Compreendendo a Dinâmica Emocional em Projetos de Código Aberto: Uma Análise das Issues

Fabricio S. Miranda[1]

[1]Instituto de Ciencias Exatas e Naturais – Universidade Federal do Pará (UFPA)
Caixa postal 479. PABX +55 91 3201-7000. Belém - Pará - Brasil

Abstract. *The main means of communication in open source projects occurs through "issues", which, although not intended to express emotions and feelings, can end up including them in interactions among collaborators. In this study, we conducted an analysis to assess toxicity in the comments of "issues" in open source projects. For this, we created a database containing authors identified as toxic and, based on this information, mapped all the "issues" in which these authors were involved. Subsequently, we submitted these "issues" to the IBM Watson Natural Language API for a more detailed analysis. After the classification by the IBM Watson Natural Language API, we selected a subset for further manual analysis.*

Resumo. *O principal meio de comunicação em projetos de código aberto ocorre através das "issues", que, embora não sejam destinadas a expressar emoções e sentimentos, podem acabar por incluí-las nas interações entre colaboradores. Neste estudo, conduzimos uma análise para avaliar a toxicidade nos comentários das "issues" em projetos open source. Para isso, criamos um banco de dados contendo autores identificados como tóxicos e, com base nessa informação, mapeamos todas as "issues" nas quais estes autores estiveram envolvidos. Posteriormente, submetemos essas "issues" à API do IBM Watson Natural Language para uma análise mais detalhada. Após a classificação realizada pela API do IBM Watson Natural Language, selecionamos um subconjunto para realizar uma análise manual adicional.*

1. Introdução

O principal meio de comunicação entre os colaboradores são as 'issues' no GitHub. Estas são utilizadas para discutir aspectos do projeto, resolver problemas e compartilhar ideias. Entretanto, há ocasiões em que essas discussões podem adquirir um tom tóxico, o que tem o potencial de desencorajar a participação da comunidade e tornar as conversas improdutivas. [Miller et al. 2022] [Joni Salminen 2020].

Este artigo investiga a relação entre a toxicidade nas 'issues' do GitHub e o comportamento dos membros envolvidos. Para tal, propõe-se mapear os membros de 100 'issues' identificadas como tóxicas [Miller et al. 2022]. Utilizando a API do GitHub, será realizada uma investigação para determinar se esses membros exibiram comportamentos semelhantes em outros projetos. Adicionalmente, empregaremos a API do IBM Watson Natural Language Understanding para a classificação automatizada desses comportamentos.

1.1. Contexto

Em comunidades de software de código aberto, diferentemente de outras, as 'issues' são utilizadas como principal meio de comunicação. Elas variam desde a correção de bugs até discussões gerais sobre o projeto. Como um indicador crucial de engajamento na comunidade, é essencial observar que discussões tóxicas têm o potencial de reduzir o envolvimento colaborativo [Joni Salminen 2020].

Para o processamento de linguagem natural, empresas como AWS, Google Cloud e IBM Watson disponibilizam diversas ferramentas. Essas ferramentas empregam técnicas de Processamento de Linguagem Natural (NLP) e Inteligência Artificial para analisar a linguagem humana, com capacidades específicas para identificar áreas de toxicidade.

A detecção e a prevenção proativas são cruciais para assegurar um ambiente seguro e saudável para todos os envolvidos. Quando utilizadas de maneira eficaz, as ferramentas disponíveis podem contribuir significativamente para alcançar esse objetivo.

1.2. Justificativa

As 'issues' no GitHub, sendo o principal meio de comunicação em muitos projetos de código aberto, tornam-se um ponto focal para diversas interações. Enquanto, por um lado, elas atuam como um indicador vital de engajamento e discussão produtiva, por outro lado, podem refletir sentimentos negativos, emoções intensas e, em alguns casos, comportamentos prejudiciais. Esta dualidade sublinha a urgência em desenvolver ferramentas e metodologias eficazes para monitorar, analisar e, idealmente, mitigar a toxicidade nas comunidades online.

Além disso, com o advento e a popularização de ferramentas automatizadas de detecção de sentimentos, tornou-se viável explorar a natureza e a extensão da toxicidade em comunidades online em grande escala. Empresas líderes no setor de tecnologia, como AWS, Google Cloud e IBM Watson, têm desenvolvido soluções que empregam técnicas avançadas de Processamento de Linguagem Natural e Inteligência Artificial para analisar a linguagem humana. Contudo, a eficácia e a precisão dessas ferramentas, especialmente no contexto específico das comunidades de código aberto, ainda requerem avaliações rigorosas.

1.3. Objetivo Geral

O objetivo deste trabalho é realizar uma busca por usuários envolvidos em 'issues' tóxicas no GitHub e desenvolver um método de detecção automatizada de comentários tóxicos. Isso será feito por meio das métricas obtidas do IBM Natural Language Understanding, que serão utilizadas para analisar e identificar padrões de toxicidade nos comentários.

1.4. Objetivos específicos

Com o objetivo de obter uma classificação automatizada de comentários tóxicos, precisamos estabelecer alguns objetivos específicos.

- **Análise Automatizada:** Realizar a análise automatizada dos comentários das 'issues' já classificadas como tóxicas no trabalho [Miller et al. 2022].
- **Criação de um Dataset:** Desenvolver um conjunto de dados abrangente para permitir a análise em diversos projetos.

- **Classificação em Massa:** Implementar a utilização de uma ferramenta de classificação em massa.
- **Validação da Classificação:** Utilizar um subconjunto de dados para validar a eficácia da classificação automatizada.

2. Trabalhos Relacionados

Para a realização deste trabalho, conduziu-se uma pesquisa acadêmica abrangente em busca de estudos relacionados ao tema. Esta pesquisa envolveu uma análise detalhada dos trabalhos existentes, destacando suas diferenças e contribuições únicas em relação ao presente estudo.

O estudo de [Miller et al. 2022] explora a dinâmica da toxicidade em discussões de projetos de código aberto no GitHub. Analisando 100 discussões marcadas por conteúdo tóxico, o estudo revela características distintas de toxicidade nesta plataforma, como arrogância, senso de direito e insultos técnicos. Métodos variados foram utilizados na coleta e análise qualitativa dessas discussões, focando na categorização de interações e identificação de padrões comportamentais.

O trabalho de [Sanei et al. 2021] analisa três bibliotecas de Machine Learning de código aberto - TensorFlow, Scikit Learn e PyTorch - utilizando dados coletados do GitHub. Esta análise inclui issues fechadas com comentários, commits e metadados, enfatizando o pré-processamento de dados de linguagem natural para aprimorar a detecção de sentimentos e tons nas discussões.

O artigo de [Huilian Sophie Qiu 2022] investiga métodos para detectar conflitos interpessoais, incluindo detectores de toxicidade para discussões de código aberto e detectores de pushback para revisões de código em empresas. Este estudo testa a eficácia desses detectores em diferentes contextos e discute como sua combinação pode aprimorar a identificação de conflitos interpessoais.

Herrmann e Klünder (2021) apresentam uma abordagem inovadora para aplicar análise de sentimentos em comunicações verbais durante reuniões de projetos de software. Combinando reconhecimento de fala e análise de sentimentos, o estudo visa processar áudio de reuniões ao vivo, classificando declarações em classes de polaridade de sentimentos. Os resultados preliminares mostram um acordo moderado entre as classificações da ferramenta e avaliações humanas.

No artigo de [Raman et al. 2020], os autores desenvolvem um classificador SVM específico para engenharia de software, treinado e validado com dados manualmente rotulados de discussões e comentários. A metodologia adapta o classificador ao contexto da engenharia de software, neutralizando termos técnicos com conotações negativas. A precisão e o recall foram avaliados através de validação cruzada e testes em um conjunto de dados separado e em issues aleatórias do GitHub, permitindo análises de tendências de toxicidade.

No artigo de [Kavaler et al. 2017], os autores estudam a complexidade linguística em discussões do GitHub e seu impacto na resolução de problemas. Utilizando modelos de linguagem específicos do projeto e um modelo global, coletaram um conjunto de dados de discussões, incluindo texto, metadados e redes sociais. A análise revelou padrões de conformidade linguística e seu impacto nos tempos de resolução de problemas.

No artigo de [Klünder et al. 2020], a metodologia consiste na análise de comunicação baseada em texto de equipes de desenvolvimento de software para identificar o humor do grupo. Utilizando técnicas de processamento de linguagem natural e aprendizado de máquina, o estudo classifica mensagens de texto e analisa 1947 mensagens de um chat de grupo no Zulip ao longo de 5,5 meses.

A pesquisa [Klünder et al. 2020] também realizou análises em 3778 frases, classificadas manualmente, para treinar um classificador de humor. A correlação entre várias métricas foi examinada, revelando padrões de comunicação e a necessidade de conjuntos de dados mais equilibrados para pesquisas futuras.

No artigo de [Novielli et al. 2018], os autores avaliam o desempenho e a confiabilidade de três ferramentas de análise de sentimentos específicas para o domínio da engenharia de software. Comparando ferramentas com anotação humana, o estudo investiga a melhoria na precisão na detecção de emoções e realiza uma análise qualitativa dos textos mal classificados.

O estudo atual se diferencia dos trabalhos anteriores na análise da toxicidade em projetos de código aberto, expandindo a pesquisa para uma amostra mais ampla e utilizando a API IBM Watson Natural Language para processar e filtrar issues, abrangendo 1583 projetos e 11.530 issues. Diferente dos estudos citados, o foco é na identificação de comportamentos negativos em uma escala mais ampla, utilizando uma metodologia que inclui filtragem de comentários e análise manual para categorizar a toxicidade.

3. Background

No âmbito da análise automatizada de emoções em textos, inicialmente escolhemos o IBM Tone Analyzer, reconhecido por sua eficiência na análise de sentimentos. Esta ferramenta se destacava pela sua capacidade de categorizar textos segundo emoções específicas, como ”entusiasmado” e ”frustrado”. Contudo, com a descontinuação do Tone Analyzer, vimo-nos na necessidade de buscar alternativas. Nesse contexto, optamos pela IBM Natural Language Understanding (NLU).

A IBM Natural Language Understanding não só preserva as funcionalidades essenciais de análise de sentimentos do Tone Analyzer, mas também amplia significativamente as possibilidades de interpretação textual. Com uma abordagem mais sofisticada e abrangente, a NLU oferece uma análise detalhada de emoções e sentimentos, permitindo uma compreensão mais profunda das nuances textuais. Além disso, esta ferramenta é capaz de identificar e analisar entidades e conceitos presentes no texto, proporcionando insights contextuais e conteúdos relacionados.

Esta transição para a IBM Natural Language Understanding representa não só uma adaptação necessária, mas também um avanço significativo em nossa capacidade de análise de textos. Com a NLU, agora podemos interpretar emoções e sentimentos de maneira mais precisa e abrangente, abrindo novas possibilidades para aplicação da análise de sentimentos em diferentes contextos.

A IBM Natural Language Understanding é uma ferramenta de processamento de linguagem natural que identifica e classifica emoções em textos. Ela distingue cinco emoções básicas: raiva, desgosto, medo, alegria e tristeza, atribuindo um score de 0 a 1 para cada uma, onde 0 indica ausência e 1, presença máxima da emoção. Scores mais

altos indicam uma maior presença da emoção analisada no conteúdo.

A NLU pode categorizar as emoções tanto no nível do documento como em frases específicas. A categorização no nível do documento fornece uma visão geral da emoção predominante, enquanto a categorização em frases permite análises mais detalhadas.

A NLU é uma ferramenta versátil com aplicações variadas, incluindo:

- Análise de mídias sociais para identificar tendências emocionais
- Avaliação de produtos e serviços para entender a satisfação dos clientes
- Análise de feedback de funcionários para identificar questões de moral

Integrar a IBM Natural Language Understanding com as issues do GitHub é uma inovação significativa na gestão e análise de projetos de software. Esta integração permite que equipes de desenvolvimento utilizem análise avançada de linguagem natural para melhor compreender as discussões e feedbacks nas issues do GitHub. Tal fusão possibilita a extração automática de insights valiosos, identificando tendências, sentimentos e preocupações chave de colaboradores e usuários. Esta análise em tempo real facilita a identificação de problemas comuns, priorização de bugs e solicitações de funcionalidades, além de oferecer uma visão clara do moral da equipe e da comunidade. A automação desses processos economiza tempo e fornece uma base de dados rica para decisões mais informadas e estratégicas.

4. Metodologia

Com o objetivo de fornecer uma compreensão abrangente e aprofundada do fenômeno da toxicidade em projetos de código aberto, este capítulo está estruturado em uma sequência de etapas . Cada etapa, detalhadamente descrita na lista a seguir, constitui um aspecto fundamental de nossa metodologia de pesquisa e análise. Iniciaremos com a seleção de 'issues' tóxicas, prosseguindo para a busca destas no GitHub. Em seguida, exploraremos os resultados obtidos e os desafios enfrentados durante o processo. Avançaremos para o desenvolvimento e aplicação de uma ferramenta de detecção automatizada, seguida pela análise e filtragem de comentários. Finalmente, a análise manual dos comentários filtrados nos proporcionará insights valiosos sobre a natureza da toxicidade nas interações online. Esta abordagem metodológica nos permitirá não apenas identificar padrões de comportamento tóxico, mas também avaliar a eficácia das ferramentas de detecção e estratégias de mitigação, resultando em um entendimento mais profundo das dinâmicas de comunicação em projetos de código aberto.

No contexto do nosso estudo, baseamos nossa pesquisa na fundação estabelecida pelo artigo anterior de [Miller et al. 2022]. Neste trabalho, foram analisadas 100 'issues' específicas de projetos de código aberto, as quais foram identificadas por sua natureza tóxica.

Para analisar issues tóxicas no GitHub, a pesquisa [Miller et al. 2022] adotou uma abordagem multifacetada para criar uma amostra representativa, levando em conta a raridade da toxicidade em issues. Quatro estratégias distintas foram implementadas: o uso de um detector de toxicidade baseado em linguagem para analisar milhões de questões e comentários; a busca por menções ao 'código de conduta' nos comentários; a identificação de issues bloqueadas por serem excessivamente acaloradas; e a recuperação de issues

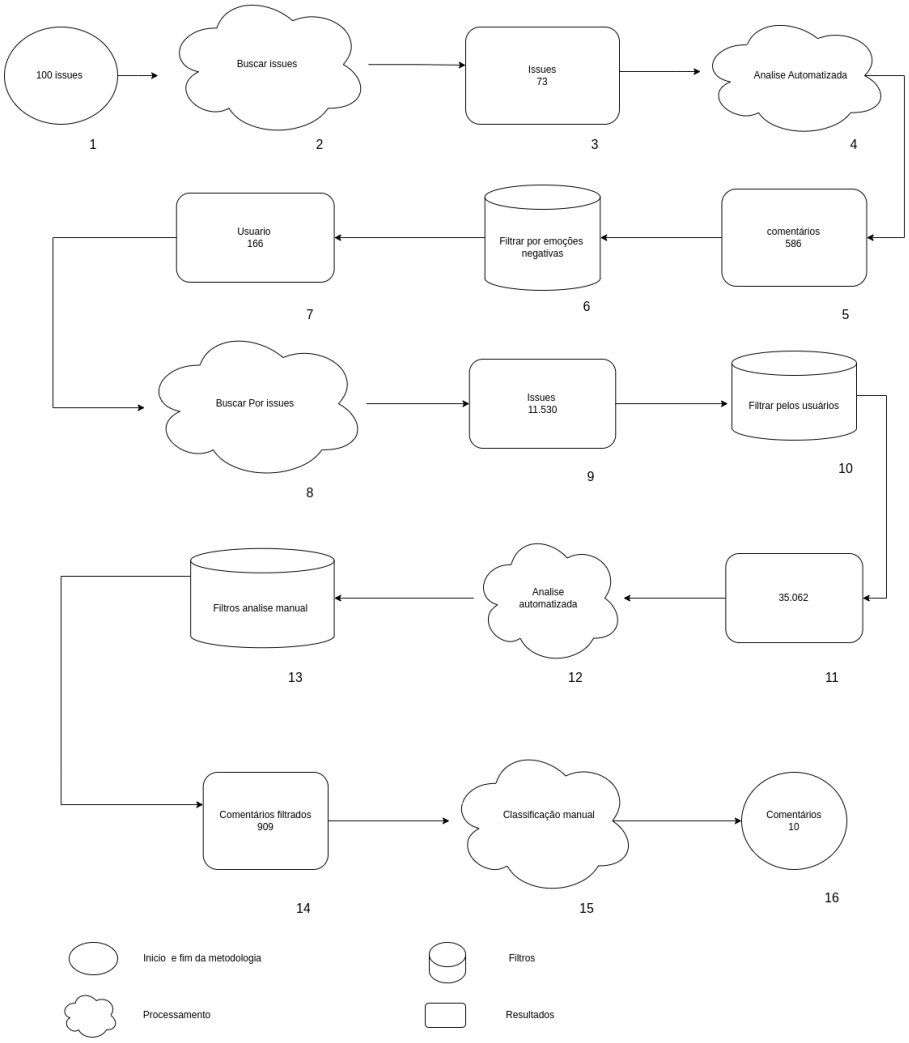

Figure 1. fluxo metodologia

excluídas. Embora cada estratégia possuísse suas próprias limitações e tendências, combinadas, elas ofereceram uma visão mais abrangente e diversificada do problema. Após a identificação inicial, foi realizada uma rotulagem manual para confirmar a toxicidade das questões, com múltiplos autores avaliando independentemente cada caso. Esse processo garantiu a confiabilidade dos resultados, alcançando um alto grau de concordância entre os avaliadores.

A partir dos resultados dessas estratégias, uma amostra final de 100 issues tóxicas foi composta [Miller et al. 2022], selecionando aleatoriamente 20 issues de cada grupo identificado, evitando duplicatas e limitando a amostra a no máximo duas issues tóxicas por repositório para preservar a diversidade dos projetos.

Para a seleção dos dados iniciais, utilizamos um conjunto previamente validado de 100 issues tóxicas conforme descrito em [Miller et al. 2022]. No entanto, durante a busca no GitHub, conseguimos acessar apenas 73 dessas issues. Importante ressaltar que nossa busca se restringiu à plataforma do GitHub, aproveitando sua API robusta e bem documentada. As dificuldades encontradas ao tentar acessar as 27 issues restantes podem

ser atribuídas a várias mudanças na plataforma. Por exemplo, issues de repositórios arquivados, que se tornam apenas para leitura, podem limitar ou impedir o acesso. Esse estado é frequentemente usado por proprietários de repositórios para preservar conteúdo existente, inibindo novas interações ou modificações. Além disso, issues excluídas por administradores ou proprietários do repositório tornam-se inacessíveis, mesmo com o URL direto. Alterações no nome do repositório ou transferências para outros usuários ou organizações também afetam o acesso, uma vez que o URL da issue muda, invalidando links anteriores. Embora seja possível obter esses conteúdos através de snapshots de páginas web, essa abordagem não oferece a mesma praticidade da API do GitHub. Assim, visando uma metodologia de busca automatizada, optamos por restringir nossa pesquisa às issues acessíveis exclusivamente via API do GitHub.

Após o processo de busca, conseguimos acessar 73 das 100 issues previamente analisadas, as quais formaram a base do nosso trabalho. Com base nessas issues, delineamos a metodologia para identificar novas issues relevantes.

Cada issue foi desmembrada em comentários individuais, os quais foram submetidos à API do IBM Watson Natural Language para análise. O resultado foi a classificação de 586 comentários.

Com o objetivo de focar nossos esforços na identificação de usuários com comportamentos negativos, aplicamos um filtro específico. Este consistiu em excluir comentários onde a emoção 'joy' recebeu a maior pontuação pela API do Watson Natural Language.

Dessa filtragem, identificamos 166 usuários que demonstraram predominantemente sentimentos negativos. Com esses usuários em mente, utilizamos a API do GitHub para buscar outras issues que esses indivíduos tenham criado ou nas quais tenham sido mencionados. Esta etapa de busca mais abrangente tem como objetivo identificar novas issues para uma análise mais aprofundada.

Durante um período de dois anos, de julho de 2020 a julho de 2022, realizamos uma investigação detalhada focada nos 166 usuários identificados no GitHub. O propósito era acompanhar suas atividades em issues de diversos projetos.

Ao longo dessa análise extensa, que cobriu um amplo intervalo temporal, identificamos a participação desses usuários em 1583 projetos distintos. A imersão nesse universo revelou 11.530 issues, abrangendo uma vasta gama de tópicos e desafios. A interação ativa nesses projetos foi evidenciada pelos 213.167 comentários registrados, refletindo um engajamento significativo por parte da comunidade.

Com a grande quantidade de dados disponíveis, focamos especificamente nos 166 usuários já identificados, visando analisar seus comportamentos em outros projetos. Dentro do total de 213.167 comentários, identificamos a participação desses usuários em 1.488 projetos, contribuindo com 35.062 comentários.

Após filtrar os comentários desses usuários específicos, encaminhamos-os para a análise pela API do IBM Watson Natural Language. Os resultados obtidos permitiram detectar padrões de toxicidade, com base nas pontuações atribuídas pela API. Devido ao volume substancial de dados, foi necessário estabelecer um critério rigoroso de seleção para proceder à análise manual dos comentários. Os detalhes desse processo de seleção são explicados na próxima seção.

Para identificar comentários potencialmente tóxicos, empregamos uma estratégia de filtragem específica. Inicialmente, definimos variáveis-chave para a análise dos comentários, considerando o número de palavras em cada um e os escores de emoções negativas e positivas fornecidos pela API do Watson Natural Language. Nossa análise abrangeu comentários com uma extensão de 10 a 50 palavras, avaliando os escores de emoções numa escala de 0 a 1, com precisão de seis dígitos. O principal foco era estabelecer limites inferiores para emoções negativas, como anger, disgust, fear e sadness, e um limite superior para a emoção positiva, joy. Este critério visava excluir comentários com predominância de emoções positivas e reter aqueles com tendência negativa, indicativos de potencial toxicidade. Além disso, exploramos a combinação de emoções, analisando todas as permutações únicas possíveis das cinco emoções identificadas pelo Watson.

Prosseguindo com a estratégia de filtragem, um passo adicional envolveu a implementação de restrições baseadas no volume de comentários, estabelecendo um intervalo de aceitação de 50 a 5000 comentários. Esse critério foi definido para garantir um volume de dados razoável, suficiente para validar a análise, mas gerenciável para uma categorização manual dentro de um período estimado de 15 dias. Ao finalizar o processo de filtragem, notou-se que apenas os limites aplicados nas emoções joy e sadness geraram resultados significativos para o estudo, ressaltando a importância dessas emoções na identificação de comentários potencialmente tóxicos. A análise também indicou que alterações nos limites de comentários por restrição tinham um impacto notável no número de combinações possíveis, sublinhando a necessidade de uma definição cuidadosa desses parâmetros.

Na nossa estratégia de filtragem para discernir comentários potencialmente tóxicos, a definição de limites para a emoção 'joy' foi um passo crucial. Estabelecemos um escore entre 0 e 0.1, associado a uma contagem de palavras por comentário variando entre 30 e 40. Esta abordagem levanta questões sobre a precisão e eficácia dos parâmetros escolhidos na análise das interações. A seleção desses limites destacou o desafio de alcançar a precisão desejada, sugerindo que a identificação de conteúdos potencialmente prejudiciais pode ser influenciada por nuances e requer um refinamento contínuo na estratégia de filtragem.

Após a aplicação dos filtros, identificamos 909 comentários que já haviam sido classificados pela API do Watson. Estes comentários foram selecionados para a análise manual subsequente.

A análise manual representou um passo crucial para validar a precisão da API. Embora a API do Watson não tenha a capacidade de classificar diretamente um comentário como tóxico, utilizamos os scores que ela fornece, avaliando cinco emoções diferentes. Com base nos comentários filtrados mencionados na seção anterior, procedemos com uma análise manual simultânea. Neste processo, os comentários foram categorizados conforme os tipos de toxicidade definidos no trabalho de [Miller et al. 2022], que incluem 'insulting', 'entitled', 'arrogant', 'trolling' e 'unprofessional'.

Durante a análise dos 909 comentários, cada um foi classificado como tóxico ou não tóxico. Implementamos uma estratégia de revisão a cada lote de 100 comentários, onde fazíamos uma pausa para alinhamento entre os pesquisadores. Em casos de discordância sobre a classificação de um comentário, era realizada uma discussão detalhada

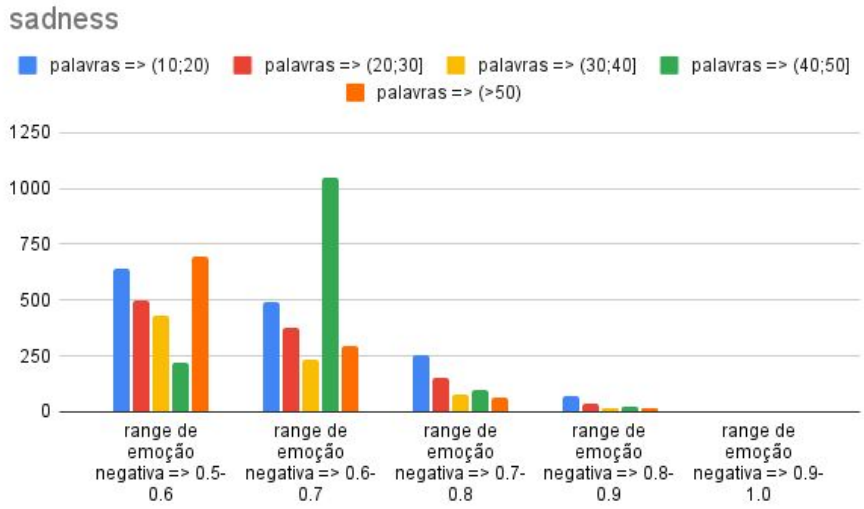

Figure 2. palavras x score joy

para alcançar um consenso. Além disso, identificamos algumas duplicatas no conjunto de dados; nestas situações, apenas o primeiro comentário identificado foi considerado para análise.

Como resultado deste processo, constatamos que apenas 10 dos 909 comentários foram unanimemente identificados como tóxicos por ambos os pesquisadores. Esta proporção — 10 comentários tóxicos em um total de 909 — sugere que a estratégia adotada de filtragem, baseada na menor pontuação da emoção 'joy', não foi eficaz para detectar comentários tóxicos. Estes representaram aproximadamente apenas 1% do total de comentários analisados, indicando a necessidade de revisão e possível ajuste da estratégia de filtragem utilizada.

Ao concluir esta seção sobre a metodologia de nosso estudo, destacamos a relevância da abordagem adotada para entender a toxicidade em comunidades de software de código aberto. Nossa cuidadosa seleção de issues nos permitiu superar os desafios dinâmicos associados ao acesso de dados em plataformas como o GitHub. A implementação da API do IBM Watson Natural Language representou um avanço significativo, facilitando não apenas a identificação de usuários com tendências negativas, mas também ampliando o escopo de nossa análise para abranger uma ampla gama de projetos e issues. Este processo detalhado, sustentado por uma estratégia de filtragem meticulosa e uma análise manual rigorosa, forneceu insights valiosos. Esses insights não apenas evidenciaram a presença de toxicidade, mas também revelaram as limitações e potenciais áreas de melhoria nas ferramentas automáticas de detecção de sentimentos.

Focando especificamente em três repositórios, que envolveram quatro issues distintas e três usuários diferentes, pudemos observar como certos tópicos ou discussões tendem a suscitar respostas tóxicas. Notadamente, a maioria desses comentários foi vinculada a um único usuário, destacando como certos indivíduos podem exercer uma influência considerável sobre o tom de uma discussão online. Os detalhes dessas observações estão resumidos na Tabela 1.

Projetos	Comentários	issues	usuários
pH7-Social-Dating-CMS	8	2	1
vscode	1	1	1
event-stream	1	1	1

Table 1. 10 comentários

5. Resultados

Dedicamos uma seção para detalhar os comentários tóxicos encontrados, explorando os tópicos específicos das discussões e as dinâmicas de interação dentro de cada repositório. Examinamos as questões de cache no repositório *pH7-Social-Dating-CMS*, os desafios de funcionalidade no*Visual Studio Code*, e as preocupações com segurança no repositório *event-stream*. Esta análise detalhada oferece uma compreensão mais profunda dos contextos e temas que podem catalisar a toxicidade em comunidades de software aberto.

Avaliamos cada comentário tóxico em detalhes, categorizando-os com base em características como arrogância, insultos, provocação e falta de profissionalismo [Miller et al. 2022]. Esta seção discute como cada tipo de toxicidade se manifesta nos comentários, o impacto desses comportamentos na comunidade e as implicações para a gestão de comunidades de software aberto.

Na análise dos comentários, identificamos a presença de toxicidade em três projetos distintos, distribuídos por quatro issues. O projeto ”pH7-Social-Dating-CMS” se destacou, apresentando o maior número de comentários tóxicos, totalizando oito. Dentre as issues analisadas, ”Cache is causing me big problems!” foi a que mais concentrou comentários tóxicos, com um total de sete.

As demais issues, por sua vez, registraram apenas um comentário tóxico cada. Esta distribuição dos comentários tóxicos sugere uma concentração de interações negativas em tópicos específicos, destacando a importância de uma análise mais detalhada para entender os contextos e as dinâmicas subjacentes que conduzem a esse padrão de comunicação.

A seguir, apresentamos uma tabela que ilustra de forma detalhada a classificação dos comentários analisados em nossa pesquisa. Esta tabela oferece uma visão clara sobre a natureza específica das interações tóxicas em projetos de software open source, categorizando cada comentário com base na sua classificação de toxicidade.

5.1. Contexto dos comentários

Nesta seção, apresentamos uma análise criteriosa de três distintas discussões em projetos de software open source, ilustrando como questões técnicas podem se transformar em interações tóxicas e conflitos pessoais. Estes casos foram selecionados por representarem um espectro diversificado de desafios e dinâmicas comunicacionais em ambientes de desenvolvimento colaborativo.

Cada caso revela um aspecto único da interação em projetos open source. Desde problemas técnicos que escalam para tensões pessoais até discussões sobre funcionalidades que são permeadas por atitudes condescendentes, os exemplos escolhidos destacam

como a toxicidade pode surgir e impactar negativamente o processo de colaboração. Além disso, evidenciam a importância da comunicação eficaz e respeitosa para a sustentabilidade e sucesso de projetos colaborativos.

Através desta análise, buscamos compreender melhor as causas e consequências da comunicação tóxica em projetos de código aberto, fornecendo insights valiosos para a melhoria das práticas de interação e colaboração dentro dessas comunidades. A Tabela 2 apresenta a lista de todos os comentários, juntamente com suas classificações de acordo com [Miller et al. 2022]. A classificação manual foi realizada com base nos critérios definidos pelo artigo.

Comentário	**Classificação**
"if some policy keeps vscode from being uploaded to the microsoft store then obv the policy is wrong..I have told microsoft a million times already to fire evryone that is working on that project..."	Arrogante
"¿ You're removal of the package broke our process, it says it can't find package on the server, it worked before with no problems.....dear god I pray whatever company you work for realizes how dumb you are"	Insulto
"@Undefined-Variables I haven't seen your site well, yes I did because I used a VPN with a address from China to hack you...What do you want us to say ?.Everything we say is never good for you."	Provocação (Trolling)
"@Undefined-Variables I wouldn't hire you...Doing setup on empty bare metal or virtual hardware ain't a deal. Doing diagnostic require knowledge...What type of test have you done to know that you have minor issues ?"	Arrogante
"@Undefined-Variables Not only the keyboard that's fucked up. The user's pretty much the same as the machine...You're what we call a code 3. The problem is 3 feet in front of the screen."	Insulto
"@Undefined-Variables You are going to be a man with this. You'll be an honorable men !.But as you seem to do not much more than crap, I doubt you'll get any fork...Maybe you shall start by the base."	Insulto e não profissional
"@Undefined-Variables You are really disconnected from earth..Look at what you wrote earlier... ***Girl haha that would explain a LOT!***..So this was your stupid comment, like many other...I don't know what do you need to understand..Go away."	Insulto e não profissional
"@Undefined-Variables You talk about being nasty ?.Well for sure when someone act the way you do, it sure attract lot of love and kisses...Why don't you go back like an hour ago and feel ashamed, cry in your pillow."	Provocação (Trolling)
"@rondlite Chuuut don't say that too loud ! The other is gonna think we are trying to make money off his Youtube / Polyfill / Polymer / Poly-me / Cache / Cash problem."	Não profissional
"@Undefined-Variables Maybe I wasn't clear enough.....Do you really think such a long running issue has any use for other to see ?.I think you shall go onto a social network or open you own slack channel ;-)"	Arrogante e provocação (Trolling)

Table 2. comentarios tóxicos

1. **Issue Cache is causing me big problems! :** No repositório pH7-Social-Dating-CMS, hospedado no GitHub, uma issue específica relacionada a problemas de cache afetando as URLs do site se destacou pela quantidade e natureza dos comentários. Nessa issue, foram identificados sete comentários classificados como tóxicos, evidenciando uma escalada de tensões que ultrapassou os limites aceitáveis de comunicação produtiva. O usuário Undefined-Variables reportou inicialmente que os links do site, incluindo cabeçalho e rodapé, estavam sendo alterados para diferentes URLs, e que o endereço IP do site também estava sendo afetado. Embora a limpeza do cache resolvesse temporariamente o problema, suspeitas surgiram quanto à possível ligação desse problema com a execução do composer, especialmente após a detecção de endereços IP internos e externos em diversos arquivos.
   As tentativas de resolver o problema envolveram modificações em temas e arquivos de template, além de discussões técnicas sobre problemas de JavaScript. No entanto, a discussão se desviou do foco técnico e mergulhou em frustrações e desafios, marcada por um tom crescentemente negativo. Apesar de haver reconhecimento por assistências recebidas, a predominância de comentários tóxicos nessa issue reflete a dificuldade de manter uma comunicação construtiva em meio a problemas técnicos complexos e a necessidade de estratégias efetivas para gerenciar e mitigar a toxicidade em ambientes colaborativos de software.
   A discussão entre os usuários Undefined-Variables, rondlite e polynamaude centra-se inicialmente em um problema técnico relacionado a URLs e endereços IP em um site, mas rapidamente se deteriora em um conflito pessoal, especialmente entre Undefined-Variables e polynamaude. Polynamaude adota um tom crítico e condescendente, questionando as habilidades técnicas de Undefined-Variables, que responde com frustração e defensividade. Enquanto rondlite tenta redirecionar o foco para a solução técnica, o desentendimento entre os outros dois usuários domina a conversa, destacando um exemplo de como a comunicação ineficaz e desrespeitosa pode gerar um ambiente tóxico e desviar a atenção das soluções técnicas.

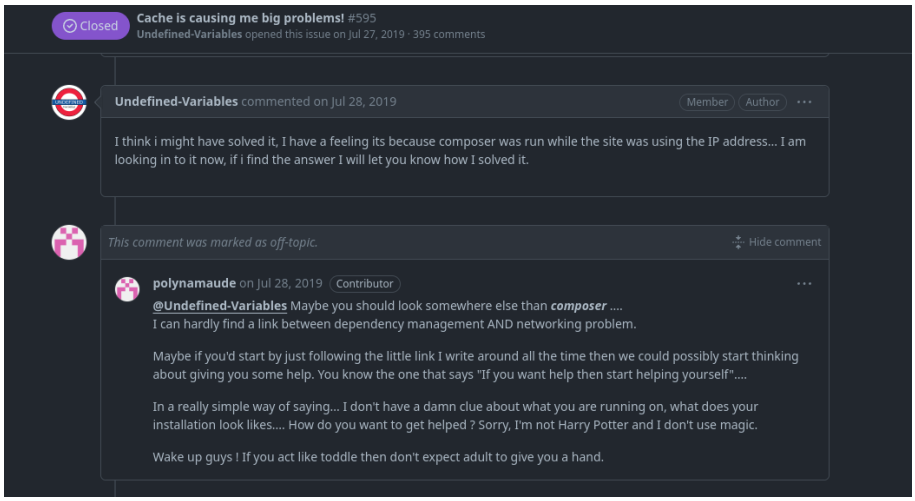

Figure 3. Issue Cache is causing me big problems!

2. **addfakeprofiles not generating profiles :** No repositório pH7-Social-Dating-CMS, o usuário rondlite relatou um problema com a funcionalidade ”Add Fake profiles”. Apesar de a interface mostrar uma barra verde indicando a adição de usuários, nenhum perfil era realmente adicionado. rondlite observou que o console de desenvolvedor mostrava o pedido ”addfakeprofiles” sendo cancelado. O problema foi identificado como estando na verificação de perfil antes da inserção no banco de dados. Ao comentar essa função, o problema foi resolvido temporariamente. Mais tarde, foi descoberto que o bug foi introduzido na versão 15.1.7 do pH7Builder e foi corrigido removendo o “@example.com” do arquivo de e-mails banidos. O tópico foi eventualmente trancado por falta de atividade recente.

   O comentário do polynamaude direcionado ao @Undefined-Variables. ele exibe características de toxicidade potencial através de seu tom condescendente e linguagem desdenhosa. A abertura ”Maybe I wasn’t clear enough...” sugere uma falha de compreensão por parte do destinatário, enquanto a questão retórica sobre a utilidade do problema discutido minimiza e invalida as preocupações do destinatário. Este conjunto de expressões contribui para a percepção de desrespeito e exclusão no comentário, alinhando-o com características de comunicação tóxica.

3. **Release in the Microsoft Store :** O problema 10759 no repositório do Visual Studio Code da Microsoft, que foi excluída, discutia a possibilidade de lançar o VS Code na Microsoft Store. Os participantes argumentaram que isso poderia facilitar a atualização e tornar o VS Code mais acessível, especialmente para plataformas como telefones e Xbox. Surgiram preocupações sobre limitações de sandboxing para aplicativos UWP e se o VS Code poderia funcionar nesse formato. Alguns sugeriram o uso como um IDE baseado em nuvem para contornar essas limitações. Houve apoio à ideia de tornar o VS Code um aplicativo Windows universal, destacando-se o benefício para usuários em regiões com acesso limitado a downloads diretos. Vários comentários refletiram entusiasmo pelo potencial de melhorar a produtividade e contribuir para a acessibilidade do desenvolvimento de software. Entretanto, houve um debate sobre a viabilidade e o futuro da Microsoft Store em si, com algumas opiniões expressando ceticismo sobre sua sustentabilidade a longo prazo.

   Contendo o comentário ”if some policy keeps vscode from being uploaded to the microsoft store then obv the policy is wrong..I have told microsoft a million times already to fire evryone that is working on that project. . . ” que é tóxico porque promove uma abordagem agressiva e desrespeitosa em relação aos profissionais envolvidos no projeto, simplifica indevidamente questões complexas e utiliza hiperbolização para enfatizar pontos de vista pessoais. Ele reflete uma falta de disposição para o diálogo construtivo e uma inclinação para a crítica destrutiva.

Nossas conclusões destacam a complexidade e os desafios na detecção de toxicidade em comunidades de software aberto. A análise sugere a necessidade de abordagens mais refinadas na detecção automatizada de emoções e sentimentos. Este estudo contribui para a compreensão da toxicidade online, enfatizando a importância de criar ambientes digitais mais saudáveis e colaborativos para o avanço do software aberto.

6. Conclusão

O estudo em questão lançou uma luz sobre a complexa questão da toxicidade em comunidades de código aberto, com um foco particular nas ”issues” do GitHub. A metodologia delineada permitiu uma análise aprofundada das interações dentro dessas comunidades, com destaque para a análise de comentários e a identificação de conteúdo potencialmente tóxico. Os resultados mostraram que, de um vasto número de comentários analisados, apenas uma pequena fração foi categorizada como tóxica por ambos os pesquisadores, sugerindo uma predominância de interações não tóxicas.

A inovação chave deste estudo foi a integração da detecção automatizada de emoções, que, apesar de sua promessa, mostrou-se limitada em eficácia, particularmente ao isolamento de comentários potencialmente tóxicos. A abordagem focada na emoção ”joy” com escores baixos não foi suficientemente assertiva para identificar comentários tóxicos com precisão. Este resultado sugere que, embora avançada, a tecnologia de detecção de emoções ainda pode necessitar de refinamentos para ser eficaz neste contexto específico.

Além disso, o estudo ressaltou a relevância de uma estratégia de filtragem bem definida, que, mesmo sendo criteriosa, deixou espaço para questionamentos sobre a precisão e eficácia dos parâmetros escolhidos. Isso indica uma necessidade contínua de refinamento nas estratégias de filtragem, talvez incorporando uma abordagem mais holística que considere múltiplas variáveis.

A análise dos comentários e a subsequente categorização manual de toxicidade também ressaltaram a importância de uma abordagem colaborativa e iterativa, onde a discussão e o alinhamento entre os pesquisadores foram cruciais para a validação dos resultados.

7. Trabalhos futuros

Realizar uma pesquisa direcionada a um grupo específico de usuários com o propósito de identificar e categorizar usuários tóxicos. Este estudo buscará desenvolver um entendimento mais profundo sobre os diferentes perfis e personas desses usuários, permitindo uma análise mais detalhada de suas características e comportamentos. Essa abordagem visa contribuir para a literatura sobre comportamento online, oferecendo insights sobre a natureza da toxicidade em ambientes digitais.

Implementar e avaliar o uso de ferramentas avançadas de detecção de emoções, como Google Cloud Natural Language API, Amazon Comprehend e Microsoft Azure Text Analytics API, para o processamento de texto em larga escala. O foco desta sugestão é realizar um estudo comparativo dessas ferramentas, analisando suas especificidades, eficiência e aplicabilidade. Esta comparação permitirá identificar a ferramenta mais adequada para analisar e interpretar as emoções expressas em grandes volumes de texto.

Explorar a utilização de Large Language Models (LLMs) para a classificação automatizada de textos. Dada a versatilidade, acessibilidade e eficiência dos LLMs em uma ampla gama de tarefas de análise de texto, desde anotação até análise crítica do discurso [T¨ornberg 2023], este estudo propõe investigar como esses modelos podem ser aplicados na análise de textos relacionados à toxicidade online. O objetivo é avaliar a capacidade

dos LLMs em comparar e complementar as ferramentas tradicionais de processamento de linguagem natural.

References

Huilian Sophie Qiu, B. V. (2022). Detecting interpersonal conflict in issues and code review: Cross pollinating open- and closed-source approaches. *Association for Computing MachineryNew YorkNYUnited States*, 1(1):71–97.

Joni Salminen, Sercan Sengün, J. C. S.-g. J. B. J. J. (2020). Topic-driven toxicity: Exploring the relationship between online toxicity and news topics. In *2020 IEEE/ACM 42nd International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 57–60.

Kavaler, D., Sirovica, S., Hellendoorn, V., Aranovich, R., and Filkov, V. (2017). Perceived language complexity in github issue discussions and their effect on issue resolution. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 72–83.

Klünder, J., Horstmann, J., and Karras, O. (2020). Identifying the mood of a software development team by analyzing text-based communication in chats with machine learning. In *Human-Centered Software Engineering*, pages 133–151. Springer International Publishing.

Miller, C., Cohen, S., Klug, D., Vasilescu, B., and Kästner, C. (2022). "did you miss my comment or what?" understanding toxicity in open source discussions. pages 710–722.

Novielli, N., Girardi, D., and Lanubile, F. (2018). A benchmark study on sentiment analysis for software engineering research. In *Proceedings of the 15th International Conference on Mining Software Repositories*. ACM.

Raman, N., Cao, M., Tsvetkov, Y., Kästner, C., and Vasilescu, B. (2020). Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions. In *2020 IEEE/ACM 42nd International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 57–60.

Sanei, A., Cheng, J., and Adams, B. (2021). The impacts of sentiments and tones in community-generated issue discussions.

T¨ornberg, P. (2023). How to use large language models for text analysis. ACM.