VN

Precision (Precisão):

$$Prec = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosPositivos}$$

Recall (Relevância):

$$Rec = \frac{VerdadeirosPositivos}{VerdadeirosPositivos + FalsosNegativos}$$

F1-Score:

$$F1 = 2x\frac{Preciso * Recall}{Preciso + Recall}$$

Verdadeiro Positivo: Quando o algoritmo acerta que o elemento avaliado pertence a classe ”Verdadeiro”

Verdadeiro Negativo : Quando o algoritmo acerta que o elemento pertence a classe ”Falso”

Falso Positivo: Quando o algoritmo classifica o elemento na classe ”Verdadeiro”, mas deveria ser na classe ”Falso”

Falso Negativo: Quando o algoritmo classifica o elemento na classe ”Falso”, mas deveria ter classificado na classe ”Verdadeiro”

A kappa é uma medida de avaliação de dados que análise a concordância entre os elementos, sendo medida no intervalo de 0-1, onde 1 é a concordância total, ou seja, quando o algoritmo acerta a classe do elemento e 0 quando o algoritmo erra por completo a classe do elemento analisado. A kappa corresponde a fórmula abaixo:

$$Kappa = \frac{Concordncia - ConcordnciaEsperada}{1 - ConcordnciaEsperada}$$

5 Resultados

O primeiro resultado dessa pesquisa foi a atualização das regras gramaticais do CoGrOO. Foram adicionadas mais 24 entradas no arquivo de regras, além de atualizar as 129 regras que já existiam. De modo que as regras gramaticais do CoGrOO somam atualmente 153. Dentre as regras adicionadas, a maior parte correspondeu às regras de concordância verbal e nominal. As regras foram adicionadas seguindo o padrão do arquivo de regras do CoGrOO conforme descrito abaixo e demonstrado na (Figura 2).

1. Índice: Refere-se ao número da regra no arquivo.
2. Ativo: Status da regra, ou seja,identifica se a regra está em funcionamento ou não.
3. Categoria: Classe gramatical pertencente, exemplo: concordância verbal
4. Grupo: Representa a regra da classe gramatical que está sendo utilizada
5. Mensagem: Mensagem de texto longa que explica a regra
6. Mensagem Curta: Mensagem de texto que será apresentada ao usuário com a explicação resumida da regra que está sendo aplicada.
7. Regra: Apresenta o formato da regra e as suas combinações.
8. Sugestão: Exibe para o usuário a(s) forma(s) correta(s) da norma.
9. Exemplo: Mostra exemplo de frase certa e errada.

Nas etapas da condução da pesquisa citadas da capítulo do método foi utilizado o método Experimento que compreende na medição de variáveis e avaliação do impacto e a influência que uma variável tem sobre outra. A variáveis de estudo analisadas foram os resultados dos algoritmos de identificação de erros gramaticais e ortográficos e a variáveis usadas como comparação foram os erros identificados manualmente por professores da UOL.

Na (Figura 3), encontra-se a diferença entre os resultados obtidos pelo algoritmo de identificação de erros gramaticais em comparação com a quantidade de erros encontrados por professores da língua portuguesa. No Gráfico, o eixo X corresponde a quantidade de erros, ou seja, a diferença entre automação X manual e o eixo Y corresponde a quantidade de redações. Para facilitar o entendimento, vejamos um exemplo: A redação A possui 10 erros encontrados pelos professores da UOL e 9 erros encontrados pelo algoritmo de identificação de erros gramaticais. Portanto, a diferença entre

Node	Content
id	154
active	true
Method	general
Type	Concordância do verbo ser indicando horas
Group	verbo ser +numeral
Message	O verbo concorda com a expressão numerica
ShortMessage	O verbo concorda com a expressão numerica
Pattern	
PatternElement	
Element	
Mask	
TagMask	
Class	finitive verb
PatternElement	
Element	
Mask	
TagMask	
Class	numeral
Number	plural
Boundaries	
Suggestion	
Replace	
index	0
TagReference	
index	0
TagMask	
Number	plural
Example	

Figura 2 – Exemplo de uma regra do CoGrOO

eles é de 10-9 = 1, ou seja, apenas 1 erro não foi detectado pelo algoritmo. A base de dados possui 2164 redações, dessas 2164 redações, 654 apresentaram apenas 1 erro. A diferenças de erros de 0 até 3 correspondem a um total de 1740, ou seja, 80.40%.

Na (Figura 4) encontramos os resultados obtidos pelo algoritmo de identificação de erros ortográficos e a divergência existente entre os erros encontrados automaticamente e os erros identificados manualmente pelos professores da UOL. No gráfico podemos observar que a diferença de apenas 1 erro foi a maior diferença, correspondendo ao total de 321 redações ou 14.8% em seguida vem a diferença de 2 erros com um total de 299 redações ou 13.8%. Ao analisarmos conjuntos de erros veremos que o grupo de 0-5 erros correspondem a 67.7% Na (Figura 5), vemos as diferenças totais encontradas automaticamente pelos algoritmos versus os erros encontrados pelos professores. A maior classe foi a diferença de 2 erros com um total de 273 redações ou 12.61% e a classe de 1 erro com um total de 247 redações ou 11.41% . A diferença de 0-5 erros é de 56.3%. Nos três gráficos, vimos que as diferenças entre os erros encontrados pelos algoritmos desenvolvidos e os erros que os próprios corretores da UOL encontraram foram similares, não houve muita divergência de erros na maior parte do banco de dados utilizado.

Utilizando esses mesmos dados gerados pelos algoritmos e as informações extraídas do banco de dados da UOL (nota geral da redação e os erros encontrados pelos

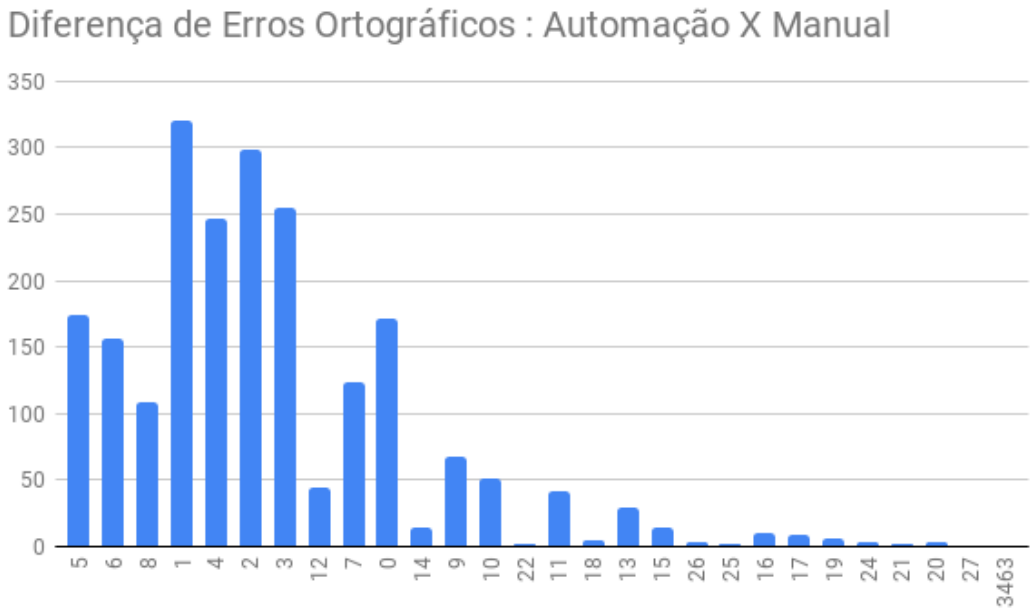

Figura 3 – Diferença entre Erros Gramaticais Encontrados pelo Algoritmo do CoGrOO X Encontrados pelos Corretores da UOL

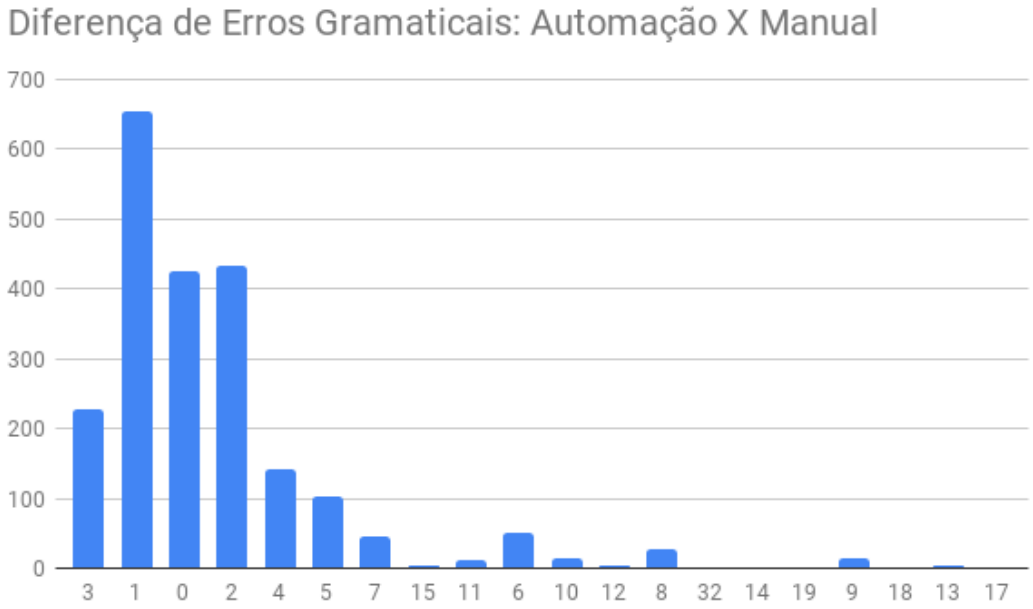

Figura 4 – Diferença entre Erros Ortográficos Encontrados pelo Algoritmo do CoGrOO X Encontrados pelos Corretores da UOL

corretores) foram usados como entrada de dados para os algoritmos de AdaBoost e XGBoost. O algoritmo gera como resultado os valores de Kappa,Acurácia, Precisão, Recall e F1-Score. Na Tabela 4 vemos os resultados da acurácia e kappa que ambos os algoritmos tiveram. O AdaBoost teve uma kappa de 0.41, para educação uma kappa de 0.3 até 0.5 é considerada um bom resultado (COHEN, 1960), já o XGBoost

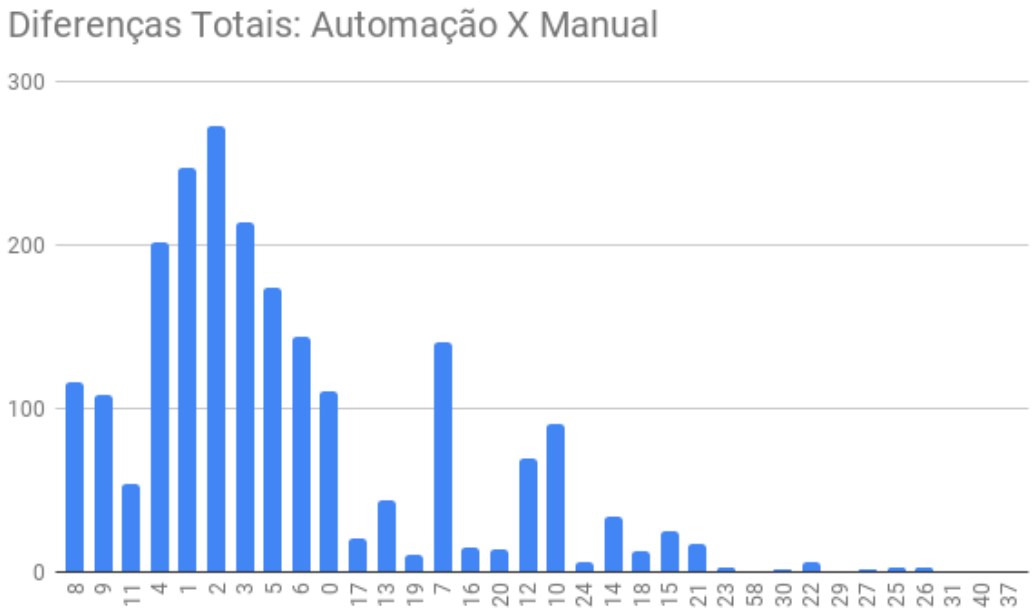

Figura 5 – Diferença total entre os erros encontrados automáticos X Encontrados por professores

teve uma kappa de 0.5, apresentando assim um desempenho melhor que o AdaBoost para classificação da nota da redação na competência 1 do ENEM.

Tabela 4 – Classificadores

Algoritmo	Acurácia	Kappa	Precisão	Recall	F1-Score
AdaBoost	0.5688	0.4111	0.44	0.48	0.45
XGBoost	0.6556	0.5160	0.73	0.67	0.69

O mesmo experimento foi feito retirando a nota geral da redação e os resultados obtidos de kappa não foram satisfatórios em ambos os algoritmos. Esses resultados podem ser observados na Tabela 5. A kappa do Adaboost e XGboost tiveram um desempenho de 0.02. Isso significa que o algoritmo esta escolhendo um classe aleatória para os resultados, classes essas que não correspondem a classe verdadeira.

Tabela 5 – Classificadores - Avaliador2

Algoritmo	Acurácia	Kappa	Precisão	Recall	F1-Score
AdaBoost	0.3775	0.0263	0.28	0.24	0.24
XGBoost	0.3852	0.0210	0.34	0.25	0.24

6 Considerações Finais

O processo de identificação de erros gramaticais e ortográficos em redações requer do aluno a análise e o estudo cuidadoso de várias regras da língua portuguesa para tentar inferir em qual boa a sua redação está e quais pontos ele precisa melhorar. Por causa disso, este artigo tendo como objetivo auxiliar o aluno no seu processo de aprendizagem propôs um desenvolvimento de ferramentas capazes de identificar automaticamente erros gramaticais e ortográficos em redações. Com base nos resultados obtidos nesta pesquisa, concluímos que a atualização do algoritmo do CoGrOO para identificação de erros gramaticais obteve um bom desempenho, correspondendo a 80.4% de acerto em um intervalo de 0 a 3 erros. E o algoritmo de identificação de erros ortográficos um percentual de 48,3% , com a mesma diferença de 0 a 3 erros em comparação com a quantidade de erros encontradas pelos professores do banco de dados da UOL. Utilizando os dois algoritmos juntos tivemos um desempenho total de 56,3% considerando erros na faixa de 0 a 5. Com base nesses dados, ambos os algoritmos podem ser utilizados pelos estudantes para identificação dos seus erros a fim de melhorar a sua escrita formal e pelos professores para ajudar na correção de redações.

A análise dos algoritmos de AdaBoost e XBoost mostrou que para classificação da nota correspondente a competência 1 da Redação do ENEM levando em consideração apenas os erros gramaticais e ortográficos não é suficiente para mensurar uma nota. Isso foi provado através das medidas de kappa que ambos os algoritmos tiveram igual a 0.02. No entanto, quando a base de dados recebe como entrada a nota geral da Redação, os algoritmos AdaBoost e XGBoost tiveram um desempenho melhor, possuindo uma kappa de 0.4 e 0.5 respectivamente, resultado esse considerado satisfatório quando se trata de educação. Portanto, o algoritmo pode ser usado como terceiro avaliador de uma redação, sendo utilizado para inferir na nota da competência 1 quando existir uma divergência de notas entre avaliadores. Como limitação da pesquisa, não tivemos acesso a uma base de dados grande e totalmente atualizada que contenha as todas as palavras da língua portuguesa, mas como solução temporária, a base de dados foi feita com a junção de quatro bases de dados que representam um dicionário da língua portuguesa com a finalidade de garantir a assertividade do algoritmo. Essa limitação impacta a base de dados que o algoritmo de Processamento de linguagem Natural usa para realização a identificação dos erros de ortográficos. Além disso, a utilização de mais redações de outras fontes de dados e diferentes avaliadores podem contribuir para um melhor treinamento da base de dados.

Portanto, a aplicação de técnicas de Processamento de Linguagem Natural uti-

lizando a língua portuguesa pode ter bons resultados para Educação sendo bastante útil no processo de aprendizagem se tornando uma boa ferramenta de estudo para alunos de ensino presencial e à distância.

Referências

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 785–794. ISBN 9781450342322. Disponível em: <https://doi.org/10.1145/2939672.2939785>. Citado na página 10.

COGROO, C. G. acoplável ao L. *CoGrOO - Corretor Gramatical acoplável ao LibreOffice*. 2014. <http://cogroo.sourceforge.net/>. Accessed on 2020-08-27. Citado 2 vezes nas páginas 7 e 10.

COHEN, J. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, Sage Publications Sage CA: Thousand Oaks, CA, v. 20, n. 1, p. 37–46, 1960. Citado na página 18.

CROSSLEY, S. et al. Pssst... textual features... there is more to automatic essay scoring than just you! In: *Proceedings of the Fifth International Conference on Learning Analytics And Knowledge*. New York, NY, USA: Association for Computing Machinery, 2015. (LAK '15), p. 203–207. ISBN 9781450334174. Disponível em: <https://doi.org/10.1145/2723576.2723595>. Citado na página 12.

FALCãO, S. B. *Linguagem da internet: do virtual para o não-virtual*. 2020. <http://www.fsma.edu.br/esfera/Artigos/Artigo_Sabrina.pdf>. Accessed on 2020-08-05. Citado na página 7.

INEP. *REDAÇÃO NO ENEM 2016: CARTILHA DO PARTICIPANTE. CARTILHA DO PARTICIPANTE*. 2017. <http://download.inep.gov.br/educacao_basica/enem/guia_participante/2016/manual_de_redacao_do_enem_2016.pdf>. Accessed on 2017-01-27. Citado na página 7.

INEP. *ENEM REDAÇÕES 2019: MATERIAL DE LEITURA*. 2019. <http://http://download.inep.gov.br/educacao_basica/enem/downloads/2020/Competencia_1.pdf>. Accessed on 2020-08-27. Citado na página 9.

ISRAEL, R.; TETREAULT, J.; CHODOROW, M. Correcting comma errors in learner essays, and restoring commas in newswire text. In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. USA: Association for Computational Linguistics, 2012. (NAACL HLT '12), p. 284–294. ISBN 9781937284206. Citado 2 vezes nas páginas 11 e 12.

KOMESU, F.; TENANI, L. *O internetês na escola*. [S.l.]: Cortez Editora, 2015. Citado na página 7.

LEITE, R. *Novas Palavras: literatura,gramática,redação e leitura*. [S.l.: s.n.], 1997. Citado na página 13.

LIDDY. Natural language processing. *Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker*, 2001. Citado na página 9.

Mishu, S. Z.; Rafiuddin, S. M. Performance analysis of supervised machine learning algorithms for text classification. In: *2016 19th International Conference on Computer and Information Technology (ICCIT)*. [S.l.: s.n.], 2016. p. 409–413. Citado 2 vezes nas páginas 11 e 12.

NOBRE, J. C. S.; PELLEGRINO, S. R. M. Avaliador automático de coesão textual em redação dissertativa – avac. *Monografia (Especialização) - Curso de Ciência da Computação, Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica (ita)*, 2010. Citado 3 vezes nas páginas 7, 11 e 12.

RASJID, Z. E.; SETIAWAN, R. Performance comparison and optimization of text document classification using k-nn and naïve bayes classification techniques. *2ND INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE AND COMPUTATIONAL INTELLIGENCE*, 2017. Citado na página 12.

ROLIM, V. B. MÉtodo supervisionado para identificaÇÃo de dÚvidas em postagens de fÓruns educacionais. *Monografia (Graduação) - Curso de Ciência da Computação da Universidade Federal Rural de Pernambuco*, 2016. Citado na página 10.

TH. PARK DC., W. D. J. T. M. S. K. Multi-class classifier-based adaboost algorithm. *Springer, Berlin, Heidelberg*, Springer, Berlin, v. 7202, n. 1, ago. 2012. Disponível em: <https://doi.org/10.1007/978-3-642-31919-8_16>. Citado na página 10.

Thanh, V. D. et al. Text classification based on semi-supervised learning. In: *2013 International Conference on Soft Computing and Pattern Recognition (SoCPaR)*. [S.l.: s.n.], 2013. p. 232–236. Citado 2 vezes nas páginas 11 e 12.

Wang, H. et al. A new linguistic feature for automated essay scoring. In: *2010 4th International Universal Communication Symposium*. [S.l.: s.n.], 2010. p. 340–344. Citado 2 vezes nas páginas 11 e 12.