Identificação de padrões de comportamento em redes sociais através de mineração de dados: Cyberbullying no Twitter

Marcelo Padilha Fontes de Barros*

Resumo

O cyberbullying nas redes sociais é nocivo para crianças e adolescentes que as utilizam. O assédio virtual pode causar danos psicológicos a eles. Infelizmente as redes sociais atualmente não contam com um método automático e eficiente de bloqueio de mensagens de cyberbullying para proteger seus usuários. Conjuntamente, a descrição atual do que seria o cyberbullying em redes sociais é confusa, e isso prejudica a criação de um software que detecte os padrões do assédio virtual com eficiência. Esse projeto define mais precisamente as características do cyberbullying, aperfeiçoa um processamento de linguagem natural resistente a ruído de redes sociais, e cria uma estratégia de detecção por mineração de dados desse comportamento na rede Twitter. Por fim, esse projeto também busca encontrar novos padrões ainda não delineados que ajudem na otimização da mineração de dados para detecção de cyberbullying.

Palavras-chave
Padrões de Comportamento, Cyberbullying, Redes Sociais, Mineração de dados.

Identifying patterns of behavior in social networks through data mining: Cyberbullying on Twitter

Abstract

Cyberbullying on social networks is harmful to children and teenagers who use them. Virtual harassment can cause psychological harm to them. Unfortunately, social networks currently do not have an automatic and efficient method of blocking cyberbullying messages to protect their users. Taken together, the current description of what cyberbullying in social networks would be is confusing, and this undermines the creation of software that efficiently detects patterns of cyber harassment. This project more precisely defines the characteristics of cyberbullying, improves noise-resistant natural language processing of social networks, and creates a data mining detection strategy for this behavior on the Twitter network. Finally, this project also seeks to find new standards not yet outlined that help optimize data mining for detecting cyberbullying.

Keywords
Behavior Patterns, Cyberbullying, Social Networks, Data Mining

I. Introdução

Bullying é uma violência que pode começar de maneira não intencional e que resulta na vitimização de um jovem que sofre maus tratos sistemáticos por um agressor e reforçadores desta agressão [2].

Segundo Patchin e Hinduja (2006) em seu trabalho Bullies Move Beyond The Schoolyard (Bullies Movem-se Além do Pátio da Escola), o cyberbullying difere do bullying tradicional principalmente pelo alcance dos infratores. Cyberbullies são capazes de estender o bullying além dos terrenos da escola e seguir os alvos em suas casas, e completam: Nós definimos o cyberbullying como dano intencional e repetitivo infligido por meio de texto eletrônico [4]. Porém, o padrão repetitivo do ataque é o mesmo do bullying clássico. Se tais mensagens forem enviadas várias vezes pela mesma pessoa para a mesma vítima, o processo é chamado Cyberbullying [18].

Através de mensagens de texto publicadas em ambientes virtuais frequentados pelos jovens, o bullying se estendeu. Uma vez que estas plataformas permitem uma troca de conteúdo não filtrada e por vezes anônima, novos problemas surgem também [18]. O assediador agora pode humilhar e agredir sua vítima por outro meio mais difuso onde há uma plateia para assisti-lo, e mesmo assim manter sua anonimidade após o ataque. Os efeitos negativos inerentes ao cyberbullying, no entanto, não são leves ou triviais e têm o potencial de causar sérios danos psicológicos, emocionais ou sociais [4].

O combate ao cyberbullying é feito individualmente através da vítima alertando um moderador, embora uma alta fração de vítimas se isola e não relata tais casos [18]. A resposta para o problema seria a utilização de um software capaz de analisar mensagens e extrair o conteúdo, confirmar o ataque, e identificar o agressor antes que esse cause mais danos. Tal aplicação não pode apenas conter algoritmos de condicionais,

ela deve possuir uma inteligência capaz de compreender o contexto do que está sendo analisado. O aprendizado de máquina através da mineração de dados é usado no processamento de quantidades massivas de dados não estruturados [3]. No entanto, os estudos de análise de perfil de usuário disponíveis atualmente são quase que totalmente direcionados à identificação de perfil de clientes para empresas. Estudos sobre detecção automática de cyberbullying são poucos e tipicamente limitados aos comentários individuais e não levam em conta o contexto [6]. Faz-se necessário um estudo sobre o tema para a criação de uma forma eficiente de detectar ataques de cyberbullying.

A extração dos dados utilizando a API do Twitter foi realizada através da biblioteca Tweepy [16]. Os dados foram armazenados utilizando o MongoDB com sua estrutura Big Data, que facilita a manipulação de grandes quantidades de informação.

A mineração de dados é um processo de descoberta de padrões em uma base [3]. Utilizando esse processo em textos de redes sociais pode-se detectar as características que correspondem ao teor abusivo, e com um algoritmo rastrear ataques repetitivos característicos do cyberbullying. O Twitter possui uma sintaxe específica devido aos textos de extensão limitada 280 (duzentos e oitenta) caracteres, a mineração de dados do Twitter precisa levar em conta essa característica para trabalhar o ruído textual [10]. Por isso, o *Word Embedding* é utilizado em conjunto com a Rede Neural como método de processamento de linguagem natural [3]. A característica de multidimensionalidade de palavras do WE trata qualquer simbologia semântica como palavra válida [10]. Usa-se uma base de treinamento rotulada para a criação do modelo de classificação.

É necessário um estudo sobre a extração de dados do Twitter, o armazenamento deles, e principalmente os métodos de mineração desses dados para a detecção de ataques de cyberbullying. Este projeto encontra informações que ajudam na descrição de tais ataques para ajudar outros projetos de detecção no futuro. Esse trabalho também propõe um estudo de caso sobre o método de detecção de cyberbullying em redes sociais através de mineração de dados.

O trabalho tem como objetivo detectar cyberbullying na rede social, Twitter. Os métodos utilizados para a detecção de cyberbullying foram Big Data, *Word Embendding*, Rede Neural e Algoritmo de Confirmação.

II. Material e Métodos

A. *Extração e Armazenamento*

O corpus utilizado no projeto foi 68.483 tweets retirados de buscas avançadas de 343 hashtags pré-selecionadas. As hashtags representam contas de políticos, celebridades, youtubers, militantes de minorias e líderes religiosos brasileiros que são os alvos dos tweets coletados. A escolha dessas contas utilizadas como alvo de tweets foi realizada manualmente, pois a fonte de informação de quais seriam os indivíduos e instituições alvos de ataques e mensagens negativas está espalhada por inúmeros sites de entretenimento e notícias como Terra, Exame, Folha, Veja, UOL e por vezes o próprio Twitter. Todos os dados extraídos são na língua portuguesa.

Para o armazenamento e a manipulação dessa enorme quantidade de informações, é usado o banco NoSQL de Big Data orientado a documento MongoDB, que utiliza o formato semiestruturado (meta-data) de dados para facilitar nas buscas de mensagens dentro do campo valor e tornar o trabalho de coleta de dados mais simples (MongoDB manual)

Os campos do documento armazenado no banco são como na figura seguinte:


```
{
        "user_id": "1197731618204659536",
        "tweet_text": "texto exemplo twitter demonstração",
        "id_tweet": "1327987918663192673",
        "created": "Fri Nov 13 22:48:44 +0000 2020",
        "target": "@alguem",
}
```


Fig. 1: Estrutura de armazenamento

Cada documento contém “user_id” para a identificação do usuário criador do tweet. O documento também possui o campo “tweet_text” contendo o texto do tweet, a campo “id_tweet” com a identidade única desse tweet, a data de criação em “created”, e a hashtag no campo “target” guardando a pesquisa que trouxe o tweet em questão. Esse campo guarda a informação para quem o tweet foi direcionado.

Para a preparação inicial dos textos, os documentos no banco de dados têm removido palavras conectivas e caracteres especiais irrelevantes ao vocabulário português. Foi utilizado o vocabulário de stopwords da língua portuguesa do NLTK [8].

B. *Processamento de linguagem natural e classificação*

No treinamento do classificador de aprendizado profundo (word embedding em conjunto com rede neural) utiliza-se o dataset rotulado de tweets para análise de sentimento disponível em kaggle.com, sobre licença CC BY-NC-AS 4.0, e desenvolvido seguindo o método de emoticons positivos e negativos usado por Go et al. (2009) [1]. O dataset contém quinhentos mil tweets na língua portuguesa rotulados como positivos de valor 1 e negativos de valor 0. Noventa por cento dos tweets do dataset são utilizados para treino e dez por cento para teste. Todos os tweets de treino e teste também têm palavras conectivas e caracteres especiais removidos.

O método de mineração de dados escolhido é um conjunto de *Word Embedding* com Rede Neural, onde o *Word Embedding* atua como uma camada inicial em uma Rede Neural de alta densidade (onde todos os neurônios da camada anterior se conectam com todos os neurônios da camada seguinte) [3]. Assim o classificador é uma rede de aprendizado profundo com 5 camadas. Essa estrutura foi escolhida conforme a análise de desempenho de Lebret e Collobert em *Word Embeddings*through Hellinger PCA (2017).

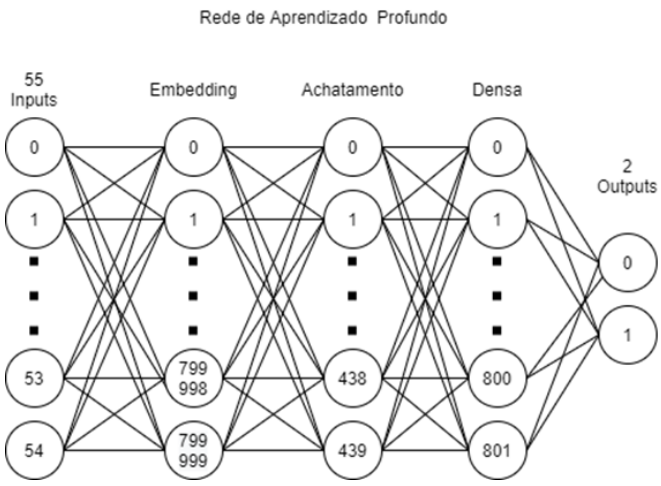

Fig. 2: Rede de Classificação

1) Camada de input: Com entrada de 55 indexes de palavras por texto.
2) Camada embedding: Configurada para 8 dimensões de coordenadas e 100.000 palavras de dicionário, e ajustada com retro propagação.
3) Camada de achatamento: Com a função de transformar o output matricial 2D de 55 x 8 posições do WE para input 1D de 440 posições lineares da Rede Neural.
4) Camada densa: Rede Neural responsável pela abstração da informação.
5) Camada de output dual: Com 0 para texto de sentimento negativo e 1 para o texto de sentimento positivo.

O classificador trabalha apenas com números, então as palavras dos textos de input precisaram ser indexadas e o index delas é quem alimenta o classificador.

O treinamento é realizado alimentando o classificador com o dataset rotulado já devidamente indexado, e após o treinamento da rede de aprendizado profundo, todos os tweets coletados são classificados pelo modelo como positivo ou negativo, e, então, estão prontos para o passo de confirmação de cyberbullying.

C. Confirmação de Cyberbullying

Após a classificação, os tweets agora rotulados como negativos são utilizados na etapa da confirmação do cyberbullying. É realizada com uma nova extração de tweets da base de dados do Twitter. Dessa vez, a busca é realizada não com hashtags, mas com "user_id" do usuário criador de cada tweet classificado como negativo. Essa nova busca retorna os últimos 100 tweets de cada usuário ou menos, caso o usuário tenha menos de 100 tweets na sua linha do tempo.

Os tweets de cada usuário são primeiramente checados se possuem no seu corpo de texto a hashtag do alvo inicial do tweet classificado como negativo. Cada confirmação de novo tweet endereçado para a hashtag alvo envia-o para a remoção de palavras conectivas e caracteres especiais, indexação, em seguida para o classificador. Se o tweet é classificado como negativo, ele soma 1 à contagem da pontuação do usuário para confirmação de cyberbullying. Usuários com pontuação maior ou igual a 10 são armazenados no banco de dados seguindo a seguinte estrutura:

```
{
        user_id: "8054035536035505809"
        target: "@alguém"
        score: 10
        tweets: ["mensagem com teor negativo 1",
                "mensagem com teor negativo 2",
                "mensagem com teor negativo 3",
                "mensagem com teor negativo 4",
                "mensagem com teor negativo 5",
                "mensagem com teor negativo 6",
                "mensagem com teor negativo 7",
                "mensagem com teor negativo 8",
                "mensagem com teor negativo 9",
                "mensagem com teor negativo 10"]
}
```

Fig. 3: Estrutura de confirmação

São guardados o "user_id" para identificação do usuário, a hashtag do alvo do cyberbullying no campo "target", a pontuação do usuário com mensagens negativas em "score", e uma lista contendo cada um dos tweets classificados como negativo para o alvo do ataque.

III. RESULTADOS E DISCUSSÃO

O Twitter limita buscas em sua base de dados a apenas 450 solicitações a cada 15 minutos [17]. A extração inicial dos tweets analisados no projeto foi lenta. Teve que ser implementada uma extração dividida em sessões. O programa captura a mensagem de erro de acesso de busca, e imprime na tela a hashtag sendo buscada, quantas mensagens foram extraídas dessa hashtag, e a hora da interrupção. Então, a extração é retomada após passados 15 minutos do bloqueio. Isso torna o processo de extração de dados lento e difícil de ser automatizado, mas não impossível de ser realizado.

```
def get_tweets(name, _error_mark):
    today = date.today()
    last_week = today - timedelta(days=7)
    count_tweets = 0
    try:
        for tweet in tweepy.Cursor(api.search, q=name+"-filter:retweets", lang="pt",
                                   since=last_week, tweet_mode='extended').items(300):
            count_tweets =+ 1
            tweet_data = {
                'name': noemoji.d_e(tweet._json['user']['name']),
                'user_id': tweet._json['user']['id_str'],
                'tweet_text': noemoji.d_e(tweet._json['full_text']),
                'id_tweet': tweet._json['id_str'],
                'created': tweet._json['created_at'],
                'target': name
            }
            try:
                twitter.insert_one(tweet_data)
            except:
                print('Database connection error')
        _error_mark = False
        return _error_mark
    except:
        print(name + " " + str(count_tweets))
        _error_mark = True
        return _error_mark
```

Fig. 4: Código extração

O armazenamento e manipulação de dados utilizando um modelo Big Data do MongoDB orientado a documentos,

permitiu a manipulação de grandes quantidades de informações sem sobrecarregar a memória, isso, graças a utilização de cursor. A conversão do formato de meta-dados BSON do banco de dados para o formato DataFrame utilizado no treinamento e classificação é simples e direto utilizando a biblioteca Pandas [9].

Os tweets possuem textos curtos pela sua limitação de caracteres, assim força a criação de reducionismos. Há muito ruído de caracteres estranhos, como, emojis e emoticons. Tais fatores tornam suas características léxicas e semânticas únicas e diferentes de outros tipos de texto LI et al [10]. Portanto, uma forma de lidar com todo esse ruído fez-se necessária. O *Word Embendding* (WE) [20] trabalha com vetorização multidimensional posicionando palavras com significado semelhante em coordenadas próximas entre elas. Isso permite o uso de reduções ou até emojis como vocabulário válido, já que esses sempre estão agrupados juntos às palavras que representam seu sentido semântico. Esse aspecto é adicionado a uma Rede Neural (RN) para ampliar a busca de contexto a todo o texto do Twitter. O WE foi transformado em uma camada adicionada a RN formando então, uma rede neural de aprendizado profundo.

```
emb_model = models.Sequential()
emb_model.add(layers.Embedding(NUM_WORDS, 8, \
    input_length=MAX_LEN))
emb_model.add(layers.Flatten())
emb_model.add(layers.Dense(2, activation='softmax'))
emb_model.summary()

emb_model.compile(optimizer='rmsprop', \
    loss='categorical_crossentropy', metrics=['accuracy'])
emb_model_result = emb_model.fit(X_train_emb, y_train_emb, \
    epochs=NUM_EPOCHS, batch_size=BATCH_SIZE, \
        validation_data=(X_valid_emb, y_valid_emb), verbose=2)
emb_model.save('model_saved')
```

Fig. 5: Código rede aprendizado profundo

A biblioteca Keras que utiliza o Tensorflow [5] permite criar uma camada de *Word Embeddings* como entrada para uma rede neural com alguns ajustes, como tokenização que transforma palavras em um index de dicionário, e normalização de textos, que consiste em ajustar um tamanho fixo para input na rede neural. Segundo Bochkarev [19], a média de palavras por texto no alfabeto latino ou cirílico é o número de caracteres dividido por 5.1. O Twitter refere a uma média de 55 palavras dentro do seu escopo fixo de 280 caracteres. Textos curtos foram estendidos com zeros, e excedentes formas truncados. Foi testado e definido, também, que o dicionário teria 100 mil palavras. Esse número é maior do que é frequentemente usado pela necessidade de acomodar os reducionismos e dialetos comuns em redes sociais. Dicionários menores utilizados nesse experimento mostraram uma perda de precisão. O tamanho reduzido acaba descartando as palavras menos utilizadas nos textos, mas que ainda são importantes para a análise.

Lebret demonstrou em seu artigo *Word Embeddings through Hellinger PCA, 2017* [14] que bons WE podem ser obtidos usando operações lineares de aprendizado. A camada embedding do Tensorflow utiliza apenas um processo de otimização por função de perda em uma matriz simples. A figura abaixo mostra como seria a estrutura de output da camada embedding.

1	1,53	-3,20	-7,93	-3,66	7,12	1,99	3,06	4,13
2	1,73	4.23	0.87	2,73	2,73	4.24	0.88	3,73
3	2.03	-2,20	-6,93	2.04	-1,66	-6,39	2.05	-1,12
4	-6,97	4.24	-6,93	9,12	-6,05	4.25	0.89	-5,97

1 = árvore 2 =você 3 = hoje 4 = não

Fig. 6: Matriz embedding

Na matriz da figura acima, cada linha representa uma palavra, e cada uma das oito colunas contém a coordenada da palavra no eixo de cada dimensão. A distância entre as palavras é ajustada gradualmente com retro propagação utilizando uma função de perda [15]. No caso desse trabalho, foi utilizada a Função de Entropia Cruzada.

$$H(T,q) = \sum_{i=1}^{N} \frac{1}{N} log_2 q(x_i) \quad (1)$$

Conforme Lebret e Collobert explicam em seu trabalho Word Embeddings through Hellinger PCA (2017), funções geométricas simples para WE tem desempenho tão bom quanto outras técnicas mais complexas de aprendizado quando usadas em conjunto com outro classificador, no caso a rede neural.

Uma camada de achatamento foi necessária porque a saída do WE é uma matriz bidimensional 55x8, e a RN recebe apenas dados arranjados em uma lista unidimensional de 440 posições.

O treinamento foi realizado com uma base de dados da mesma fonte que os textos que são classificados pelo modelo, mensagens do Twitter. A intenção é também reduzir os efeitos de *overfitting*. Uma vez que, o escopo desse projeto é apenas o ambiente do Twitter, um overfitting para esse conjunto de dados deixa de ser um problema.

Redes neurais são ferramentas poderosas de aprendizado de máquina, porém são caixas pretas. Suas soluções são pesos em conexões entre neurônios, sendo difícil de proporcionar explicações detalhadas sobre a solução do problema [7]. Mesmo assim, sua eficiência supera sua limitação neste projeto.

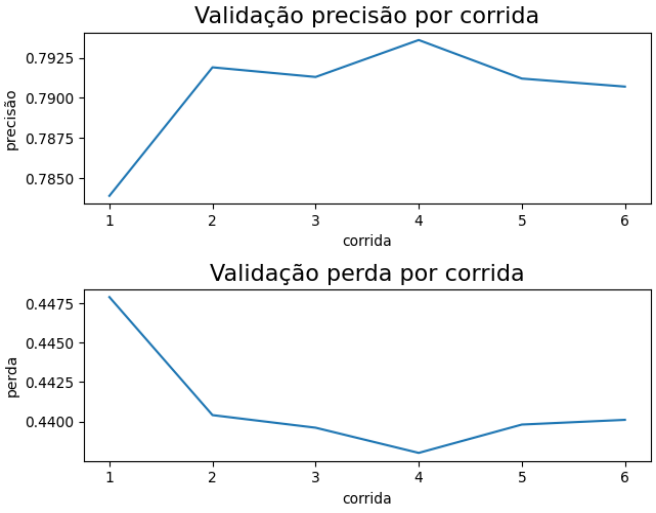

Fig. 7: Precisão e perda

A figura 6 mostra que o ponto máximo de precisão e menor perda é na quarta corrida ou geração. O *overfitting* iniciou-se a partir da quinta geração. *Overfitting* é falha em

generalização, quando um modelo funciona com um conjunto de dados, mas não com outro [7], ou seja, o modelo está tão otimizado para um determinado conjunto de dados, que passa a falhar quando dados um pouco fora dos padrões de treino são utilizados. Para mantermos nossa melhor performance devemos parar o treinamento quando esse alcança seu valor de precisão máximo, no caso a quarta corrida. Essa técnica é conhecida como *early stopping* [3].

Outra técnica utilizada, dessa vez para evitar um erro estatístico no processo de treinamento, é a *10-fold cross-validation*. Foi utilizada a biblioteca StratifiedKFold do Scikit-learn [13].

```
skf = StratifiedKFold(n_splits=NUM_FOLDS, shuffle=True)
skf.get_n_splits(df.tweet_text, df.sentiment)

for train_index, test_index in skf.split(df.tweet_text, df.sentiment):
    X_train, X_test = df.tweet_text[train_index], df.tweet_text[test_index]
    y_train, y_test = df.sentiment[train_index], df.sentiment[test_index]
```

Fig . 8: Cross-validation code

Nesse procedimento a base de treinamento é dividida em dez partes proporcionais, cada uma delas montada a partir de documentos aleatórios da base. O teste é executado com cada uma das partes, no total de dez sequências. A precisão do modelo é uma média obtida da soma das precisões das sequências de testes [3].

Sequências	1	2	3	4	5	6	7	8	9	10	Média
Precisão	0,7929	0,7943	0,7976	0,7908	0,7937	0,7968	0,7959	0,7926	0,7873	0,7931	0,7935

Fig. 9: 10-fold cross-validation

O resultado do modelo desse projeto possui uma precisão de 0,7935.

Mas, apenas a avaliação de mensagens com teor negativo de um usuário não é comprovação de ato de cyberbullying. Porém, se tais mensagens forem enviadas várias vezes pela mesma pessoa para a mesma vítima, o processo é chamado Cyberbullying [18].

Cada documento, agora avaliado como contendo mensagem de teor negativo, é utilizado para uma nova pesquisa na base de dados do Twitter. O campo de "user_id" é usado como o parâmetro de uma nova busca de tweets do usuário. Não há o campo com nome do usuário devido a política de uso dos dados do Twitter, o qual não permite a exposição dos dados pessoais de usuários por quem utiliza sua base de dados. Outro campo do registro utilizado é "target", que possui a hashtag do alvo do tweet inicial. Tendo encontrado repetição de mensagens àquele alvo, todas elas são classificadas pelo modelo já treinado. Se houver um total maior do que dez mensagens de teor negativo direcionadas ao mesmo usuário, o cyberbullying está confirmado.

IV. Conclusões

É realmente possível utilizar processos de mineração de dados para encontrar usuários que praticam cyberbullying em redes sociais, como foi provado nesse projeto. Para isso, deve-se levar em consideração trabalhos que descrevem as características próprias do cyberbullying de repetição de ataque de um usuário a outro, ou um grupo específico. Um modelo que não aborda a diferença entre texto abusivo e cyberbullying, não realiza um bom trabalho.

Processos de treinamento não assistidos são ineficientes no ambiente textual complexo de redes sociais. Há muitos caminhos de contexto que podem ser tomados que não são relevantes para o trabalho. Uma base de dados rotulada para treinamento é essencial para desenvolver um modelo eficaz.

Inicialmente, foi tentada uma técnica de aprendizado de máquina com treinamento não assistido de agrupamento utilizando K-means. Esta técnica consiste em criar centroides e deixar que o computador encontre a semelhança dos dados analisados calculando a distância euclidiana entre esses dados e os centroides [3]. Embora essa técnica seja utilizada quando se possui uma grande quantidade de dados não rotulados para serem analisados, ela mostrou-se vaga. Não há como validar a taxa de precisão do método sem uma base de dados rotulado, e, por isso, essa técnica de aprendizado não assistido foi abandonada pela técnica de aprendizado assistido, mais apropriada para esse projeto.

Outra técnica pretendida, essa de aprendizado assistido, foi o Classificador de Árvore de Decisão. Essa técnica atrai pela simplicidade do seu uso e fácil visualização. Os vetores criados no *Word Embedding* não precisam ser trabalhados para utilizar esse processo de mineração [12]. No entanto, foi observado em testes que sua taxa de precisão ficou abaixo de sessenta por cento, muito baixo para o nível de confiança exigido para esse projeto. Conforme o site de documentação do Scikit [12] deixa claro, árvores de decisão podem criar estruturas muito complexas que não generalizam bem os dados, e pequenas variações na base resultam em uma árvore totalmente diferente. Como o ambiente de textos de Twitter possui características léxicas e semânticas inerentes, essa técnica também foi descartada.

O uso do *Word Embedding* se mostrou crucial para permitir o trabalho com textos complexos, como os de Twitter. Mesmo usando apenas operações lineares simples, seu uso em conjunto rede neural faz o conjunto ter uma eficiência maior que a soma com uma das partes [11].

O processo de classificação pode ser aprimorado com o uso de um dataset de treinamento rotulado em uma faixa de espectro de sentimento, ao invés de binário 0 ou 1. Assim, poderia ser ainda mais preciso no tipo de texto que se quer classificar.

Futuros trabalhos utilizando essa solução poderiam analisar o desempenho no uso de outros tipos de modelos de aprendizado de *Word Embedding* como o Skip-Gram, Bag-of-Words ou CBOW. Também poderia ser analisada a eficiência de outros vários tipos de redes neurais na classificação em conjunto com WE.

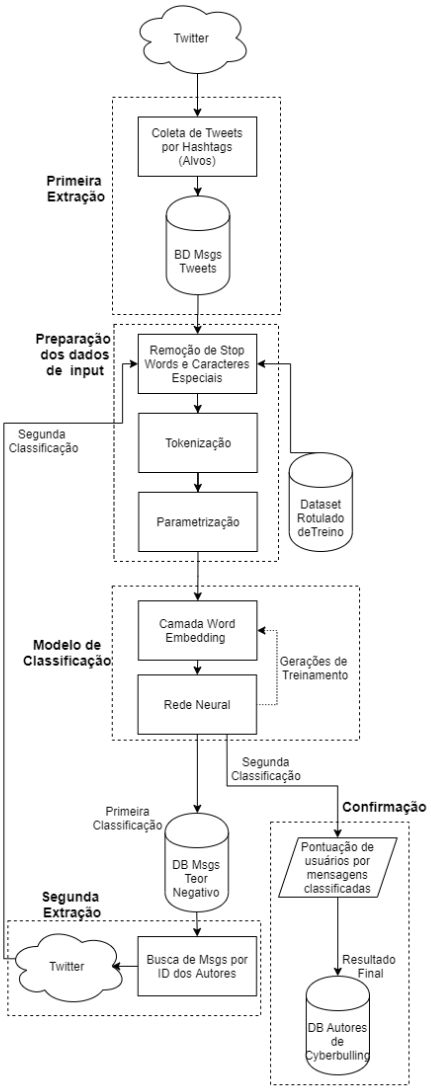

Fig. 10: Diagrama do projeto (do autor)

V. BIBLIOGRAFIA

[1] Alec Go, Richa Bhayani and Lei Huang, "Twitter Sentiment Classification using Distant Supervision." 2008.

[2] Guilherme Welter Wendt, Débora Martins de Campos e Carolina Saraiva de Macedo Lisboa, "Agressão entre pares e vitimização no contexto escolar: bullying, cyberbullying e os desafios para a educação contemporânea. " Universidade do Vale do Rio dos Sinos, 2010.

[3] Ian H. Witten and Eibe Frank, "Data Mining: Practical Machine Learning Tools and Techniques." 2ª Edição, 2005.

[4] Justin W. Patchin and Sammer Hinduja, "Bullies Move Beyond the Schoolyard A Preliminary Look at Cyberbullying*"*. 2006.

[5] Keras API reference. Dispinível online: <https://keras.io/api/>. Acesso em: 13 dez. 2020.

[6] Maral Dadvar, et al. "Improving cyberbullying detection with user context." *European Conference on Information Retrieval*, 2013.

[7] Michael J.A. Berry and Gordon S. Linoff, "Data Mining Techniques for Marketing, Sales, and Costumers Relationship Management", 2ª Edição, 2004.

[8] Natural Language ToolKit (NLTK). Disponível online: https://www.nltk.org/book/ch02.html. Acesso em: 14 Dez. 2020.

[9] Pandas Documentation. Disponível online: <https://pandas.pydata.org/docs/user_guide/index.html#user-guide>. Acesso em: 14 Dez. 20.

[10] Quanzhi Li, at al. "Data Sets: *Word Embeddings*Learned from Tweets and General Data." *International AAAI Conference on Web and Social Media*, 2017

[11] Rémi Lebret and Ronan Collobert, "*Word Embeddings*through Hellinger PCA." *Conference of the European Chapter of the Association for Computational Linguistics 2014,* Revisão 2017.

[12] Scikit-learn Decision Tree Regression. Disponível online: <https://scikit-learn.org/stable/modules/tree.html#tree>. Acesso em: 14 Dez. 2020

[13] sklearn.model_selection.StratifiedKFold Disponível online: <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html>

[14] Tensorflow. tf.keras.layers.Embedding. Disponível online: <https://www.tensorflow.org/api_docs/python/tf/keras/layers/Embedding.> Acesso em:14 Dez. 20.

[15] Tensorflow. tf.keras.losses. Disponível online: <https://www.tensorflow.org/api_docs/python/tf/keras/losses/categorical_crossentropy>. Acesso em: 14 dez. 20.

[16] Tweepy Documentation. Disponível online: <http://docs.tweepy.org/en/latest/>. Acesso em: 14 Dez. 2020.

[17] Twitter Developer. Rate limits: Standart v1. 1. Disponível online: <https://developer.twitter.com/en/docs/twitter-api/v1/rate-limits#:~:text=Standard%20API%20v1.&text=Please%20note%20%2D%20The%20300%20per,during%20a%203%20hour%20period>. Acesso em: 14 dez. 20.

[18] Uwe Bretschneider, Thomas Wöhner and Ralf Peters, "Detecting Online

Harassment in Social Networks." *Thirty Fifth International Conference on Information Systems*, 2014.
[19] V.V. Bochkarev, A.V. Shevlyakova and V.D. Solovyev "Average word length dynamics as indicator of cultural changes in society." *Social Evolution and History Vol. 14, Nº 2*, 2012
[20] Yoshua Bengio, at al. "Neural Probabilistic Language Model." *Journal of Machine Learning Research 3*, 2003.

Marcelo Padilha Fontes de Barros é formando no curso de bacharelado em Ciência da Computação (2020) pela Universidade de Caxias do Sul (UCS, Caxias do Sul). Atualmente é desenvolvedor de aplicações web Full-Stack em Flask.