**UNIVERSIDADE FEDERAL DA PARAÍBA - CENTRO DE INFORMÁTICA**

**Processamento de Linguagem Natural**

**Professor Yuri Malheiros**

# ANÁLISE E CLASSIFICAÇÃO DE DISCURSO DE ÓDIO E TOXIDADE NO REDDIT UTILIZANDO O DATASET HATEBR E PROCESSAMENTO DE LINGUAGEM NATURAL

**Cleydson de Souza Ferreira Junior - Felipe Gontijo Siqueira**

**JOÃO PESSOA**

**2025**

## 1 APRESENTAÇÃO DO PROBLEMA

A ascensão das redes sociais e plataformas de conteúdo, como o Facebook, Instagram, X (antigo Twitter) e Reddit, transformou a maneira como as pessoas se comunicam, compartilham informações e formam comunidades. O Reddit, em particular, com sua estrutura baseada em comunidades de interesses específicos (os "subreddits"), promove discussões aprofundadas sobre uma vasta gama de tópicos. No entanto, o anonimato relativo e a escala massiva dessas plataformas também criaram um ambiente propício para a disseminação de comportamentos antissociais, incluindo toxicidade, assédio e, mais criticamente, o discurso de ódio.

O discurso de ódio online não é apenas uma violação dos termos de serviço da maioria das plataformas, mas também representa uma ameaça significativa à segurança e ao bem-estar dos usuários, podendo silenciar vozes marginalizadas e normalizar a intolerância. A moderação de conteúdo, embora essencial, enfrenta desafios monumentais:

- **Volume:** A quantidade de conteúdo gerado a cada segundo torna a moderação manual humana impraticável em larga escala.
- **Subjetividade e Contexto:** A distinção entre crítica, sarcasmo e discurso de ódio genuíno é complexa e altamente dependente do contexto cultural e situacional.
- **Evolução da Linguagem:** Gírias, eufemismos e "*dog whistles*" (apitos de cachorro) são constantemente criados para contornar os sistemas de moderação existentes.

Nesse cenário, o Processamento de Linguagem Natural (PLN) emerge como uma ferramenta poderosa. Sistemas automatizados podem ser treinados para analisar grandes volumes de texto e identificar padrões associados a comportamentos tóxicos, funcionando como uma primeira linha de defesa ou como uma ferramenta de apoio para moderadores humanos. Este projeto se insere exatamente nesse contexto, buscando aplicar técnicas de PLN para criar um sistema capaz de analisar o histórico de um usuário e quantificar a prevalência de discurso de ódio em sua comunicação, oferecendo um panorama claro de seu comportamento na plataforma.

## 2 OBJETIVOS

O objetivo principal deste projeto é desenvolver uma solução ponta a ponta para a análise e classificação de discurso de ódio e toxidade no histórico de comentários de um usuário do Reddit utilizando o *dataset* HateBR que, embora tenha sido construído a partir de comentários de Instagram, o problema-alvo deste projeto é estimar e visualizar a toxicidade de perfis em redes sociais, exigindo generalização do modelo entre domínios. Para alcançar este objetivo principal, foram definidos os seguintes objetivos específicos:

- Construir um classificador supervisionado em português baseado no HateBR capaz de estimar a probabilidade de ódio/ofensividade em comentários curtos;
- Integrar a coleta de dados do Reddit via PRAW (*Python Reddit API Wrapper*), respeitando credenciais, limites e convenções da API, para analisar os n comentários públicos mais recentes de um usuário;
- Reportar, no aplicativo, métricas de perfil como total analisado, percentual de comentários odiosos, nível médio de toxicidade e distribuição de atividade por subreddit e por dia.

## 3 DADOS UTILIZADOS E PRÉ-PROCESSAMENTO

O conjunto de dados HateBR foi carregado diretamente do repositório oficial no Github em formato CSV, contendo 7.000 comentários anotados por especialistas e utilizado aqui na camada binária para detecção de conteúdo ofensivo ou não-ofensivo. Na carga inicial, são inspecionadas as primeiras linhas e a distribuição de classes para uma verificação rápida da integridade e do balanceamento, assegurando aderência às anotações documentadas pelos autores do HateBR.

- **Representação textual:** utiliza-se um vetorizador TF-IDF, ajustando exclusivamente no conjunto de treino e transformando o conjunto de teste com o vocabulário aprendido, o que converte documentos em matrizes esparsas de características TF-IDF adequadas a modelos lineares;

- **Modelo de classificação:** treina-se uma regressão logística sobre a representação TF-IDF de treino, explorando um classificador linear regularizado e compatível com dados esparsos de alta dimensionalidade;
- **Avaliação:** o desempenho é medido com acurácia e com o relatório de classificação, que sumariza precisão, revocação e F1 por classe;
- **Objetos para produção:** ao final, os artefatos necessários para inferência são vetorizador treinado e o classificador, serializados para uso no aplicativo de análise de perfis.

## 4 METODOLOGIA

A arquitetura assume um *pipeline* clássico de PLN supervisionado: treinamento *offline* com HateBR, serialização de vetorizador e classificador com Joblib e, em produção, inferência probabilística para cada comentário, seguida de agregação de métricas de perfil. Essa abordagem se alinha aos *baselines* disponibilizados pelos autores do HateBR, que combinam representações baseadas em n-gramas/TF-IDF com classificadores lineares e de margem, oferecendo boa interpretabilidade e custo computacional reduzido para cenários de tempo real.

A decisão de projeto por um classificador com regressão logística e predição probabilística permite expor não apenas o rótulo binário, mas também um "nível de toxicidade" contínuo que melhora a análise agregada e a priorização de casos, sem dependência de modelos de larga escala mais custosos.

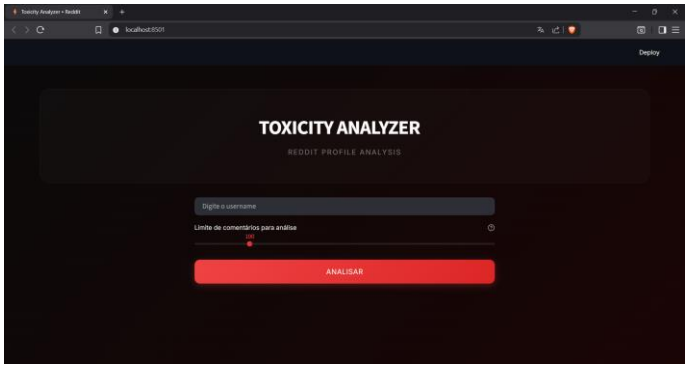
### 4.1 Técnicas utilizadas

- **Representação textual:** vetorizador serializado compatível com o modelo TF-IDF, conforme prática recomendada nos *baselines* HateBR e compatível com a API usada em produção;
- **Classificador probabilístico:** modelo com suporte a predição probabilística, permitindo rótulo binário e escore contínuo, salvo no arquivo "modelo_odio.joblib" e carregado na inicialização da aplicação;
- **Wrapper:** A integração utiliza PRAW como ferramenta de coleta (*wrapper*) oficial da comunidade para a API do Reddit, com autenticação do

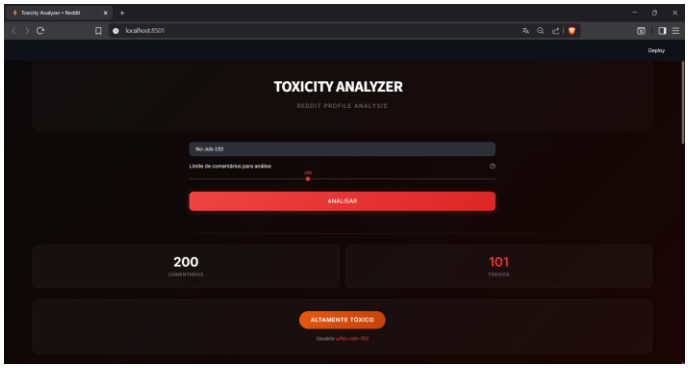
desenvolvedor definida por variáveis de ambiente, seguindo as convenções da documentação e dos *endpoints* de listagem e paginação descritos na referência da API;

- **Agregação e UI analítica:** cálculo de percentual de comentários odiosos, nível médio de toxicidade (média das probabilidades), "top subreddits" por atividade e comentários mais tóxicos, combinando interpretabilidade com acompanhamento longitudinal simples.

### 4.2 Experimento: Utilização prática com a API do Reddit



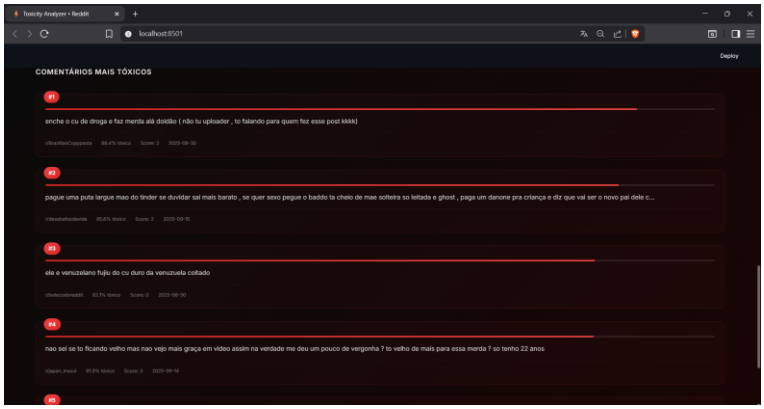

A aplicação, desenvolvida em Streamlit e hospedada pelo Streamlit Cloud, é iniciada localmente com o campo para digitar o usuário que será analisado, um *scroll* para delimitar a quantidade de comentários analisados e o botão de análise, que inicia a coleta dos "n comentários" públicos mais recentes de um perfil via PRAW, e aplica o classificador a cada item para obter rótulo/escala de toxicidade.




Após a solicitação de análise, para interpretação prática, o *app* mapeia a quantidade e porcentagem de comentários tóxicos/odiosos a faixas cognitivamente simples ("Extremamente Tóxico" ≥ 70%, "Altamente Tóxico" ≥ 50%, "Moderadamente

Tóxico” ≥ 25%, “Levemente Tóxico” ≥ 10%, “Perfil Limpo” < 10%), oferecendo um sumário direto do risco reputacional do perfil.


Como complemento visual, a interface também mostra os “subreddits preferidos” do usuário e sua atividade ao longo do tempo. Essas visualizações podem ajudar a identificar possíveis traços de personalidade do usuário, comunidades tóxicas e períodos turbulentos na rede.




Por fim, como principal critério de validação textual, um “top 5 comentários mais tóxicos” é apresentado, sendo possível analisar se o modelo está identificando bem os comentários com certo grau de toxidade. É importante destacar que a aplicação não seleciona apenas de forma binária, mas também indica “o quão tóxico” é um comentário, em porcentagem.

## 5 RESULTADOS

O sistema entrega, para cada perfil analisado, um conjunto de indicadores: total de comentários processados, contagem e percentual classificados como odiosos, nível médio de toxicidade em escala percentual, “subreddits” mais ativos e os cinco

comentários com maiores escores, permitindo diagnósticos rápidos e comparações ao longo do tempo.

Como linha de base para expectativa de desempenho do classificador, os autores do HateBR reportam resultados robustos com técnicas clássicas sobre o corpus (F1 aproximado em 0,85 para tarefas correlatas), o que sugere adequação da estratégia de vetorizarador + classificador linear/probabilístico para o domínio de comentários curtos em português.

Por fim, observou-se possíveis melhorias futuras: *fine-tuning* de modelos modernos de linguagem (por exemplo, variantes BERT adaptadas ao HateBR) pode ser explorado para ganhos adicionais; aplicação em outras redes sociais, como Instagram e X, bem como em plataformas de conversa, como Whatsapp, Telegram e Discord; monitoramento contínuo com processamento paralelo de múltiplos perfis e/ou plataformas para análises ampliadas em casos judiciais.

## REFERÊNCIAS


VARGAS, Francielle; CARVALHO, Isabelle. HateBR: A benchmark dataset for offensive language and hate speech in Brazilian Portuguese. GitHub, 2025. Disponível em: https://github.com/franciellevargas/HateBR.

VARGAS, Francielle; CARVALHO, Isabelle; GÓES, Fabiana Rodrigues de; PARDO, Thiago; BENEVENTO, Fabrício. HateBR: A Large Expert Annotated Corpus of Brazilian Instagram Comments for Offensive Language and Hate Speech Detection. In: LANGUAGE RESOURCES AND EVALUATION CONFERENCE, 13., 2022, Marseille. Proceedings [...]. Marseille: European Language Resources Association, 2022. p. 7174–7183. Disponível em: https://aclanthology.org/2022.lrec-1.777/.

PRAW. Python Reddit API Wrapper – Documentation. Read the Docs, 2025. Disponível em: https://praw.readthedocs.io/en/stable/.

PRAW Developers. PRAW – The Python Reddit API Wrapper. GitHub, 2025. Disponível em: https://github.com/praw-dev/praw.

JOBLIB Development Team. Joblib – Python persistence and parallel computing. Read the Docs, 2025. Disponível em: https://joblib.readthedocs.io/en/stable/.