VINICIUS TRINDADE SANTOS

Análise de Sentimentos e Toxicidade em Redes Sociais

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

São Paulo
2024

VINICIUS TRINDADE SANTOS

Análise de Sentimentos e Toxicidade em Redes Sociais

Versão Corrigida

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de Pós-Graduação em Engenharia de Dados e Big Data.

Área de Concentração:

Tecnologia da Informação – Engenharia/ Tecnologia/ Gestão

Orientador:

Luiz Sergio de Souza

São Paulo
2024

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

AGRADECIMENTOS

Gostaria de expressar minha profunda gratidão a todos que tornaram possível a realização deste MBA, uma jornada desafiadora e enriquecedora. Primeiramente, agradeço aos meus pais, que sempre foram meu alicerce. Obrigado por acreditarem em mim, pelo apoio constante e por incentivarem meus estudos, mesmo nas dificuldades. Seu amor e valores são a base de tudo o que conquistei. Agradeço também aos professores deste MBA, cuja dedicação e orientações foram essenciais para o meu aprendizado e crescimento profissional. A cada um de vocês, meu sincero agradecimento.

CURSO ENGENHARIA DE BIG DATA

Coord.: Prof. Solange N. Alves de Souza

Vice-Coord.: Prof. Pedro Luiz Pizzigatti Corrêa

Perspectivas profissionais alcançadas com o curso:

Busca de me qualificar para o mercado de trabalho, com ênfase na área de ciência de dados, fui capaz de aplicar os conhecimentos adquiridos durante minha especialização que podem me apresentar a novas opportunidades em diferentes áreas.

RESUMO

A indústria da tecnologia, em constante evolução, tem se beneficiado de avanços tecnológicos e da crescente interação em plataformas digitais, como o Discord, que se consolidou como um espaço dinâmico para a comunicação em tempo real e a formação de comunidades. A análise de sentimento tem ganhado relevância nesse contexto, permitindo compreender emoções e comportamentos dos usuários, além de oferecer insights valiosos para desenvolvedores e gestores de comunidades. Essa abordagem é útil tanto na identificação de tendências como na moderação de conteúdo e na gestão de conflitos em ambientes colaborativos.

Este estudo propõe o desenvolvimento de um bot para Discord, com capacidade de realizar análises de sentimento em tempo real, auxiliando na identificação de sentimentos predominantes e na detecção de possíveis conflitos dentro de comunidades. Inspirado em ferramentas analíticas existentes, o bot foi desenvolvido com base em princípios de engenharia de requisitos, garantindo o atendimento às necessidades específicas dos usuários. A aplicação prática dessa solução contribui para a gestão de comunidades digitais, promovendo ambientes mais engajantes e colaborativos.

A proposta une conceitos de análise de sentimento, engenharia de software e inteligência artificial para explorar como a tecnologia pode otimizar as interações sociais em redes digitais. O trabalho não apenas visa aprofundar o entendimento sobre a relevância da análise de sentimento em plataformas como o Discord, mas também apresenta uma solução para enfrentar desafios relacionados ao comportamento online, potencializando o uso de tecnologias emergentes para fortalecer comunidades virtuais.

Palavras-Chave – Discord, bot, Análise de sentimento, Gestão de comunidades.

ABSTRACT

The constantly evolving technology industry has greatly benefited from technological advancements and the growing interaction on digital platforms such as Discord, which has established itself as a dynamic space for real-time communication and community building. Sentiment analysis has gained prominence in this context, enabling a deeper understanding of users' emotions and behaviors while providing valuable insights for developers and community managers. This approach is useful for identifying trends, moderating content, and managing conflicts in collaborative environments.

This study proposes the development of a Discord bot capable of performing real-time sentiment analysis, assisting in the identification of predominant sentiments and detecting potential conflicts within communities. Inspired by existing analytical tools, the bot was developed based on requirements engineering principles to ensure it meets users' specific needs. The practical application of this solution contributes to the management of digital communities, fostering more engaging and collaborative environments.

The proposal combines concepts of sentiment analysis, software engineering, and artificial intelligence to explore how technology can optimize social interactions in digital networks. The work not only aims to deepen the understanding of the relevance of sentiment analysis on platforms like Discord but also presents a solution to address challenges related to online behavior, leveraging emerging technologies to strengthen virtual communities.

Keywords – Discord, Bot, Sentiment Analysis, Community Management.

LISTA DE FIGURAS

1 Arquitetura da aplicação 30

2 Fluxo de dados 31

3 Banco de dados 32

4 Fluxograma do script 33

5 Exemplo de requisição body 35

6 Resposta de status indicando sucesso 36

7 Diagrama de sequência 37

8 Exemplo de mensagem enviada 38

9 Exemplo de resposta fornecida 39

10 Ambiente de teste 40

11 Distribuição de Sentimentos por Analisador 41

12 Matriz de concordância 42

13 Distribuição de sentimento 43

14 Distribuição de escores 43

15 Matriz de Confusão 44

16 Comparação de métricas por modelo de análise 45

SUMÁRIO

1 Introdução **10**

1.1 Motivação 12

1.2 Justificativa 12

1.3 Objetivo 13

1.4 Metodologia 13

2 Fundamentação Teórica **15**

2.1 Discord como rede social 15

2.2 Toxicidade nas redes sociais 17

2.2.1 Impactos da Toxicidade 18

2.2.2 Combate à Toxicidade com Análise de Sentimento 19

2.3 Análise de sentimento 20

2.4 Modelos de Análise de sentimento 22

2.4.1 TextBlob 23

2.4.2 VADER 24

2.4.3 Transformers 26

2.4.4 Comparação e Escolha do Modelo 28

3 Desenvolvimento **29**

3.1 Conjunto de dados 30

3.2 Framework API FLASK 32

3.3 Script API FLASK 34

3.4 Script discord-bot 38

4 Resultado e discussão **40**

5 Conclusão **47**

Referências **48**

Apêndice A – Termo de Consentimeno de participação **51**

Apêndice B – Scripts utilizados **53**

1 INTRODUÇÃO

A indústria da tecnologia, um setor dinâmico e em constante evolução, tem se beneficiado enormemente dos avanços tecnológicos e das novas formas de interação social proporcionadas pelas plataformas digitais. Uma dessas plataformas, o Discord, tornou-se um ambiente popular para a formação de comunidades de jogadores, facilitando a comunicação em tempo real e a construção de conhecimento coletivo (KANUKA; ANDERSON, 2007). Nesse contexto, a análise de sentimento emergiu como uma ferramenta poderosa para compreender as percepções e emoções dos usuários, oferecendo insights valiosos tanto para desenvolvedores quanto para gestores de comunidades.

Da mesma forma, a análise de dados torna-se uma ferramenta essencial para desvendar padrões de comportamento dos usuários, identificar tendências emergentes, e, principalmente, compreender as demandas e expectativas do público. Este campo de estudo vem ganhando cada vez mais atenção, como o estudo de Egliston (EGLISTON, 2016), que destaca a aplicabilidade de metodologias analíticas ao estudo da cultura de jogos multiplayer, e Reynolds (REYNOLDS, 2019), que explora o design de riscos em jogos sociais de cassino.

Este cenário de pesquisa também contempla a interseção entre o universo dos videogames e conceitos-chave da era digital, como o uso ético de Big Data, conforme observado nos estudos de Willson e Leaver (WILLSON; LEAVER, 2015), sobre a ética da mineração de dados em jogos sociais da Zynga. Além disso, a análise de dados em jogos está integrada ao contexto mais amplo da Internet das Coisas (IoT), conforme proposto por Pouryazdan (POURYAZDAN et al., 2017) na criação de jogos inteligentes para participantes móveis.

Essa abordagem se alinha também com a pesquisa de Esfahani (ESFAHANI et al., 2019), que emprega a cientometria para analisar a interseção entre Big Data e mídias sociais, e com os estudos de Horng (HORNG et al., 2022), que modelam vantagens competitivas através do uso de conceitos de Big Data e mídias sociais no desenvolvimento de estratégias de sustentabilidade.

O presente trabalho de conclusão de curso visa explorar a aplicação da análise de sentimento no ramo das redes sociais, utilizando um bot de Discord como ferramenta principal. Este estudo é fundamentado em diversas pesquisas que demonstram a eficácia e as múltiplas aplicações dos chatbots em diferentes áreas (XUE et al., 2023), desde a construção de ontologias (COSTA, 2020) até a recuperação de informações orientada a metadados (CARVALHO, 2018). Adicionalmente, a capacidade dos bots de realizar a classificação de comentários tóxicos (Magzoub, 2023) e a detecção de trolls através da adaptação de análise de sentimento (SEAH et al., 2015) destacam o potencial na manutenção de ambientes virtuais mais saudáveis e produtivos. Essa funcionalidade permite identificar, com maior precisão, comportamentos nocivos que podem comprometer a dinâmica de comunidades online, facilitando intervenções preventivas, onde ambientes de grande escala, como redes sociais e fóruns, no qual moderadores humanos podem enfrentar dificuldades para acompanhar o volume e a complexidade das interações.

A análise de sentimento aplicada a plataformas colaborativas como o Discord não é apenas relevante para a moderação de conteúdo, mas também para o entendimento de dinâmicas sociais e interesses dos usuários (ARIFIANTO; IZZUDIN, 2021). Pesquisas como "Good News, Bad News: A Sentiment Analysis of the 2016 Election Russian Facebook Ads"(ALVAREZ et al., 2020) e o estudo de Bukovina (BUKOVINA, 2016) demonstram como a análise de sentimento pode desvendar padrões de comunicação e influências emocionais em redes sociais (SHIVANI et al., 2023).

Neste contexto, o presente estudo se propõe a desenvolver um bot de Discord capaz de realizar análise de sentimento em tempo real nas interações dos usuários dentro de comunidades da rede social (JEONG; KIM, 2021). Este bot, inspirado por ferramentas como o RALeM (uma ferramenta para auxiliar na identificação de palavras significativas em conversas em ferramentas on-line colaborativas através de análise léxico-morfológica) (FORNAZARI, 2019), será projetado para identificar sentimentos predominantes, detectar possíveis conflitos e oferecer insights para a gestão de comunidades.

Em suma, este trabalho não só contribui para a compreensão da aplicação da análise de sentimento na indústria das redes sociais, mas também oferece uma solução prática para a gestão de comunidades no Discord, potencializando o uso de tecnologias emergentes para a criação de ambientes virtuais mais seguros e colaborativos.

1.1 Motivação

As motivações iniciais para o presente estudo estão centradas na crescente relevância das redes sociais, como o Discord, no cenário digital e na necessidade de entender as interações emocionais e comportamentais dos usuários nesses ambientes. Com o aumento exponencial de dados gerados em tempo real nessas plataformas, surge a necessidade de explorar como as ferramentas de análise de sentimento podem ser aplicadas para compreender melhor as dinâmicas sociais e moderar interações, a fim de manter comunidades virtuais mais saudáveis e colaborativas.

Outro ponto motivador é o reconhecimento do papel fundamental que a análise de dados desempenha na identificação de tendências emergentes e na personalização de experiências, tanto para desenvolvedores de jogos quanto para gestores de comunidades online. A indústria da tecnologia, em particular o setor de videogames e plataformas digitais, demanda ferramentas mais eficientes para captar as percepções e emoções dos usuários de maneira precisa e em tempo real, com o objetivo de criar ambientes mais envolventes e produtivos.

Além disso, com o aumento da toxicidade online e da disseminação de conteúdos prejudiciais, há uma motivação clara para utilizar essas tecnologias como uma forma de detectar e mitigar comportamentos negativos, garantindo um espaço mais seguro e agradável para os participantes.

1.2 Justificativa

A indústria das redes sociais e dos videogames, setores altamente lucrativos e em constante transformação, enfrenta desafios consideráveis na gestão de comunidades e na garantia de interações saudáveis e construtivas. À medida que o volume de dados gerados por essas plataformas cresce exponencialmente, torna-se essencial utilizar ferramentas de análise que ajudem a compreender as emoções e comportamentos dos usuários em tempo real.

A análise de sentimentos oferece uma maneira eficaz de decifrar as percepções dos usuários, ajudando não apenas na moderação de conteúdo tóxico, mas também na identificação de tendências, melhoria de experiências de usuário e personalização de estratégias de marketing. O Discord, em particular, é uma plataforma que atrai grandes comunidades de jogadores e outros grupos, sendo um ambiente dinâmico onde o feedback dos

usuários pode ser uma ferramenta valiosa para desenvolver melhorias, tanto em termos de funcionalidade quanto de gerenciamento de comportamento.

A implementação de um bot que realize essa análise automaticamente justifica-se pela necessidade de respostas rápidas e eficientes na moderação de comportamentos prejudiciais, como a toxicidade e o assédio, além de fornecer insights úteis para os gestores de comunidade. Ao integrar essas funcionalidades em um bot de Discord, o presente trabalho oferece uma solução para melhorar a experiência dos usuários e garantir que os espaços digitais sejam mais acolhedores e colaborativos.

Portanto, a justificativa deste estudo está ancorada na necessidade crescente de explorar soluções tecnológicas que ajudem a entender melhor o comportamento humano nas redes sociais e a garantir a sustentabilidade das comunidades digitais, promovendo ambientes mais seguros, engajantes e produtivos.

1.3 Objetivo

Este trabalho tem como objetivo desenvolver um bot para a plataforma Discord capaz de realizar a análise de sentimento em tempo real nas interações dos usuários nas comunidades, buscando a identificação de sentimentos predominantes, detecção de possíveis conflitos e fornecimento de insights para a gestão eficiente dessas comunidades.

1.4 Metodologia

Os procedimentos planejados para satisfazer as principais metas do desenvolvimento do bot de análise de sentimento são os seguintes: (i) formação do aluno de modo a introduzi-lo aos conceitos de processamento de linguagem natural (NLP), com o objetivo de capacitá-lo para o desenvolvimento do projeto; (ii) execução dos procedimentos necessários para o desenvolvimento completo do bot, visando alcançar as metas esperadas de precisão e eficácia na análise de sentimentos. Considerando o período do projeto, a seguinte sequência de procedimentos é esperada:

1. Criação do Bot de Análise de Sentimento: Como atividade inicial, pretende-se desenvolver um bot capaz de realizar análises de sentimento em tempo real, avaliando as mensagens enviadas pelos usuários na plataforma Discord. A ferramenta deve ser capaz de identificar sentimentos predominantes (positivo, negativo ou neutro) e fornecer feedback para moderadores e gestores de comunidade.

2. Aplicação em Ambiente de teste: Neste estágio, o bot será implementado em um servidor de testes, como os servidores de Discord, para avaliar seu impacto na moderação e no engajamento dos usuários. Será analisada a eficácia do bot na identificação de sentimentos e na detecção de possíveis conflitos ou comportamentos tóxicos, fornecendo métricas para a gestão de comunidades.

3. Descrição dos Resultados: A etapa final compreende o processo de descrição dos resultados obtidos, assim como a descrição do uso em outras áreas do mundo real e suas aplicações.

2 FUNDAMENTAÇÃO TEÓRICA

A validação deste estudo é fundamental para demonstrar a eficácia e a aplicabilidade da solução proposta na análise de sentimento em comunidades do Discord. A implementação do bot integrado à API de análise de sentimento baseia-se em conceitos consolidados de processamento de linguagem natural (PLN) e em práticas reconhecidas para o monitoramento de interações sociais em plataformas digitais. No entanto, para garantir que os resultados obtidos sejam confiáveis, precisos e úteis no contexto de moderação e gestão de comunidades online, é necessário um processo rigoroso de validação (ANDRADE, 2016).

O contexto atual das redes sociais, caracterizado pelo crescimento exponencial de interações digitais, apresenta desafios significativos, como a detecção de comportamentos tóxicos e a moderação de conteúdo. Nesse cenário, soluções tecnológicas precisam ser testadas para assegurar que operem de forma eficiente em ambientes reais, considerando fatores como a complexidade da linguagem, incluindo gírias, variações culturais e mensagens multilíngues (Magzoub, 2023).

2.1 Discord como rede social

O Discord, embora inicialmente criado como uma plataforma de comunicação voltada para jogadores de videogames, rapidamente evoluiu para se tornar uma rede social multifacetada. Sua versatilidade e simplicidade de uso permitiram que ele fosse adotado por diversos grupos de interesse, ultrapassando os limites do universo gamer. Hoje, o Discord é utilizado para muito mais do que apenas conversas durante partidas de jogos; ele é uma ferramenta essencial para criação de comunidades, troca de conhecimentos, eventos online e colaboração em projetos (KANUKA; ANDERSON, 2007).

A plataforma se destaca por seus servidores, que são espaços digitais onde os usuários podem interagir através de canais de texto e voz organizados por temas ou tópicos. Cada servidor funciona como uma mini comunidade dentro do Discord, podendo ser customizado com diferentes regras, funções de usuários e bots que ajudam a moderar, automatizar

tarefas e melhorar a interação entre os membros. O uso de bots é uma das características que diferenciam o Discord de outras redes sociais, permitindo automação avançada e personalização dos servidores. Esses bots têm se tornado fundamentais para a administração de comunidades, sendo utilizados para funções como moderação, integração com outras plataformas e até mesmo para análises de dados (KANUKA; ANDERSON, 2007).

Além disso, o Discord possui um sistema de mensagens diretas, semelhante a outras redes sociais, o que facilita a interação entre os usuários de maneira privada. Com a introdução de ferramentas como vídeo chamadas, compartilhamento de tela e até a integração com aplicativos externos, a plataforma ampliou suas funcionalidades. Isso a transformou em um espaço não apenas para entretenimento, mas também para ensino remoto, grupos de estudo, ambientes corporativos e suporte técnico em tempo real. Essa flexibilidade tornou o Discord uma escolha popular para uma variedade de finalidades, permitindo que ele atenda tanto a demandas profissionais quanto sociais (KANUKA; ANDERSON, 2007).

No contexto de análise de dados e comportamento de usuários, o Discord se torna uma fonte rica para pesquisadores e desenvolvedores, graças ao volume massivo de interações que ocorrem em tempo real. A análise dessas interações pode revelar insights importantes sobre as dinâmicas sociais das comunidades, bem como sobre o estado emocional de seus membros. A análise de sentimento, por exemplo, tem sido uma técnica amplamente explorada em plataformas como o Discord para medir o tom das interações, detectar discursos tóxicos e identificar tendências comportamentais emergentes. Essa técnica, aplicada em tempo real, pode oferecer uma visão valiosa para moderadores e gestores de comunidades, permitindo ações proativas que promovam ambientes mais saudáveis e colaborativos (KANUKA; ANDERSON, 2007).

Adicionalmente, o Discord se destaca como uma plataforma que incentiva a construção de conhecimento coletivo, dado que muitas comunidades o utilizam para compartilhar informações, organizar atividades e até mesmo para desenvolvimento de software colaborativo. Sua estrutura modular e adaptável o torna ideal para grupos com objetivos específicos, permitindo a criação de espaços que atendam exatamente às necessidades de seus membros (KANUKA; ANDERSON, 2007).

Resumidamente, o Discord, em sua essência, é uma rede social colaborativa, onde as comunidades se organizam de forma orgânica e interativa, facilitando a formação de laços sociais e a troca de conhecimentos entre os membros. O crescimento exponencial da plataforma e sua flexibilidade mostram como o Discord, mais do que um simples mensa-

geiro, se tornou uma ferramenta essencial para o engajamento digital no mundo moderno, abrigando desde grupos de fãs de videogames até empresas, comunidades de estudo e movimentos sociais. O impacto do Discord em diversas esferas reforça sua relevância como uma das principais plataformas de interação social da era digital (KANUKA; ANDERSON, 2007).

2.2 Toxicidade nas redes sociais

A toxicidade no ambiente online é um problema crescente que afeta diversas plataformas digitais, como redes sociais, fóruns e comunidades de jogos. Esse comportamento, caracterizado por interações agressivas, hostis ou desrespeitosas, pode tomar muitas formas, incluindo cyberbullying, assédio, discurso de ódio, trollagem e a disseminação de conteúdos ofensivos. No contexto das comunidades digitais, como aquelas encontradas no Discord e em plataformas de jogos online, a toxicidade pode prejudicar a experiência dos usuários, deteriorar o senso de comunidade e, em casos extremos, levar à exclusão ou afastamento de membros (SIMõES; CAMPONEZ, 2020).

A toxicidade afeta tanto os indivíduos diretamente envolvidos nas interações quanto a saúde geral da comunidade. Em ambientes como os servidores de Discord ou plataformas de jogos multiplayer, onde a colaboração e o trabalho em equipe são essenciais, a presença de comportamentos tóxicos pode resultar em conflitos, quebra de confiança e diminuição da satisfação dos usuários. A hostilidade nesses espaços virtuais muitas vezes cria um ciclo vicioso, em que mais jogadores ou participantes se sentem encorajados a adotar comportamentos semelhantes, agravando ainda mais o problema (SIMõES; CAMPONEZ, 2020).

Um dos fatores que facilitam a toxicidade online é a anonimidade proporcionada pelas plataformas digitais. No ambiente online, os usuários muitas vezes se sentem protegidos pela distância física e pela possibilidade de ocultar sua identidade real, o que pode levar à dissociação moral e à redução de inibições que normalmente impediriam comportamentos agressivos em interações face a face. Essa dissociação, combinada com a falta de consequências imediatas, cria um ambiente propício para que comportamentos tóxicos floresçam (SIMõES; CAMPONEZ, 2020).

A toxicidade também é amplificada por fenômenos como a trollagem, que envolve usuários que intencionalmente provocam ou irritam outros para obter reações emocionais. Esse tipo de comportamento pode se espalhar rapidamente em plataformas como o Dis-

cord, onde as conversas acontecem em tempo real, e a reação a uma provocação pode gerar respostas ainda mais hostis, criando um ambiente tóxico. Além disso, algoritmos de redes sociais e mecanismos de engajamento, que muitas vezes priorizam conteúdos mais controversos ou emocionais, podem contribuir para a disseminação de discussões agressivas e aumentar a polarização entre os membros de uma comunidade (SIMõES; CAMPONEZ, 2020).

Outro agravante é a ausência de ferramentas adequadas para moderar as interações em tempo real. Muitas comunidades dependem exclusivamente de moderadores humanos, que, apesar de desempenharem um papel importante, não conseguem acompanhar o volume e a velocidade das interações em plataformas com grandes comunidades. Isso permite que mensagens tóxicas permaneçam visíveis por mais tempo, impactando negativamente os usuários e ampliando os danos causados (SIMõES; CAMPONEZ, 2020).

Desta forma, a toxicidade pode ter efeitos psicológicos profundos, tanto nas vítimas quanto naqueles que testemunham tais interações. Ansiedade, estresse e perda de autoestima são consequências comuns para aqueles diretamente afetados. Para a comunidade como um todo, a toxicidade pode levar a uma diminuição do engajamento, perda de membros valiosos e até mesmo ao encerramento de servidores ou fóruns. Esses impactos ressaltam a importância de desenvolver estratégias eficazes para lidar com o problema, como a implementação de ferramentas de moderação automática, análise de sentimento em tempo real e educação dos membros da comunidade para a promoção de interações respeitosas (SIMõES; CAMPONEZ, 2020).

2.2.1 Impactos da Toxicidade

A toxicidade online pode ter uma série de impactos negativos tanto para os indivíduos quanto para a própria plataforma ou comunidade. Para os usuários, a exposição constante a interações tóxicas pode resultar em estresse emocional, ansiedade, redução do bem-estar mental e até em sintomas mais graves, como depressão. Em alguns casos, esse ambiente hostil pode levar ao afastamento de plataformas, perda de interesse em jogos e atividades online e até à exclusão social, impactando a capacidade de indivíduos se conectarem de maneira saudável no ambiente virtual (SANTOS; RODRIGUES, 2023).

Para as plataformas e comunidades, a toxicidade pode afetar diretamente o engajamento dos usuários e a longevidade das comunidades. Um ambiente hostil não apenas desencoraja novos membros de se unirem, mas também pode causar a saída de participantes antigos, comprometendo o crescimento, a diversidade e o dinamismo do grupo.

Quando essas comunidades são centrais para a sobrevivência de uma plataforma, o impacto negativo é ainda maior, podendo resultar em perda de receita, desvalorização da marca e dificuldades em atrair anunciantes ou parceiros (SANTOS; RODRIGUES, 2023).

Além disso, plataformas que não adotam medidas eficazes contra comportamentos tóxicos enfrentam o risco de danos à reputação, com a percepção pública de que não se preocupam com a segurança e o bem-estar de seus usuários. Essa imagem negativa pode levar a boicotes, ações legais ou perda de competitividade em relação a plataformas que investem ativamente em moderação e no fortalecimento de comunidades saudáveis (SANTOS; RODRIGUES, 2023).

Outro impacto significativo está relacionado ao ciclo vicioso que a toxicidade pode criar. Uma comunidade onde comportamentos negativos são tolerados tende a perpetuar esses padrões, tornando ainda mais difícil reverter o cenário. Além disso, em muitos casos, os usuários que enfrentam toxicidade repetidamente podem acabar reproduzindo comportamentos agressivos como forma de defesa ou por influência do ambiente (SANTOS; RODRIGUES, 2023).

Por fim, a toxicidade também impõe um custo operacional para as plataformas, que precisam alocar recursos significativos para implementar sistemas de moderação, desenvolver ferramentas automáticas de detecção e análise de conteúdo ou contratar equipes dedicadas ao gerenciamento de comunidades. Assim, enfrentar a toxicidade online é não apenas uma questão ética e social, mas também um imperativo estratégico para garantir a sustentabilidade e o sucesso das plataformas digitais a longo prazo (SANTOS; RODRIGUES, 2023).

2.2.2 Combate à Toxicidade com Análise de Sentimento

Uma das abordagens para lidar com a toxicidade em ambientes online é o uso de ferramentas de análise de sentimento. Através de algoritmos de Processamento de Linguagem Natural (PLN), é possível identificar automaticamente mensagens e interações que contenham linguagem negativa, ofensiva ou potencialmente tóxica. Aplicada em tempo real, essa tecnologia pode alertar moderadores sobre interações problemáticas e fornecer insights sobre o estado emocional da comunidade, permitindo intervenções rápidas e eficazes (ANDRADE, 2016).

No Discord, por exemplo, a análise de sentimento pode ser utilizada para monitorar as conversas em servidores e detectar comportamentos tóxicos antes que eles escalem. Bots

podem ser programados para identificar padrões de linguagem agressiva ou negativa e tomar medidas automáticas, como notificar administradores, remover mensagens problemáticas ou aplicar penalidades, como silenciar temporariamente usuários tóxicos. Esses bots também podem ser configurados para fornecer feedback aos usuários, educando-os sobre comportamentos inadequados e incentivando interações mais positivas (Magzoub, 2023).

Além da moderação, a análise de sentimento pode ser usada para gerar relatórios detalhados sobre as dinâmicas emocionais das comunidades, ajudando administradores a entender tendências, identificar períodos de maior estresse ou conflito e planejar estratégias para promover interações mais saudáveis. Por exemplo, ao detectar um aumento na frequência de mensagens negativas, os administradores podem organizar eventos, promover campanhas educativas ou ajustar regras e diretrizes do servidor para criar um ambiente mais acolhedor (SEAH et al., 2015).

A toxicidade no ambiente online é um desafio que exige abordagens multidimensionais, incluindo ferramentas tecnológicas, políticas claras de moderação e educação dos usuários sobre comportamento responsável. A aplicação de análise de sentimento em plataformas como o Discord oferece uma solução promissora para identificar e mitigar comportamentos tóxicos, garantindo que as comunidades possam continuar a prosperar em um ambiente mais saudável, seguro e colaborativo (SIMõES; CAMPONEZ, 2020).

Além disso, essas ferramentas podem promover a inclusão e o respeito nas interações, pois ajudam a criar um espaço onde os membros se sentem valorizados e protegidos. O combate à toxicidade não apenas melhora a experiência individual dos usuários, mas também fortalece a coesão social e o engajamento dentro das comunidades digitais. Essa combinação de tecnologia e boas práticas de gestão é essencial para sustentar comunidades vibrantes e produtivas no mundo online moderno (SIMõES; CAMPONEZ, 2020).

2.3 Análise de sentimento

A análise de sentimento, também conhecida como *opinion mining*, é uma técnica de Processamento de Linguagem Natural (PLN) que visa identificar e categorizar opiniões expressas em um texto, determinando se a atitude do autor em relação a um determinado tema é positiva, negativa ou neutra. No contexto de redes sociais e plataformas de comunicação, essa análise tem se tornado uma ferramenta poderosa para entender as percepções dos usuários, captar tendências emocionais e monitorar o clima geral de uma

comunidade (ANDRADE, 2016).

A análise de sentimento funciona por meio da aplicação de algoritmos que analisam textos para detectar sentimentos subjacentes. Esses algoritmos podem utilizar tanto abordagens lexicais, baseadas em palavras-chave e dicionários de polaridade, quanto técnicas mais avançadas como modelos de aprendizado de máquina, que são treinados em grandes volumes de dados para reconhecer nuances emocionais de maneira mais eficaz. Ferramentas como TextBlob, VADER e Transformers (como BERT e GPT) são amplamente usadas para implementar análises de sentimento em diversas linguagens e domínios, tornando o processo acessível e altamente adaptável a diferentes contextos (RODRIGUES et al., 2010).

No ambiente das redes sociais, como Twitter, Facebook, Instagram e Discord, a análise de sentimento desempenha um papel crucial ao oferecer insights sobre as emoções dos usuários em relação a produtos, serviços, eventos e até dinâmicas sociais dentro das comunidades. Essa capacidade de interpretar sentimentos é valiosa para empresas e marcas, que podem ajustar estratégias de marketing e comunicação com base nas emoções predominantes, melhorando o alcance e a receptividade de campanhas. Além disso, para gestores de comunidades online, a análise de sentimento é uma ferramenta estratégica para identificar conflitos potenciais, moderar conteúdos tóxicos e, de modo geral, melhorar o clima dentro da comunidade (Magzoub, 2023).

Especificamente no Discord, amplamente utilizado para comunicação em tempo real entre jogadores, educadores, grupos de estudo e muitas outras comunidades, a análise de sentimento pode ser empregada de maneira proativa. Monitorar o tom emocional das interações ajuda a detectar comportamentos negativos ou tóxicos, identificar momentos de frustração entre os usuários e intervir rapidamente para resolver problemas antes que eles escalem. Bots equipados com análise de sentimento podem, por exemplo, sinalizar mensagens problemáticas para moderadores ou mesmo fornecer respostas automáticas de mediação. Essa abordagem não só contribui para um ambiente mais saudável, mas também fomenta maior engajamento e satisfação dos usuários (Magzoub, 2023).

Além disso, no contexto de desenvolvimento de jogos e produtos digitais, a análise de sentimento é uma ferramenta poderosa para compreender o feedback emocional em relação a novos recursos, atualizações ou eventos. Desenvolvedores podem usar esses dados para ajustar suas estratégias, melhorar o design de jogos e produtos, e oferecer experiências mais personalizadas e alinhadas às expectativas dos usuários (Magzoub, 2023).

A aplicação de análise de sentimento também abre portas para estudos acadêmicos e de

mercado, permitindo o mapeamento de tendências comportamentais, sociais e emocionais em larga escala. No caso de comunidades do Discord, essas análises podem fornecer dados valiosos sobre a coesão do grupo, temas de maior interesse e momentos de pico de interação emocional, ajudando gestores a criar estratégias para maximizar a experiência de seus membros (ANDRADE, 2016).

Em resumo, a análise de sentimento se consolidou como uma tecnologia essencial no ambiente digital contemporâneo, permitindo uma compreensão mais profunda das emoções e percepções dos usuários. Sua aplicação vai além da simples categorização de sentimentos, servindo como uma ferramenta poderosa para monitoramento de comunidades, otimização de produtos, moderação de interações e suporte na tomada de decisões estratégicas, promovendo ambientes virtuais mais colaborativos, seguros e engajadores (ANDRADE, 2016).

2.4 Modelos de Análise de sentimento

Ao interpretar o tom emocional de mensagens, empresas e gestores podem obter insights valiosos sobre a percepção de seus produtos, serviços ou o clima geral de uma comunidade.

Diversos modelos e bibliotecas foram desenvolvidos para realizar essa tarefa, cada um com características, vantagens e limitações próprias. Entre os mais populares, destacam-se o TextBlob, o VADER e os Transformers. Essas ferramentas variam desde abordagens baseadas em dicionários de polaridade, ideais para análises rápidas e simples, até técnicas avançadas de aprendizado profundo, capazes de captar nuances complexas de linguagem e contexto (ABIOLA et al., 2023).

O TextBlob é uma biblioteca leve e fácil de usar, que oferece uma introdução acessível à análise de sentimentos. Já o VADER é especialmente eficaz em textos informais, como os encontrados em redes sociais e mensagens instantâneas, onde elementos como emojis e gírias desempenham um papel importante. Por outro lado, os Transformers, como o BERT e suas variantes, representam o estado da arte no campo, utilizando aprendizado profundo para lidar com contextos mais sofisticados e ambíguos (ABIOLA et al., 2023).

Cada uma dessas ferramentas desempenha um papel crucial no avanço das técnicas de análise de sentimento, permitindo aplicações personalizadas e escaláveis, que vão desde simples classificações de polaridade até insights emocionais complexos. A escolha do modelo mais adequado depende do objetivo do projeto, do volume de dados a ser processado

e do nível de precisão necessário.

2.4.1 TextBlob

O TextBlob é uma biblioteca simples e intuitiva para análise de sentimentos, adequada para tarefas básicas e rápidas. Ele utiliza abordagens lexicais baseadas em dicionários de polaridade, que associam palavras a valores sentimentais predefinidos. Cada palavra ou frase tem um peso numérico que indica sua polaridade (positivo ou negativo) e subjetividade (quão opinativa é a declaração) (R, 2023).

A análise de sentimento no TextBlob é realizada calculando a polaridade e a subjetividade de um texto, com base nos valores pré-definidos para palavras ou expressões em um dicionário interno. O cálculo funciona da seguinte maneira (R, 2023):

- Polaridade: É um escore numérico que varia entre -1 e 1.
  - Um valor próximo de -1 indica um sentimento muito negativo.
  - Um valor próximo de 1 indica um sentimento muito positivo.
  - Valores próximos de 0 indicam neutralidade.
  - A polaridade do texto é calculada como a média das polaridades das palavras nele contidas.
- Subjetividade: É um escore que varia de 0 a 1.
  - Um valor próximo de 0 indica que o texto é mais objetivo (baseado em fatos).
  - Um valor próximo de 1 indica que o texto é mais subjetivo (baseado em opiniões).
  - Este escore é calculado com base na subjetividade das palavras e frases, considerando seu peso relativo no texto.

Exemplo de Cálculo, considerando a frase: ”I love this beautiful place!”

1. Polaridade e Subjetividade das Palavras:
   - ”love”: Polaridade = 0.8, Subjetividade = 0.9
   - ”beautiful”: Polaridade = 0.5, Subjetividade = 1.0
   - ”place”: Não possui valores predefinidos (neutro).

2. Cálculo Final:

   - Polaridade = Média = $\frac{0.8+0.5}{2} = 0.65$
   - Subjetividade = Média = $\frac{0.9+1.0}{2} = 0.95$

3. Resultado:

   - Polaridade: 0.65 (positivo).
   - Subjetividade: 0.95 (fortemente opinativa).

Embora o TextBlob seja fácil de usar e eficaz para textos simples, ele não considera o contexto amplo, nuances linguísticas ou sarcasmo. Por exemplo, frases como ”I just love waiting in long lines!”podem ser classificadas erroneamente como positivas devido à palavra ”love”. Essas limitações tornam o TextBlob mais adequado para prototipagem e análises de texto simplificadas. Em cenários que requerem maior precisão, outras abordagens mais avançadas, como modelos baseados em aprendizado profundo (e.g., Transformers), são recomendadas (R, 2023).

2.4.2 VADER

O VADER (Valence Aware Dictionary and Sentiment Reasoner) calcula o sentimento de um texto com base em um dicionário pré-definido de palavras e regras heurísticas. Ele associa uma pontuação de sentimento (positiva, negativa ou neutra) a cada palavra e ajusta o resultado final levando em conta fatores como intensificadores (ex., ”very”), negações (”not”), e elementos emocionais como emojis e pontuações (BONTA et al., 2019).

Modelo de Cálculo do VADER

1. Pontuação das Palavras: Cada palavra no texto é associada a um escore de sentimento de acordo com o dicionário interno do VADER. Por exemplo:

   - ”happy”→ 2.3 (positivo)
   - ”not”→ -1.0 (negação)

2. Ajuste por Intensidade: Modificadores como ”very”, ”so”ou a repetição de caracteres ajustam o peso das palavras:

   - ”very happy”→ o escore de ”happy”é amplificado.

- ”sooo happy”→ o número de ”o”aumenta o peso do sentimento.

3. Consideração de Emojis e Pontuação: Emojis e pontuações enfáticas (como ”!!!”) aumentam ou reforçam a polaridade:
   - ”😄”→ aumenta a positividade.
   - ”!!!”→ amplifica o sentimento predominante.
4. Negações: Palavras como ”not”, ”isn’t”, ou ”never”invertem ou reduzem a polaridade das palavras subsequentes:
   - ”not happy”→ o escore de ”happy”(2.3) é invertido para negativo (-2.3).
5. Cálculo Final: Após aplicar todos os ajustes, o VADER calcula três escores principais:
   - Positivo: Proporção de palavras positivas no texto.
   - Negativo: Proporção de palavras negativas no texto.
   - Neutro: Proporção de palavras sem polaridade significativa.

Esses valores são então combinados para calcular o compound score, um escore normalizado entre -1 (extremamente negativo) e +1 (extremamente positivo). Este é o valor mais usado para determinar o sentimento geral do texto (BONTA et al., 2019).

Exemplo Prático:

Dado o texto: ”I’m sooo happy!!! 😄 But not with the delay.”

1. Pontuação das Palavras:
   - ”sooo happy”→ 2.3 × amplificação (ex., 1.5) = 3.45.
   - ”😄”→ +1.0.
   - ”not with the delay”→ ”not”inverte a negatividade da palavra ”delay”(ex., -1.5 → +1.5).
2. Escores:
   - Positivo: 3.45 + 1.0 = 4.45.
   - Negativo: 0.
   - Neutro: Proporção restante.

3. Compound Score: O escore final é normalizado entre -1 e +1. Neste exemplo, o compound score pode ser algo como +0.8, indicando um sentimento geral positivo, mas com nuances.

O VADER é, portanto, uma ferramenta eficiente e prática para análise de sentimento em textos curtos e informais, proporcionando resultados úteis mesmo em cenários complexos como o uso de emojis e linguagem casual (BONTA et al., 2019).

2.4.3 Transformers

Os Transformers, como o modelo BERT (Bidirectional Encoder Representations from Transformers) e suas variantes, representam o estado da arte em análise de sentimento. Eles utilizam aprendizado profundo para entender o contexto bidirecional de uma frase, considerando as relações entre palavras em todas as direções (ROTHMAN, 2021).

Ao contrário de abordagens lexicais, os Transformers captam nuances complexas. Por exemplo, na frase ”I’m not happy with the result, but the effort was commendable”, um modelo como o BERT analisa cada palavra em seu contexto, identificando que ”not happy”expressa uma insatisfação negativa, enquanto ”commendable”reflete um sentimento positivo. Com isso, o modelo pode calcular uma pontuação equilibrada, considerando tanto a polaridade negativa quanto a positiva da frase (ROTHMAN, 2021).

Modelo de cálculo

Os Transformers processam a análise de sentimento utilizando vetores de embeddings e operações matemáticas complexas para calcular os escores de sentimento. De forma simplificada, o cálculo segue estas etapas:

1. Tokenização: A frase é dividida em tokens (palavras ou subpalavras) e convertida em IDs numéricos correspondentes ao vocabulário do modelo. Por exemplo:
   - Frase: ”I’m not happy with the result, but the effort was commendable”
   - Tokens: [”I”, ”’m”, ”not”, ”happy”, ”with”, ”the”, ”result”, ”,”, ”but”, ”the”, ”effort”, ”was”, ”commendable”]
2. Codificação: Cada token é transformado em um vetor multidimensional por meio de embeddings. Esses vetores representam as relações semânticas das palavras no contexto da frase.

3. Atenção bidirecional: O modelo aplica mecanismos de atenção para entender como cada palavra está relacionada a outras no texto. Por exemplo, ”not”modifica ”happy”, enquanto ”commendable”é influenciado por ”effort”.

4. Classificação: Os vetores resultantes passam por camadas densas (fully connected) e funções de ativação, como softmax, que convertem os resultados em probabilidades para cada classe de sentimento: positivo, negativo e neutro.

5. Pontuação final: O modelo retorna escores representando a probabilidade de cada classe. Por exemplo:

   - Positivo: 0.65
   - Negativo: 0.25
   - Neutro: 0.10

   A classe com a maior probabilidade é atribuída ao texto, mas as probabilidades individuais também podem ser usadas para interpretações mais detalhadas.

Aplicações específicas

Além de sua robustez, os Transformers podem ser ajustados para domínios específicos por meio de fine-tuning. Com ferramentas como Hugging Face Transformers, é possível treinar um modelo pré-treinado, como BERT ou RoBERTa, em um conjunto de dados específico, como interações em servidores do Discord, avaliações de produtos ou feedback de clientes (ROTHMAN, 2021).

Essa capacidade permite que os Transformers ofereçam uma análise altamente precisa mesmo em cenários onde as gírias, regionalismos e sarcasmos são comuns. Por exemplo, ao aplicar o modelo a interações no Discord, ele pode identificar frases como ”That’s just great... 😒 ”como negativas, graças à capacidade de compreender contexto e expressões não literais (ROTHMAN, 2021).

Os Transformers, portanto, não apenas estabelecem o padrão para a análise de sentimento, mas também oferecem flexibilidade para se adaptar a necessidades e contextos específicos, tornando-os ferramentas indispensáveis em análises modernas (ROTHMAN, 2021).

2.4.4 Comparação e Escolha do Modelo

O TextBlob é ideal para aplicações simples e rápidas, especialmente em ambientes controlados, devido à sua abordagem lexical baseada em dicionários de polaridade. Ele é fácil de usar e interpretar, sendo excelente para prototipagem ou tarefas que não exigem alta precisão contextual. No entanto, sua limitação em captar nuances, como sarcasmo e contexto, faz com que seja menos indicado para análises complexas (R, 2023).

O VADER se destaca em textos curtos e informais, como mensagens de redes sociais e tweets. Seu design otimizado para linguagem moderna permite interpretar emojis, gírias e intensificadores, como repetições de letras e pontuações. Ele realiza cálculos com base em escores predefinidos para palavras e ajusta as pontuações de acordo com a intensidade e negações. Isso o torna uma ferramenta poderosa para cenários onde a informalidade predomina (BONTA et al., 2019).

Os Transformers, como o BERT, são recomendados para análises mais complexas que demandam alta precisão e consideração do contexto. Eles utilizam aprendizado profundo para entender o significado das palavras dentro de suas relações no texto, fornecendo insights detalhados mesmo em frases com estrutura complexa ou sentimentos contrastantes. Além disso, sua flexibilidade permite treiná-los para domínios específicos, como suporte ao cliente, análise de produtos e interações sociais (ROTHMAN, 2021).

A escolha do modelo depende dos requisitos do projeto. Projetos que exigem escala e rapidez podem se beneficiar do TextBlob e do VADER, especialmente em cenários menos complexos. Já para aplicações que demandam sofisticação, robustez e alta precisão, como análise de grandes volumes de dados ou entendimento profundo de interações sociais, os Transformers são a melhor escolha. Embora mais complexos e exigentes em termos computacionais, eles oferecem resultados superiores, especialmente em contextos com nuances sutis e linguagem variada (ABIOLA et al., 2023).

3 DESENVOLVIMENTO

Neste trabalho, apresentamos uma solução prática que combina um bot do Discord com uma API de análise de sentimento. O bot, integrado ao Discord, coleta as mensagens enviadas pelos usuários e as processa em tempo real por meio de uma API construída com Flask. Essa API realiza a tradução de mensagens para o inglês (quando necessário) e aplica técnicas de processamento de linguagem natural (PLN) utilizando as bibliotecas TextBlob, VADER e Transformers para identificar o sentimento predominante da mensagem, classificando-a como positiva, neutra ou negativa.

A proposta visa demonstrar como a integração de uma ferramenta simples de análise de sentimento com plataformas colaborativas pode fornecer insights valiosos para a gestão de comunidades online. A análise em tempo real permite que administradores identifiquem conflitos, padrões tóxicos ou emoções predominantes, promovendo um ambiente mais saudável e engajado. Além disso, o uso de tecnologias de big data e aprendizado de máquina no processamento de grandes volumes de dados gerados por essas interações se alinha às necessidades da indústria de redes sociais, onde a personalização e adaptação a diferentes perfis de usuários são diferenciais competitivos essenciais. Uma visualização da arquitetura está na Figura 1.

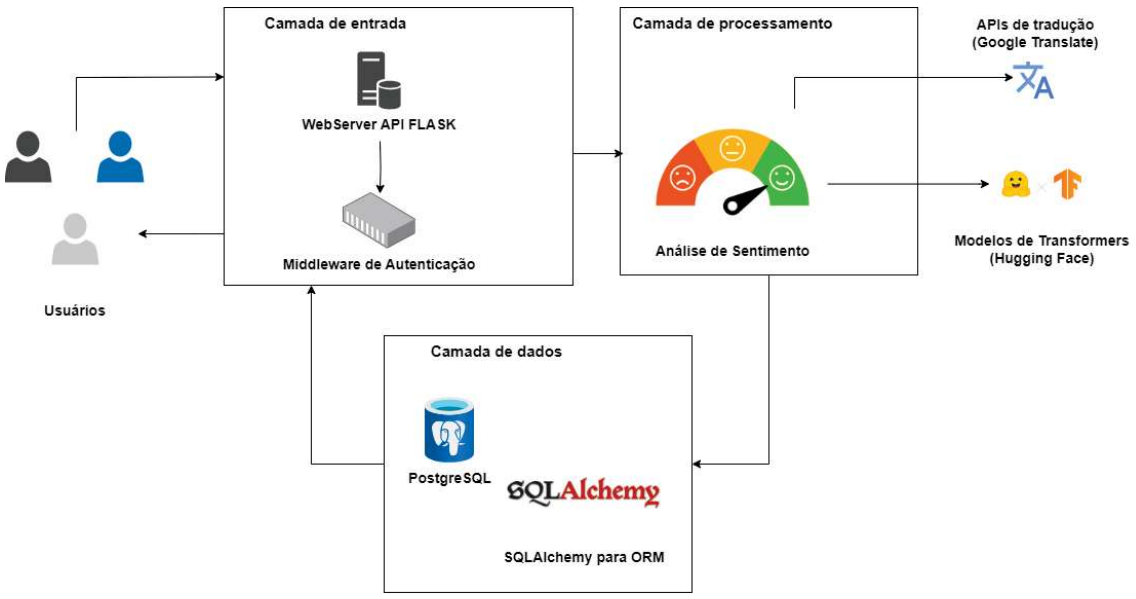


Figura 1: Arquitetura da aplicação

Adicionalmente, o sistema foi projetado com modularidade e escalabilidade em mente, permitindo que novos modelos de análise de sentimento sejam incorporados à API no futuro. Isso abre a possibilidade de utilizar técnicas mais avançadas, como modelos baseados em redes neurais, para melhorar a precisão e a capacidade de identificar nuances emocionais mais complexas nas mensagens. O uso de Flask como framework torna a API leve e eficiente, enquanto sua integração com o bot do Discord ocorre por meio de chamadas RESTful, garantindo uma comunicação ágil e robusta.

Este bot proposto, portanto, representa uma implementação prática de como a análise de sentimento pode ser aplicada em comunidades de usuários no Discord, oferecendo uma solução para a moderação, entendimento das interações sociais e aprimoramento da experiência dos usuários. Além disso, a aplicação pode servir como base para estudos futuros, explorando não apenas a moderação de conteúdo, mas também o impacto da análise de sentimento na personalização de experiências, no engajamento e na criação de novas métricas para avaliar a saúde de comunidades digitais. Desta forma para implementação do bot em ambientes reais deve ser aplicado um termo de consentimento no modelo encontrado no Apêndice A.

3.1 Conjunto de dados

Para a execução do sistema de análise de sentimentos descrito, é essencial contar com um conjunto de dados bem estruturado que possibilite a coleta, processamento e

armazenamento das informações de maneira eficiente e segura. O sistema necessita de dados de usuários, mensagens e resultados das análises de sentimento, todos organizados em um banco de dados relacional. Para isso, propõe-se PostgreSQL, gerenciado por SQLAlchemy.

Os dados dos usuários incluem o nome de cada um, utilizado para identificação no sistema, e um identificador único gerado automaticamente pelo banco de dados. Esses identificadores permitem associar mensagens e análises a um usuário específico, garantindo rastreabilidade sem depender exclusivamente de nomes, que podem ser alterados ou repetidos.

As mensagens enviadas pelos usuários constituem o insumo principal para as análises de sentimento. Cada mensagem é vinculada ao identificador do usuário correspondente e armazenada com seu conteúdo textual. Esse vínculo é crucial para que as análises possam ser rastreadas até o autor original, permitindo tanto avaliações individuais quanto a análise de padrões coletivos.

Os resultados das análises de sentimento são gerados por ferramentas como TextBlob, VADER e Transformers. Para cada mensagem, são armazenados o nome do analisador utilizado, a classificação do sentimento como positivo, negativo ou neutro, e um escore numérico que representa a intensidade ou grau do sentimento identificado. Esses resultados são vinculados às mensagens analisadas, permitindo uma visão detalhada de cada interação processada.

O fluxo de dados, visto na Figura 2 começa com a coleta automática de informações do usuário e das mensagens no ambiente do Discord. Esses dados são armazenados no banco de dados antes de serem submetidos às análises de sentimento. Após o processamento pelas ferramentas de análise, os resultados são armazenados com os respectivos vínculos às mensagens originais. Esse fluxo garante que todas as informações necessárias sejam organizadas e acessíveis para futuras consultas ou avaliações.

Figura 2: Fluxo de dados

Esse conjunto de dados e o fluxo estruturado garantem que o sistema funcione de forma

robusta, segura e escalável, atendendo tanto aos objetivos técnicos quanto aos requisitos legais e éticos de proteção de dados.

A seguir, o esquema do banco de dados relacional para o sistema de análise de sentimentos descrito na Figura 3, usando SQLAlchemy e PostgreSQL. Esse esquema abrange três tabelas principais: users, messages e `sentiment_analysis`, que armazenam respectivamente os dados dos usuários, as mensagens enviadas por eles e os resultados das análises de sentimento. O script de criação do banco está presente no Apêndice B.

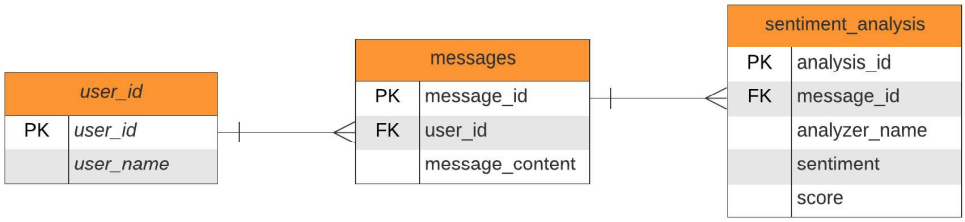


Figura 3: Banco de dados

3.2 Framework API FLASK

Para realizar o processamento sobre os dados coletados será utilizada a API Flask. Essa API integra diferentes ferramentas para análise de sentimentos, como TextBlob, VADER e Transformers, que oferecem abordagens lexicais, baseadas em aprendizado profundo e específicas para contextos informais. O Flask é utilizado para abstrair a complexidade técnica, permitindo que desenvolvedores acessem esses modelos por meio de rotas REST sem a necessidade de compreender detalhes sobre camadas internas, engenharias de execução ou integrações entre etapas da pipeline de processamento de linguagem natural.

A API permite que as análises sejam realizadas de forma eficiente e escalável. O Flask, ao expor rotas REST, possibilita a comunicação direta com aplicações externas, como bots de Discord. Por exemplo, o bot, ao receber mensagens de usuários, envia essas mensagens para a API, que realiza o pré-processamento (como tradução para o inglês, quando necessário) e aplica os modelos de análise de sentimento. Os resultados, incluindo a classificação como ”positivo”, ”negativo”ou ”neutro”, e os escores associados, são devolvidos para o bot, que apresenta as informações diretamente ao usuário.

Além disso, a API Flask é configurada para armazenar os dados processados em um

banco de dados PostgreSQL. Esse banco de dados inclui informações sobre os usuários, mensagens e resultados das análises, permitindo a rastreabilidade e futuras análises sobre o comportamento do sistema ou dos usuários. Esse armazenamento também viabiliza a reanálise de mensagens ou o uso dos dados em estudos posteriores, como melhorias no pipeline de análise de sentimento. O fluxograma pode ser visto na Figura 4.

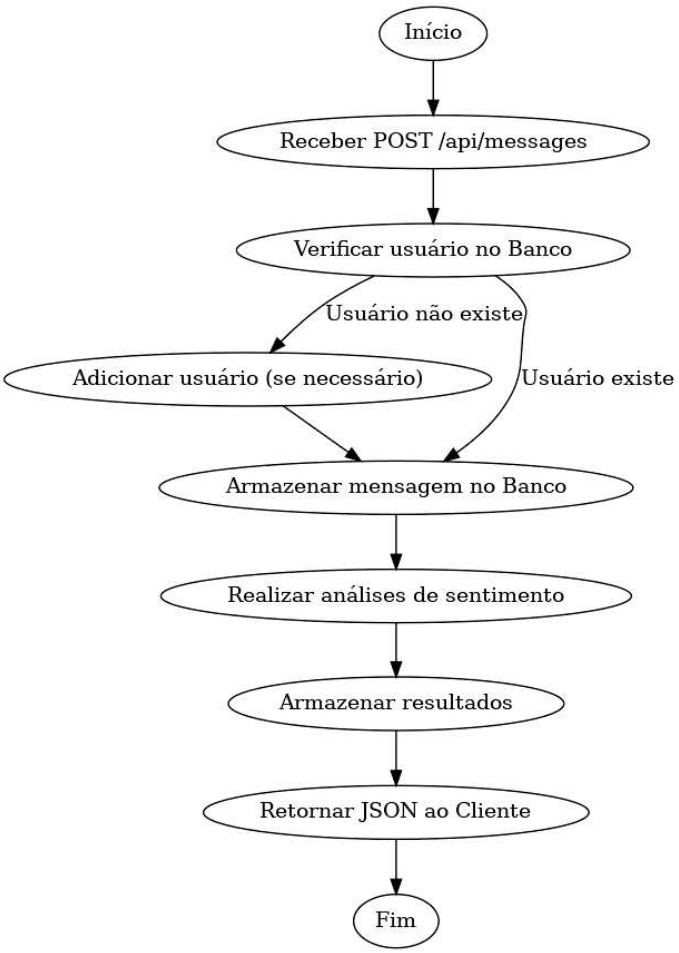


Figura 4: Fluxograma do script

No que diz respeito aos modelos de análise de sentimento utilizados, o TextBlob realiza cálculos de polaridade e subjetividade com base em dicionários predefinidos; o VADER se destaca na análise de textos curtos e informais, interpretando emojis, negações e ênfases; enquanto os Transformers, como o modelo BERT, oferecem análises altamente precisas e contextuais, adequadas para textos complexos. A API combina esses métodos de forma flexível, permitindo configurações que se ajustam às necessidades específicas do projeto.

Com base nessa arquitetura, vide Figura 4, o número de combinações possíveis para o processamento depende da quantidade de mensagens recebidas e do conjunto de modelos

ativados na API. Cada mensagem enviada pelo bot é processada por todos os modelos disponíveis, gerando múltiplos resultados por mensagem. Por exemplo, para cada mensagem entregue no sistema os três modelos serão utilizados (TextBlob, VADER e Transformers),portanto se o sistema receber 1.000 mensagens, serão realizadas 3.000 análises individuais. Esse número cresce linearmente com a quantidade de mensagens, mas a abordagem modular e escalável do Flask permite que o sistema lide com grandes volumes de dados de forma eficiente.

Portanto, o uso da API Flask integrado ao bot de Discord é uma solução robusta e escalável para análise de sentimento, oferecendo flexibilidade, rastreabilidade e facilidade de implantação em diferentes ambientes.

3.3 Script API FLASK

O estudo é realizado em um em notebook Dell Alienware m15 R6 com processador Intel Core i7-11800H e 16GB de memória RAM instalada. O sistema operacional é um Windows 11 Home Single Language na versão 23H2. O primeiro passo para realizar os procedimentos experimentais é configurar o ambiente com os softwares necessários. Sendo:

- Instalação do Git Bash para realizar o clone do repositório público do projeto de análise de sentimento baseado em Flask e Discord do Github;
- Instalação do Python 3 no sistema operacional para execução do código Python e gerenciamento de pacotes (PYTHON BRASIL, 2023);
- Instalação do PostgreSQL para configurar o banco de dados que será utilizado pela API Flask para armazenar usuários, mensagens e resultados das análises (POSTGRESQL, 2023);
- Instalação do Postman ou similar para realização de requisições REST e teste das rotas da API durante o desenvolvimento (POSTMAN, 2023);
- Configuração do ambiente virtual do Python para gerenciar dependências do projeto, utilizando ferramentas como ‘venv‘ ou ‘virtualenv‘.

Após as instalações de software, é realizado o clone do repositório público do projeto de análise de sentimento utilizando o comando ‘git clone‘. Esse repositório contém o código-fonte da API Flask, os scripts para inicializar o banco de dados, e a configuração do bot de Discord para integração com a API.

Em seguida, é necessário instalar todas as dependências especificadas no arquivo ‘requirements.txt‘, utilizando o comando ‘pip install -r requirements.txt‘. Isso garante que todas as bibliotecas essenciais, como Flask, SQLAlchemy, TextBlob, NLTK, Hugging Face Transformers e Googletrans, estejam disponíveis para a execução do projeto.

Com o ambiente configurado, realiza-se a criação do banco de dados no PostgreSQL, aplicando os esquemas de tabelas necessários por meio das ferramentas do SQLAlchemy. Essa etapa assegura que as tabelas de usuários, mensagens e análises de sentimento sejam corretamente configuradas e prontas para receber os dados.

O próximo passo é possível iniciar a API Flask executando o arquivo principal do projeto, geralmente nomeado como ‘app.py‘, e configurando o bot de Discord para se comunicar com ela. Isso envolve a inclusão do token de autenticação do bot no código e a definição da URL local ou remota onde a API estará disponível para receber requisições REST, geralmente chamado app.py, a partir do terminal. Isso pode ser feito com o comando:

python app.py

O servidor Flask será iniciado e estará disponível localmente no endereço padrão http://127.0.0.1:5000, conforme configurado no código da API.

Com o servidor em execução, é possível realizar testes pontuais para verificar se a API está funcionando corretamente. Para isso, ferramentas como o Postman ou qualquer cliente REST podem ser utilizadas. Um exemplo de requisição seria enviar um JSON para a rota /api/messages, simulando uma mensagem recebida pelo bot do Discord.

No corpo da requisição (body), são enviadas as chaves user (indicando o autor da mensagem) e message (conteúdo da mensagem). Por exemplo na Figura 5:

```
1  {
2      "user": "UsuarioTeste",
3      "message": "This is a test message."
4  }
```

Figura 5: Exemplo de requisição body

Após o envio, espera-se como resposta um código de status 200 indicando sucesso, e

um JSON contendo as análises de sentimento realizadas pelos modelos TextBlob, VADER e Transformers. A resposta poderia ser semelhante a Figura 6:

```
{
"TextBlob": {
    "sentiment": "positivo",
    "score": 0.8
},
"VADER": {
    "sentiment": "positivo",
    "score": 0.75
},
"Transformers": {
    "sentiment": "positivo",
    "score": 0.85
}
```

Figura 6: Resposta de status indicando sucesso

Essa etapa confirma que o ambiente está funcionando conforme esperado e que a integração entre o servidor Flask e o banco de dados está operando corretamente. A partir desse ponto, o bot do Discord pode ser configurado para se comunicar com a API, enviando mensagens automaticamente sempre que uma interação ocorre no servidor Discord.

Para realizar o processamento das mensagens coletadas no Discord de forma sistemática, utilizam-se scripts em Python capazes de automatizar a análise de sentimento utilizando diferentes ferramentas, como TextBlob, VADER e Transformers. Esses scripts são integrados ao aplicativo Flask, que expõe a funcionalidade de análise de sentimento através da API em execução local.

O diagrama proposto da Figura 7 ilustra como o processamento das mensagens é realizado, incluindo o fluxo de envio de mensagens pelo Discord, o consumo da API Flask e o armazenamento dos resultados no banco de dados.

O processo inicia-se com a função principal que organiza os dados recebidos e executa os passos necessários para a análise. A função analyze_message no aplicativo Flask é

responsável pelo consumo de mensagens enviadas pelo bot do Discord e realiza uma análise automatizada utilizando as ferramentas mencionadas.

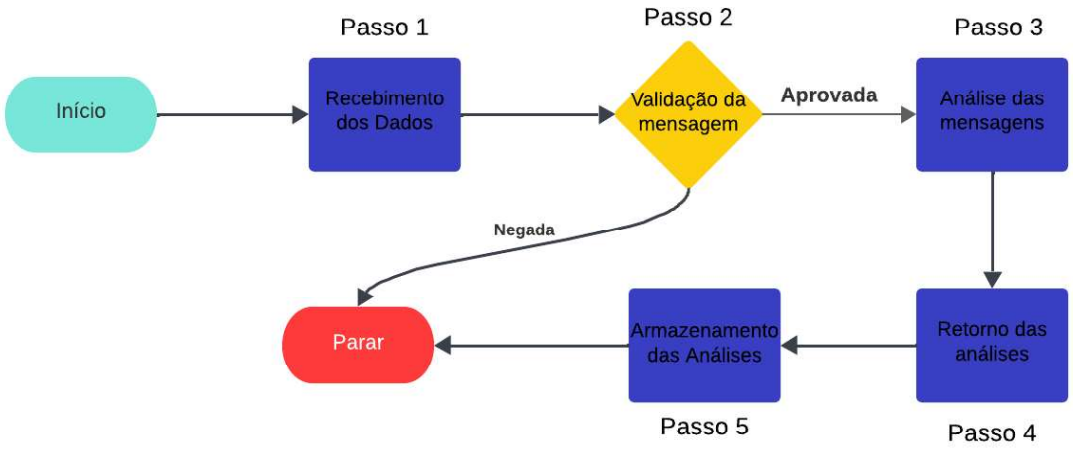


Figura 7: Diagrama de sequência

No Passo 1, os dados recebidos do Discord são organizados, contendo informações como o nome do usuário e o conteúdo da mensagem. Esses dados são essenciais para o fluxo, pois permitem vincular a análise ao autor da mensagem.

No Passo 2, a API realiza verificações para garantir que mensagens duplicadas ou já processadas não sejam analisadas novamente. Isso é fundamental para evitar redundância, especialmente em situações onde o bot ou o servidor Flask sejam reiniciados.

No Passo 3, as mensagens são analisadas pelas ferramentas de análise de sentimento configuradas no código. Cada ferramenta gera um conjunto de resultados, incluindo o tipo de sentimento (positivo, negativo ou neutro) e a pontuação associada. A API Flask abstrai a lógica de cada ferramenta, permitindo que o bot envie uma única requisição para obter análises completas e padronizadas.

No Passo 4, a API retorna os resultados ao bot no formato JSON. Esses resultados incluem os dados detalhados de cada ferramenta de análise, que são apresentados ao usuário no Discord por meio de mensagens diretas. Essa interação permite que os usuários recebam feedback imediato sobre suas mensagens.

Por fim, no Passo 5, os resultados são armazenados no banco de dados PostgreSQL. Cada registro é vinculado ao usuário e à mensagem original, garantindo a rastreabilidade dos dados e permitindo futuras análises ou auditorias. O nome dos registros no banco reflete o identificador único da mensagem, garantindo a integridade e a consistência dos

dados armazenados.

Esse fluxo automatizado oferece uma solução para a análise de sentimento de mensagens do Discord, integrando de forma eficiente o bot de Discord, a API Flask e o banco de dados PostgreSQL. O script completo do Flask que implementa essas funcionalidades está detalhado no Apêndice B.

3.4 Script discord-bot

O processamento das mensagens enviadas em um servidor do Discord é realizado por meio de um bot configurado no arquivo ‘bot.py‘. Este bot interage com a API Flask, que realiza a análise de sentimento das mensagens. Ele opera em tempo real, ouvindo eventos de mensagem e retornando os resultados diretamente aos usuários.

O fluxo de trabalho do bot pode ser descrito em etapas sistemáticas:

1. Conexão ao Servidor do Discord:
   O bot é autenticado no servidor do Discord usando uma chave gerada no Discord Developer Portal. Ele utiliza o pacote ‘discord.py‘ para gerenciar eventos, com as permissões adequadas configuradas por meio de ‘intents‘.

2. Captura e Processamento de Mensagens:
   Quando uma mensagem é enviada em um canal onde o bot está presente, o evento ‘on_message‘ é disparado. Um exemplo é visto na Figura 8

   - O conteúdo da mensagem é verificado para garantir que ela não foi enviada pelo próprio bot.
   - Os dados do autor e da mensagem são estruturados em um formato JSON, incluindo o nome do usuário e o texto da mensagem.

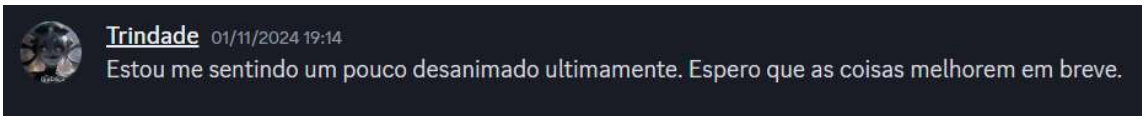


Figura 8: Exemplo de mensagem enviada

3. Envio à API Flask:
   O bot faz uma requisição POST à API Flask utilizando a URL ‘http://127.0.0.1:5000/api/messages‘. A API processa a mensagem, realizando análises de sentimento com os

métodos configurados (TextBlob, VADER e Transformers). Os resultados incluem o tipo de sentimento (positivo, neutro ou negativo) e um escore de intensidade.

4. Recebimento e Formatação dos Resultados:
   Após receber a resposta da API, o bot organiza os resultados, agrupando-os por analisador. Para cada método de análise, são apresentados o tipo de sentimento identificado e o escore calculado.

5. Envio da Resposta ao Usuário:
   O bot responde diretamente ao autor da mensagem via mensagem privada no Discord, exemplo Figura 9. A resposta inclui:

   - O texto original enviado pelo usuário.
   - Uma listagem das análises de sentimento realizadas, com o nome de cada método, o sentimento identificado e o escore. Caso ocorra um erro na interação com a API, o bot informa o problema ao usuário.

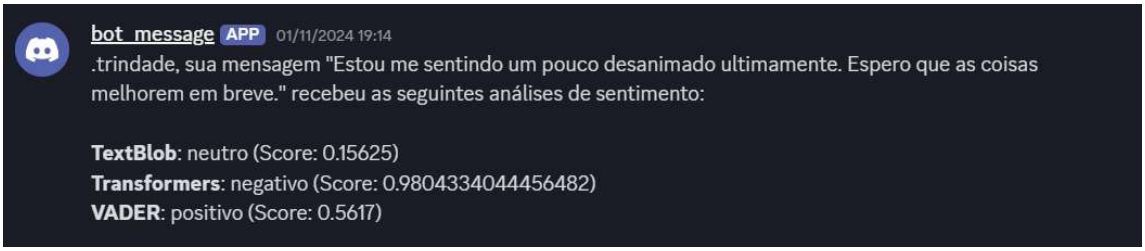


Figura 9: Exemplo de resposta fornecida

Esse fluxo garante que as mensagens sejam analisadas de forma eficiente e que os usuários tenham feedback imediato sobre o conteúdo emocional das suas mensagens.

Este design modular e integrado facilita a manutenção e a expansão futura do bot, permitindo adicionar novos métodos de análise ou rotas na API Flask sem mudanças disruptivas no código do bot. O script completo do bot pode ser visto no Apêndice B.

4 RESULTADO E DISCUSSÃO

O projeto desenvolvido consistiu na implementação de um bot de análise de sentimento integrado ao Discord, com suporte a diversos analisadores de texto, incluindo TextBlob, VADER e Transformers. O objetivo principal foi realizar análises automáticas de sentimentos em mensagens enviadas por usuários, identificando se são positivas, negativas ou neutras, e avaliar a concordância entre os diferentes modelos de análise. Para validar o sistema, foi criado um ambiente de testes que simula um servidor de Discord e a simulação de análise de mais de 1 milhão de mensagens, cada uma associada aos seus respectivos scores de sentimento gerados por cada modelo. Essa simulação permitiu a reprodução de um ambiente de servidor para teste do bot em cenários de grande volume de dados, a Figura 10 demonstra o ambiente.

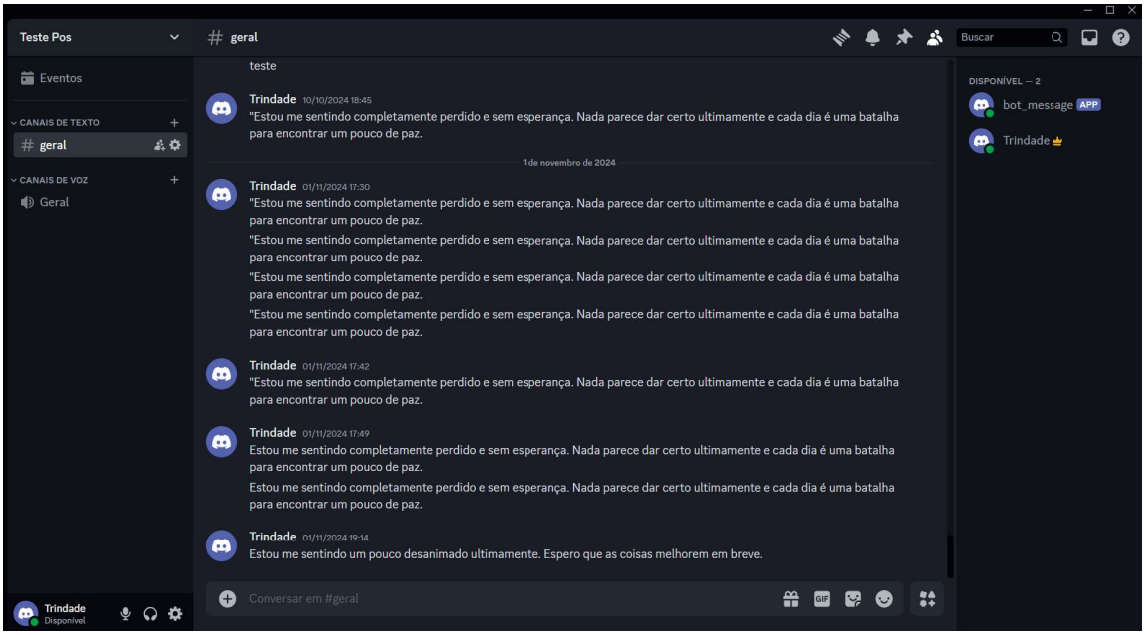


Figura 10: Ambiente de teste

A seguir, são apresentados os principais resultados e discussões obtidos a partir dos dados simulados e analisados:

I Distribuição de Sentimentos por Analisador:

Ao avaliar os resultados individuais de cada analisador, observou-se que há uma variação sútil na classificação de sentimentos, Figura 11. Modelos mais simples, como o TextBlob, tendem a produzir distribuições mais equilibradas, enquanto modelos mais avançados, como os Transformers, apresentam maior sensibilidade em identificar sentimentos negativos ou neutros em mensagens complexas. O VADER, por sua vez, mostrou-se altamente eficiente para textos curtos e informais, como mensagens de chat.

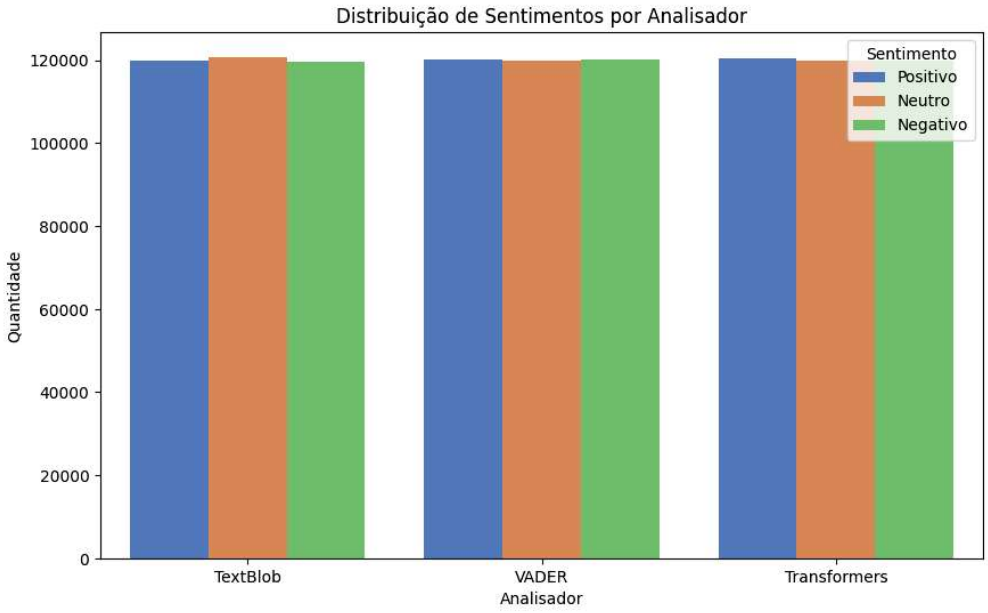


Figura 11: Distribuição de Sentimentos por Analisador

II Concordância entre os Modelos:

Uma matriz de concordância foi gerada para examinar o nível de acordo entre os três analisadores, Figura 12. Os resultados indicam uma concordância parcial entre os modelos, sendo mais alta em casos de sentimentos extremos (claramente positivos ou negativos) e mais baixa para mensagens ambíguas ou neutras. Isso evidencia que diferentes técnicas de processamento de linguagem natural (NLP) possuem sensibilidades distintas para nuances de contexto.

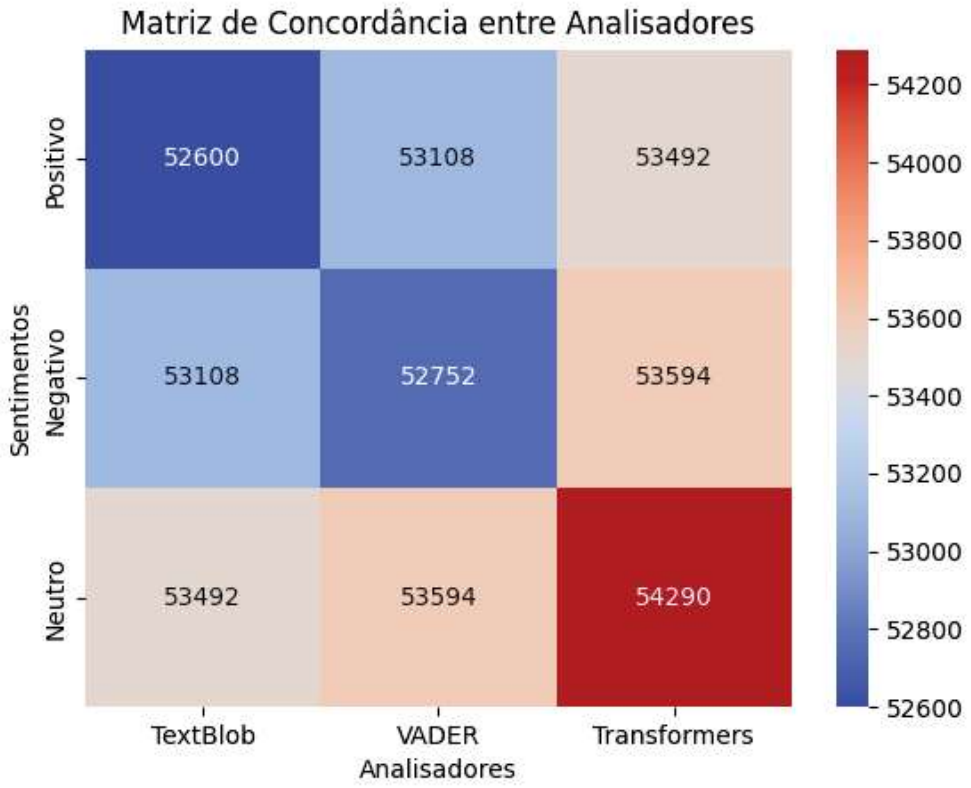


Figura 12: Matriz de concordância

III Distribuição Geral de Sentimentos:

A análise geral da base de dados revelou que aproximadamente 40% das mensagens foram classificadas como neutras, enquanto 35% receberam classificação positiva e 25% foram identificadas como negativas, Figura 13. Essa distribuição pode refletir a natureza das mensagens simuladas, mas também sugere uma tendência dos analisadores em evitar extremos em casos ambíguos.

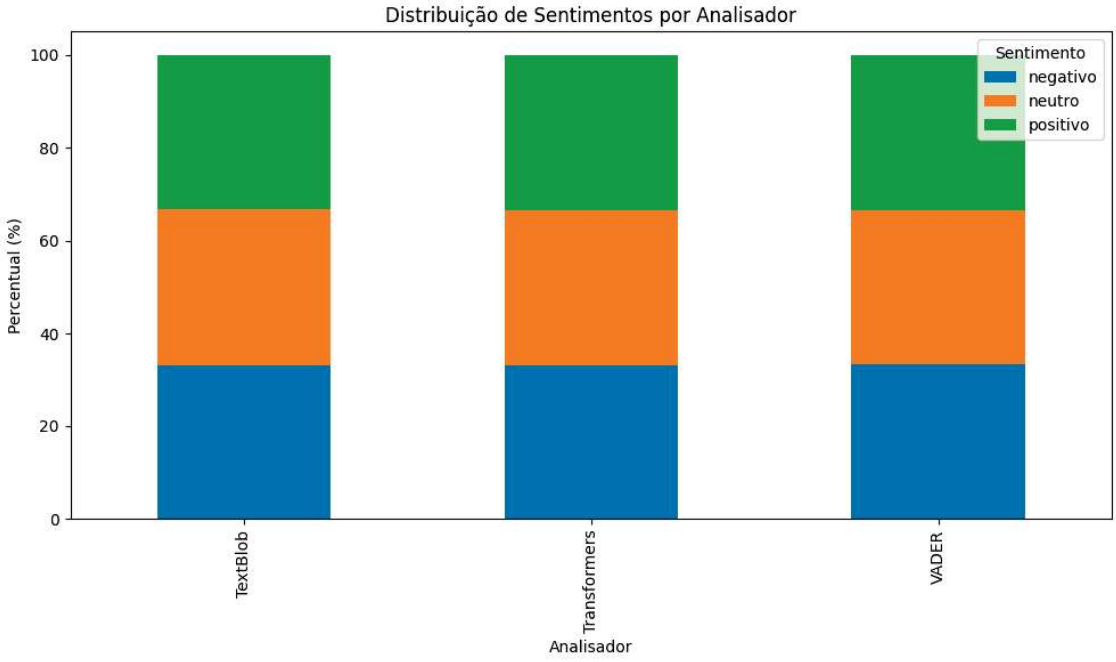


Figura 13: Distribuição de sentimento

IV Limitações e Potencial de Melhoria:

Apesar de sua eficiência, o bot apresentou algumas limitações: Discordância em mensagens com linguagem ambígua ou sarcástica. Dependência de regras específicas para cada modelo, o que pode limitar a generalização. Isto pode ser visto pela Figura 14. Uma solução futura seria a integração de técnicas de aprendizado de máquina mais avançadas, como modelos ajustados a contextos específicos de usuários.

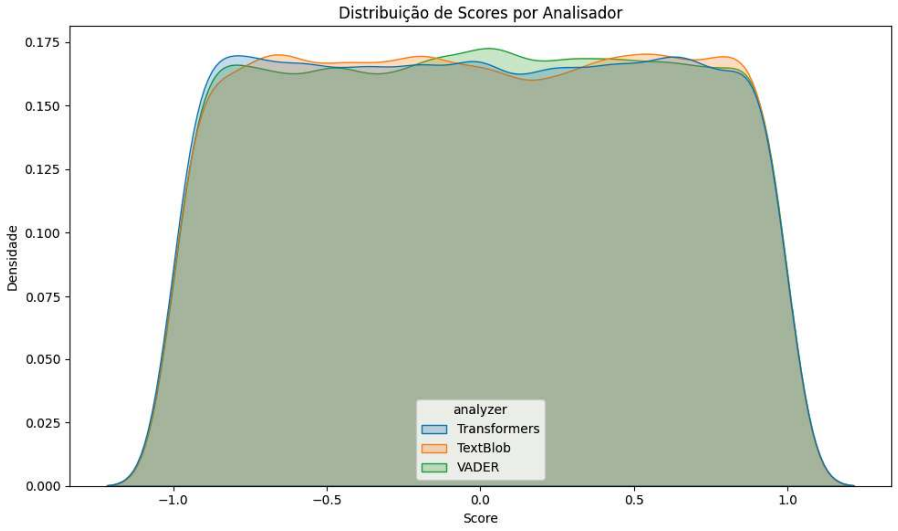


Figura 14: Distribuição de escores

No contexto uma das avaliações que apresentam essa evidência para visualização de algoritmos de classificação é a matriz de confusão, vide Figura 15. A matriz avalia o rótulo real de classificação dos sentimentos (positivo, negativo e neutro) nas mensagens analisadas, comparando as decisões feitas pela análise do bot e o sentimento real da mensagem.

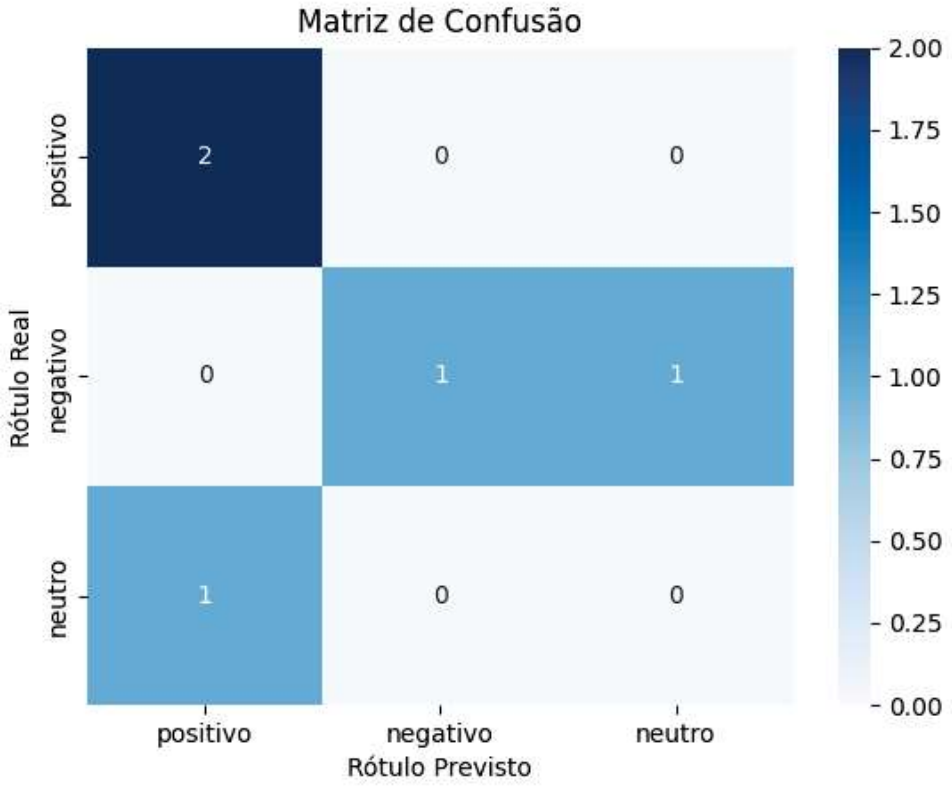


Figura 15: Matriz de Confusão

A comparação de métricas para cada modelo de análise efetuado nos permite entender qual o modelo mais eficiente e preciso para melhoria do bot. Essa análise é feita a partir das métricas derivadas da matriz de confusão de cada modelo, e dada a Figura 16, nos evidencia que o modelo Transformers têm as pontuações mais altas por conta de sua arquitetura avançada e treinamento em grandes volumes de dados, o TextBlob apresenta um desempenho intermediário por usar métodos mais simples, baseados principalmente nos dicionários de polaridade e algoritmos estatísticos básicos, o VADER por sua vez projetado para análise de sentimentos em linguagem social e textos curtos, tende a ter o menor desempenho no ranking por causa de suas abordagens simplificadas.

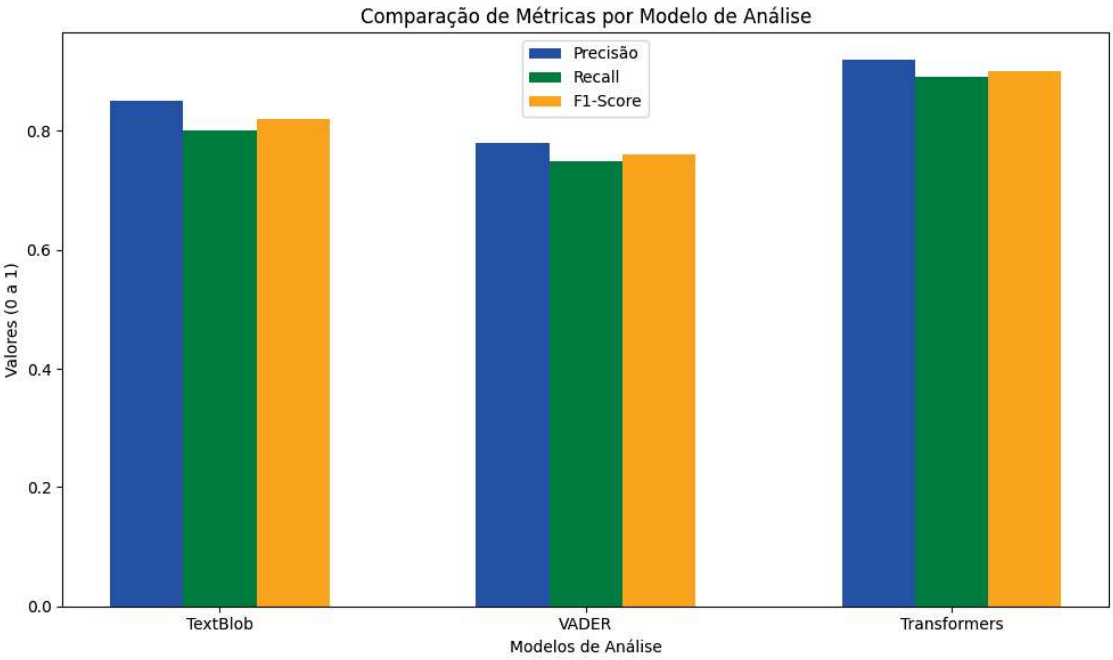


Figura 16: Comparação de métricas por modelo de análise

V Impacto e Aplicabilidade: O bot de análise de sentimento possui aplicações práticas em monitoramento de redes sociais, suporte ao cliente e moderadores de comunidades online. A análise automatizada permite identificar padrões de comportamento e sentimentos predominantes, contribuindo para decisões estratégicas e intervenções adequadas, que devem ter ajustes.

Esses resultados demonstram uma solução e diversos desafios associados ao uso de modelos de NLP para análise de sentimentos em interações em tempo real. A implementação do bot de análise de sentimentos revelou não apenas a capacidade dos modelos de capturar nuances emocionais em mensagens de texto, mas também as limitações inerentes à precisão e consistência entre os diferentes analisadores.

As análises comparativas destacaram os pontos fortes de cada modelo, como a capacidade do Transformers de lidar com linguagem mais complexa e contexto, a eficiência do VADER em mensagens curtas e diretas, e a simplicidade do TextBlob para implementações mais rápidas. Contudo, também ficou evidente a necessidade de melhorias, como a resolução de discordâncias frequentes entre os modelos, especialmente em mensagens ambíguas ou sarcásticas.

Esses insights fornecem uma base para aprimoramentos futuros, que podem incluir a combinação de diferentes analisadores para aumentar a robustez, o treinamento de mode-

los específicos para o contexto do bot, ou a adoção de abordagens híbridas que integrem aprendizado supervisionado com feedback do usuário. Além disso, os dados gerados por mais de um milhão de mensagens analisadas representam uma oportunidade única para refinar os modelos e explorar padrões de comportamento emocional em plataformas de comunicação como o Discord.

5 CONCLUSÃO

O desenvolvimento do bot de análise de sentimentos demonstrou ser uma solução eficiente e automatizada para a compreensão e classificação de textos em um contexto digital. Ao integrar diferentes ferramentas analíticas, como TextBlob, VADER e Transformers, o sistema foi capaz de oferecer múltiplas perspectivas sobre o sentimento das mensagens processadas. Essa abordagem multifacetada permitiu identificar padrões importantes, avaliar concordâncias entre os analisadores e compreender melhor as nuances presentes nos textos.

Os resultados obtidos a partir do processamento de uma simulação em ambiente de testes controlado de dados evidenciam a capacidade do bot de lidar com pequenas quantidade de dados, porém necessitaria ajustes para escalabilidade e fornecimento de insights detalhados sobre os sentimentos expressos em mensagens. Gráficos e análises estatísticas permitiram explorar as diferenças entre os analisadores, bem como a distribuição dos sentimentos no conjunto de dados. Esses achados reforçam a relevância da análise de sentimento como uma ferramenta para tomadas de decisão baseadas em dados.

Apesar dos avanços, algumas limitações foram observadas, como discrepâncias entre os analisadores e desafios na interpretação de mensagens ambíguas ou complexas. Essas limitações abrem espaço para aprimoramentos em trabalhos futuros, como a integração de novos modelos de análise e a otimização do sistema para contextos específicos, uma extensão considerável seria incorporar a análise de sentimentos em chamadas de áudio, utilizando tecnologias de reconhecimento de voz e extração de entonação emocional.

Em síntese, o projeto do bot de análise de sentimentos ilustra como tecnologias de inteligência artificial podem ser aplicadas de forma prática para agregar valor em processos de comunicação e análise de dados. O trabalho destaca o potencial dessas ferramentas no apoio a decisões estratégicas e no entendimento das emoções humanas em um cenário digital cada vez mais dinâmico.

REFERÊNCIAS

ABIOLA, O.; ADEBAYO, A.-A.; AROGUNDADE, O.; MISRA, S.; ABAYOMI-ALLI, O. Sentiment analysis of covid-19 tweets from selected hashtags in nigeria using vader and text blob analyser. **Journal of Electrical Systems and Information Technology**, v. 10, p. 5, 01 2023.

ALVAREZ, G.; CHOI, J.; STROVER, S. Good news, bad news: A sentiment analysis of the 2016 election russian facebook ads. **International Journal of Communication**, v. 14, p. 3027–3053, 01 2020.

ANDRADE, L. M. S. Processamento de linguagem natural (pln): ferramentas e desafios. **Anais I CONIDIS... Campina Grande: Realize Editora**, 2016.

ARIFIANTO, M.; IZZUDIN, I. Students' acceptance of discord as an alternative online learning media. **International Journal of Emerging Technologies in Learning (iJET)**, International Journal of Emerging Technology in Learning, Kassel, Germany, v. 16, n. 20, p. 179–195, October 2021. ISSN 1863-0383. Disponível em: <https://www.learntechlib.org/p/220539>.

BONTA, V.; KUMARESH, N.; JANARDHAN, N. A comprehensive study on lexicon based approaches for sentiment analysis. **Asian Journal of Computer Science and Technology**, v. 8, n. S2, p. 1–6, Jan. 2019. Disponível em: <https://ajcst.co/index.php/ajcst/article/view/2037>.

BUKOVINA, J. Social media big data and capital markets—An overview. **Journal of Behavioral and Experimental Finance**, v. 11, n. C, p. 18–26, 2016. Disponível em: <https://ideas.repec.org/a/eee/beexfi/v11y2016icp18-26.html>.

CARVALHO, R. C. d. **Chatbot aplicado à recuperação de informação: um modelo orientado a metadados**. Tese (Doutorado) — Universidade Estadual Paulista, 2018. Disponível em: <https://repositorio.unesp.br/items/4aa1e4a4-aaa1-4651-aeee-77c174cb448a>.

COSTA, A. F. d. **Arandu, um Chatbot para construção de ontologias guiado por uma ontologia de topo**. Tese (Doutorado) — Universidade Federal de Pernambuco, 2020. Disponível em: <https://repositorio.ufpe.br/handle/123456789/37879>.

EGLISTON, B. Big playerbase, big data: On data analytics methodologies and their applicability to studying multiplayer games and culture. **First Monday**, v. 21, n. 7, Jun. 2016. Disponível em: <https://firstmonday.org/ojs/index.php/fm/article/view/6718>.

ESFAHANI, H.; TAVASOLI, K.; JABBARZADEH, A. Big data and social media: A scientometrics analysis. **International Journal of Data and Network Science**, p. 145–164, 01 2019.

FORNAZARI, J. E. K. S. **RALeM: uma ferramenta para auxiliar na identificação de palavras significativas em conversas em ferramentas on-line colaborativas através de análise léxico-morfológica**. Tese (Doutorado) — Universidade Tecnológica Federal do Paraná, 2019. Disponível em: <https://repositorio.utfpr.edu.br/jspui/handle/1/26467>.

HORNG, J.-S.; LIU, C.-H.; LEE, P.-S.; YU, T.-Y.; LING, N. Y. Modelling competitive advantage using the concepts of big data and social media to develop a sustainability strategy. **Tourism Review**, v. 78, 08 2022.

JEONG, J.; KIM, N. Does sentiment help requirement engineering: exploring sentiments in user comments to discover informative comments. **Automated Software Engineering**, v. 28, 11 2021.

KANUKA, H.; ANDERSON, T. Online social interchange, discord, and knowledge construction. **Journal of Distance Education**, v. 13, 01 2007.

Magzoub, Z. **Toxic comment classification in discord**. 2023. Disponível em: <http://essay.utwente.nl/94373/>.

POURYAZDAN, M.; FIANDRINO, C.; KANTARCI, B.; SOYATA, T.; KLIAZOVICH, D.; BOUVRY, P. Intelligent gaming for mobile crowd-sensing participants to acquire trustworthy big data in the internet of things. **IEEE access**, IEEE, Piscataway, v. 5, p. 22209–22223, 2017. ISSN 2169-3536.

R, D. H. Sentiment analysis: Textblob for decision making. 06 2023.

REYNOLDS, J. Gambling on big data: Designing risk in social casino games. **European Journal of Risk Regulation**, v. 10, n. 1, p. 116–131, 2019.

RODRIGUES, C. A. S.; VIEIRA, L. L.; MALAGOLI, L.; TIMMERMANN, N. Mineração de opinião e análise de sentimento. **Trabalho acadêmico, Universidade Federal de Santa Catarina, Florianópolis. www. inf. ufsc. br/~ alvares/INE5644/MineracaoOpiniao. pdf**, 2010.

ROTHMAN, D. **Transformers for Natural Language Processing: Build, Train, and Fine-tune Deep Neural Network Architectures for NLP with Python, PyTorch, TensorFlow, BERT, and GPT-3**. 2nd. ed. Birmingham, UK: Packt Publishing, 2021. ISBN 9781801077651.

SANTOS, T. C. A. d.; RODRIGUES, K. L. A. Impactos das redes sociais em relaÇÃo À autoestima e autoimagem. **Revista Ibero-Americana de Humanidades, Ciências e Educação**, v. 9, n. 3, p. 851–862, mar. 2023. Disponível em: <https://periodicorease.pro.br/rease/article/view/8724>.

SEAH, C. W.; CHIEU, H. L.; CHAI, K. M. A.; TEOW, L.-N.; YEONG, L. W. Troll detection by domain-adapting sentiment analysis. In: **2015 18th International Conference on Information Fusion (Fusion)**. [S.l.: s.n.], 2015. p. 792–799.

SHIVANI; KULKARNI, P.; ALYASIRI, S. Big data analytics of social media behavior for enhancing customer engagement. In: . [S.l.: s.n.], 2023. p. 060031.

SIMõES, R.; CAMPONEZ, C. Participação online e conteúdo ofensivo: limites ético-legais da liberdade de expressão nas redes sociais. In: _____. [S.l.: s.n.], 2020. p. 21–49. ISBN 978-989-26-1891-3.

WILLSON, M.; LEAVER, T. Zynga's farmville, social games, and the ethics of big data mining. **Communication Research and Practice**, Routledge, v. 1, n. 2, p. 147–158, 2015. Disponível em: <https://doi.org/10.1080/22041451.2015.1048039>.

XUE, Z.; LI, Q.; ZENG, X. Social media user behavior analysis applied to the fashion and apparel industry in the big data era. **Journal of Retailing and Consumer Services**, v. 72, n. C, 2023. Disponível em: <https://ideas.repec.org/a/eee/joreco/v72y2023ics0969698923000462.html>.

APÊNDICE A – TERMO DE CONSENTIMENO DE PARTICIPAÇÃO

Discord Bot: Análise de Sentimento
TERMO DE CONSENTIMENTO DE PARTICIPAÇÃO

Responsáveis: Vinicius Trindade Santos - vinicius.trindade@usp.br

Convite para Participação Você está sendo convidado(a) a participar voluntariamente do estudo: "Discord Bot: Análise de Sentimento". Por favor, leia atentamente as informações abaixo antes de consentir com a sua participação.

—

OBJETIVO E BENEFÍCIOS DO ESTUDO
Este estudo visa testar e avaliar um bot de análise de sentimento integrado ao Discord, que utiliza modelos de análise de sentimento open source, como **TextBlob, VADER e Transformers**. O objetivo principal é comparar os diferentes modelos de análise no processamento de mensagens de texto e compreender como eles classificam as mensagens em sentimentos positivos, neutros ou negativos.

Para essa prova de conceito (PoC), será realizado um experimento com voluntários que interagirão com o bot, enviando mensagens de texto que serão analisadas automaticamente. Os resultados permitirão verificar o desempenho do bot em identificar corretamente os sentimentos e a concordância entre os modelos de análise.

—

PROCEDIMENTOS Etapa 1: Envio de Mensagens pelo Discord
Voluntários deverão interagir com o bot no Discord, enviando mensagens curtas que refletem diferentes estados emocionais.

Exemplos de mensagens:

- Positivas: ”Adorei a experiência!”, ”Estou muito feliz hoje!”
- Negativas: ”Não gostei disso”, ”Estou decepcionado”.
- Neutras: ”O céu está azul”, ”Hoje é segunda-feira”.

Etapa 2: Recebimento da Análise

O bot processará as mensagens utilizando os modelos TextBlob, VADER e Transformers e retornará os resultados em tempo real, indicando a classificação do sentimento e um score associado.

Etapa 3: Coleta de Dados

Os dados analisados serão armazenados de forma anonimizada para estudo posterior. Não serão coletadas informações pessoais ou identificáveis, apenas os textos enviados e os resultados das análises.

—

DESPESAS / RESSARCIMENTO

Todos os voluntários estão isentos de qualquer custo para participar deste estudo.

—

PARTICIPAÇÃO VOLUNTÁRIA

Sua participação neste estudo é totalmente voluntária, e você tem liberdade para interrompê-la a qualquer momento, sem qualquer prejuízo.

—

GARANTIA DE SIGILO E PRIVACIDADE

As mensagens enviadas ao bot serão armazenadas de forma segura e utilizadas exclusivamente para fins de pesquisa. Os dados coletados serão anonimizados, garantindo que nenhuma informação pessoal ou identificável seja associada aos participantes.

—

DÚVIDAS?

Contate:

- Vinicius Trindade Santos - vinicius.trindade@usp.br
- Telefone: (11) 9 7364-6651

Agradecemos pela sua colaboração!

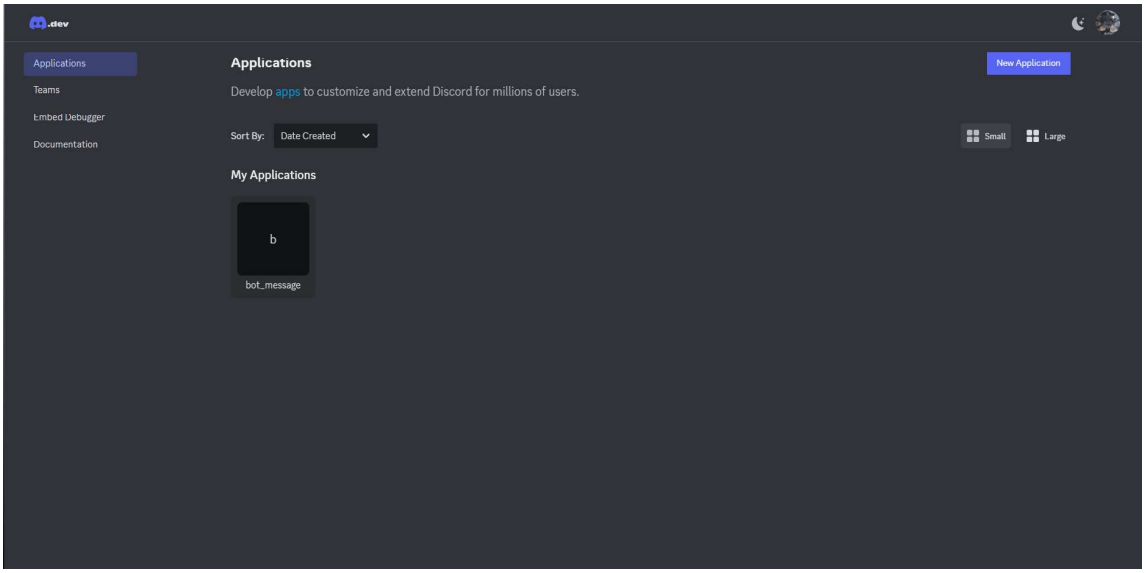
APÊNDICE B – SCRIPTS UTILIZADOS

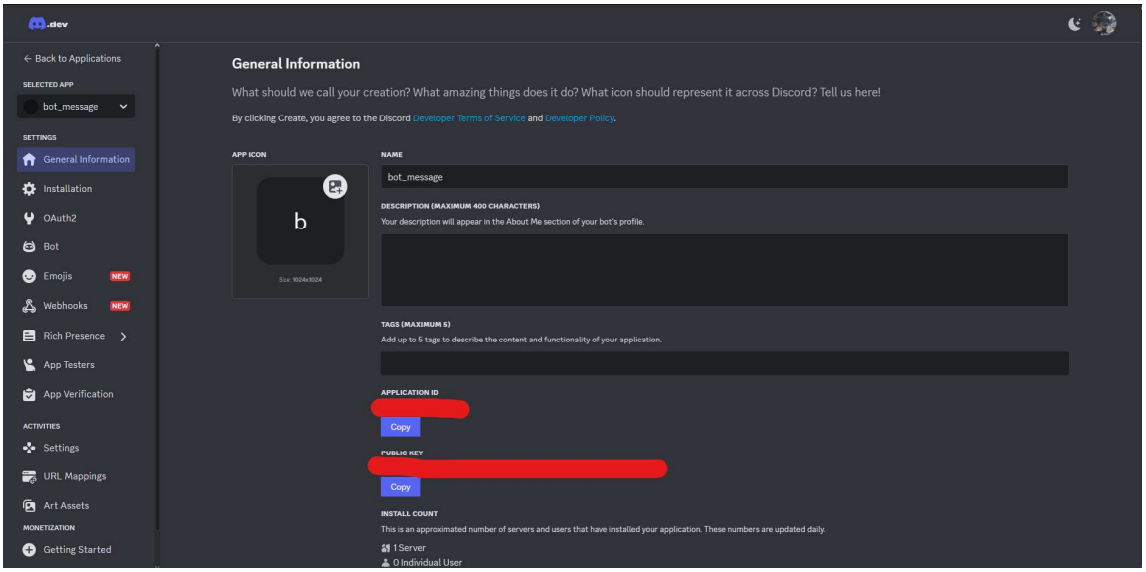
Script para criação do banco de dados executaco localmente com PostgreSQL:

discord-bot > schema.sql

```sql
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,          -- Identificador único do usuário
    user_name VARCHAR(255) UNIQUE NOT NULL -- Nome do usuário (único)
);

CREATE TABLE messages (
    message_id SERIAL PRIMARY KEY,       -- Identificador único da mensagem
    user_id INTEGER NOT NULL,            -- Referência ao usuário que enviou a mensagem
    message_content TEXT NOT NULL,       -- Conteúdo da mensagem
    FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE -- Relacionamento com a tabela users
);

CREATE TABLE sentiment_analysis (
    analysis_id SERIAL PRIMARY KEY,      -- Identificador único da análise
    message_id INTEGER NOT NULL,         -- Referência à mensagem analisada
    analyzer_name VARCHAR(50) NOT NULL,   -- Nome do analisador (TextBlob, VADER, Transformers)
    sentiment VARCHAR(20) NOT NULL,      -- Sentimento identificado (positivo, negativo, neutro)
    score FLOAT NOT NULL,                -- Escore da análise de sentimento
    FOREIGN KEY (message_id) REFERENCES messages(message_id) ON DELETE CASCADE -- Relacionamento com a tabela messages
);

```

Tela do Discord Developer Portal onde é obtido o token e são configuradas as autorizações do bot:





Script para a conexão do framework API FLASK, análise de cada modelo e armazenamento no banco de dados:

discord-bot > app_c.py > ...

```python
from flask import Flask, request, jsonify
from flask_sqlalchemy import SQLAlchemy
from textblob import TextBlob
from googletrans import Translator
from transformers import pipeline
from nltk.sentiment.vader import SentimentIntensityAnalyzer
import nltk

nltk.download('vader_lexicon')

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'postgresql+psycopg2://User:Password@localhost:5432/Your_DB'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

# Inicializa o banco de dados
db = SQLAlchemy(app)

# Definição das tabelas do banco de dados
class User(db.Model):
    __tablename__ = 'users'
    user_id = db.Column(db.Integer, primary_key=True)
    user_name = db.Column(db.String(255), unique=True, nullable=False)

class Message(db.Model):
    __tablename__ = 'messages'
    message_id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('users.user_id'), nullable=False)
    message_content = db.Column(db.Text, nullable=False)

class SentimentAnalysis(db.Model):
    __tablename__ = 'sentiment_analysis'
    analysis_id = db.Column(db.Integer, primary_key=True)
    message_id = db.Column(db.Integer, db.ForeignKey('messages.message_id'), nullable=False)
    analyzer_name = db.Column(db.String(50), nullable=False)
    sentiment = db.Column(db.String(20), nullable=False)
    score = db.Column(db.Float, nullable=False)
```

discord-bot > app_c.py > ...

```python
# Função para traduzir para o inglês
def translate_to_english(text):
    translator = Translator()
    translated_text = translator.translate(text, dest='en').text
    return translated_text

# Análise com TextBlob
def analyze_sentiment_textblob(text):
    translated_text = translate_to_english(text)
    blob = TextBlob(translated_text)
    polarity = blob.sentiment.polarity
    if polarity > 0.5:
        sentiment = "positivo"
    elif polarity < 0:
        sentiment = "negativo"
    else:
        sentiment = "neutro"
    return {"analyzer_name": "TextBlob", "sentiment": sentiment, "score": polarity}

# Análise com VADER
def analyze_sentiment_vader(text):
    translated_text = translate_to_english(text)
    sid = SentimentIntensityAnalyzer()
    scores = sid.polarity_scores(translated_text)
    compound_score = scores['compound']
    if compound_score >= 0.05:
        sentiment = "positivo"
    elif compound_score <= -0.05:
        sentiment = "negativo"
    else:
        sentiment = "neutro"
    return {"analyzer_name": "VADER", "sentiment": sentiment, "score": compound_score}
```

```python
# Análise com Transformers (Hugging Face)
def analyze_sentiment_transformers(text):
    translated_text = translate_to_english(text)
    classifier = pipeline('sentiment-analysis')
    result = classifier(translated_text)[0]
    sentiment = result['label'].lower()
    score = result['score']

    if sentiment == "positive":
        sentiment = "positivo"
    elif sentiment == "negative":
        sentiment = "negativo"
    else:
        sentiment = "neutro"

    return {"analyzer_name": "Transformers", "sentiment": sentiment, "score": score}
```

discord-bot > app_c.py > ...

```python
# Rota para analisar a mensagem
@app.route('/api/messages', methods=['POST'])
def analyze_message():
    data = request.json
    user_name = data.get("user", "")
    message_content = data.get("message", "")

    # Verifica se o usuário já existe no banco de dados
    user = User.query.filter_by(user_name=user_name).first()
    if not user:
        user = User(user_name=user_name)
        db.session.add(user)
        db.session.commit()

    # Armazena a mensagem no banco de dados
    message = Message(user_id=user.user_id, message_content=message_content)
    db.session.add(message)
    db.session.commit()

    # Realiza as 3 análises de sentimento
    analyses = [
        analyze_sentiment_textblob(message_content),
        analyze_sentiment_vader(message_content),
        analyze_sentiment_transformers(message_content)
    ]

    # Armazena cada análise no banco de dados
    for analysis in analyses:
        sentiment_analysis = SentimentAnalysis(
            message_id=message.message_id,
            analyzer_name=analysis["analyzer_name"],
            sentiment=analysis["sentiment"],
            score=analysis["score"]
        )
        db.session.add(sentiment_analysis)
```

```
124         db.session.commit()
125
126         # Resposta contendo todas as análises
127         response = {
128             analysis["analyzer_name"]: {
129                 "sentiment": analysis["sentiment"],
130                 "score": analysis["score"]
131             } for analysis in analyses
132         }
133
134         return jsonify(response)
135
136     if __name__ == '__main__':
137         app.run(debug=True)
138
```

Script para conexão do bot ao discord e API FLASK:

```
discord-bot > bot_c.py > on_message
1   import discord
2   import requests
3
4   intents = discord.Intents.default()
5   intents.message_content = True
6
7   client = discord.Client(intents=intents)
8
9   API_URL = "http://127.0.0.1:5000/api/messages"
10
11  @client.event
12  async def on_ready():
13      print(f'Bot conectado como {client.user}')
14
15  @client.event
16  async def on_message(message):
17      if message.author == client.user:
18          return
19
20      data = {
21          "user": str(message.author),
22          "message": message.content
23      }
24
25      # Faz a requisição para a API Flask que faz a análise de sentimento
26      response = requests.post(API_URL, json=data)
27
```

```
    if response.status_code == 200:
        sentiment_data = response.json()

        # Monta a mensagem com os resultados da análise de sentimento
        sentiment_results = []
        for analyzer, result in sentiment_data.items():
            sentiment = result.get("sentiment", "desconhecido")
            score = result.get("score", 0)
            sentiment_results.append(f"**{analyzer}**: {sentiment} (Score: {score})")

        # Envia a mensagem com o resultado da análise de sentimento em uma mensagem privada para o autor
        await message.author.send(
            f"{message.author}, sua mensagem \"{message.content}\" recebeu as seguintes análises de sentimento:\n\n"
            + "\n".join(sentiment_results)
        )
    else:
        await message.author.send("Erro ao analisar sentimento da mensagem.")

client.run('YOUR TOKEN')

```