PROTÓTIPO DE SISTEMA MODERADOR DE CONTEÚDO COM INTERAÇÕES POR DEEP LEARNING

Cícero Marcelo de Oliveira[1]; Rodrigo Guimarães de Aro[2];Vitor Manzato Darakjian[3];

Resumo
Com a popularização das redes sociais, a presença do discurso de ódio sofreu um aumento exponencial. Nesse cenário, o dia a dia de usuários se torna cada vez mais estressante. O objetivo do projeto é aplicar métodos de filtragem utilizando *deep learning* para demonstrar a existência de formas práticas de coibir o aumento do discurso de ódio. Para simular o uso da aplicação em um cenário real, foi desenvolvido um protótipo de *blog* e sistema administrativo utilizando conceitos de API aliados à linguagem Swift e seu *framework* Vapor. Os métodos de filtragem foram obtidos por meio de bibliotecas Python, e sua eficácia foi comprovada a partir de testes realizados através de informações reais coletadas de uma base de dados. Verificou-se, assim, a capacidade da ferramenta de obter resultados consistentes e satisfatórios.
Palavras-chave: Python. Discurso de ódio. Vapor.

Abstract
With the popularization of social networks, the presence of the hate speech in them has increased exponentially. In this scenario, the daily life of users becomes increasingly stressful. The objective of this project is to apply filtering methods using deep learning to demonstrate practical ways to curb the increase in hate speech.
To simulate the use of the application in a real scenario, a blog prototype and administrative system were developed using API concepts allied to the Swift language and its framework, Vapor. Filtering methods were obtained through Python libraries, and their effectiveness was proven from tests performed through real information collected from a database. The ability of the tool to obtain consistent and satisfactory results was verified.
Key-words: Python. Hate speach. Vapor.

Introdução

A presente proposta tem como justificativa o aumento no que diz respeito aos discursos de ódio nas redes sociais, o que tem se tornado cada vez mais comum no dia a dia das pessoas, ocasionando-lhes problemas de ordem considerável nos mais diversos aspectos da vida cotidiana.

O combate e a prevenção são ferramentas possíveis de serem utilizadas com o auxílio da tecnologia, especialmente quando aliado ao *deep learning* no intuito de identificação de tais comentários.

[1] Mestre em Engenharia Elétrica (área de Sistemas Inteligentes) pela Universidade Estadual Paulista (UNESP), campus Ilha Solteira/SP; professor nos cursos de Bacharelado em Sistemas de Informação da Universidade do Estado de Minas Gerais (UEMG), campus Frutal e de Ciência da Computação da Universidade Paulista (UNIP), campus São José do Rio Preto/SP. E-mail: cicero.oliveira@uemg.br.
[2] Graduado em Ciência da Computação pela Universidade Paulista (UNIP), campus São José do Rio Preto/SP. Desenvolvedor Sênior de Aplicações iOS. E-mail: rooguima@icloud.com.
[3] Graduado em Ciência da Computação pela Universidade Paulista (UNIP), campus São José do Rio Preto/SP. Desenvolvedor Full Stack Rails. E-mail: vitordarakjian@gmail.com.

Métodos de filtragem de conteúdos ofensivos podem e devem ser utilizados visando a diminuição de ocorrências dos tão comuns discursos de ódio no ambiente virtual.

Diante disso, foi desenvolvido um protótipo de sistema de comentários para que fosse aprimorada a filtragem necessária, além de a realização de testes de eficiência do sistema, o que será apresentado ao final do presente artigo.

1 Referencial Teórico

1.1 Aplicação e API

Na área de programação, a sigla *API* refere-se ao termo *"Application Programming Interface"*. Muito utilizada hoje em dia, as *APIs* facilitam o desenvolvimento e o uso de certas tecnologias durante a criação de aplicações.

Para Nijim e Pagano (2014), uma *API* pode ser descrita como contêineres de cargas levando conteúdos entre portos. Complementam ainda que uma *API* conecta os processos, serviços e informações de sua empresa com negócios de parceiros, equipes internas e desenvolvedores autônomos de forma segura e fácil.

O desenvolvimento de uma *API*, assim como qualquer aplicação, necessita de uma linguagem de programação. O Swift é uma linguagem de programação desenvolvida pela Apple e anunciada em 2014 durante a *WWDC* (Apple Worldwide Developers Conference), inicialmente focada no desenvolvimento de aplicações para iOS, macOS e watchOS. A partir da versão 2.2 da linguagem, se tornou *open source* e no dia 3 de dezembro de 2015, tanto a linguagem, quanto suas bibliotecas de suporte, o depurador e o gerenciador de pacotes foram publicados no GitHub. (Apple, 2023)

Para ser utilizado no ambiente *web*, o Swift, assim como diversas outras linguagens, requer um *framework* e atualmente, um dos *frameworks* mais populares para essa linguagem é o Vapor. Como constatado por Hussain e Crowford (2017), trata-se de tecnologias relativamente novas.

Não menos importante que o *server-side*, o *front-end*, ou *client-side*, é responsável pela conexão *API*-usuário. Lançado em fevereiro de 2014 o VueJS é um *framework* JavaScript *open source* que oferece ferramentas para o desenvolvimento de interfaces web utilizando componentes e funcionalidades como *hot-reloading* e *declarative UI*. (Vuejs, 2023)

1.2 Deep Learning

Deep learning, no conceito de LeCun *et al*. (2015), são métodos de aprendizagem profunda baseados em múltiplos níveis de representação, obtidos pela composição de módulos simples e não lineares, que transformam representações de baixo nível em representações de níveis mais altos e abstratos.

Técnicas convencionais de aprendizado de máquina são limitadas na sua habilidade de processar dados em sua forma "crua", ou seja, dados não tratados. Por décadas, desenvolver um sistema de reconhecimento de padrões ou de aprendizado de máquina necessitava de conhecimentos específicos em engenharia e um domínio considerável para desenvolver um extrator de recursos que transformasse os dados não tratados (como os *pixels* de uma imagem) em uma representação organizada que poderia ser absorvida por um subsistema que gerasse e classificasse a saída do sistema. (Lecun et al., 2015).

Ponti e da Costa (2017) explicam que as redes neurais convolucionais (do inglês *Convolutional Neural Networks* (CNN)), *Deep Belief Networks*, *Restricted Boltzmman Machines e Autoencoders* (AE) começaram a aparecer como base para métodos do estado da arte em diversas aplicações.

A utilização de redes neurais e métodos de *deep learning* já se mostrou bastante eficaz no tratamento de dados que, posteriormente, serão utilizados para solucionar problemas reais.

Com essas técnicas, pode-se solucionar um dos problemas que está cada vez mais presente nas páginas da *Internet*: uma onda de comentários tóxicos.

1.3 Abusos na Internet

A crescente onda de redes sociais está trazendo problemas que antes não eram tão explícitos. Muito discutido atualmente, o discurso de ódio que se encontra nessas redes vem se mostrando cada vez mais presente no dia a dia de seus usuários.

Com esse crescente problema, o termo "*shitstorm*" veio à tona. Segundo o dicionário Oxford (2016, on-line) da língua, designa um “evento frenético ou desastroso, uma comoção ou um tumulto”, o significado no dicionário alemão indica um fenômeno específico, no qual há uma indignação manifestada por intermédio da *Internet* no qual os usuários, geralmente, propagam comentários injuriosos (Pereira; Caldas, 2017).

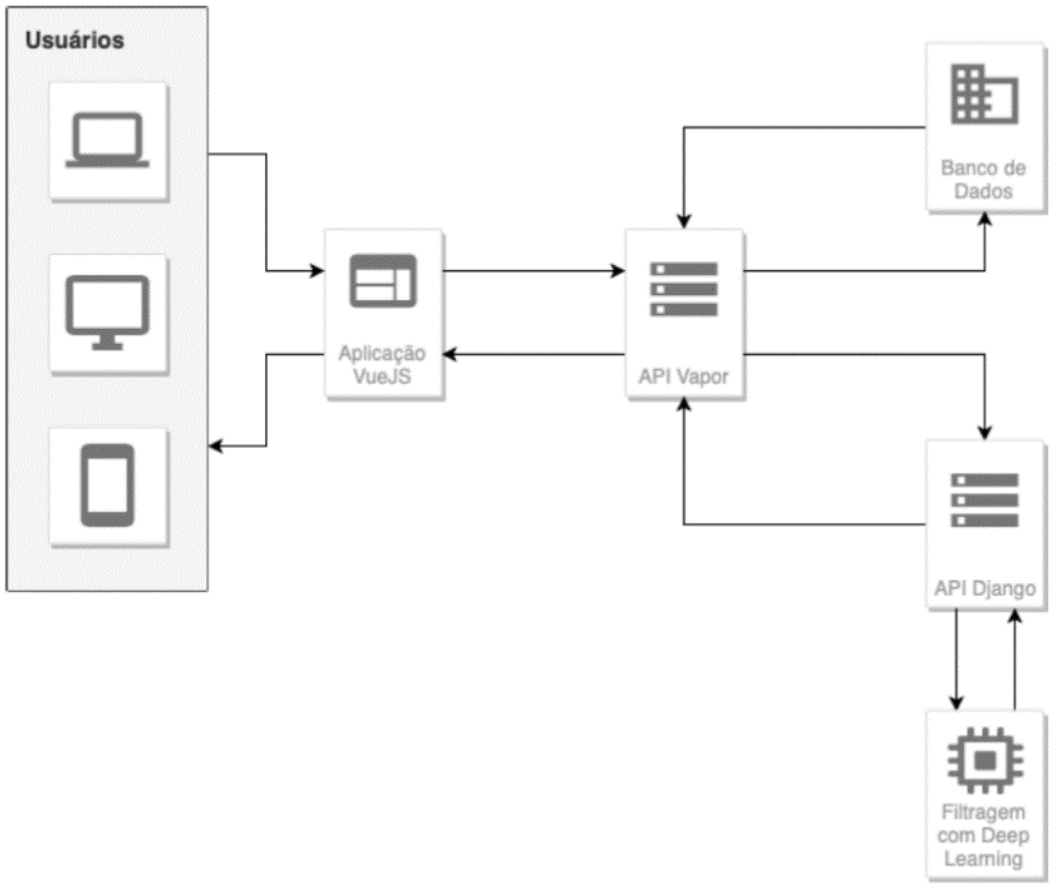
Essa relação comum nas redes sociais tem caracterizado cada vez mais humilhação desigual representada pelo *bullying*, onde agressores mostram seu poder com palavras de

intimidação e encontram nos mais fracos o alvo para a violência, a humilhação, as chacotas e as ameaças (Guerra; Klass, 2017).

2 Materiais e Métodos

No presente projeto, o foco se deu no desenvolvimento de uma plataforma de filtragem de comentários ofensivos tendo sido criada uma *API*, uma aplicação para utilização dos usuários e administradores, um banco de dados para armazenar todos os dados criados pelos usuários e uma ferramenta que, utilizando *deep learning*, ficou responsável por fazer o processo de filtragem e aprovação dos comentários. Na figura 1, se encontra detalhado separadamente o que foi necessário realizar em cada área e suas funcionalidades para que os objetivos fossem atingidos.

Figura 1 - Fluxograma da aplicação



Fonte: Dados dos autores (2023)

Considerando como foco inibir comentários ofensivos que disseminam ideias racistas e preconceituosas a qualidade e a quantidade de discussões encontradas nesses ambientes aumentaria, tornando o espaço de comentários mais atrativo para os usuários.

2.1 API

Para a criação da *API* foi escolhida a linguagem de programação Swift e o *framework* Vapor.

A escolha da Swift se deu pois é uma linguagem moderna e que oferece muito poder aos desenvolvedores, tendo pouco mais de quatro anos do seu anúncio oficial. A linguagem tem como principal foco a área de aplicações diretamente para sistemas operacionais do ambiente Apple, sendo eles o iOS, macOS, watchOS e tvOS.

O *framework* Vapor entrou em cena para auxílio na utilização da linguagem no desenvolvimento web, sendo responsável por disponibilizar toda a base para que a *API* pudesse ser desenvolvida focando diretamente no desenvolvimento das funcionalidades.

A escolha por utilizar esse *framework* frente à outros, como o Perfect, Kitura e Zewo, foi feita baseada em analises de facilidade de uso, ferramentas disponíveis e toda a comunidade.

2.2Aplicação

Para o desenvolvimento da aplicação o escolhido foi o *framework* JavaScript VueJS e seu uso está relacionado ao seu poder de desenvolvimento alinhado com sua flexibilidade.

Suas funcionalidades proporcionam velocidade e simplicidade para o desenvolvimento de aplicações. Além disso, o *framework* é bastante conhecido na comunidade e, com isso, diversas discussões e artigos podem ser encontradas na *Internet*.

A aplicação funciona de forma conjunta à *API*, recebendo seus dados e apresentando-os nas *views*.

Por sua vez, essas foram construídas utilizando as ferramentas disponibilizadas pelo *framework* e seguindo as regras contidas em seu *guideline*, o VueJS especifica convenções que, na maioria dos casos, devem ser seguidas para se construir um código limpo, reutilizável e com menor possibilidade de erros.

Para o design e estilização das páginas, foi utilizado um *template* pronto dentre os diversos disponíveis na Internet, que possibilitou a padronização das páginas da aplicação, definindo todo a CSS e classes HTML que foram utilizados para o desenvolvimento das páginas.

2.2.1 Área administrativa

A área administrativa do sistema é o local onde todo o controle da plataforma está localizado, sendo que, por meio dela é possível visualizar estatísticas e informações de usuários, artigos e comentários, sendo de acesso restrito aos usuários Administradores.

Na sequência, a descrição de cada módulo presente na área:

2.2.1.1 Usuários

Área do sistema responsável por gerenciar os usuários cadastrados.

(a) Cadastrar: Permite o cadastro de novos usuários, definindo seu nome, email, nome de usuário, senha, tipo e status.

(b) Editar: Possiblidade de editar cadastros de usuários já existentes, podendo alterar seu nome, email, nome de usuário, senha, tipo e status.

(c) Listar: Listagem de todos os usuários cadastrados no sistema, permitindo o acesso à interface de edição, exclusão e verificar quais usuários estão liberados ou não para acessar a aplicação utilizando como base o status.

(d) Excluir: Permite que seja feita a exclusão de usuários.

2.2.1.2 Artigos

Área do sistema responsável por cadastrar e gerenciar os artigos postados na plataforma.

(a) Cadastrar: Permite a criação de novos artigos, definindo seu título e corpo.

(b) Editar: Possiblidade de editar artigos cadastrados anteriormente, podendo alterar seu título e o conteúdo do corpo.

(c) Listar: Listagem de todos os artigos cadastrados no sistema, permitindo o acesso à interface de edição, exclusão, verificar a quantidade de comentários e visualiza-los.

(d) Excluir: Permite que seja feita a exclusão de artigos.

2.2.1.3 Comentários

(a) Cadastrar: Permite a listagem dos comentários com base em vários filtros, como: “Últimos comentários”; “Comentários no artigo”; “Comentários bloqueados”; “Comentários liberados”, além de possibilitar o acesso à avaliação manual de comentários e à interface de exclusão.

(b) Avaliar: Possibilita fazer a avaliação (bloqueio e desbloqueio) manual do comentário, podendo alterar o que o método de filtragem definiu.

(c) Excluir: Permite que seja feita a exclusão de comentários.

2.2.2 Área de acesso público

É a parte do sistema responsável pelo acesso de usuários “comuns”, sendo que, por meio dela, o usuário pode criar uma conta, comentar e ler os artigos publicados.

Abaixo segue listado descritivamente cada módulo presente na área:

2.2.2.1 Usuários

(a) Login: Permite que usuários que já possuam conta no sistema entrem por meio de seu email/nome de usuário e senha.

(b) Cadastrar: Permite que pessoas que acessem a área pública do sistema criem suas próprias contas.

(c) Editar: Possibilita que, após entrar, seja possível alterar dados da conta como nome, email, nome de usuário e senha.

(d) Excluir: Possibilita que após entrar, seja possível fazer a exclusão da conta.

2.2.2.2. Artigos

(a) Cadastrar: Permite a criação de novos artigos, definindo seu título e corpo.

(b) Editar: Possibilita a edição de artigos cadastrados anteriormente, podendo alterar seu título e o conteúdo do corpo.

(c) Listar: Listagem de todos os artigos criados pelo usuário logado, permitindo o acesso à interface de edição e exclusão.

(d) Excluir: Permite que seja feita a exclusão do artigo.

2.2.2.3. Comentários

(a) Cadastrar: Permite que o usuário logado no sistema faça comentários no artigo que ele estão [rever concordância verbal] lendo no momento.

(b) Listar: Listagem de todos os comentários criados pelo usuário logado, visualizando quais de seus comentários foram liberados e bloqueados, além de permitir o acesso à interface de edição e exclusão.

(c) Editar: Possiblidade de editar os comentários feitos anteriormente, podendo alterar o conteúdo do corpo.

(d) Excluir: Permite que seja feita a exclusão do comentário.

2.3 Banco de Dados

A escolha pelo SGBD PostgreSQL se dá por ser um dos mais conhecidos do mundo quando se trata de código aberto, sendo amplamente compatível com inúmeras linguagens de programação no mercado, como é o caso da Swift através do framework Vapor.

A seguir estão listados todos os campos que estão disponíveis no banco de dados da plataforma. O quadro 1 refere-se à tabela em que os dados dos usuários cadastrados estarão armazenados.

Quadro 1 - Lista Users

NOME	TIPO	OBRIGATÓRIO?	DEFAULT
Id	int	SIM	
Name	varchar	SIM	
Email	varchar	SIM	
Username	varchar	SIM	
Password	varchar	SIM	
Type	int	SIM	
Status	bool	SIM	

Fonte: Dados dos autores (2023).

O quadro 2 abaixo é responsável por apresentar as informações que serão gravadas na tabela de artigos.

Quadro 2 - Lista Articles

NOME	TIPO	OBRIGATÓRIO?	DEFAULT
id	int	SIM	
title	text	SIM	
body	text	NÃO	
date	datetime	SIM	Current datetime
idUser	int	SIM	

Fonte: Dados dos autores (2023).

O quadro 3 a seguir demonstra a tabela do banco de dados em que novos comentários serão registrados.

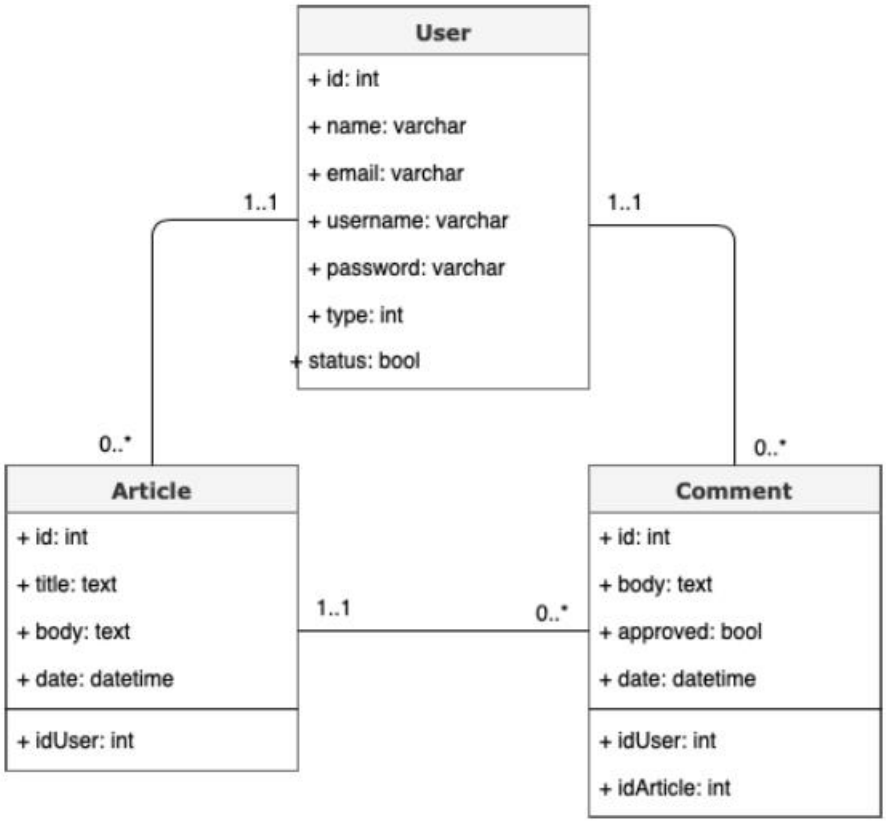
Quadro 3 - Lista Comments

NOME	TIPO	OBRIGATÓRIO?	DEFAULT
id	int	SIM	
body	text	SIM	
approved	bool	SIM	
date	datetime	SIM	Current datetime
idUser	int	SIM	
idArticle	int	SIM	

Fonte: Dados dos autores (2023).

Representado na figura 2, o diagrama de classes do banco de dados do projeto.

Figura 2 - UML - Diagrama de Classes



Fonte: Dados dos autores (2023).

Na figura acima, estão informações como o nome de cada tabela, seus atributos e o relacionamento entre cada uma.

2.4 Ferramenta de Filtragem

A ferramenta de filtragem da plataforma foi desenvolvida através de técnicas de *deep learning*, sendo técnicas que consistem em métodos de *Machine Learning* baseados em representações de aprendizado de dados.

A criação da ferramenta se constituiu da linguagem de programação Python, junto de uma de suas bibliotecas *profanity_filter*, *criada* para auxiliar a detecção e a filtragem de palavras comumente usadas para disseminar o ódio, em conjunto com essa biblioteca e a linguagem, também foram usados *Spacy models*, que consistem em modelos previamente treinados utilizando *deep learning*, e instalados no Python por meio do sistema de pacotes da linguagem.

Para que a comunicação com a ferramenta de filtragem seja possível foi necessário a criação de uma nova *API*, que assim como o restante da ferramenta foi desenvolvida em Python, porém, dessa vez aliado ao *framework* Django, que assim como o Vapor, é um conjunto de ferramentas que auxilia o desenvolvimento de plataformas web.

2.5 Ferramentas

A plataforma de versionamento de códigos utilizada foi o Git. Dentre as diversas ferramentas disponíveis que utilizam o Git como base foi selecionada a mais conhecida, o GitHub e nele, foram armazenados todo o código-fonte da *API*, da aplicação e da ferramenta de filtragem, além dos arquivos de criação do banco de dados e demais arquivos de configurações necessários.

Em conjunto com o Github foi utilizado um software de interface gráfica de git, chamado GitKraken, por meio do qual foi possível utilizar o sistema de armazenamento e versionamento oferecido pelo serviço.

Para o desenvolvimento do código foram utilizados três softwares, sendo eles: Xcode, PyCharm e Visual Studio Code. O primeiro foi utilizado no macOS para o desenvolvimento da API; o segundo foi utilizado tanto no macOS quanto no Ubuntu para o desenvolvimento da ferramenta de filtragem; e o último, utilizado no Ubuntu para a criação da API e da Aplicação, bem como no macOS para a criação da Aplicação.

Em complemento a todas as ferramentas já apresentadas, foi utilizado também o terminal, sendo que, por meio dele foi possível criar a base da API utilizando o *framework* Vapor, além de ser necessário para instalar o Swift no Ubuntu.

3 Discussão e Resultados

Para se obter um resultado mais próximo à realidade os comentários utilizados para a realização dos testes foram retirados de uma base de dados do website Reddit contendo informações reais. Os comentários foram selecionados para que a análise seja mais precisa e detalhada.

No arquivo original diversas informações como "autor", "comentário", "categoria", "pontuação" e "status" estão listadas, mas para a realização dos testes apenas a coluna "comentário" foi utilizada.

O quadro 4 a seguir é uma exemplificação da base de dados tratada que será utilizada para verificar a eficácia do método de filtragem, nele estão listados os comentários selecionados e a qualificação de cada um.

Quadro 4 - Lista - Base de comentários classificada

COMENTÁRIO	É OFENSIVO?
Amazing cinematographer one of the best to ever do it	NÃO
He was behind the podium getting his dick sucked in the police academy	SIM
You ve finally had your life changed	NÃO
How fucking bad was my fever last night	SIM
Donnie yen is the shit	SIM
What the fuck	SIM
Leo didint fake it legend	NÃO
Star wars episode viii hen ai nt dead fucker	SIM
Well that s fuck 1ng supid and par for the course i guess	SIM
Dave franco is still fucking hot	SIM
Wow that was actually a really cool read	NÃO
Excitting new poster for bloodfeast sex boobs	SIM
Holy shit do i get to do it	SIM
The finniest part was when it said allinew movie	NÃO
Am i the only one who thinks targeting a rating is bulish 1t	SIM
Based off the mother fucking 1ng awesome book also by that name please be the case	SIM

Fonte: Dados dos autores (2023).

Para efeitos de comparação foram geradas duas tabelas, uma com os resultados da filtragem utilizando *deep learning* e outra sem a utilização dele.

Todas as etapas necessárias para a obtenção dos resultados, com exceção da seleção de comentários, foram realizadas em um *"notebook jupyter"*, um arquivo da biblioteca de mesmo nome utilizado para executar linhas de código diretamente de um navegador.

A porção de código representada na figura 3 se refere aos passos utilizados para transformar a base de dados original em um novo arquivo *CSV* com as informações necessárias pela ferramenta de filtragem.

Figura 3 - Código - Tratamento de dados

```
import petl as etl

csv = etl.fromcsv('reddit_dataset_movies.csv')
csv = etl.cut(csv, 'commentary')
csv = etl.extendheader(csv, ['status',
                             'deep_filtered_comment',
                             'deep_new_status',
                             'normal_filtered_comment',
                             'normal_new_status'])

etl.tocsv(tabela, 'reddit_dataset_movies_output.csv')
```

Fonte: Dados dos autores (2023).

Com o arquivo tratado, é possível executar a ferramenta de filtragem nos dados contidos nele, para isso será utilizado um loop, onde para cada iteração uma linha será enviada aos métodos de filtragem (com e sem *deep learning*). Cada método retornará um valor booleano especificando se o comentário é tóxico ou não e o comentário, filtrado ou não. Essas

informações serão inseridas nas respectivas colunas e após a iteração de todas as linhas, um novo arquivo CSV será gerado e salvo.

Os passos citados acima estão demonstrados na figura 4 a seguir.

Figura 4 - Código – Filtragem

```
from profanity_filter import ProfanityFilter
import petl as etl

csv       = etl.fromcsv('reddit_dataset_movies_output.csv')
header    = csv[0]
dict_temp = etl.dicts(csv)
dict_list = []

deep    = ProfanityFilter(deep_analysis=True)
normal  = ProfanityFilter(deep_analysis=False)

for row in dict_temp:
  dict_list.append(row)

for row in dict_list:
    row['deep_filtered_comment'] = deep.censor(row['commentary'])
    row['deep_new_status']       = '1' if deep.is_profane(row['commentary']) else '0'

    row['normal_filtered_comment'] = normal.censor(row['commentary'])
    row['normal_new_status']       = '1' if normal.is_profane(row['commentary']) else
'0'
tabela = etl.fromdicts(dict_list, header=header)
etl.tocsv(tabela, 'profanity_filter_output.csv')
```

Fonte: Dados dos autores (2023).

Para melhor visualização a tabela foi dividida em outras duas, uma com os resultados obtidos da filtragem com *deep learning* e outra em que foi utilizado o método simples.

Exemplificando-as, as tabelas a seguir, destacadas no quadro 5, possuem 15 (quinze) das linhas de cada arquivo gerado, demonstrando as informações obtidas. Nota-se que as informações foram alteradas para melhor visualização.

Quadro 5 - Lista comparativas com deep learning (quadro a) e sem (quadro b)

COMENTÁRIO FILTRADO	CORRETO?
Amazing cinematographer one of the best to ever do it	SIM
He was behing the podium getting his **** sucked in the police academy	SIM
You ve final had your life changed	SIM
How ****** bad was my fever last night	SIM
Donnie yen is the ****	SIM
What the ****	SIM
Ieo didnt fake it legend	SIM
Star wars episode viii han ai nt dead *****	SIM
Well that s ****** stupid and par for the course i guess	SIM
Dave franco is still ****** hot	SIM
Wow that was actually a really cool read	SIM
Exciting new pôster for bloodfeast *** ****	SIM
Holy **** do i get to do it	SIM
The finniest part was when it said allnew movie	SIM
Am i the only one who thinks targeting a rating is ******t	SIM
Based off the mother ****** awesome book also by that name please be the case	SIM

(a)

COMENTÁRIO FILTRADO	CORRETO?
Amazing cinematographer one of the best to ever do it	SIM
He was behing the podium getting his **** sucked in the police academy	SIM
You ve final had your life changed	SIM
How ****** bad was my fever last night	SIM
Donnie yen is the ****	SIM
What the ****	SIM
Ieo didnt fake it legend	SIM
Star wars episode viii han ai nt dead *****	SIM
Well that s ****** stupid and par for the course i guess	NÃO
Dave franco is still ****** hot	SIM
Wow that was actually a really cool read	SIM
Exciting new pôster for bloodfeast *** ****	SIM
Holy **** do i get to do it	SIM
The finniest part was when it said allnew movie	SIM
Am i the only one who thinks targeting a rating is ******t	NÃO
Based off the mother ****** awesome book also by that name please be the case	NÃO

(b)

Fonte: Dados dos autores (2023).

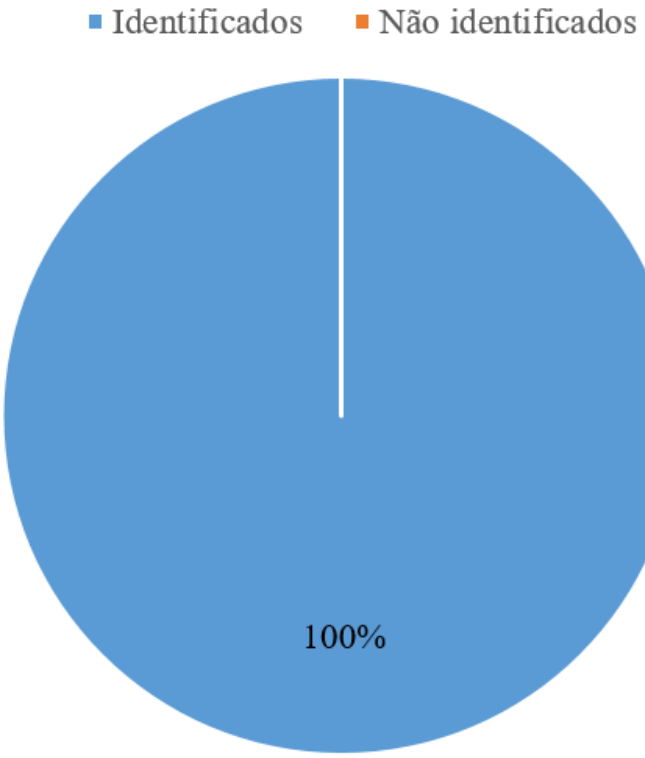
3.1 Resultados obtidos

Para realizar a análise dos resultados foi utilizado, de forma simples, uma planilha de dados para comparar os status de cada saída com a classificação realizada manualmente.

Ao analisar o arquivo filtrado utilizando o método com *deep learning* verificou-se que das 50 (cinquenta) frases selecionadas, todas foram corretamente checadas e, filtradas quando se fez necessário.

Como pode ser visto na figura 5 em que o gráfico demonstra a porcentagem das informações elencadas acima.

Figura 5 - Porcentagem de acertos e erros com deep learning



Fonte: Dados dos autores (2023).

Porém, em testes realizados previamente, foram identificados alguns pontos e frases em que a ferramenta peca.

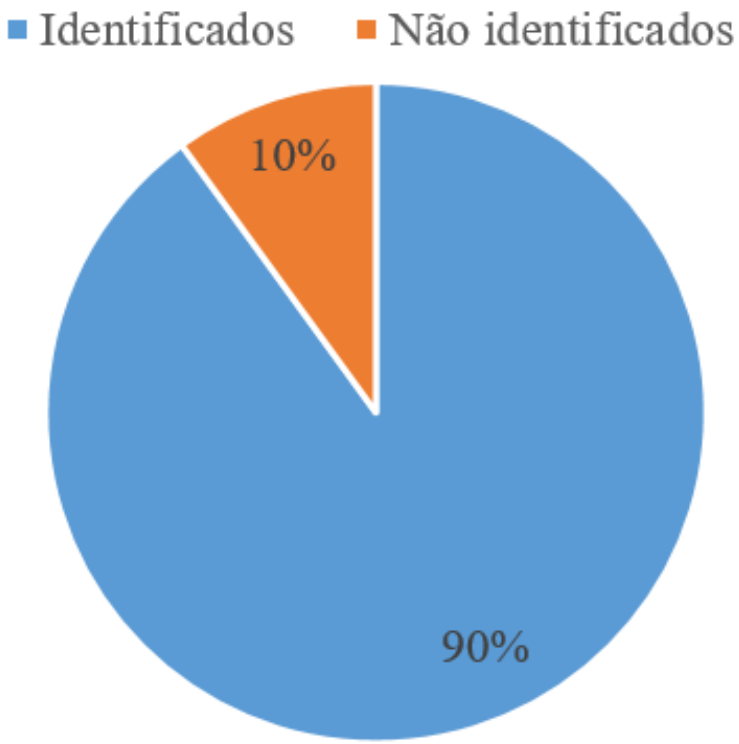
Em ambientes que a escrita se torna muito informal, onde a utilização de abreviações e gírias é algo comum, identificar possíveis palavras tóxicas se torna uma tarefa complexa. Tomando como exemplo a frase: "my mom shares the same bihday as @user bihday snake! see you this weekend".

A palavra "bihday" foi classificada como imprópria pois "bih" é uma forma informal de se escrever "bitch", dessa forma a ferramenta considerou, de forma incorreta, a frase como tóxica. Entretanto para a correta execução da filtragem, antes da verificação de sub palavras, o contexto e a palavra como um todo deveria ser levado em conta.

Quando as frases selecionadas foram submetidas ao processo sem a utilização do deep learning, foi possível notar uma queda na identificação correta durante a checagem, o que resultou em uma diminuição de 10% na eficácia. Com isso, das 50 (cinquenta) frases, apenas 40 (quarenta) foram classificadas corretamente.

A figura 6 abaixo demonstra graficamente as informações citadas.

Figura 6 - Gráfico - Porcentagem de acertos e erros sem deep learning



Fonte: Dados dos autores (2023).

Tomando como base o método sem o *deep learning*, é possível observar desvantagens em relação ao método que o utiliza. Palavras flexionadas ou com vogais trocadas para numerais dificilmente serão identificadas sem a utilização de técnicas de aprendizado de máquina. Um exemplo que é bastante encontrado é a troca da vogal "i" pelo numeral "1": *"That's bullsh1t!"*

Em ambos os casos não foram identificados falsos negativos e falsos positivos.

4 Considerações Finais

A criação da plataforma esbarrou em alguns desafios, principalmente o aprendizado de novas tecnologias e do estudo aprofundado de conteúdos sobre toxidade de comentários que não são explorados no dia a dia pelos integrantes do projeto. Visando minimizar as dificuldades que pudessem aparecer no decorrer do projeto, os integrantes demonstraram pré disposição em estudar de maneira aprofundada os temas que fossem necessários e, além disso, analisar e inibir as dificuldades que pudessem ser encontradas futuramente.

Levando em conta o conhecimento prévio adquirido pelos componentes do projeto em outras tecnologias, a curva de aprendizado e o estudo necessário para a criação da *API*, da aplicação, do banco de dados e o manuseio das ferramentas se mostravam aceitáveis para o tempo de desenvolvimento disponíveis.

O protótipo desenvolvido atendeu as expectativas gerando resultados satisfatórios e permitindo a coleta de informações para análise das propostas sugeridas. O método de filtragem utilizado demonstrou sua eficácia nas mais diversas situações impostas.

Ao longo do levantamento bibliográfico, foi detalhado de forma concisa o conhecimento que foi necessário para a criação total da plataforma, e além disso, também foi apresentado na seção de desenvolvimento como se planejou e desenvolveu o projeto para alcançar o resultado final, detalhando de forma clara e precisa os passos que foram necessários para que o projeto pudesse funcionar como esperado, entregando ao final uma utilização simples e eficaz de toda a plataforma.

Referências

APPLE, **About Swift**. Disponível em: https://www.swift.org/about/. Acesso em: 04 mar. 2024.

GUERRA, V.M.L.; KLAS, S.S. Um olhar sobre o cyberbullying: entre a periferia social e o preconceito, o percurso identitário do sujeito. **Cadernos de estudos culturais**, Campo Grande, v. 9, n. 17, p.195-212, jan.-jun. 2017.Disponível em https://periodicos.ufms.br/index.php/cadec/article/view/4235. Acesso em: 04 mar. 2024.

HUSSAIN, T.; CRAWFORD, T. A comparison of server side scripting technologies. In: **Proceedings of the International Conference on Software Engineering Research and Practice (SERP)**. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2017. p. 69-76. Disponível em: https://www.semanticscholar.org/paper/A-Comparison-of-Server-Side-Scripting-Technologies-Crawford/ad4da4d8cb10b77096794eac72c549d8e8d47b8c. Acesso em: 04 mar. 2024.

LECUN, Y.; BENGIO, Y; HINTON, G. E. Deep Learning. **Nature**, Vol. 521, p. 436-444. Disponível em: https://www.nature.com/articles/nature14539. Acesso em: 04 mar. 2024.

PAGANO, B; NIJIM, S. **APIs for dummies**: Apigee special edition. Hoboken: John Wiley & Sons, Inc. 2014.

PEREIRA, L.I.; CALDAS, C.O.L. O fenômeno Shitstorm: Internet, intolerância e violação de direitos humanos. **Interfaces Científicas - Humanas e Sociais**, Aracaju, v. 6, n. 1, p. 123-134, jun. 2017. Disponível em: https://periodicos.set.edu.br/humanas/article/view/3540. Acesso em: 04 mar. 2024.

PONTI, M. A.; COSTA, G. B. P. Como funciona o deep learning. In: **Tópicos em gerenciamento de dados e informações 2017**. Uberlândia: SBC, 2017. Disponível em: https://sol.sbc.org.br/livros/index.php/sbc/catalog/book/31. Acesso em: 04 mar. 2024.

VUEJS, **Readme**, Disponível em: https://github.com/vuejs/vue. Acesso em: 04 mar. 2024.