# RIT: uma Biblioteca para Análise da Relevância Temática dos Comentários Postados no *Issue Tracking* do GitHub

Estela Miranda Batista
Federal University of Viçosa
Florestal, Brasil
estela.batista@ufv.br

Melissa Araújo
Federal University of Viçosa
Florestal, Brasil
melissa.araujo@ufv.br

Fábio Trindade Ramos
Federal University of Viçosa
Florestal, Brasil
fabio.ramos@ufv.br

Gláucia Braga e Silva
Federal University of Viçosa
Florestal, Brasil
glaucia@ufv.br

Luiz Eugênio Coelho Neto
Federal University of Viçosa
Florestal, Brasil
luiz.e.neto@ufv.br

Santiago G. de Oliveira Souza
Federal University of Viçosa
Florestal, Brasil
santiago.souza@ufv.br

## ABSTRACT

The mining of software repositories generates useful and relevant knowledge for project management with direct application in effort estimates, task distribution and resource allocation. This work presents the RIT Library, which mines communication data from issue tracking repositories on GitHub, in order to calculate the thematic relevance of issue comments. To validate the library, a two-step validation procedure was applied: a validation of the results by comparing them with the opinion of business experts; and a proof of concept, which exemplifies the use of the library in a client application, highlighting how the different views of the generated data can be useful for future analyzes in external tools.

## KEYWORDS

Issue comments, Text Mining, Cosine Similarity

## 1 INTRODUÇÃO

A comunicação em projetos de software tem sido um dos objetos de estudo em diversos trabalhos na literatura [3, 13–15], sendo um fator crítico para os resultados de qualquer projeto. A troca de informações é importante para que os desenvolvedores compreendam o trabalho a ser feito e desenvolvam soluções apropriadas. Grande parte da comunicação fica concentrada nos repositórios de *Issue Tracking* [3], nas discussões em torno de questões (*issues*) relacionadas ao processo de controle de mudanças. Após serem reportadas, as *issues* recebem comentários, com informações implícitas e explícitas sobre o andamento de suas resoluções [15]. A qualidade desses comentários tem influência direta no entendimento do problema, no esforço dispendido e no sucesso (ou insucesso) de sua resolução.

Nesse contexto de análise das comunicações em *issue tracking*, este trabalho apresenta a Biblioteca RIT (**R**elevância em **I**ssue **T**racking), que aplica a métrica de relevância temática sobre os comentários das *issues* em repositórios do GitHub, com o intuito de classificar os mais relevantes para uma dada discussão. De acordo com Machado et al. [11], a Relevância Temática pode ser definida como uma métrica que determina a relevância de um texto no contexto do tema da discussão onde está inserido. Dessa forma, o valor da Relevância Temática dos comentários das *issues* será determinado pela quantidade de termos encontrados no texto dos comentários que são relevantes para a discussão como um todo.

A escolha de se analisar dados de comunicação provenientes das mensagens trocadas em ambientes de *Issue Tracking* se justifica porque estes dados ainda são pouco explorados apesar de seu valor no contexto do trabalho colaborativo e da gestão dos projetos. O GitHub foi escolhido tanto pela sua popularidade, com mais de 61 milhões de repositórios de software criados por mais de 16 milhões de desenvolvedores cadastrados[1], quanto pelo seu uso cada mais mais expressivo enquanto fonte para mineração de dados [8, 17].

Visto que as análises feitas nas comunicações de projetos de desenvolvimento de software podem revelar indicadores importantes para identificação de pessoas desenvolvedoras-chave, bem como falhas nas interações entre os envolvidos [13], os resultados gerados pela aplicação da métrica de relevância temática podem ser usados por gerentes de projetos diretamente ou podem compor dados para outras análises mais completas. Do ponto de vista prático, a biblioteca RIT pode trazer indicadores quali-quantitativos importantes sobre a participação dos membros, como número de comentários postados e relevância temática dos comentários postados. Esses indicadores podem ser úteis em tarefas distribuição de tarefas e alocação de recursos[6]. Do ponto de vista científico, os resultados podem ser úteis em pesquisas que explorem a relação entre a relevância temática dos comentários e outros fatores do desenvolvimento como tempo de resolução das *issues*, questões de gênero e produtividade.

O artigo está organizado da seguinte forma: a seção 2 traz a fundamentação teórica sobre comunicações em *Issue Tracking*. Na seção 3, tem-se os trabalhos relacionados. A métrica de relevância temática é apresentada na seção 4. A seção 5 traz uma visão geral da Biblioteca RIT, com alguns artefatos de modelagem e detalhes técnicos de implementação. Na seção 6, tem-se a validação da biblioteca RIT. A seção 7 traz as ameaças à validade do trabalho e a seção 8, as considerações finais.

## 2 ISSUE TRACKING E COMUNICAÇÕES NO GITHUB

Ambientes de *Issue Tracking* são comumente utilizados para gerenciar as tarefas de desenvolvimento em torno da resolução de questões (*issues*) reportadas por desenvolvedores e usuários de sistemas [9]. O uso desses ambientes melhora o fluxo de conhecimento por conceder aos colaboradores a possibilidade de destacar gargalos no

[1] https://octoverse.github.com/

processo e compartilhar opiniões de interesse mútuo entre os envolvidos [4]. Além disso, constituem um meio de comunicação em si e, por consequência, tornam-se um repositório importante do histórico das discussões acerca do trabalho realizado [3].

O ambiente de *Issue Tracking* do GitHub é chamado de GitHub Issues[2]. Nesse ambiente, os colaboradores de um projeto discutem questões relacionadas ao sistema sendo desenvolvido, por meio de *issues*, que podem representar erros (bugs), melhorias ou novos requisitos. Ao criar uma nova *issue*, o usuário deve preencher obrigatoriamente o campo de título, embora outros campos esteja disponíveis (Figura 1). Campos como descrição, *Labels*, que categorizam os tópicos, e *Assignees*, contendo os responsáveis pela resolução da *issue*, também estão disponíveis, mas não são obrigatórios. Existem também outros campos gerados automaticamente pela plataforma, como o autor, a data de criação e o estado (*status*). Quando uma *issue* é aberta, o valor atribuído ao seu estado será *open*, mas pode ser definido como *closed* quando a mesma for dada como resolvida.

Como o Github é uma plataforma aberta, seu workflow mostra-se pouco burocrático, já que qualquer usuário pode discutir e contribuir para o progresso de um produto de software. Com isso, muitos dos campos que definem uma *issue* são opcionais e não são preenchidos, na maioria das vezes. Uma alternativa para complementar essas lacunas envolve minerar os textos das *issues*, que incluem o título, a descrição e os comentários associados, para que informações complementares sejam capturadas/extraídas.

## 3 TRABALHOS RELACIONADOS

A mineração de repositórios de software, têm sido objeto de estudo de diversos trabalhos na literatura [8, 17], sendo a maioria das pesquisas voltada aos dados dos sistemas de controle de versão. Ferreira *et al.* [7] [6] desenvolveram a ferramenta *Developer Tracker App*, que permite a mineração de repositórios Git, aplicando medidas quantitativas, tais como, o grau de importância de um determinado desenvolvedor no projeto, medido em função da participação em *commits*, por exemplo. A biblioteca RIT também auxilia na avaliação da participação das pessoas desenvolvedoras no projeto, mas em termos da relevância dos comentários postados em *issue tracking*. Ortu et al. [14] também exploram dados de *Issue Tracking*, em um estudo empírico que apresenta resultados sobre o impacto de alguns fatores na resolução das *issues*. Krasniqi [10] apresenta uma ferramenta, chamada RETRORANK, para recomendar os comentários de *issues* mais relevantes para sua solução. Dentre as técnicas usadas no estudo para essa recomendação, encontra-se a relevância semântica entre os comentários, usando um modelo baseado em grafos. A RIT também avalia a relevância de comentários, mas usa a similaridade de cossenos no cálculo da métrica.

Neto and Silva [13] propuseram a ferramenta ColMiner para análise das comunicações no *Issue Tracking* do GitHub, baseada na aplicação da métrica de Relevância Temática, de forma adaptada, para se avaliar a qualidade dos comentários postados nas *issues*. A ferramenta aplica essa métrica em conjunto com outras análises dos dados de *Issue Tracking* e de controle de versão, com o intuito de identificar desenvolvedores chave para o projeto. O estudo de Batista et al. [2] também utiliza essa mesma métrica em um estudo comparativo de desigualdade de gênero no issue tracking do GitHub. A biblioteca RIT também endereça a aplicação da métrica de Relevância Temática no contexto de *Issue Tracking* e será construída com base no trabalho de Neto and Silva [13].

[2]https://github.com/features/issues

## 4 MÉTRICA DE RELEVÂNCIA TEMÁTICA EM ISSUE TRACKING

A métrica de Relevância Temática foi originalmente proposta para análise de comunicações em fóruns de discussão educacionais [1, 11]. No entanto, seu uso no contexto de *Issue Tracking* já foi reportado na literatura [13]. Nesse contexto, os textos dos comentários de uma *issue* são usados para o cálculo da Relevância Temática com base nos conceitos relevantes extraídos do título e da descrição da *issue*. Neto and Silva [13] definem que o cálculo da Relevância Temática deve ser feito a partir do número de conceitos de cada comentário, sua frequência na discussão e seus relacionamentos. A Equação 1 apresenta a fórmula para cálculo da métrica [13], em que $S_{CI}$ consiste na semelhança entre o comentário e a *issue* (título e descrição); $S_{CD}$ equivale à semelhança entre o comentário e a discussão (definida pelo comentário e o título da *issue*); e $S_{CC}$, representa a semelhança entre o comentário e seu anterior (caso exista).

$$RT = max\{S_{CI}, S_{CD}, S_{CC}\} \quad (1)$$

Como a Relevância Temática corresponde ao valor máximo entre as semelhanças $S_{CI}$, $S_{CD}$ e $S_{CC}$, o cálculo da semelhança em si representa um componente chave da métrica.

Duas técnicas são comumente utilizadas na literatura [1, 5, 16] para cálculo de similaridade entre textos, no contexto de fóruns de discussão: a similaridade por cossenos e a comparação de grafos.

Segundo Brandão et al. [5], a similaridade por cossenos utiliza o método de vetorizar os textos para fazer uma análise angular entre os conteúdos textuais. Calcula-se o cosseno do ângulo entre os vetores que representam as interações para determinar sua similaridade. A técnica envolve o estudo do conteúdo textual vetorizado e realiza inferências por meio da distância angular dos vetores [5].

Já a comparação de grafos, utiliza grafos para representar os termos mais relevantes de um dado conteúdo textual, de forma que os vértices representam os termos mais relevantes, e as arestas conectam os termos que aparecem em conjunto no texto, podendo possuir ou não um valor que referencie o número de vezes que esses termos aparecem em conjunto. Sobre os grafos, aplicam-se métricas referentes às ligações entre vértices, como a quantidade de vizinhos, para expressar a correlação entre estes [1]. A técnica é baseada em atributos que avaliam como palavra se relaciona com o contexto em que está inserida, tal como o peso e a frequência dos termos em uma sentença e as suas semelhanças com as métricas de outros grafos [16].

Nesse sentido, ao utilizar os grafos, aplicam-se diversas equações matemáticas para determinar a relação das palavras armazenadas nos nós, o que não ocorre ao se utilizar a similaridade por cossenos. Dessa forma, o cálculo da similaridade por cossenos mostra-se mais simples em termos de implementação e custo computacional, sendo adotada no contexto desta pesquisa.


**Figura 1: Exemplo de *Issue* no GitHub**

### 4.1 Modificação no cálculo da Métrica Relevância Temática

Para automatizar o cálculo da métrica de Relevância Temática, foram testadas as duas técnicas de cálculo de similaridade de textos: comparação de grafos, usada no trabalho de referência [13], e similaridade por cossenos.

A comparação de grafos foi usada de forma similar ao trabalho de referência [13], mantendo-se a dependência à ferramenta externa SOBEK [16] para geração dos grafos. Foi feita uma pequena adaptação técnica, substituindo-se o acesso via *Web service* por uma versão offline da ferramenta. Embora a adaptação tenha simplificado a arquitetura, ainda existem problemas com desempenho, em virtude do volume de grafos gerados. Para cada *issue*, são gerados os seguintes grafos: um para a *issue*, representada pelo texto do título e da descrição; um para cada comentário postado; e um para a discussão como um todo, representada por uma concatenação do título, da descrição e dos comentários anteriores ao analisado.

Já para aplicação da técnica de similaridade por cossenos, todas as equações da métrica foram verificadas para se avaliar a necessidade de adaptações. As verificações foram feitas testando-se as fórmulas com alguns exemplos de *issues*. Ao aplicar a fórmula da Equação 1, apresentada na Seção 4, em alguns testes, observou-se que a similaridade do comentário em relação ao seu comentário anterior ($S_{CC}$) assumia, em diversos momentos, o valor zero no cálculo da similaridade de cossenos, mostrando-se irrelevante para o cálculo da relevância temática. Assim, após alguns testes, foi possível notar que a similaridade do comentário em relação à discussão($S_{CD}$) e a similaridade do comentário em relação à *issue* ($S_{CI}$) eram suficientes para compor a equação de cálculo da relevância temática. Assim, de forma similar à equação original apresentada por Azevedo [1], optou-se pela média aritmética de $S_{CI}$ e $S_{CD}$, como apresentado na Equação 2. Ressalta-se que a discussão corresponde a uma concatenação do título, da descrição e dos comentários anteriores da *issue*.

$$RT = \frac{S_{CI} + S_{CD}}{2} \quad (2)$$

Para os testes comparativos realizados, o uso da técnica de cossenos provocou impactos significativos em termos do desempenho, quando comparada à técnica de similaridade por grafos, uma vez sua aplicação é mais simples em termos de processamento, pois não requer a utilização de ferramentas externas. Além disso, a técnica de cossenos apresenta uma alta precisão em relação à classificação humana, como observado por Medeiros et al. [12].

Dessa forma, neste trabalho, o cálculo da métrica de Relevância Temática será feito com base técnica de similaridade por cossenos, para cálculo de $S_{CI}$ e $S_{CD}$, que serão usados na Equação 2. O procedimento detalhado de validação desta adaptação será apresentado na Seção 6.1.

## 5 BIBLIOTECA RIT

De acordo com o SEVOCAB (*Software and Systems Engineering Vocabulary*)[3], uma biblioteca de software é "uma coleção controlada de software e documentação relacionada projetada para auxiliar no desenvolvimento, uso ou manutenção de software". A biblioteca RIT apresenta-se como uma biblioteca de software responsável por calcular a relevância temática de comentários postados no *issue tracking* do GitHub, servindo como ferramenta de apoio à análise das comunicações ocorridas neste ambiente. Não faz parte do escopo da RIT a apresentação gráfica dos resultados ou qualquer análise estatística sobre os mesmos. Para ser utilizada, a biblioteca pode ser integrada ao código de outras ferramentas de suporte ao gerenciamento de projetos de software. Outro uso potencial para a RIT consiste na utilização dos dados gerados em ferramentas específicas para análise de dados.

Esta seção apresenta a biblioteca RIT, suas funcionalidades e sua arquitetura, modelada em classes e componentes. Por fim, são apresentados alguns detalhes técnicos de implementação.

### 5.1 Especificação e Modelagem da RIT

A biblioteca RIT possui as seguintes macro-funcionalidades:

[3]https://pascal.computer.org/

(1) Extração de dados de *issues*, a partir de repositórios do GitHub;
(2) Aplicação do cálculo da relevância temática sobre cada comentário das *issues*; e
(3) Exportação dos resultados com as relevâncias calculadas em arquivos no formato CSV.

A Figura 2 apresenta uma visão lógica da biblioteca, por meio das classes que a representam, organizadas em camadas.

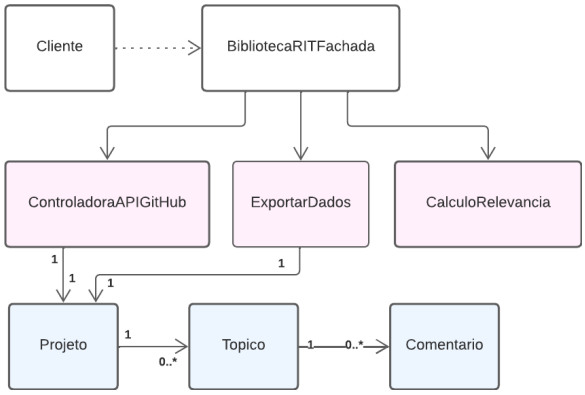


**Figura 2: Diagrama de Classes da RIT**

Conforme pode ser observado na Figura 2, a classe **BibliotecaRITFachada** encapsula o código da biblioteca e fornece, ao *Cliente*, uma interface de acesso às suas macro-funcionalidades, por meio do padrão de projeto *Facade*[4]. As responsabilidades de cada classe encontram-se descritas a seguir:

- **BibliotecaRITFachada:** responsável por utilizar as demais classes e controlar todo o fluxo das operações, desde a extração de dados, segundo as entradas do código *Cliente*, até a exportação dos dados;
- **ControladoraAPIGitHub:** faz requisições a partir de filtros disponibilizados pela API do GitHub, para obter os dados das *issues*, segundo os parâmetros fornecidos pelo código *Cliente*;
- **CalculoRelevancia:** automatiza a aplicação da métrica de relevância temática, usando a técnica de similaridade por cossenos, sobre todos os comentários das *issues* extraídas para um dado repositório;
- **ExportarDados:** permite a exportação dos dados obtidos ao final do processo do cálculo de relevância para arquivos no formato CSV. Nessa classe, existem 05 visões para geração dos dados, a partir de filtros selecionados pelo *Cliente*, relacionados a status das *issues*, autoria e data do comentário.
- **Projeto, Topico e Comentario:** classes que representam os objetos do domínio, instanciados na etapa de extração e exportados, junto com os respectivos valores de relevância calculados (um para cada comentário da *issue*).
  - *Topico*: abstrai uma *issue*, sendo representada pelos seguintes atributos: ID da *issue*, login do autor, login do usuário ao qual foi atribuído a *issue*, data de criação e fechamento, título, número de identificação visível ao usuário, descrição, status da *issue* e os seus respectivos comentários.
  - *Comentario*: representa os comentários de uma *issue* e contém os seguintes atributos: ID do comentário, ID do tópico ao qual este comentário pertence, o comentário, data de postagem, a relevância temática e a reputação do autor.
  - *Projeto*: representa atributos de um repositório, sendo eles: nome de usuário proprietário do repositório, nome do repositório e uma lista de *issues* (objetos *Topico*).

## 5.2 Arquitetura e Detalhes Técnicos de Desenvolvimento

O código da biblioteca RIT foi desenvolvido na linguagem Python, a partir do diagrama de classes da Figura 2. O código foi produzido sob a licença MIT[5], o que favorece tanto seu uso quanto possíveis extensões e adaptações.

A Figura 3 apresenta o diagrama de componentes da biblioteca, em que os componentes internos estão representados na cor roxa; componentes referentes às bibliotecas auxiliares da linguagem Python, na cor rosa; e um componente representando o código *Cliente*, na cor azul.

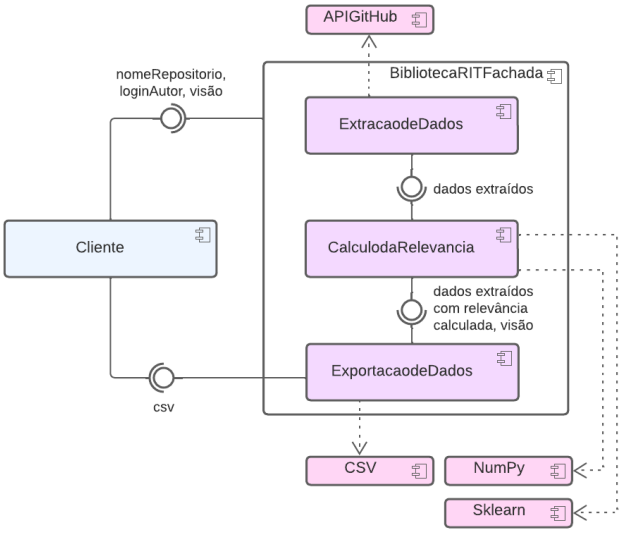


**Figura 3: Diagrama UML de Componentes da RIT**

Ao receber uma requisição do *Cliente*, contendo o nome do repositório a ser analisado, o nome do usuário dono do repositório e a visão de exportação escolhida, a classe **BibliotecaRITFachada** acionará o componente **ExtracaodeDados**, que enviará os dados para a **ControladoraAPIGitHub**. Essa última irá realizar requisições para a API do GitHub[6]. A API permite a extração de *issues*

[4]https://refactoring.guru/pt-br/design-patterns/facade

[5]https://opensource.org/licenses/MIT
[6]https://docs.github.com/pt/rest

segundo o seu status (*issues* abertas e *issues* fechadas). A **ControladoraAPIGitHub** recebe os dados e retorna um objeto da classe **Projeto**, que armazena esses dados. É importante notar que estes dados não estão estruturados em sua forma original, então deve-se armazená-los em coleções de documentos estruturados [18], como por exemplo, vetores de palavras ou grafos. Para isso, o retorno das requisições, que estão em formato JSON, são atribuídos às suas respectivas classes (**Projeto**, **Topico** e **Comentario**) que são responsáveis pela estruturação destes conteúdos textuais.

Em um segundo momento, os dados extraídos, agora representados pelos objetos **Projeto**, **Topico** e **Comentario**, são então enviados para o componente **CalculodaRelevancia**, que acionará a classe **CalculoRelevancia**, responsável por realizar o cálculo da relevância temática de cada comentário de cada *issue* (objeto Topico) extraída. Para isso, são utilizadas duas outras bibliotecas, a **Sklearn**[7] e a **NumPy**[8], responsáveis pelo cálculo da similaridade de cossenos, uma vez que geram a vetorização automática do texto. Importante destacar que informações contidas nos comentários que não são texto da mensagem em si (imagens, trechos de código, conteúdo em links externos, entre outros) são descartados nesta etapa.

Por fim, após extraídos os dados com as respectivas relevâncias temáticas calculadas, eles são enviados para o componente **ExportacaodeDados**, juntamente com a visão de exportação escolhida. Esse componente acionará a classe **ExportarDados** para gerar o arquivo CSV a ser entregue ao código *Cliente*. Os dados são gerados com auxilio da biblioteca **CSV**[9], de forma a realizar a transformação dos dados salvos em cada uma das classes citadas anteriormente em colunas selecionadas para esse trabalho. O arquivo CSV gerado, segundo uma das cinco visões, possui 9 colunas:

(1) Número da *issue* a que o comentário pertence
(2) Título da *issue* a que o comentário pertence
(3) Descrição da *issue* a que o comentário pertence
(4) Data de criação da *issue* a que o comentário pertence
(5) Número do comentário
(6) Texto do comentário
(7) Data de postagem do comentário
(8) Valor da Relevância Temática do comentário
(9) Autor do comentário postado

Tais colunas foram selecionadas baseadas em necessidades encontradas para a análise dos dados. O número de cada *issue*, e sua data foram inseridos para que o usuário possa filtrar *issues* específicas. O título e a descrição da *issue* foram adicionados a cada um dos comentários para que o usuário possa realizar outras análises, caso necessário. De mesma forma que para as *issues*, os comentários possuem data, relevância temática calculada e o autor do comentário para que possam ser feitas filtragens especificas, e análises dos dados, e o seu texto salvo para que outras análises possam ser feitas posteriormente.

## 6 VALIDAÇÃO DA BIBLIOTECA RIT

A validação da biblioteca RIT foi executada em duas etapas: a primeira, com um procedimento de validação por especialistas para verificar a precisão e a confiabilidade do cálculo da Relevância Temática após a substituição pela técnica de similaridade por cossenos; e a segunda, com uma prova de conceito para ilustrar o uso da biblioteca em um dado contexto.

### 6.1 Validação da adaptação da Métrica de Relevância Temática

Considerando as alterações apresentadas sobre a abordagem [13] para cálculo da métrica de relevância temática, foi realizado um procedimento de validação em termos do impacto na precisão dos resultados. Esse procedimento abrangeu uma comparação dos valores obtidos automaticamente com a opinião de especialistas, que atribuíram manualmente valores de relevâncias temáticas em um conjunto de *issues*. O grupo de especialistas contou com um total de 12 voluntários, sendo 3 engenheiros de *software*; 2 especialistas de domínio, ou seja, que participam da *issue* em questão; e 7 pessoas desenvolvedoras, com níveis de conhecimento variados.

Para a validação, foi enviado um questionário que endereçava uma determinada *issue*, e campos em que os especialistas atribuíam uma relevância de 0 a 4 para cada comentário daquela *issue*. Ao todo, foram utilizadas 12 *issues*, com 59 comentários no total. Cada *issue* foi analisada por 3 especialistas, sendo: dois com perfil técnico, representados por profissionais e acadêmicos com experiência em desenvolvimento de software; e o terceiro, representado pela equipe de desenvolvimento da biblioteca, que computou um único valor de relevância, calculado pela média das relevâncias de cada membro. Embora com menor peso, a equipe foi considerada como alternativa para ampliar a quantidade de especialistas (1 a mais por *issue*), uma vez que encontrar voluntários para este tipo de validação é um desafio. Além disso, os resultados da equipe convergiam com os dos demais especialistas, não sendo identificado nenhum tipo de viés nas análises.

A Tabela 1 apresenta um exemplo da validação para uma *issue*, em que a relevância final corresponde à média aritmética entre os valores de relevância definidos pelos especialistas e a média de relevância atribuída pela equipe de desenvolvimento, sendo representada pela coluna *#Relevância Atribuída*.

**Table 1: Exemplo de Avaliação Manual de Relevância Temática dos Comentários de uma *Issue***

| Comentário | #Média Equipe | #Especialista 1 | #Especialista 2 | #Relevância Atribuída |
|---|---|---|---|---|
| 1 | 3.8 | 3 | 4 | 3.6 |
| 2 | 2.0 | 2 | 2 | 2.0 |
| 3 | 4.0 | 4 | 3 | 3.7 |
| 4 | 1.0 | 1 | 0 | 0.7 |

As relevâncias atribuídas pelos especialistas foram comparadas com valores automaticamente calculados em 4 diferentes versões, conforme descrito a seguir:

(1) A abordagem original proposta por Azevedo [1], usando da similaridade de grafos;
(2) A abordagem proposta por Azevedo; [1], adaptada para uso da similaridade de cossenos;

[7] https://scikit-learn.org/
[8] https://numpy.org/
[9] https://docs.python.org/3/library/csv.html

(3) A abordagem proposta por Neto and Silva [13], usando da similaridade de grafos;
(4) A abordagem descrita por Neto and Silva [13], usando da similaridade de cossenos.

Para realizar as comparações entre as abordagens, foi usada a métrica do Erro Absoluto Médio (MAE -*Mean Absolute Error*), que representa a média da diferença absoluta entre entre os valores reais e os previstos no conjunto de dados. A Equação 3 apresenta a fórmula para cálculo do MAE, em que $y$ representa o valor real, e $\hat{y}$ o valor previsto.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \|y - \hat{y}\| \tag{3}$$

Além do MAE, foi também usada a métrica do Erro Quadrado Médio (MSE - *Mean Squared Error*), que representa a média quadrada entre os valores reais e os previstos no conjunto de dados, como apresentado na Equação 4.

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y - \hat{y})^2 \tag{4}$$

Os valores de MAE e MSE para cada uma das abordagens podem ser observados na Tabela 2.

**Table 2: MAE e MSE das Abordagens Propostas**

| Abordagem | MAE | MSE |
|---|---|---|
| (1) | 37.88% | 19.22% |
| (2) | 8.45% | 1% |
| (3) | 21.02% | 6.69% |
| (4) | 9.57% | 1.85% |

Os resultados mostram que as abordagens que utilizam a técnica de similaridade por grafos (1 e 3) alcançaram valores altos tanto para MAE quanto para MSE, demonstrando uma baixa precisão. Olhando para as abordagens que melhor se destacaram, com menores valores de MAE e MSE, as abordagens 2 e 4, que aplicaram a técnica de cossenos para cálculo da similaridade, obtiveram valores próximos. A abordagem 4 foi escolhida para ser utilizada na biblioteca RIT, uma vez que além de ter apresentado alta precisão, com os segundos menores valores de MAE e MSE, o cálculo da relevância temática é compatível com a Equação 2, apresentada na Seção 4.1.É importante ressaltar que a escolha também considerou o fator apresentado anteriormente, de que em muitos casos o valor de $S_{CC}$, usada na abordagem 2, assumia valores zero, o que em bases de dados maiores pode causar *outliers*, conforme verificado durante a validação.

## 6.2 Prova de Conceito

Para demonstrar o funcionamento da biblioteca RIT, foi desenvolvido um código *Cliente* simples, que acessa as funcionalidades da biblioteca por meio da classe *BibliotecaRITFachada*. A interface em modo texto deste *Cliente* pode ser visualizada na Figura 4.

Ao executar o código do *Cliente*, o usuário irá inserir informações sobre o repositório do qual deseja extrair as informações. Essas

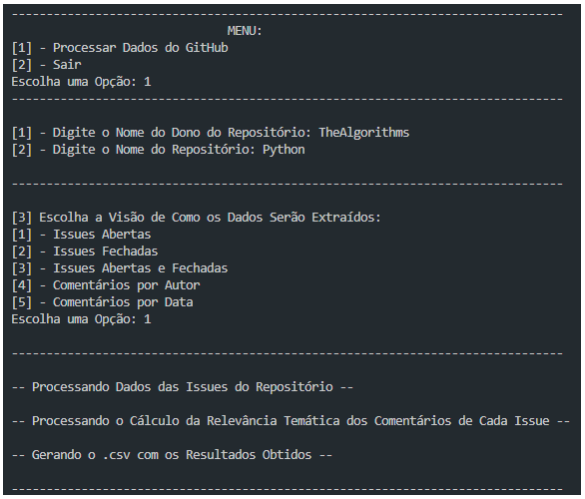


**Figura 4: Exemplo de Uso da Biblioteca RIT**

informações serão usadas na invocação do método *processarDadosGitHub()*, apresentado na linha 1 do Código 1. Esse método retorna um objeto da classe *Projeto*. De posse da instância de *Projeto*, realiza-se a chamada do método *calcularRelevanciaTematicaGitHub()*, que realiza o processamento dos dados para calcular a relevância de cada comentário, como apresentado na linha 2 do Código 1. Por fim, o usuário deve selecionar uma das 05 visões para exportação de dados disponíveis:

(1) Exportar *Issues* Abertas
(2) Exportar *Issues* Fechadas
(3) Exportar *Issues* Abertas e Fechadas
(4) Exportar Comentários de um determinado Autor
(5) Exportar Comentários realizados em uma Data Especificada

Após a escolha da visão de exportação, realiza-se a chamada do método *gerarCSVGitHub()*, apresentado na linha 3 do Código 1, em que *tipoExportacao* é um inteiro correspondente a uma das cinco visões.

```
1 projeto = BibliotecaRITFachada.processarDadosGitHub(usuario,
      repositorio, tipoIssue)
2 BibliotecaRITFachada.calcularRelevanciaTematicaGitHub(
     projeto)
3 BibliotecaRITFachada.gerarCSVGitHub(projeto, tipoExportacao)
```

**Listing 1: Demontração da Invocação de Métodos**

As visões são criadas para facilitar a visualização dos dados e seu uso subsequente em ferramentas de análises de dados. Dessa forma, para ilustrar uma aplicação da biblioteca RIT, os dados gerados pela aplicação *Cliente* produzida foram inseridos na ferramenta Jupyter Notebook[10]. Os dados gerados pela aplicação *Cliente* são do repositório *Ruby on Rails*[11], sendo que os dados foram extraídos no dia 18 de Julho de 2022. A visão de exportação escolhida foi a (1) *Exportar Issues Abertas*, o que reflete também no filtro aplicado para extração dos dados. Ou seja, só foram extraídas *issues* abertas naquele momento do projeto.

[10]https://jupyter.org/
[11]https://github.com/rails/rails

Para exemplificar o uso dos dados gerados pela RIT, foram construídos alguns gráficos no Jupyter Notebook. A Figura 5 apresenta um gráfico com o número de comentários postados em cada uma das 10 primeiras *issues*. Visto que o projeto escolhido tem atualizações constantes, e foram selecionados comentários apenas das 10 primeiras *issues* listadas, as datas são próximas entre si. Tal gráfico pode ser utilizado para entender o fluxo de trabalho de uma equipe de desenvolvimento ao longo de um determinado ciclo (como uma *Sprint*, por exemplo), sendo possível verificar quantos comentários foram postados num dado intervalo de tempo (datas inicial e final). Além disso, a filtragem por data também pode ser usada para se avaliar a relevância temática dos comentários postados num período de tempo, para se investigar questões como qualidade nas comunicações em função da etapa do desenvolvimento, qualidade das mensagens em determinados dias da semana, entre outras.

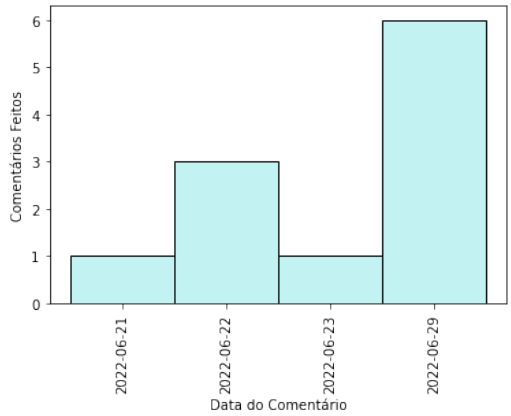


**Figura 5: Número de Comentários Postados em Cada Dia**

Já na Figura 6, tem-se um boxplot com as relevâncias dos comentários das *issues* extraídas. Nesse gráfico, é possível observar uma concentração de valores de relevância temática bem baixos para as *issues* avaliadas. Para essas *issues* analisadas, existem muitos comentários que possuem imagens e trechos de código, fazendo com que se tenha pouco texto analisado e, consequentemente, baixos valores de relevância temática.

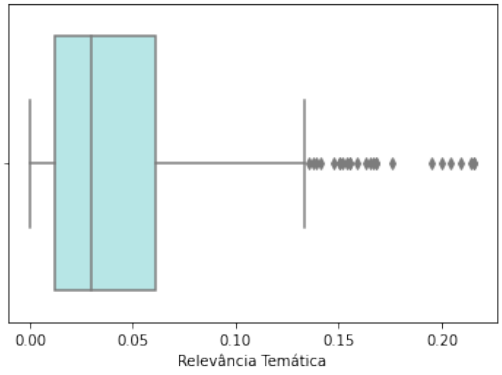


**Figura 6: Boxplot das Relevâncias dos Comentários Postados**

Para ilustrar a geração de outros dados, a visão de exportação foi alterada para (4) *Exportar Comentários de um determinado Autor*.

A Figura 7 apresenta um gráfico com o número de comentários postados por uma pessoa desenvolvedora em cada dia. Nele, é possível observar o quão atuante e participativa essa pessoa é nas discussões no projeto. Nesse caso, também podem ser cruzados os dados da relevância temática calculada com as datas filtradas, de forma que se possa identificar a produtividade e a qualidade da comunicação de uma pessoa desenvolvedora ao longo de um determinado projeto de software.

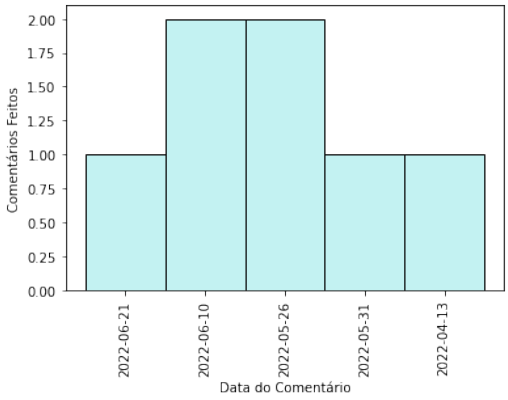


**Figura 7: Número de Comentários Postados em Cada Dia por uma Pessoa Desenvolvedora**

Do ponto de vista da qualidade dessa participação, a Figura 8 ilustra um boxplot com os dados das relevâncias temáticas dos comentários postados por essa pessoa desenvolvedora.

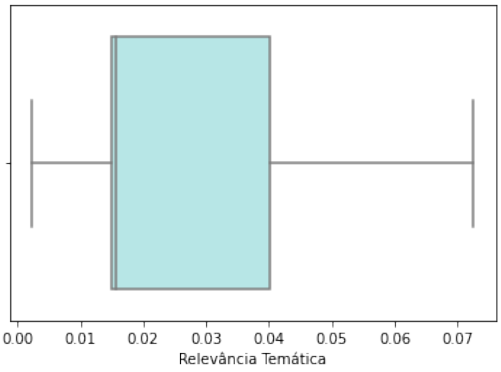


**Figura 8: Boxplot das Relevâncias dos Comentários Postados por uma Pessoa Desenvolvedora**

Do ponto de vista gerencial, estes gráficos podem ser úteis para apoiar os gerentes de projeto nas análises das comunicações, de forma a identificar desenvolvedores-chave nos projetos e facilitar alocações futuras.

## 7 AMEAÇAS À VALIDADE

Em relação ao uso da métrica de Relevância Temática, ainda existem poucos estudos sobre sua aplicação no contexto de *Issue Tracking*. Além disso, a natureza da mensagem neste contexto, em que tem-se comentários com imagens, trechos de código, conteúdo em links externos e mensagens de agradecimento, faz com que muitos valores de relevância sejam relativamente baixos. Esses elementos ou são ignorados na versão atual do cálculo da métrica ou não geram similaridade com a temática da *issue*, embora, em geral, sejam relevantes para a discussão como um todo.

## 8 CONSIDERAÇÕES FINAIS

Este trabalho apresentou a RIT, uma biblioteca que automatiza o cálculo da métrica de Relevância Temática sobre comentários postados no *Issue Tracking* do GitHub. A biblioteca mostra-se útil como ferramenta auxiliar para análise de dados de comunicação em projetos de software e também em outras tarefas gerenciais como estimativas de esforço, distribuição de tarefas e alocação de recursos. Os dados com os valores de relevâncias calculados são exportados em formato intercambiável, o que facilita seus usos subsequentes em ferramentas de análises de dados e em modelos de previsão, por exemplo.

Como trabalhos futuros, destacam-se: a) investigar como tratar conteúdos não textuais dos comentários (imagens, trechos de código, links) e avaliar o impacto destes no cálculo da relevância temática; b) aplicar a biblioteca em uma amostra representativa de projetos para fins de avaliação da métrica de relevância temática no contexto de *issue tracking*; c)investigar novas visões para exportação dos dados e o uso de um ou mais padrões de projeto que deixem a funcionalidade de exportação ainda mais modular, extensível e de fácil manutenibilidade; e d) adaptar a biblioteca para uso em outras ferramentas de *Issue Tracking* e/ou para uso em outros contextos, como para análise de discussões no StackOverflow.

## DISPONIBILIDADE DE ARTEFATOS

O código da biblioteca RIT está disponível em um repositório aberto na plataforma do GitHub[12]. Para utilizá-lo, basta realizar o clone do repositório e gerar um *Personal Acess Token*, que deve ser inserido nos arquivos da biblioteca.

## REFERENCES

[1] Breno Fabrício Terra Azevedo. 2011. *Minerafórum : um recurso de apoio para análise qualitativa em fóruns de discussão.* Tese de Doutorado em Informática na Computação. Universidade Federal do Rio Grande do Sul.

[2] Estela Batista, Gláucia Braga e Silva, and Thais Silva. 2022. Diversidade de Gênero em Projetos Open Source: um Estudo da Relevância dos Comentários Postados em Issues do GitHub. In *Anais do XVI Women in Information Technology* (Niterói). SBC, Porto Alegre, RS, Brasil, 197–202. https://doi.org/10.5753/wit.2022.222628

[3] Dane Bertram, Amy Voida, Saul Greenberg, and Robert Walker. 2010. Communication, Collaboration, and Bugs: The Social Nature of Issue Tracking in Small, Collocated Teams. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work* (Savannah, Georgia, USA) *(CSCW '10)*. Association for Computing Machinery, New York, NY, USA, 291–300. https://doi.org/10.1145/1718918.1718972

[4] Elizabeth Bjarnason, Baldvin Gislason Bern, and Linda Svedberg. 2022. Inter-team communication in large-scale co-located software engineering: a case study. *Empirical Software Engineering* 27, 2 (2022), 1–43. https://doi.org/10.1007/s10664-021-10027-z

[5] Maicom Sergio Brandão, Moacir Godinho-Filho, Walther Azzolini Junior, Bruna Christina Battissacco, and Josadak Astorino Marçola. 2022. Melhoria da categorização de produtos a partir do uso de algoritmos de aprendizado de máquina e medidas de similaridade. *Revista Produção Online* 21, 4 (mar. 2022), 2093–2124. https://doi.org/10.14488/1676-1901.v21i4.4483

[6] Matheus Silva Ferreira, Paulo Pereira Júnior, and Heitor Augustus Xavier Costa. 2021. Understanding Developers' Work - A Visual Approach for Project Managers. In *XX Brazilian Symposium on Software Quality* (Virtual Event, Brazil) *(SBQS '21)*. Association for Computing Machinery, New York, NY, USA, Article 16, 10 pages. https://doi.org/10.1145/3493244.3493256

[7] Matheus Silva Ferreira, Paulo Afonso Júnior, and Heitor Costa. 2021. Developer Tracker App: Uma Ferramenta para Visualizar o Trabalho dos Desenvolvedores. *ACM International Conference Proceeding Series*, 27–32. https://doi.org/10.1145/3474624.3476009

[8] Georgios Gousios and Diomidis Spinellis. 2017. Mining Software Engineering Data from GitHub. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, Buenos Aires, Argentina, 501–502. https://doi.org/10.1109/ICSE-C.2017.164

[9] Maliheh Izadi, Kiana Akbari, and Abbas Heydarnoori. 2022. Predicting the objective and priority of issue reports in software repositories. *Empirical Software Engineering* 27, 2 (2022), 1–37. https://doi.org/10.1007/s10664-021-10085-3

[10] Rrezarta Krasniqi. 2021. Recommending Bug-fixing Comments from Issue Tracking Discussions in Support of Bug Repair. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. 812–823. https://doi.org/10.1109/COMPSAC51774.2021.00114

[11] Crystiano José Richard Machado, Alexandre Magno Andrade MacIel, Rodrigo Lins Rodrigues, and Ronaldo Menezes. 2019. An approach for thematic relevance analysis applied to textual contributions in discussion forums. *International Journal of Distance Education Technologies* 17 (2019), 37–51. Issue 3. https://doi.org/10.4018/IJDET.2019070103

[12] Danielle C. Medeiros, José Eustáquio Rangel de Queiroz, and Joseana M. F. R. Araújo. 2014. Análise de Funções de Similaridade para Verificação do Conteúdo de Mensagens em Fóruns de Discussão. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)*. SBC, Dourados, Mato Grosso do Sul, Brazil, 144. https://doi.org/10.5753/cbie.sbie.2014.144

[13] Luiz Eugênio Coelho Neto and Gláucia Braga e Silva. 2018. ColMiner: A Tool to Support Communications Management in an Issue Tracking Environment. In *Proceedings of the XIV Brazilian Symposium on Information Systems* (Caxias do Sul, Brazil) *(SBSI'18)*. Association for Computing Machinery, New York, NY, USA, Article 50, 8 pages. https://doi.org/10.1145/3229345.3229398

[14] Marco Ortu, Tracy Hall, Michele Marchesi, Roberto Tonelli, David Bowes, and Giuseppe Destefanis. 2018. Mining Communication Patterns in Software Development: A GitHub Analysis. In *Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering* (Oulu, Finland) *(PROMISE'18)*. Association for Computing Machinery, New York, NY, USA, 70–79. https://doi.org/10.1145/3273934.3273943

[15] Sandra L. Ramírez-Mora, Hanna Oktaba, and Helena Gómez-Adorno. 2020. Descriptions of issues and comments for predicting issue success in software projects. *Journal of Systems and Software* 168 (oct 2020), 110663. https://doi.org/10.1016/j.jss.2020.110663

[16] Eliseo Reategui, Miriam Klemann, Daniel Epstein, and A Lorenzatti. 2011. Sobek: A text mining tool for educational applications. In *Proceedings of the International Conference on Data Science (ICDATA)*. The Steering Committee of The World Congress in Computer Science, Computer …, 1.

[17] Samaneh Saadat, Olivia B. Newton, Gita Sukthankar, and Stephen M. Fiore. 2020. Analyzing the Productivity of GitHub Teams based on Formation Phase Activity. In *2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE, Melbourne, Australia, 169–176. https://doi.org/10.1109/WIIAT50758.2020.00027

[18] Ah-Hwee Tan, Heng Mui, and Keng Terrace. 2000. Text Mining: The state of the art and the challenges.

[12] https://github.com/BibliotecaRIT/BibliotecaRIT