UNIVERSIDADE DE PASSO FUNDO

Programa de Pós-Graduação em Computação Aplicada

Dissertação de Mestrado

UM SISTEMA DE RECOMENDAÇÃO PARA USUÁRIOS DAS PLATAFORMAS DE CROWDSOURCING

TIAGO MORAES FERREIRA

UPF

UNIVERSIDADE DE PASSO FUNDO
INSTITUTO DE CIÊNCIAS EXATAS E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO APLICADA

UM SISTEMA DE RECOMENDAÇÃO PARA USUÁRIOS DAS PLATAFORMAS DE CROWDSOURCING

Tiago Moraes Ferreira

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Computação Aplicada na Universidade de Passo Fundo.

Orientador: Prof. Dr. Cristiano Roberto Cervi
Coorientador: Prof. Dr. Alexandre Lazaretti Zanatta

Passo Fundo
2020

CIP – Catalogação na Publicação

F383s Ferreira, Tiago Moraes
Um sistema de recomendação para usuários das plataformas de *crowdsourcing* [recurso eletrônico] / Tiago Moraes Ferreira. – 2020.
1.8 MB ; PDF.

Orientador: Prof. Dr. Cristiano Roberto Cervi.
Coorientador: Prof. Dr. Alexandre Lazaretti Zanatta.
Dissertação (Mestrado em Computação Aplicada) – Universidade de Passo Fundo, 2020.

1. Software – Desenvolvimento. 2. Sistemas de recomendação 3. Crowdsourcing. I. Cervi, Cristiano Roberto, orientador. II. Zanatta, Alexandre Lazaretti, coorientador. III. Título.

CDU: 004.41

Catalogação: Bibliotecária Juliana Langaro Silveira – CRB 10/2427

UPF

PPGCA
Programa de Pós-Graduação
em Computação Aplicada
Instituto de Ciências Exatas e Geociências | ICEG

ATA DE DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO DO ACADÊMICO

TIAGO MORAES FERREIRA

Aos vinte e quatro dias do mês de junho do ano de dois mil e vinte, às quatorze horas, realizou-se, no Instituto de Ciências Exatas e Geociências, prédio B5, da Universidade de Passo Fundo (UPF), a sessão pública de defesa do Trabalho de Conclusão de Curso "Um Sistema de Recomendação para Usuários das Plataformas de Crowdsourcing", de autoria de Tiago Moraes Ferreira, acadêmico do Curso de Mestrado em Computação Aplicada do Programa de Pós-Graduação em Computação Aplicada – PPGCA. Segundo as informações prestadas pelo Conselho de Pós-Graduação e constantes nos arquivos da Secretaria do PPGCA, o aluno preencheu os requisitos necessários para submeter seu trabalho à avaliação. A banca examinadora foi composta pelos doutores Cristiano Roberto Cervi, Alexandre Lazaretti Zanatta, Ana Carolina Bertoletti De Marchi e Igor Steinmacher. Concluídos os trabalhos de apresentação e arguição, a banca examinadora considerou o candidato **APROVADO**. Foi concedido o prazo de até quarenta e cinco (45) dias, conforme Regimento do PPGCA, para o acadêmico apresentar ao Conselho de Pós-Graduação o trabalho em sua redação definitiva, a fim de que sejam feitos os encaminhamentos necessários à emissão do Diploma de Mestre em Computação Aplicada. Para constar, foi lavrada a presente ata, que vai assinada pelos membros da banca examinadora e pela Coordenação do PPGCA.

Prof. Dr. Cristiano Roberto Cervi – UPF
Presidente da Banca Examinadora
(Orientador)

Prof. Dr. Alexandre Lazaretti Zanatta – UPF
(Coorientador)

Prof. Dr. Ana Carolina Bertoletti De Marchi – UPF
(Avaliador Interno)

Prof. Dr. Igor Steinmacher – Northern Arizona University
(Avaliador Externo)

Prof. Dr. Rafael Rieder
Coordenador do PPGCA

UM SISTEMA DE RECOMENDAÇÃO PARA USUÁRIOS DAS PLATAFORMAS DE CROWDSOURCING

RESUMO

O sucesso na utilização do desenvolvimento de software em plataformas de crowdsourcing depende de um grande número de desenvolvedores estar engajado que registrem e submetam tarefas de forma recorrente. Um dos principais desafios enfrentados pelos usuários neste tipo de plataforma é a dificuldade na seleção de tarefas devido ao grande número disponível simultaneamente, bem como encontrá-las de acordo com seu perfil. Este trabalho propõe um sistema de recomendação com objetivo de recomendar tarefas em tempo real de acordo com o histórico de participação do usuário. Para isto foi desenvolvido um modelo de análise de similaridade entre tarefas (SIM-Crowd), um modelo de avaliação do histórico do usuário (UHR-Crowd) e um algoritmo responsável por agrupar os modelos e gerar as recomendações (RA-Crowd). Para avaliação do sistema de recomendação foram realizados experimentos com objetivo de medir a precisão e cobertura, bem como as métricas propostas pelos modelos e melhorar as configurações iniciais do sistema. Conclui-se que os objetivos foram atendidos uma vez que foi possível gerar recomendações com uma boa taxa de precisão e cobertura utilizando como meio os modelos propostos. Apesar dos experimentos serem realizados com dados históricos extraídos da plataforma, estima-se que pelo detalhamento do processo, é demonstrado a possibilidade de gerar recomendações em tempo real.

Palavras-chave: Busca de Tarefas, Sistemas de Recomendação, Software Crowdsourcing.

A RECOMMENDATION SYSTEM FOR CROWDSOURCING PLATFORM USERS

ABSTRACT

The success in software development on crowdsourcing platforms depends on many developers who are involved in registering and submitting tasks every time. One of the main challenges faced by users on this type of platform is the difficulty in choose tasks, due to the large number available and finding tasks according to their profile. This work presents a recommendation system with the objective of recommending tasks in real-time based on the user's last activities in these platforms that use TopCoder as a study base. For that, a task similarity analysis model (SIM-Crowd), a user history evaluation model (UHR-Crowd), and an algorithm responsible for grouping the models and generating the analyzes (RA-Crowd) were developed. To evaluate the recommendation system, experiments were carried out to measure the accuracy and coverage and the metrics propose by the models to improve the initial configuration of the system. The objectives were achieved because it is possible to generate recommendations with a good precision and coverage rate using the proposed models as a means. Although the experiments are carried out with historical data extracted from the platform, it is estimated that by detailing the process, the possibility of generating recommendations in real time is demonstrated.

Keywords: Task Search, Recommendation Systems, Crowdsourcing Software

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

SQ – Submission Qualitty

SR – Submission Rate

EC – Effort Concentration

FC – Filtragem Colaborativa

FBC – Filtragem Baseada em Conteúdo

FH – Filtragem Híbrida

FH – Filtragem Híbrida

CSD - Desenvolvimento de Software *Crowdsourced*

AMT - Amazon Mechanical Turk

SUMÁRIO

1. INTRODUÇÃO

A internet permitiu um ambiente propício para a colaboração, onde as organizações puderam ser beneficiadas com as informações que eram compartilhadas na rede. O aumento de mão de obra qualificada, redução de vida dos produtos e de um mercado de trabalho mais volátil, fez com que a informação se tornasse um bem valioso, acarretando na criação e evolução de inteligência colaborativa, como o *crowdsourcing* [1]. Nos últimos anos, um número crescente de empresas que trabalham com desenvolvimento de software voltaram-se a empregar um ecossistema descentralizado para este desenvolvimento semelhante a comunidades de código aberto com o objetivo de aumentar sua produção de software [2].

Crowdsourcing é um modelo emergente de solução de problemas distribuídos, baseado na combinação de humanos e computadores [3]. O termo foi publicado pela primeira vez pelo jornalista Jeff Howe [4] em seu artigo *"Rise of crowdsourcing"*, na revista *Wired* no ano de 2006, expondo um novo modelo de comportamento social de coletividade, no qual pessoas de diversas áreas e níveis de experiência se reúnem para executar tarefas, que são levadas à multidão por meio de chamadas abertas. Como um paradigma emergente, o desenvolvimento de software *crowdsourced* (CSD), que deriva seu conceito de *crowdsourcing*, utiliza também um formato de chamada aberta para atrair desenvolvedores para realização de tarefas de desenvolvimento de software, tais como arquitetura, desenvolvimento de componentes, testes entre outros [5] [6].

O sucesso no desenvolvimento de software *crowdsourced* (CSD), depende muito de uma grande multidão de trabalhadores de software confiáveis que registrem-se e submetam tarefas de forma recorrente, geralmente em troca de ganhos financeiros [6]. Entre as plataformas de *crowdsourcing* que suportam o desenvolvimento de software podem ser citadas TopCoder[1], uTest[2], AMT[3], Freelancer[4], entre outras. O TopCoder destaca-se sendo o maior portal de competição

[1] TopCoder Website: http://www.topcoder.com/
[2] uTest Website: http://www.utest.com/
[3] Amazon Mechanical Turk Website: www.mturk.com/
[4] Freelancer Website: http://www.freelancer.com/

no desenvolvimento de software do mundo, entre seus clientes, podem ser citados Google, Microsoft, Facebook e AOL [7].

1.1. MOTIVAÇÃO

Projetado para permitir ampla acessibilidade e autoseleção de tarefas, a maioria das plataformas CSD permitem que o desenvolvedor selecione livremente as tarefas de acordo com suas habilidades pessoais e seus interesses [5], porém segundo Yang *et al* [6] isto acarreta em alguns desafios: a demora na seleção manual de tarefa em virtude do grande número de tarefas disponíveis simultaneamente; o solicitante de tarefas geralmente não ter visibilidade e controle sobre os desenvolvedores que estão trabalhando nas tarefas por ele disponibilizada. Estes fatores influenciam na identificação da melhor correspondência usuário-tarefa.

Quanto à forma como são disponibilizadas as tarefas em CSD, e os problemas causados por este formato, Mão *et al* [5] avaliam sob duas perspectivas:

1) **Perspectiva do trabalhador:** considerando que há muitas tarefas disponibilizadas simultaneamente em plataformas como o TopCoder, é trabalhoso para um desenvolvedor escolher uma tarefa mais adequada ao seu perfil;

2) **Perspectiva da plataforma:** desafio de buscar os melhores desenvolvedores disponíveis para cada tarefa, sendo este um fator chave para uma entrega com qualidade.

Yuen, King e Leung [8] acrescentam não fazer sentido que a quantidade de tempo gasto na seleção de uma tarefa seja igual ou até mesmo superior ao tempo de desenvolvimento desta tarefa em determinadas situações. Em seu trabalho, Yuen, King e Leung [8] citam o caso do escritor *Rachael King* que ganhou USD $ 4,38 por oito horas de trabalho utilizando a plataforma AMT. Nesta experiência o escritor relata que passou a maior parte do tempo procurando uma tarefa coerente com seu perfil, sendo que a recompensa não correspondeu ao esforço utilizado.

Chilton *et al* [9] relatam que a maioria dos usuários geralmente visualizam apenas algumas tarefas publicadas recentemente em plataformas de *crowdsourcing* como a AMT. No de seu estudo, Chilton *et al* [9] demonstram que uma tarefa com posicionamento favorável nos resultados da pesquisa foi concluída 30 vezes mais rápido e por uma recompensa menor do que quando sua posição era desfavorável. Ainda segundo, Chilton *et al* [9] a recomendação de tarefas ajuda os solicitantes a

coletar os resultados concluídos das tarefas em um período de tempo mais curto sem manipular a posição de suas tarefas, e as tarefas podem ser apresentadas aos usuários que realmente tenham interesse.

Embora plataformas de CSD como o TopCoder geralmente permitam a inscrição para alertas sobre novas tarefas [5], elas não fornecem informações sobre recomendação personalizada para tais alertas. Para Yuen, King e Leung [10], a recomendação de tarefas pode ajudar a trabalhadores encontrarem uma tarefa “mais facilmente” e de uma maneira “mais rápida”, incentivando a colaboração e participação nas plataformas de *crowdsourcing*. Yang *et al* [6] também afirmam que é fundamental o apoio a tomada decisão do trabalhador quanto a seleção de tarefas nestas plataformas como forma de aumentar o engajamento e participação.

Entende-se que um sistema de recomendação que possa filtrar previamente as tarefas de acordo com o perfil do usuário e seu histórico de participação, poderia ser utilizado como uma ferramenta de apoio a tomada de decisão dos participantes de plataformas de *crowdsourcing* como o TopCoder, contribuindo assim com o principal objetivo deste tipo de plataforma que é a utilização da inteligência coletiva como forma de colaboração.

Diante do exposto, chegou-se ao seguinte problema de pesquisa: como desenvolver um sistema para recomendar tarefas de acordo com o perfil do trabalhador em uma plataforma de crowdsourcing?

1.2. OBJETIVOS

Tendo em vista a dificuldade existente na seleção de tarefas em plataformas de *crowdsourcing,* este trabalho têm por objetivo apresentar uma abordagem de sistema de recomendação como forma de apoio a decisão para seleção de tarefas no desenvolvimento de software nas plataformas de crowdsourcing, utilizando o TopCoder como referência. Para isto foram elencados alguns objetivos específicos:

- *Identificar atributos relevantes*: identificação dos atributos mais relevantes associados as tarefas e histórico de participação dos usuários na plataforma TopCoder;
- *Criar modelo de similaridade*: criação de um modelo para identificar a medida de similaridade entre tarefas;

- *Criar modelo de avaliação do usuário:* gerar um modelo de avaliação quanto a participação dos usuários neste tipo de plataforma;
- *Definir mecanismo recomendação:* definição de um mecanismo de recomendação que utilize o modelo de similaridade de tarefas e o modelo de avaliação do histórico de participação do usuário;
- *Realizar experimentos:* Realização de experimentos utilizando o sistema de recomendação desenvolvido;
- *Avaliar o sistema de recomendação:* Avaliação dos experimentos com o objetivo de identificar se as recomendações são abrangentes e consistentes.

1.3. OUTRAS CONTRIBUIÇÕES

1.3.1. Artigo Publicado

Durante o mestrado, foi publicado um artigo no Simpósio Brasileiro de Sistemas de Informação, evento anual e itinerante organizado pela SBC (Sociedade Brasileira de Computação), responsável por congregar grande parte da comunidade nacional com interesse nesta área[5]. O artigo com o título: *CrowdRec: Desenvolvimento de um protótipo de sistema de recomendação para plataformas de crowdsourcing utilizando Google Venture Design Sprint* propõe a prototipação de um aplicativo de recomendação de tarefas, com design orientado a experiência do usuário (UX). Como metodologia utilizou-se o *Google Venture Design Studio* e a ferramenta *Quant-UX* para realizar a análise dos resultados.

Após a construção do protótipo foi realizada uma pesquisa com um grupo de usuários, onde utilizando a metodologia TAM (adaptação do modelo da Teoria da Ação Racional (TAR) [11]), foi possível obter algumas variáveis como facilidade de uso, utilidade do uso e variáveis externas como sugestão de melhorias ao protótipo proposto. Por fim, concluiu-se que o protótipo apresentou bons resultados, sendo que a maior parte dos usuários considerou a ferramenta de fácil utilização. Também foram identificadas a necessidade de adequação em algumas telas por serem pouco intuitivas dado os objetivos sugeridos na mesma.

[5]https://www.sbc.org.br/2-uncategorised/1817-sbsi-simposio-brasileiro-de-sistemas-de-informacao

1.3.2. Participação em co-orientação

Como aluno de Mestrado do Programa de Pós-Graduação em Computação Aplicada da Universidade de Passo Fundo, foi dado auxílio na co-orientação do aluno Regis Sganzerla, com trabalho de conclusão do curso de Ciência da Computação na área de software crowdsourcing, com título indefinido.

1.4. ORGANIZAÇÃO DO TEXTO

Com o objetivo de descrever a abordagem de recomendação proposta e os experimentos realizados, o presente trabalho encontra-se organizado da seguinte forma:

- Capitulo 2 apresenta os principais conceitos que fundamentam esta dissertação, iniciando com a contextualização sobre crowdsourcing e na sequência sobre sistemas de recomendação;
- Capitulo 3 são apresentados os trabalhos relacionados com o tema que foram encontrados na literatura;
- Capitulo 4 é detalhada a abordagem proposta, os modelos utilizados bem como o algoritmo de recomendação utilizado nesta abordagem;
- Capitulo 5 descreve os experimentos realizados, os resultados obtidos e a discussão sobre os mesmos;
- Capitulo 6 conclui o trabalho com um resumo geral das contribuições e a sugestão para trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem por objetivo apresentar a fundamentação teórica utilizada para o desenvolvimento do presente trabalho.

2.1. CROWDSOURCING

O termo *crowdsourcing* refere-se a um processo de conectar um grande grupo de pessoas, com o objetivo de contribuir com um propósito específico tendo como vantagem a acessibilidade econômica e de infraestrutura [12]. O termo foi publicado pela primeira vez pelo jornalista Jeff Howe [4] e expõe um novo modelo de comportamento social e coletividade, no qual pessoas de diversas áreas e níveis de experiência se reúnem para executar tarefas, que são levadas à multidão por meio de chamadas abertas. Em seu livro *Crowdsourcing: o poder das multidões*, Howe [13] evidencia que o termo refere-se à criação, execução, inovação e solução, que podem vir de qualquer lugar, de forma voluntária e gratuita até obtenção de algum benefício.

Segundo Brabham [14], comunidades online são fontes férteis de criatividade e de talentos, com isso pesquisas acadêmicas cresceram para verificar o motivo do crescimento do *crowdsourcing*. As empresas assumiram algumas funções que antes eram desempenhadas por funcionários realizando a distribuição do trabalho para outros trabalhadores em uma chamada aberta para a comunidade online [14].

Brabham [14] relata ainda que em 2008 ocorreram as primeiras tentativas acadêmicas para definir o *crowdsourcing*. Essas definições eram conflitantes e explicavam o *crowdsourcing* de acordo com os participantes e as razões para suas participações, conforme as ferramentas utilizadas, as características organizacionais, o grau de complexidade ou o nível de participação do usuário. Essas tentativas levaram a várias definições e interpretações para o *crowdsourcing*. Para Li e Hongjuan [15], *crowdsourcing* é um ato de terceirizar tarefas que tradicionalmente eram realizadas por um empregado ou contratado e disponibilizá-las a um grande e indefinido grupo de pessoas ou à comunidade por meio de uma chamada aberta. As tarefas são divulgadas ao público desconhecido pela comunidade online e o público utiliza uma plataforma para envio das soluções. A solução final é obtida por quem propôs a tarefa e o ganhador recebe incentivos em dinheiro [15].

Li e Hongjuan [15] descrevem em linhas gerais o processo do *crowdsourcing* como:

- O *crowdsourcer* (empresas e organizações) propõem a tarefa e a tornam disponível pela internet ou outros meios de comunicação e ao mesmo tempo apresentam de forma clara os requisitos da solução e a recompensa para os vencedores;
- O público apresenta soluções potenciais;
- O *crowdsoucer* determina a melhor solução e recompensa o vencedor da tarefa;
- *Crowdsoucer* possui a melhor solução.

Diante deste novo modelo de distribuição de solução de problemas, fez-se necessário a criação de um elo entre o *crowdsoucer* e a multidão. Ou seja, a criação de plataformas para auxílio no gerenciamento, não apenas de pequenas tarefas, mas projetos inteiros, partindo desde sua documentação, até sua entrega final. Desta forma, possibilitou ao *crowdsoucer* encontrar talentos fora de seus limites físicos, além de vantagens como: redução nos custos, visibilidade de sua marca/empresa, soluções diversificadas, melhor qualidade, acesso à criatividade e, como esperado, a resolução do seu problema [16].

Algumas plataformas especializam-se apenas em determinada fase do processo, seja na coleta dos requisitos, no design, no desenvolvimento ou na validação do escopo de uma demanda. Por exemplo, a BugFinders[6] é uma plataforma de *crowdtesting*, onde os usuários participam apenas para encontrar defeitos em sistemas, em sua maioria e-commerce, sendo remunerados conforme gravidade do defeito reportado. Os participantes necessitam seguir padrões para reportar os defeitos encontrados e, principalmente, saber o que faz parte do escopo.

Segundo Zanatta [12], o crowdsourcing é uma maneira diferenciada de como a multidão oferta serviços, ideias e conteúdo, sendo assim, o desenvolvimento de software pode ser considerado com uma possibilidade no uso deste modelo.

2.1.1. Software Crowdsourcing

[6] https://www.bugfinders.com

Movimentos de softwares *open source* demonstraram que a produção de software pode ser realizada fora da empresa, sendo possível aproveitar talentos espalhados de forma global. O sucesso atraiu interessados no modelo de *crowdsourcing*, no qual empreendedores podem utilizar o poder da multidão em torno de uma meta de produção. Essa forma de trabalho demonstrou-se eficiente e um número crescente de empresas a têm a utilizado [17]. Alguns exemplos deste tipo de iniciativa são o Sistema Operacional Linux, o Apache[7], o Rails[8] e o Firefox[9] que foram construídos por um número diverso de pessoas espalhadas geograficamente de forma colaborativa e sem recompensa financeira [12].

Alguns dos objetivos e motivações apontados por organizações que utilizam o software crowdsourcing são: qualidade do software, aquisição rápida, identificação de talentos, redução de custos, soluções diversificadas, criação de ideias, ampliar a participação, educação dos participantes e marketing [18] [19].

Segundo Zanatta [12], o projeto de software crowdsourcing deve ser executado a partir de quatro elementos fundamentais: o cliente, a plataforma, a multidão e a tarefa. Desta forma, para que a plataforma execute seu papel no processo é fundamental que possua processos adaptados ou específicos para o desenvolvimento de software. Diversas plataformas buscaram permitir esse modelo de crowdsourcing aplicado ao desenvolvimento de software, sendo uma delas o TopCoder.

Para Shenck e Guitard [20], o TopCoder é uma das principais plataformas para projetos de software no mundo, reunindo profissionais e amadores que realizam tarefas comerciais e que competem entre si visando uma compensação financeira. Yang *et al* [6] também destaca a importância do TopCoder elencando como "a plataforma de software mais popular em software crowdsourcing". Segundo [21], o TopCoder é uma plataforma que abrange todo o processo de desenvolvimento de software como a obtenção de requisitos, desenvolvimento, teste e evolução.

O processo de desenvolvimento de software no TopCoder, de uma forma resumida, funciona da seguinte maneira: o cliente (*requester*) anuncia um problema que precisa ser solucionado pela multidão (*crowd*); esse problema é fragmentado em problemas menores e bem definidos, conhecidos como tarefas (*tasks*), desta forma

[7] https://www.apache.org/
[8] http://rubyonrails.org/
[9] https://www.mozila.org/pt-BR/firefox/new/

qualquer programador cadastrado na plataforma pode se registrar para tentar desenvolver uma solução, gerando uma competitividade entre os programadores participantes; entre todas as soluções enviadas, é escolhida a que atender melhor os requisitos do projeto e for melhor avaliada quanto sua qualidade; o conjunto e tarefas solucionadas formam o projeto completo, que é certificado pelo TopCoder e entregue ao cliente [13].

Uma ferramenta que pode auxiliar e incentivar a utilização de plataformas de crowdsourcing bem como o desenvolvimento de software utilizando estas plataformas são os Sistemas de Recomendação, que será conceituado na próxima seção.

2.2. SISTEMAS DE RECOMENDAÇÃO

Devido à grande quantidade e diversidade de informações disponíveis na internet, torna-se difícil escolher a melhor informação a ser consumida. Os processos de busca e recuperação da informação acabam sendo complicados e nem sempre a pessoa está preparada ou tem a experiência necessária para elaborar as estratégias de busca com assertividade [22]. Uma das ferramentas utilizadas para amenizar esse tipo de problema são os sistemas de recomendação, os quais facilitam o encontro de dados durante os processos de busca e navegação, oferecendo aos usuários indicações daqueles produtos e serviços que poderiam estar associados ou relacionados às suas necessidades [23].

Segundo Rassweiler [24], uma definição para sistema de recomendação seria "*um software utilizado em contextos onde existem usuários e itens, sendo que o objetivo deste software é estimar quais são os itens que cada usuário têm a maior chance de vir a consumir*". Para Ricci [25], os sistemas de recomendação podem auxiliar os usuários no processo de tomada de decisão em relação a diversas ações, como comprar itens, assistir filmes, ouvir música, ler notícias, entre outras.

Um dos primeiros aplicativos experimentais na área foi proposto por Goldberg [26], o *Tapestry*, criado por pesquisadores da empresa *Xerox* no *Palo Alto Research Center.* Um sistema experimental de envio de e-mails que permitia a personalização dos e-mails recebidos pelos usuários por meio da aplicação de filtros baseados não apenas no conteúdo dos documentos, mas também envolvendo uma avaliação dos leitores a respeito destes documentos. Resnick e Varian [13]

posteriormente foram responsáveis pela introdução do termo sistema de recomendação.

Os sistemas de recomendação são muito úteis em situações quando o usuário não possui conhecimento suficiente para tomar uma decisão sobre um domínio específico ou quando ele não é capaz de avaliar todas as opções disponíveis, geralmente devido ao grande volume de opções. Um princípio básico destes sistemas é usar informações disponíveis do perfil do usuário, dos itens e as interações entre eles com objetivos de fazer previsões sobre quais novos itens podem interessar este usuário.

Os sites de comércio eletrônico são um exemplo da utilização de sistemas de recomendação tendo como objetivo aumentar a quantidade de vendas, usando a personalização de conteúdo e a indicação de itens com uma maior taxa de aceitação dos usuários. Introduzido em julho de 1996 o *My Yahoo* foi o primeiro website a utilizar a personalização em grandes proporções, utilizando a estratégia de customização onde o usuário indicava explicitamente vários critérios de preferência de exibição de sua página [27]. Atualmente vários domínios utilizam este tipo de técnica com o intuito de melhorar a experiência do usuário, como *streaming* de vídeos do Netflix, Youtube e Vimeo, *streaming* de música como Spotify, Deezer e Pandora, redes sociais como Twitter, Facebook e Instagram, entre outros [24].

As estratégias de personalização mais utilizadas trazem ao usuário ofertas combinadas ("clientes que compraram item X também compraram item Y"), itens de sua preferência, itens mais vendidos em suas categorias favoritas, entre outras. Para possibilitar quaisquer destas formas de personalização, é necessário coletar e armazenar informações sobre os usuários.

Para atender à necessidade dos usuários, um sistema de recomendação geralmente possui [28]:

- **Dados prévios ou armazenados**: corresponde a toda informação que o sistema armazena para utilizar no processo de recomendação;
- **Dados de entrada ou do usuário**: tratam-se de informações fornecidas pelo usuário para que o processo de recomendação possa ser iniciado;
- **Algoritmo de recomendação**: é responsável por combinar dados prévios e dados de entrada para que sejam fornecidas as devidas recomendações.

A personalização de navegação do ponto de vista computacional consiste em um conjunto de funções, como por exemplo monitorar toda a interação do usuário

enquanto o mesmo realiza a navegação, e, com base nas informações coletadas, manter um modelo do usuário contendo informações como dados demográficos, itens visualizados, interesses, preferências, entre outros. Apresentar links ou fragmentos dentro de uma página, assegurando que seu conteúdo inclua a informação apropriada para aquele usuário conforme o modelo proposto acima.

Um dos grandes desafios dos sistemas de recomendação é realizar a combinação correta entre os que estão recomendando e aqueles que estão recebendo a recomendação, ou seja, definir e descobrir este relacionamento de interesses [29].

No contexto de sistemas de recomendação, pode-se destacar duas classificações quanto a sua aplicação:

- **Recomendações Não Personalizadas:** Técnica que não considera as particularidades do usuário. Recomendações sobre itens mais vendidos, itens mais pesquisados, ou com melhor avaliação geral são alguns exemplos deste tipo de recomendação;
- **Recomendações Personalizadas:** Nesta técnica é levado em consideração o perfil do usuário, desta forma geram resultados que em geral são melhor aceitos pelos mesmos. Gostos pessoais, preferências, avaliações, entre outros aspectos devem ser levados em consideração.

Segundo Adomavicius [30], é possível classificar os sistemas de recomendação em três grupos definidos a partir do método de filtragem utilizado, sendo filtragem baseada em conteúdo, filtragem colaborativa e filtragem híbrida. O método de filtragem pode ser resumido como um processo que separa os objetos aderentes ao perfil e contexto do usuário. Bobadilla *et al* [31] introduz outros dois métodos encontrado na literatura, sendo a filtragem demográfica e a filtragem social como uma possível tendência na área. Nesta seção são apresentadas as técnicas mais comuns encontradas na literatura citadas por Adomacicius [30] e Ricci *et al* [32]: filtragem baseada em conteúdo, filtragem colaborativa e filtragem híbrida.

2.2.1. Filtragem Baseada em Conteúdo

Segundo Herlocker [33], por muitos anos os cientistas têm direcionado seus esforços para aliviar o problema ocasionado com a sobrecarga de informações propondo projetos que integram tecnologias que automaticamente reconhecem e

categorizam as informações. Esta técnica é chamada de Filtragem Baseada em Conteúdo - FBC [25] por realizar uma seleção baseada na análise de conteúdo dos itens e no perfil do usuário.

A abordagem baseada em conteúdo tem suas raízes na área de recuperação de informação sendo que muitos sistemas baseados em filtragem de conteúdo focam na recomendação de itens com informações textuais, como documentos e websites [34]. As melhorias sobre os sistemas tradicionais de recuperação de informação vieram com a utilização do perfil do usuário, que contém suas preferências e necessidades.

Desta forma, a FBC parte do princípio de que os usuários tendem a interessar-se por itens similares aos que demonstraram interesse no passado, definindo então, a similaridade entre os itens [33]. O processo de recomendação neste modelo, consiste em contrastar o perfil do usuário com o perfil dos itens e recomendar a este usuários itens com perfis mais similares aos dele ou ainda contrastar o perfil dos itens que o usuário demonstrou preferência no passado com cada item que deseja-se avaliar a preferência do usuário [24].

Uma etapa importante na concepção de um modelo de sistema de recomendação utilizando a técnica de filtragem baseada em conteúdo é a definição de quais atributos serão utilizadas para construir os perfis de itens e usuários. A partir das preferências dos usuários por cada item, pode-se determinar diferentes pesos para os atributos para em seguida realizar a análise de similaridade sobre os mesmos [24].

Quando as fontes de atributos são textuais, uma técnica muito comum neste tipo de abordagem é a indexação de frequência de termos (*term frequency indexing*), usando o algoritmo *TF-IDF* (*Term Frequency-Inverse Document Frequency*), onde o objetivo é aumentar o peso do termo que ocorre frequentemente em um documento (*TF*) e que ocorre raramente no restante dos documentos (*TF-IDF*). Antes de calcular a similaridade entre textos e o perfil do usuário ou dos itens, deve observar-se o pré-processamento de textos, onde ocorrem atividades tais como remoção de *stopwords*, análise léxica e *stemming*, que evitam palavras que não são relevantes sejam consideradas palavras chaves ou comparadas às disponibilizadas pelo usuário. Cazella [35] detalha estas atividades:

- **Stopwords:** Remoção de termos que não são relevantes, como artigos, preposições, conjunções e pronomes, visando eliminar estas palavras dos

textos. São eliminados, por exemplo: “mas”, “mesmo”, “qualquer”, “seja”, “ainda”;

- **Análise Léxica:** Dígitos, sinais de pontuação, acentos e hífens podem ser removidos neste processo, e todas as letras do texto podem ser transformadas para maiúsculas ou minúsculas;
- **Stemming:** É realizado considerando cada palavra isoladamente, tentando reduzi-la a sua provável raiz. São analisadas características como grau, gênero e número com o objetivo eliminar os sufixos e prefixos das palavras transformando-as em sua forma primitiva. Por exemplo, informação e informática tem, como raiz, a palavra “informa”.

Para Burke [28], as principais vantagens na utilização da filtragem baseada em conteúdo são:

- **Independência do Usuário:** as recomendações podem ser construídas sem depender da similaridade entre os usuários existentes no sistema;
- **Transparência:** a descrição dos atributos que compõem o perfil dos itens são responsáveis por originar as recomendações, desta forma o processo de recomendação é simplificado e transparente;
- **Novos Itens:** a filtragem baseada em conteúdo é capaz de recomendar itens novos, mesmo sem nenhuma avaliação do usuário, baseado na similaridade entres os itens.

Em contrapartida, esta abordagem também possui algumas limitações, entre elas destacam-se a super especialização (*overspecialization*) e a partida fria (*cold-start*) [30].

A super especialização é identificada quando um sistema de recomendação não consegue sugerir novos itens ao usuário, isto ocorre porque ao sistemas buscar itens com maior similaridade ao perfil e contexto do usuário, este método pode ignorar itens que, de certa forma, também são aderentes ao usuário, porém não possuem altos índices de similaridade. Nestes casos a filtragem baseada em conteúdo tende a não gerar recomendações com serendipidade, ou seja, que surpreendam o usuário [24].

Outra limitação desta abordagem, é a “partida fria”, sendo objeto de amplo estudo em sistemas de recomendação e de forma resumida está relacionado ao que recomendar quando não existem informações sobre histórico do usuário. Como a

filtragem baseada em conteúdo, utiliza as escolhas passadas do usuário esta técnica é altamente dependente destas informações históricas, logo sem estas informações o método não consegue identificar recomendações iniciais.

Segundo Rassweiler [24], uma maneira de mitigar as limitações encontradas na filtragem baseada em conteúdo é o uso de ontologias (limitação de análise de conteúdo) ou outros tipos de análise semântica. Quanto a super especialização pode ser resolvido inserindo recomendações randômicas com auxílio de algoritmos genéticos, ou ainda, utilizando uma abordagem híbrida [36].

No presente trabalho, utilizou-se a filtragem baseado em conteúdo como técnica de recomendação, uma vez que foi analisado o histórico do usuário para recomendar tarefas que possui uma medida de similaridade satisfatória comparado a novas tarefas disponibilizadas.

2.2.2. Filtragem Colaborativa

A filtragem colaborativa (FC) é uma técnica que consiste em utilizar informações sobre o comportamento dos usuários, como itens comprados e avaliações de itens, de maneira a calcular quais produtos os usuários têm maior probabilidade de estarem interessados [25]. Segundo Barth [37], esta técnica deve ser utilizada quando pode-se assumir que um usuário se comporta de forma semelhante a outros, ou quando itens possuem a mesma demanda.

Na Figura 1 pode-se observar o comportamento onde o usuário A e o usuário B tem perfis similares, por terem acessado os mesmos itens (A e B), logo o item C é recomendado para o usuário A com base na similaridade dos dois perfis.

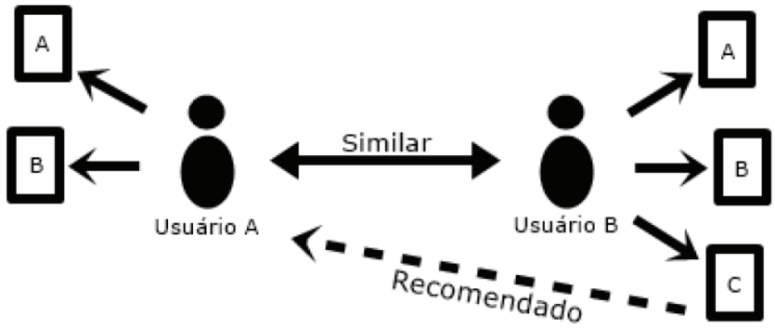

Figura 1. Exemplo de sistema de recomendação com filtragem colaborativa [38].

A abordagem da filtragem colaborativa foi desenvolvida para suprir as desvantagens apontadas na filtragem baseada em conteúdo [33], desta forma esta abordagem diferencia-se da filtragem baseada em conteúdo exatamente por não exigir a compreensão ou reconhecimento do conteúdo dos itens.

O termo filtragem colaborativa (FC) foi usado pela primeira vez por David Goldberg [26] em um artigo intitulado *Using Collaborative Filtering to Weave na Information Tapestry*, Goldberg desenvolveu um sistema chamado *Tapestry* que permitia que os usuários fizessem anotações em documentos, marcando-os como interessante ou desinteressante, a partir da análise dessas informações, o sistema conseguia filtrar estes documentos para seus usuários.

Segundo Herlocker [33] os primeiros sistemas de filtragem colaborativa requeriam usuários para especificar o relacionamento de predição entre suas opiniões, ou de modo explícito indicar os itens de seu interesse. Um usuário de um sistema deve, portanto, pontuar cada item experimentado, indicando o quanto este item combina com a sua necessidade. Estas pontuações são coletadas para grupos de pessoas, permitindo que cada usuário se beneficie das pontuações (experiências) apresentadas por outros usuários.

A técnica de filtragem colaborativa pode ser descrita em três fases [33], explicadas do ponto de vista das recomendações realizadas para um usuário, também chamado de usuário alvo (*target*):

1. Calcular o peso de cada usuário do sistema em relação à similaridade com o usuário alvo (métrica de similaridade);

2. Selecionar um subconjunto de usuários com maiores similaridades (vizinhos) para considerar na predição;

3. Normalizar as avaliações e computar as predições ponderando as avaliações dos vizinhos com seus pesos.

Em geral a filtragem colaborativa é implementada utilizando uma matriz, onde os itens são armazenados nas colunas e os usuários nas linhas. Esta matriz contêm as avaliações que os usuários informaram para cada item com o objetivo de estimar o quanto um usuário com comportamento similar pode também se interessar por estes itens. Na Figura 2 é apresentado um exemplo de uma matriz utilizada na filtragem colaborativa.

	Item(i_1)	Item(i_2)	Item(i_3)	Item(i_4)	Item(i_5)
Usuário(u_1)	1 ★	2★★	5★★★★★		
Usuário(u_2)	2★★		3★★★	4★★★★	
Usuário(u_3)		5★★★★★	2★★	1★	
Usuário(u_4)	2★★		2★★	3★★★	
Usuário(u_5)	3★★★	5★★★★★	2★★	2★★	

Figura 2. Matriz utilizada na filtragem colaborativa [39].

A partir da matriz de usuários e itens é necessário realizar o cálculo de similaridade com o objetivo de medir o coeficiente de correlação ou alguma medida de distância como: *Pearson*, Cosseno, *Jaccard* e *Spearman.* Ainda existe a possibilidade de utilizar outras técnicas para cálculo de similaridade como *Clustering*, *Redes Bayesianas*, aprendizado de máquina, mineração de dados, entre outras.

Uma das vantagens da técnica de FC é que possui um modelo conceitual de operação de "fácil entendimento", possibilitando analisar itens a serem recomendados sem preocupar-se com o conteúdo destes itens, focando nas avaliações dos mesmos [33]. A FC possui outra vantagem em relação a FBC, que é a possibilidade de apresentar aos usuários recomendações inesperadas. O usuário poderia receber recomendações de itens que não estavam sendo pesquisados de forma ativa [29].

A filtragem colaborativa também possui algumas limitações, dentre as quais pode-se citar [28][30]:

- **Partida Fria (*cold start*):** quando um usuário é novo no sistema o seu perfil não é similar a nenhum outro, isso acontece também com itens novos adicionados ao banco de dados, que não será recomendado até ser avaliado por algum usuário;
- **Pontuações Esparsas:** se o número de usuários for pequeno, existe a possibilidade de as pontuações de similaridade serem muito esparsas;
- **Preferências Atípicas (*outliers*):** caso as preferências do usuário sejam exóticas de acordo com o domínio, existirá dificuldade de encontrar outros usuários com preferências similares.

Com base nas limitações apresentadas pela filtragem baseada em conteúdo – FBC e da filtragem colaborativa, surge e filtragem híbrida com o objetivo de maximizar as vantagens apresentadas em cada abordagem e minimizar as desvantagens de cada uma.

2.2.3. Filtragem Híbrida

A filtragem híbrida busca por meio da combinação de duas ou mais técnicas, tirar proveito das vantagens de cada uma delas de modo a desenvolver um sistema que recomende o conteúdo mais adequado para o usuário.

Um exemplo comum é utilizar a filtragem colaborativa com a filtragem baseada em conteúdo. A combinação dessas duas técnicas possibilita que o sistema seja beneficiado pelos bons resultados para usuários incomuns, pela precisão da recomendação independentemente do número dos usuários (FBC), e pela descoberta de similaridades entre os usuários, além da recomendação relacionada com o histórico do usuário (FC)[29].

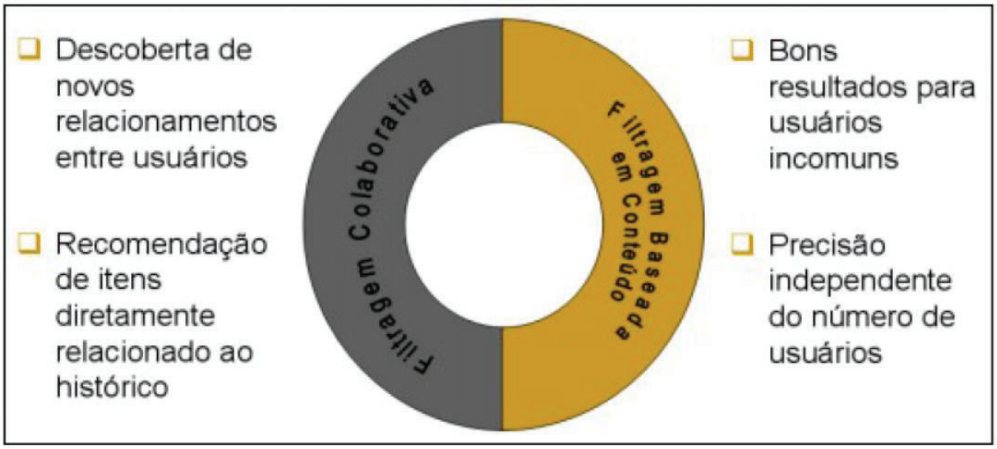

Figura 3. Filtragem Híbrida [29].

Na filtragem híbrida (FH), as principais estratégias utilizadas, no que diz respeito à forma como componentes serão combinados para gerar as recomendações segundo Barbosa [40], são:

- **Ponderada:** nesta abordagem a FBC e a FC são implementadas separadamente e uma combinação linear é feita com os resultados;
- **Mista:** nesta abordagem as recomendações geradas pelas duas técnicas são combinadas no processo final de recomendação, de tal forma que as duas recomendações sejam apresentadas ao usuário na mesma lista;
- **Combinação sequencial:** nesta abordagem a FBC cria os perfis dos usuários e, posteriormente, estes perfis são usados no cálculo da similaridade da FC;
- **Comutação:** nesta abordagem o sistema utiliza algum critério, como por exemplo a confiança no resultado, para comutar ou chavear entre a FBC e

a FC. Pode-se também realizar a comutação de uma técnica nos pontos de desvantagens da outra técnica.

A utilização da Filtragem Híbrida pode auxiliar em problemas comuns no contexto de sistemas de recomendação como o de recomendar um novo item, porém outros problemas ainda persistem como recomendar itens para um usuário novo sem histórico em seu perfil. Uma possível solução para o problema da partida fria é proposto no trabalho de Marques [41] com utilização de informações semânticas sobre os itens bem como a utilização destas informações como apoio no cálculo de similaridade com objetivo de recomendações adequadas ao perfil do usuário, incluindo itens novos do acervo que sejam relevantes.

2.2.4. Métricas de avaliação de sistemas de recomendação

Após gerar as recomendações, é importante que seja realizada a avaliação das mesmas. Esta avaliação auxilia na realização de ajustes com objetivo de reduzir possíveis erros e consequentemente melhorar a experiência do usuário no sistema de recomendação.

Segundo Gunawardana e Shani [42], um das métricas utilizadas para avaliação de predições em sistemas de recomendação são as métricas de conjunto que baseiam-se na teoria dos conjuntos, sendo elas: Precisão (*Precision*), Revocação (*Recall*), F-Measure e Cobertura (*Coverage*). Herlocker [33] também destaca estas métricas como mais populares na avaliação de sistemas de recuperação de informação, detalhando o seu conceito:

- **Precisão:** responsável por verificar a quantidade de itens recomendados que são do interesse do usuário em relação a conjunto de todos os itens que são recomendados;
- **Revocação:** indica a quantidade de itens de interesse do usuário que são apresentados na lista de itens relevantes;
- **F-Measure:** média harmônica entre a precisão e a revocação combinando-se os dois valores em um para simplificar possíveis comparações;
- **Cobertura:** calculada pela proporção de itens que são aptos a serem recomendados em relação ao conjunto de todos os itens conhecidos pelo sistema de recomendação.

O cálculo da precisão pode ser observado na Equação (1), definida por Huang et al. [43].

$$precision = \frac{|R_g \cap R_r|}{R_r} \quad (1)$$

O cálculo da revocação pode ser observado na Equação (2), definida por Huang et al. [43].

$$recall = \frac{|R_g \cap R_r|}{R_g} \quad (2)$$

Sendo,

- R_g: conjunto total relevante;
- R_r: conjunto de recomendações do sistema;
- $|R_g \cap R_r|$: conjunto de recomendações corretas ou relevantes.

Já o cálculo da F-measure é apresentado na Equação (3), também definida por Huang et al. [43].

$$F - measure = 2\left(\frac{precision.recall}{precision + recall}\right) \quad (3)$$

Para o cálculo da cobertura é utilizado a Equação (4), definida por Ge, Delgado-Battenfeld e Nannach [44]:

$$cobertura = \frac{|U_p|}{U} \quad (4)$$

Sendo,

- U_p: conjunto de todos os usuários em que o sistema pode gerar recomendações;
- U: conjunto de todos os usuários existentes no sistema;

Para uma correta avaliação do sistema de recomendação deve-se procurar um equilíbrio entre as métricas utilizadas, não avaliando uma única métrica de forma isolada, pois espera-se que além de sistema ser capaz de gerar recomendações com

precisão, o mesmo dever possuir uma cobertura abrangente, contemplando a maioria dos usuários e itens do sistema.

3. TRABALHOS RELACIONADOS

Neste capítulo serão discutidos alguns trabalhos relacionados, sendo apresentados pontos em comum e diferenciais com o trabalho proposto.

Mao *et al* [5] analisam o problema da correspondência correta entre tarefa-trabalhador no desenvolvimento de software em plataformas de crowdsourcing sob duas perspectivas. A primeira é a perspectiva do trabalhador, que encontra dificuldade em encontrar uma tarefa de acordo com seu perfil, visto que são inúmeras tarefas disponíveis simultaneamente. A segunda é da plataforma, no desafio de obter os melhores desenvolvedores disponíveis para determinado tipo de tarefa. Como alternativa para mitigar este problema, a sugestão de Mao *et al* [5] é aplicar técnicas de recomendação para disponibilizar aos desenvolvedores tarefas de acordo com seu perfil utilizando dados históricos. A plataforma escolhida para o estudo foi a TopCoder, abrangendo 2 tipos de tarefas de desenvolvimento de software (desenvolvimento e arquitetura) onde foram aplicados algoritmos utilizados para aprendizado de máquina como Árvores de Decisão, Naive Bayes e KNN e a abordagem de filtragem baseada em conteúdo, tendo como saída um modelo de recomendação.

Segundo Mao *et al* [5], como métrica de avaliação para este modelo foi utilizado tanto a precisão da recomendação como também a diversidade de tarefas recomendadas. Quanto aos resultados obtidos, o modelo de recomendação proposto foi superior em todos os conjuntos de dados analisados, tanto na recomendação de tarefas para desenvolvedores qualificados como também a recomendação para desenvolvedores iniciantes. Quanto a avaliação de desempenho o algoritmo mais eficaz foi de a Árvore de Decisão C.45, obtendo uma melhor média harmônica tanto em precisão quanto diversidade de recomendações. Ainda nos resultados, Mao *et al* [5] listam algumas sugestões como realizar uma análise criteriosa quanto ao algoritmo utilizado para o aprendizado de máquina, levando em consideração o contexto em que as tarefas estão inseridas, como também avaliar a necessidade de diversificar as recomendações de tarefas inclusive para usuários iniciantes.

Semelhante ao realizado por Mao *et al* [5], o presente trabalho têm como objetivo recomendar tarefas para os usuários das plataformas de crowdsourcing com base no seu histórico de utilização, porém foi avaliado apenas a perspectiva do trabalhador, não se atendo a recomendações que sirvam aos interesses da

plataforma. Algumas funções utilizadas para o cálculo de similaridade de tarefas bem como a técnica de filtragem baseada em conteúdo propostas por Mao et al [5], foram utilizadas no presente trabalho com algumas adaptações detalhadas posteriormente.

Yang *et al* [6] realizam uma investigação em um estudo empírico dos fatores que influenciam o comportamento do trabalhador no contexto de plataformas para desenvolvimento de software crowdsourced (CSD), tendo como objetivo proporcionar um sistema de apoio a decisão afim de melhorar o sucesso e a eficiência nestas plataformas. O estudo realizado por Yang *et al* [6] utiliza um conjunto de dados extraídos do Topcoder durante o período de janeiro de 2014 à março de 2015. As etapas utilizadas no estudo, segundo Yang *et al* [6], consistem na filtragem dos dados e pré-processamento onde são eliminadas informações relacionadas a tarefas incompletas, tarefas incomuns e trabalhadores inativos; análise de similaridade, onde para cada nova tarefa é verificado um conjunto de tarefas semelhantes do passado; construção do modelo de previsão utilizando o algoritmo de florestas randômicas juntamente com a ferramenta de análise WEKA. Como métricas para avaliar o desempenho, foram utilizadas a precisão (*precision*), revocação (*recall*), f-measure e taxa de abandono.

Como resultado, Yang *et al* [6] elenca os 10 principais fatores de impacto na decisão do trabalhador em registrar-se em uma tarefa, submeter ou desistir. Yang *et al* [6] constata ainda que a adição de atributos dinâmicos melhora o desempenho levando em consideração as métricas propostas, tendo como percentuais de probabilidade de acerto em seu modelo de predição, 95%, 94% e 96% para precisão média, revocação média e f-measure respectivamente. Quanto à eficácia das recomendações realizadas, Yang *et al* [6] observam que a abordagem de classificação utilizada pode reduzir significativamente a taxa de desistência propondo tarefas relevantes para os trabalhadores. Yang et al [6] conclui ainda que comparada a taxa média de abandono de tarefa relatada no início de trabalho de 82,9%, os resultados sugerem que este tipo de apoio dinâmico para os trabalhadores que utilizam plataformas de crowdsourcing é fundamental para alcançar um aumento significativo na taxa de submissão e consequentemente a redução da taxa de abandono.

O presente trabalho assemelha-se com o estudo de Yang *et al* [6] no intuito de utilizar atributos dinâmicos associados aos trabalhadores como histórico de participação em tarefas para realizar as recomendações. As análises e conclusões observadas nos principais fatores de impacto que influenciam na decisão de um

trabalhador, bem como as métricas utilizadas para validação também serão levadas em consideração no trabalho proposto. Como diferencial, em comparação ao trabalho de Yang *et al* [6], propõe-se um sistema de recomendação com uma modelagem de similaridade de tarefas e análise de perfil do usuário a serem realizados em tempo real, ou seja, que o sistema seja capaz de recomendar tarefas a partir do momento que o usuário interage com a plataforma.

Yuen, King e Leung [8], também analisam o problema relacionado ao tempo gasto nas plataformas de crowdsourcing para seleção de tarefas de acordo com as preferências do usuário, onde em alguns casos o tempo de procura é maior que o tempo gasto para realização para determinada tarefa. Para Yuen, King e Leung [8] o histórico de iteração do usuário com a plataforma torna possível extrair suas preferências e a partir disto realizar a recomendações de tarefas. Com o objetivo de entender melhor a relação de seleção de tarefas com as preferências do usuário, foi realizado um estudo empírico com dados coletados na plataforma *Amazon Mechanical Turk* (MTurk)[10]. Entre os principais resultados obtidos neste estudo destacam-se [8]: (i) 86% dos usuários selecionam tarefas com base na recompensa financeira; (ii) 65% dos usuários têm preferência em selecionar uma tarefa similar a tarefas realizadas no passado; (iii) 67% dos usuários não preferem selecionar uma tarefa que é similar a tarefas ao seu histórico que foram reprovadas no passado. Diante disto, Yuen, King e Leung [8] indicam que o histórico de participação do usuário pode auxiliar na seleção de tarefas em plataformas de crowdsourcing.

Com base no estudo empírico, Yuen, King e Leung [8] propõe um *framework* para recomendação de tarefas levando em consideração o histórico de desempenho do usuário e também no histórico de pesquisa por tarefas na plataforma, usando fatoração matricial aplicando técnicas de filtragem colaborativa. Para avaliar o desempenho do *framework* proposto o conjunto de dados exige o histórico de pesquisa do usuário, como este dado é de propriedade dos administradores das plataformas de crowdsourcing os autores propõem a construção de um sistema próprio para validação. Este trabalho se relaciona ao proposto no objetivo em facilitar a seleção de tarefas em plataforma de crowdsourcing, utilizando como auxílio o histórico de participação e preferências do usuário, porém diferencia-se quanto a

[10] https://www.mturk.com/

plataforma utilizada para o estudo (MTurk) bem como a técnica de recomendação utilizada (filtragem colaborativa).

O estudo de Chilton *et al* [9] também têm por objetivo o entendimento de como é o processo de seleção de tarefas em plataformas de crowdsourcing, utilizando com base de estudo a *Amazon Mechanical Turk.* Foram realizados dois métodos complementares para obter informações de como os usuários pesquisam tarefas na *MTurk*, sendo o primeiro método uma análise sobre a inferência de dados utilizados em pesquisas com diferentes critérios disponíveis na plataforma e o segundo método uma pesquisa realizada com 250 usuários sobre como eles classificam e filtram as tarefas disponíveis na *MTurk*. Como resultado, Chilton *et al* [9] relatam a importância de uma tarefa ter uma posição favorável neste tipo de plataforma, onde o experimento realizado demonstrou que uma tarefa bem posicionada foi concluída 30 vezes mais rápido e com um valor menor de recompensa financeira que quando sua posição não era favorável. A relação deste trabalho ao que foi proposto expressa-se pela preocupação em posicionar uma tarefa da forma adequada para que seja disponibilizada a usuários que possam ter interesse na sua realização. Além da plataforma utilizada no estudo, o presente trabalho diferencia-se ao utilizar a abordagem de sistemas de recomendação como apoio a tomada de decisão do usuário quanto a seleção de tarefas.

Por fim, observa-se que um ponto em comum nos trabalhos relatados é o objetivo em apoiar o usuário em plataformas de *crowdsourcing* quanto a seleção de tarefas adequadas ao seu perfil seja propondo modelos de recomendação de tarefas, bem como estudos empíricos sobre os fatores que influenciam nestas escolhas. O objetivo deste trabalho também é propor uma alternativa através de um sistema de recomendação, uma das principais diferenças que propõe-se aos demais é realizar estas recomendações em tempo real, ao usuário acessar a plataforma, para isto na próxima seção será detalhada a abordagem proposta.

4. SISTEMA DE RECOMENDAÇÃO PROPOSTO

Este capítulo apresenta o RS-Crowd (*Recommender System for Crowdsourcing*), um sistema de recomendação para usuários da plataforma TopCoder, destacando suas características, métricas utilizadas, bem como o algoritmo de recomendação. Desta forma, na primeira seção indicada como 4.1, é apresentada uma visão geral sobre a abordagem proposta e uma figura que sintetiza as etapas do processo. Já na seção 4.2 e nas subseções contidas na mesma, é apresentado o modelo de similaridade entre tarefas, onde são detalhadas as equações utilizadas para identificar as medidas de similaridades de cada atributo do modelo bem como sua exemplificação.

Na sequência do trabalho, na seção 4.3 descrito o modelo de avaliação de histórico do usuário, também detalhando as equações por meio de exemplos para obter-se as métricas relacionadas a participação do usuário em tarefas similares. Por fim, o algoritmo de recomendação proposto é demonstrado na seção 4.4 onde são apresentadas as principais funções responsáveis pela análise do histórico de participação o usuário, a relação deste histórico com as novas tarefas resultando uma lista de tarefas recomendadas que podem ser do interesse do usuário.

4.1. VISÃO GERAL DO SISTEMA PROPOSTO

O objetivo deste trabalho foi propor um sistema de recomendação de tarefas para usuários que utilizam a plataforma de crowdsourcing como o TopCoder por exemplo. O intuito é oportunizar a estes usuários a recomendação de tarefas personalizadas de acordo com seu histórico de participação na plataforma. A escolha do TopCoder, deve-se a sua popularidade destacada por Yang *et al* [6] e realçada também por Shenck e Guitard [20], elencando-a como "*a plataforma de software mais popular em software crowdsourcing*".

Para a construção desta abordagem utilizou-se como técnica de recomendação a filtragem baseada em conteúdo, pois o processo de recomendação desta técnica consiste em contrastar o perfil dos itens que o usuário demonstrou preferência no passado com cada novo item a ser avaliado. Nesta abordagem os itens são as tarefas disponíveis na plataforma TopCoder. Na Figura 4 são representadas,

de forma sintetizada, as etapas e as interações da abordagem proposta entre o usuário, a aplicação desenvolvida e a plataforma TopCoder.

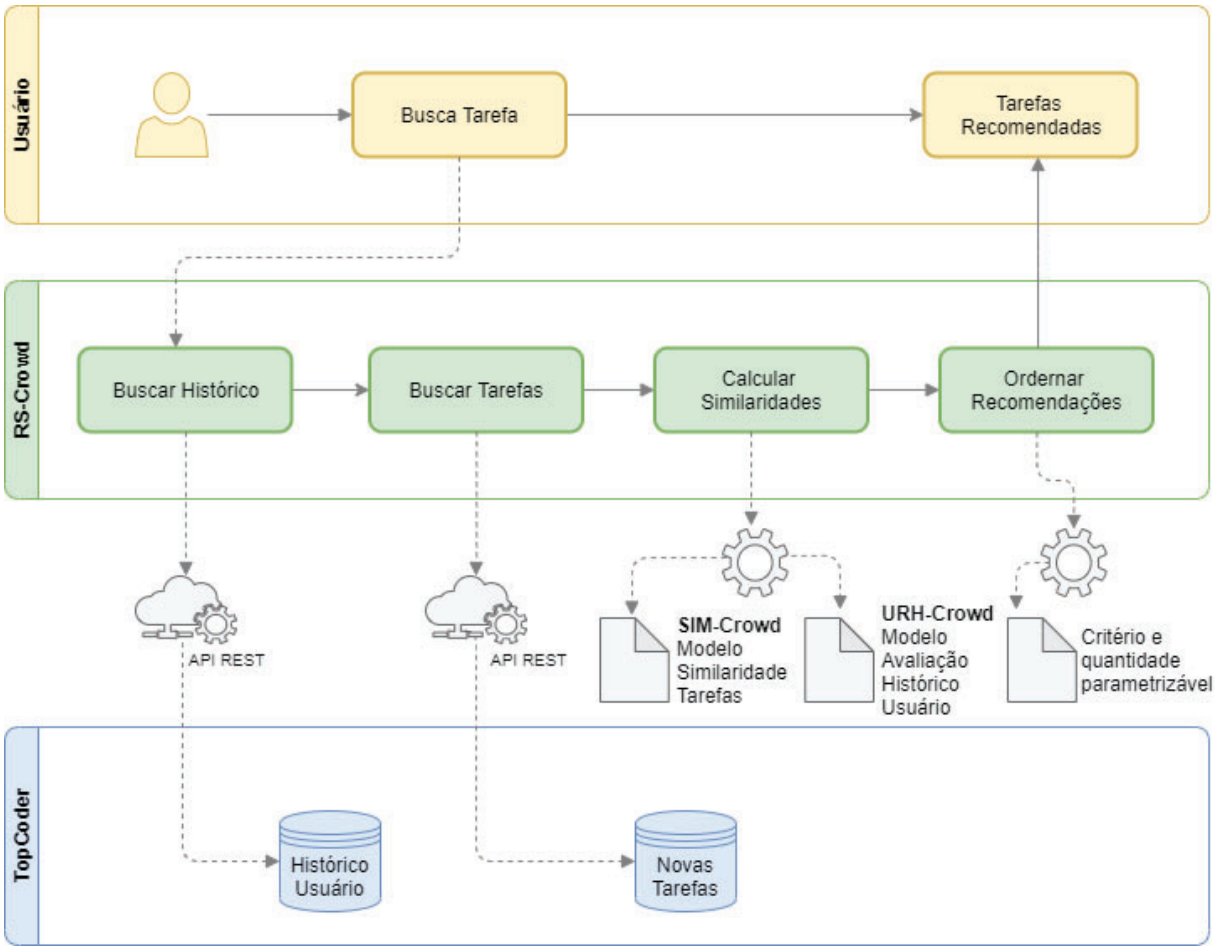

Figura 4. Representação da abordagem proposta.

O processo tem início no instante em que o usuário realiza a busca por uma tarefa disponível na plataforma TopCoder. Neste momento, o RS-Crowd realiza a busca do histórico de participação do usuário na plataforma com as informações referentes as tarefas em que o usuário interagiu no passado na categoria *"Developer"*. Isto não é feito somente para as que ele possivelmente possa ter vencido, mas também informações sobre registros e submissões que ele realizou, pois são importantes nesta etapa. Na sequência, o RS-Crowd faz a busca por novas tarefas disponíveis para registro na plataforma. Esta busca é realizada em tarefas específicas da categoria "*Developer*", categoria selecionada por estar estritamente relacionada ao desenvolvimento de software crowdsourcing (crowdsourced) relacionando-se com o objetivo presente neste trabalho. Ambas as buscas são realizadas em tempo real por requisições HTTP no padrão API/REST para plataforma TopCoder. As requisições

são realizadas desta forma para atender um dos objetivos do trabalho que é recomendar em tempo real tarefas para o usuário. Isto é possível porque a plataforma TopCoder disponibiliza o consumo destes micros serviços de forma pública[11].

De posse do histórico de participação do usuário e das tarefas disponíveis para registro, é realizada a análise de similaridade entre as novas tarefas e as tarefas que o usuário em questão interagiu no passado. Para esta análise foi definido o modelo de similaridade de tarefas denominado SIM-Crowd, que é detalhado na seção 4.2. No processo de identificação da similaridade são avaliados alguns atributos selecionados das tarefas e a partir de medidas definidas de acordo com a natureza de cada atributo, obtêm-se uma métrica de similaridade entre as mesmas. Ainda nesta etapa, as tarefas são submetidas ao modelo de avaliação do histórico do usuário, UHR-Crowd, tendo como entrada as tarefas já identificadas como similares no modelo anterior. Desta forma o UHR-Crowd, é responsável por avaliar a média geral de submissões, a média de submissões em tarefas similares e a média da pontuação de submissões em tarefas similares. Com a obtenção desta métrica, acredita-se que seja possível aprimorar as recomendações, pois de acordo com Yang *et al* [6], estas medidas estão entre os principais fatores que influenciam na seleção de tarefas pelo usuário.

A última etapa disponível na abordagem proposta é a ordenação das recomendações a serem disponibilizadas ao usuário. Nesta etapa sugere-se que o critério de ordenação e a quantidade de tarefas para recomendação sejam parametrizáveis. No caso dos critérios a abordagem permite definir uma ordem entre os seguintes índices já calculados nas etapas anteriores: (i) similaridade entre tarefas, (ii) média de submissão, (iii) média de submissão em tarefas similares e (iv) média de pontuação em tarefas similares. Em relação a quantidade de tarefas a serem recomendadas, a abordagem permite um número inteiro, sendo uma configuração parametrizável.

Ao final do processo, obtêm-se uma lista de tarefas disponíveis para registro na plataforma TopCoder.

[11] https://tcapi.docs.apiary.io/

4.2. SIM-CROWD - MODELO DE SIMILARIDADE ENTRE TAREFAS

O SIM-Crowd (*Similaritys in Crowdsourcing*) é um modelo que possibilita a identificação de similaridade entre tarefas e foi desenvolvido utilizando-se atributos associados a cada tarefa, sendo um aspecto quantitativo, baseado no histórico de participação do usuário na plataforma TopCoder, comparado com novas tarefas disponíveis para registro. Os atributos são utilizados para verificação de similaridade entre tarefas, onde sua importância é atribuída por um somatório ponderado. Estes atributos foram selecionados com base no estudo de Mao *et al* [5] e Yang *et al* [6] que também analisaram a similaridade entre tarefas no TopCoder.

Na Figura 7 é apresentada uma representação do SIM-Crowd, onde observa-se que a partir da tarefa existente no histórico do usuário os atributos selecionados são comparados com a nova tarefa através de métricas individuais e ao final é realizado o somatório ponderado gerando a medida de similaridade entre as mesmas.

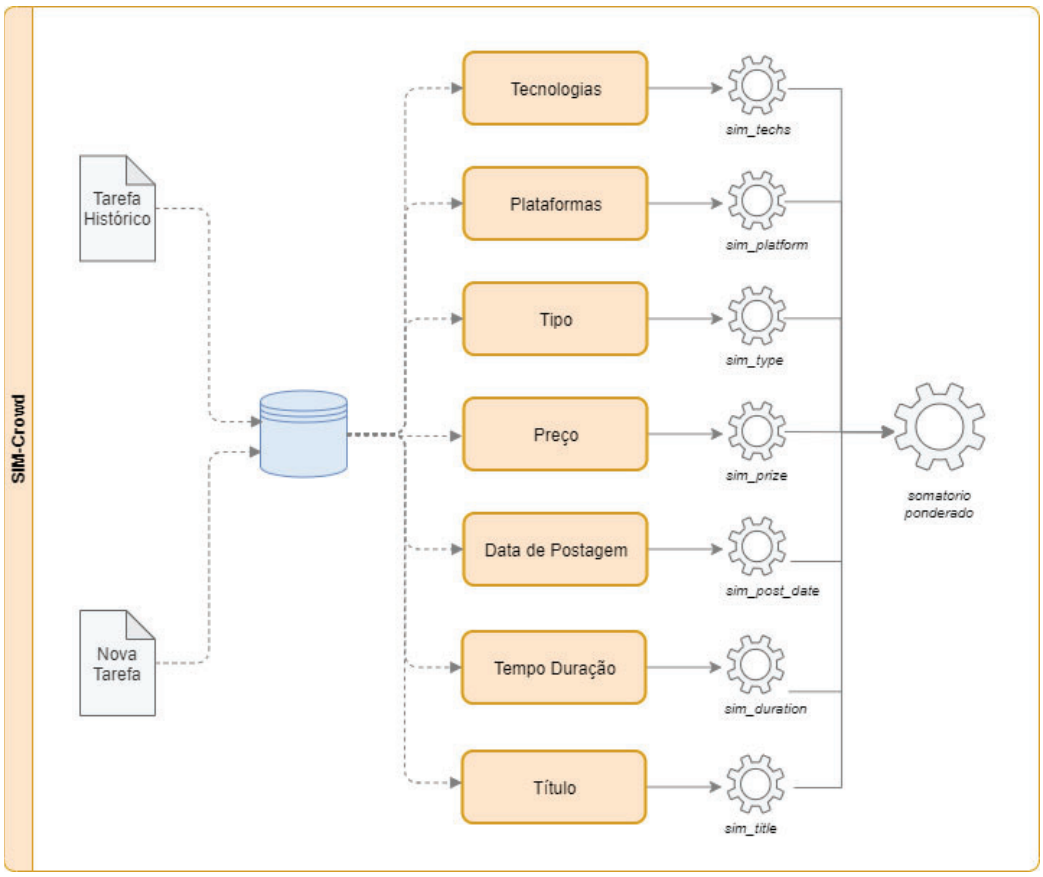

Figura 5. Representação do SIM-Crowd

Neste contexto, a principal função do modelo de similaridade é especificar como os atributos são comparados para a obtenção do grau de similaridade. Uma breve descrição dos atributos é apresentada na Tabela 1.

Tabela 1. Descrição dos atributos do SIM-Crowd

Atributo	Descrição	Exemplo
Tecnologia	Tecnologias em que a tarefa exige algum nível de conhecimento do usuário. Uma tarefa pode ter mais de uma tecnologia atribuída.	JAVA, JDBC
Plataforma	Tipo de plataforma que a tarefa está vinculada. Podem existir tarefas com mais de uma plataforma.	Mobile
Tipo	O tipo é uma subcategoria atribuída a tarefa. Cada tarefa pode estar vinculada a um tipo somente.	CODE
Tempo Duração	Tempo em dias disponível para execução da tarefa, calculado pela diferença entre a data abertura para registro e data final de submissão da tarefa.	30
Preço	Valor da recompensa disponibilizada para quem vencer a tarefa.	R$ 1.500,00
Data Postagem	Data de abertura da tarefa para registro.	15/05/2019
Título	Título que descreve de forma resumida o objetivo da tarefa.	Merlin Query Builder - Call API to get Input Names

Para obter as medidas de similaridades entre os atributos propostos no modelo de similaridade de tarefas, foram utilizadas algumas funções presentes nos artigos de Mao *et al* [5] e de Yang *et al* [6], com algumas adaptações que podem ser observados na Tabela 2 e serão detalhadas a seguir. No cálculo de similaridade dos atributos tecnologias e título utilizou-se a mesma função proposta por Mao *et al* [5], sem nenhuma alteração. Para o cálculo do preço, tempo de duração e data de postagem realizou-se a substituição da função de distância entre os elementos presente no trabalho de Yang *et al* [6] e Mao *et al* [5] pela função de similaridade conforme conversão demonstrada no trabalho de Raquel [45], onde de uma forma simplificada é apresentada a adaptação na Equação (5).

$$sim(x, y) = 1 - d(x, y) \quad (5)$$

Sendo,

- $sim(x, y)$: função de similaridade entre dois objetos, x e y.

- $d(x, y)$: função de distância entre dois objetos, x e y;

Esta adaptação foi necessária, pois os atributos selecionados no SIM-Crowd diferem dos atributos do modelo proposto por Yang *et al* [6] e Mao *et al* [5] quanto a quantidade e formato das informações obtidas. A quantidade de atributos selecionados influencia diretamente no cálculo total de similaridade já que a métrica final é obtida pelo somatório de todos cálculos individuais de similaridade. Para este trabalho não foi selecionado o atributo com a descrição da tarefa, pois identificou-se ter menor relevância em relação aos demais. Esta percepção foi possível devido ao conteúdo do texto associado às tarefas não ter, em muitos casos, correspondência semântica ao objetivo da tarefa. Verificou-se também a diferença no formato da informação representada pelos atributos, como por exemplo, no atributo plataforma, ao invés da informação ser única, como representada no trabalho de Yang *et al* [6], na representação deste modelo a informação pode ter múltiplos valores. Esta divergência foi identificada ao realizar-se análise nos dados coletados no TopCoder, sendo necessária alteração quanto a medida de similaridade associada a este atributo. Para o cálculo de similaridade do tipo de tarefa foi utilizada a função proposta por Yang *et al* [6] sem alterações.

Na Tabela 2 é realizada a comparação entre as funções do trabalho de Yang *et al* [6], Mao *et al* [5] e o SIM-Crowd.

Tabela 2. Comparação entre funções das medidas de similaridade

Atributo	Yang	Kemao	Sim-Crowd
Tecnologias	$Match(Techi;\ Techj\ ) \div NumberOfTechsMax$	$\frac{Match(Tech_i, Tech_j)}{NumberOfTechs_{Max}}$	$\frac{Match_{(tx,ty)}}{MaxTechs_{(tx,ty)}}$
Plataformas	$PLi\ ==\ PLj\ ?\ 1 : 0$	NA	$\frac{Match_{(tx,ty)}}{MaxPlatform_{(tx,ty)}}$
Preço	$(Prizei\ -\ Prizej\ ) \div PrizeMax$	$\frac{(Payment_i\ -\ Payment_j\ )}{Payment_{Max}}$	$1 - \left(\frac{Prize_x\ -\ Prize_Y}{Prize_{Max}}\right)$
Tipo	$Typei == Typej ?\ 1 :\ 0$	NA	$Type_x == Type_y\ ?\ 1 : 0$
Data Postagem	NA	$\frac{(Date_i\ -\ Date_j\ )}{Date_{MaxDiff}}$	$1 - \left(\frac{Date_{tx}\ -\ Date_{ty}}{Date_{MaxDiff}}\right)$
Tempo Duração	NA	$\frac{(Duration_i\ -\ Duration_j\ )}{Duration_{Max}}$	$1 - \left(\frac{Dur_{tx}\ -\ Dur_{ty}}{Dur_{Max}}\right)$
Título	NA	$\frac{Tit_x\ .Tit_y}{\|\|Tit_x\|\|\|\|Tit_y\|\|}$	$\frac{Vet_{tx}\ .Vet_{ty}}{\|\|Vet_{tx}\|\|\|\|Vet_{ty}\|\|}$

As métricas apresentadas na Tabela 2 são detalhadas nas subseções a seguir.

4.2.1. Similaridade baseada nas tecnologias atribuídas à tarefa

A medida de similaridade entre as tecnologias atribuídas a duas tarefas ($sim\_techs_{(tx,ty)}$) é calculada pela identificação da quantidade de tecnologias atribuídas a tarefa tx, que também estão atribuídas à tarefa ty. Este valor deve ser dividido pelo maior número de tecnologias atribuídas entre as duas tarefas, conforme a Equação (6). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre as tecnologias atribuídas às tarefas, consequentemente, quanto mais próximo a 0, menor a similaridade entre as tecnologias atribuídas às tarefas.

$$sim\_techs_{(tx,ty)} = \frac{Match_{(tx,ty)}}{MaxTechs_{(tx,ty)}} \quad (6)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Match_{(tx,ty)}$: quantidade de tecnologias de tx existentes em ty;
- $MaxTechs_{(tx,ty)}$: quantidade máxima de tecnologias entre tx e ty.

Na Tabela 3 é demonstrado um exemplo do cálculo de similaridade de tecnologias entre duas tarefas (tx, ty). Como pode ser observado, a similaridade entre a tarefa tx que possui as tecnologias “Java”, “JDBC”, “MVC” e a tarefa ty, que possui as tecnologias “HTML”, “SQL” e “Java” atribuída a si é de **0,33**.

Tabela 3. Exemplo de cálculo de similaridade de tecnologias entre tarefas

Tecnologias Tarefa *tx*	Tecnologias Tarefa *ty*	sim_techs
Java, JDBC, MVC	HTML, SQL, Java	**0,33**
$sim\_techs_{(tx,ty)} = \frac{1}{3} = 0{,}33$		

Portanto, considerando a comparação das tecnologias atribuídas entre as tarefas (tx, ty), das três tecnologias atribuídas em cada uma das tarefas, somente uma tecnologia é igual (neste exemplo a tecnologia “Java”), ou seja um terço das tecnologias destas tarefas são iguais ou em termos percentuais pode afirmar-se que a tarefa tx é 33% similar a tarefa ty. Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.2. Similaridade baseada nas plataformas atribuídas à tarefa

A medida de similaridade entre as plataformas atribuídas a duas tarefas $(sim\_platform_{(tx,ty)})$ é calculada pela identificação da quantidade de plataformas atribuídas a tarefa tx que também estão atribuídas à tarefa ty. Este valor deve ser dividido pelo maior número de plataformas atribuídas entre as duas tarefas, conforme Equação (7). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre as plataformas atribuídas às tarefas, consequentemente quanto mais próximo a 0, menor a similaridade entre as plataformas atribuídas às tarefas.

$$sim\_platform_{(tx,ty)} = \frac{Match_{(tx,ty)}}{MaxPlatform_{(tx,ty)}} \quad (7)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Match_{(tx,ty)}$: quantidade de plataformas de tx existentes em ty;
- $MaxPlatform_{(tx,ty)}$: quantidade máxima de plataformas entre tx e ty.

Para um melhor entendimento da métrica proposta, na Tabela 4 é demonstrado um exemplo do cálculo de similaridade de plataformas entre duas tarefas (tx, ty). Como pode ser observado, a similaridade entre a tarefa tx que possui as plataformas “Mobile” e “Desktop” e a tarefa ty que possui a plataforma “Mobile” atribuída a si é de **0,50**.

Tabela 4. Exemplo de cálculo de similaridade de plataformas entre tarefas

Plataformas Tarefa *tx*	Plataformas Tarefa *ty*	sim_platform
Mobile, Desktop	Mobile	**0,50**
$sim\_platform_{(tx,ty)} = \frac{1}{2} = 0{,}50$		

Portanto considerando a comparação das plataformas atribuídas entre as tarefas (tx, ty), somente uma plataforma é igual (neste exemplo a plataforma "*Mobile*"), ou seja em termos percentuais pode afirmar-se que a tarefa tx é 50% similar a tarefa ty. Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.3. Similaridade baseada no tipo da tarefa

A medida de similaridade entre o tipo atribuído a duas tarefas $(sim\_type_{(tx,ty)})$ é calculada pela comparação entre o tipo da tarefa tx com o tipo da tarefa ty, uma vez que este valor sempre será unitário, ou seja, cada tarefa só pode ter um tipo atribuído. O resultado é 1, caso os tipos forem iguais, ou 0, caso os tipos forem diferentes. Desta forma, o valor 1 indica que os tipos atribuídos entre as tarefas são similares, conforme Equação (8).

$$sim\_type_{(tx,ty)} = Type_x == Type_y \; ? \; 1 : 0 \qquad (8)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Type_x$: tipo da tarefa tx;
- $Type_y$: tipo da tarefa ty.

Na Tabela 5 é demonstrado um exemplo do cálculo de similaridade de tipos entre duas tarefas (tx, ty). Como pode ser observado, a similaridade entre a tarefa tx que possui o tipo "CODE" e a tarefa ty que possuí o tipo "CODE" é de **1**.

Tabela 5. Exemplo de cálculo de similaridade tipos entre tarefas

Tipo Tarefa *tx*	Tipo Tarefa *ty*	sim_type
CODE	CODE	**1**
$sim\_type_{(tx,ty)} = CODE == CODE = 1$		

Portanto considerando a comparação dos tipos atribuídos entre as tarefas (tx, ty), elas são do mesmo tipo (neste exemplo o tipo "CODE"), ou seja, em termos percentuais pode afirmar-se que a tarefa tx é 100% similar a tarefa ty. Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.4. Similaridade baseada no preço da tarefa

A medida de similaridade entre o preço atribuído a duas tarefas $(sim\_prize_{(tx,ty)})$ é calculada pela diferença entre o preço atribuído à tarefa tx e o preço da tarefa ty, dividido pelo maior preço entre as duas tarefas. Ao resultado subtrai-se 1, conforme Equação (9). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre os preços atribuídos às tarefas, consequentemente, quanto mais próximo de 0, menor a similaridade entre os preços.

$$sim\_prize_{(tx,ty)} = 1 - \left(\frac{Prize_x - Prize_Y}{Prize_{Max}} \right) \quad (9)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Prize_x$: preço atribuído a tarefa tx;
- $Prize_y$: preço atribuído a tarefa ty;
- $Prize_{Max}$: maior preço entre as duas tarefas.

A Tabela 6 apresenta um exemplo do cálculo de similaridade de preços entre duas tarefas (tx, ty). Como pode ser observado, a similaridade entre a tarefa tx que possui o preço de R$ 2.100,00 e a tarefa ty que possuí o preço R$ 1.500,00 é de **0,71**.

Tabela 6. Exemplo de cálculo de similaridade de preços entre tarefas

Preço Tarefa *tx*	Preço Tarefa *ty*	sim_prize
R$ 2.100,00	R$ 1.500,00	**0,71**
$sim\_prize_{(tx,ty)} = 1 - \frac{2100 - 1500}{2100} = 0{,}71$		

Portanto considerando a comparação dos preços atribuídos entre as tarefas (tx, ty), o grau de similaridade entre elas é igual a 0,71, considerando que este valor está mais próximo a 1 do que próximo a 0, pode afirmar-se que o preço entre as duas tarefas é similar. Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.5. Similaridade baseada na data de postagem

A medida de similaridade entre datas de postagem atribuída a duas tarefas $(sim\_post\_date_{(tx,ty)})$ é calculada pela diferença da data de postagem atribuída a tarefa tx pela data de postagem da tarefa ty em dias, dividido pela diferença entre a data tarefa mais antiga pela tarefa mais recente, dentre todas as tarefas comparadas. Ao resultado subtrai-se 1, conforme Equação (10). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre a data de postagem atribuída às tarefas, consequentemente quanto mais próximo a 0, menor a similaridade.

$$sim\_post\_date_{(tx,ty)} = 1 - \left(\frac{Date_{tx} - Date_{ty}}{Date_{MaxDiff}}\right) \quad (10)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Date_{tx}$: data de postagem atribuída à tarefa tx;
- $Date_{ty}$: data de postagem atribuída à tarefa ty;
- $Date_{MaxDiff}$: diferença entre a data de postagem da tarefa mais antiga pela data de postagem da tarefa mais recente entre todas as tarefas comparadas.

Na Tabela 7 é demonstrado um exemplo do cálculo de similaridade entre datas de postagem de duas tarefas (tx, ty). Como pode ser observado, a similaridade entre a data de postagem da tarefa tx 06/06/2019 e a data de postagem da tarefa ty 15/04/2019 é de **0,41**, considerando que a diferença máxima entre todas as datas de postagem das tarefas é de 120 dias.

Tabela 7. Exemplo de cálculo de similaridade entre datas de postagem de tarefas

Data Postagem Tarefa *tx*	Data Postagem Tarefa *ty*	sim_post_date
16/06/2019	06/04/2019	**0,41**
$sim\_post\_date_{(tx,ty)} = 1 - \frac{70}{120} = 0{,}41$		

Comparando as datas de postagens atribuídas entre as tarefas (tx, ty), o grau de similaridade entre elas é igual a **0,41**, considerando que este valor está mais próxima de 0 do que próximo a 1, pode afirmar-se que a data de postagem entre as duas tarefas não é similar . Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.6. Similaridade baseada no tempo de duração

A medida de similaridade entre os tempos de duração de duas tarefas $sim\_duration_{(tx,ty)}$ é calculada pela diferença do tempo de duração da tarefa tx pelo tempo de postagem da tarefa ty (em dias), dividida pelo maior tempo de duração entre as duas tarefas. Conforme visto na Tabela 1, o tempo de duração é obtido pela diferença entre a data final de submissão e a data de início de registro na tarefa. Ao resultado subtrai-se 1, conforme Equação (11). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre a data de postagem atribuída às tarefas, consequentemente, quanto mais próximo a 0, menor a similaridade.

$$sim\_duration_{(tx,ty)} = 1 - \left(\frac{Duration_{tx} - Duration_{ty}}{Duration_{Max}}\right) \quad (11)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $Duration_{tx}$: tempo de duração da tarefa tx;
- $Duration_{ty}$: tempo de duração da tarefa ty;
- $Duration_{Max}$: maior tempo de duração entre as tarefas tx e ty.

Para um melhor entendimento da métrica proposta, na Tabela 8 é demonstrado um exemplo do cálculo de similaridade entre o tempo de duração de duas tarefas (tx, ty). Como pode ser observado, a similaridade entre o tempo de duração da tarefa tx 30 dias e o tempo de duração da tarefa ty 80 dias é de **0,37**.

Tabela 8. Exemplo de cálculo de similaridade entre tempo de duração de tarefas

Duração Tarefa *tx*	Duração Tarefa *ty*	sim_duration
30	80	**0,37**
$sim\_duration_{(tx,ty)} = 1 - \left(\frac{30-80}{80}\right) = 0{,}37$		

Comparando o tempo de duração atribuído entre as tarefas (tx, ty), o grau de similaridade entre elas é igual a **0,37**, considerando que este valor está mais próxima de 0 do que próximo a 1, pode afirmar-se que o preço entre as duas tarefas não é similar . Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.7. Similaridade baseada no título

Para realizar o cálculo de similaridade entre os títulos de duas tarefas, tratando-se de um atributo textual, foram empregadas técnicas de mineração de textos, mais especificamente a indexação de frequência de termos (*term frequency indexing*), que usa o algoritmo TF-IDF (*Term Frequency-Inverse Document Frequency*). Com base nesta técnica, os títulos das tarefas foram convertidos em vetores de palavras, para serem refinados no processo de extração de análise léxica, *stemming* e remoção de "*stop words*" conforme mencionado na seção 2.2.1. Os pesos de cada termo são calculados, gerando dois vetores com os respectivos pesos dos termos de cada título. A partir destes vetores, a similaridade é calculada usando a

métrica de similaridade de cossenos, conforme detalhado na Equação (12). O resultado é um número entre 0 e 1, sendo que quanto mais próximo a 1, maior a similaridade entre os títulos, consequentemente, quanto mais próximo a 0, menor a similaridade.

$$sim\_title_{(tx,ty)} = \frac{Vet_{tx} \,.\, Vet_{ty}}{\|Vet_{tx}\| \left\|Vet_{ty}\right\|} \tag{12}$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- Vet_{tx}: vetor com os pesos calculados usando TF-IDF do título da tarefa tx;
- Vet_{ty}: vetor com os pesos calculados usando TF-IDF do título da tarefa ty;
- $\|Vet_{tx}\|$: $\sqrt{Vet_{tx}{}^2}$
- $\left\|Vet_{ty}\right\|$: $\sqrt{Vet_{ty}{}^2}$

Na Tabela 9 é demonstrado um exemplo do cálculo de similaridade entre o título de duas tarefas (tx, ty). Como pode ser observado, os vetores calculados usando TF-IDF para as tarefas tx e ty são, respectivamente [0.3, 0.0, 0.5] e [0.5, 0.4, 0.3] gerando um grau de similaridade de **0,73**.

Tabela 9. Exemplo de cálculo de similaridade entre títulos de tarefas

TF-IDF Título Tarefa *tx*	TF-IDF Título Tarefa *ty*	sim_duration
[0.3, 0.0, 0.5]	[0.5, 0.4, 0.3]	**0,73**
$sim\_title_{(tx,ty)} = \frac{(0.3\times0.5)+(0.0\times0.4)+(0.5\times0.3)}{\sqrt{0.34}+\sqrt{0.5}} = 0{,}73$		

Portanto a comparação dos títulos atribuído entre as tarefas (tx, ty), o grau de similaridade entre elas é igual a **0,73**, considerando que este valor está mais próximo a 1 do que próximo a 0, pode afirmar-se que o título entre as duas tarefas é similar. Este valor é utilizado no somatório ponderado de todas as medidas de similaridades de tarefas.

4.2.8. Somatório ponderado de similaridades das tarefas

Para obter-se a medida total de similaridade entre as tarefas é necessário realizar o somatório entre as medidas de similaridade dos atributos calculados nas seções anteriores, ou seja, tecnologia, plataforma, tipo, preço, data de postagem, tempo de duração e título associado a cada tarefa.

O somatório das medidas de similaridade foi realizado de forma ponderada, ou seja, alguns atributos possuem um peso maior em relevância aos demais. Neste modelo de similaridade entre tarefas foi utilizado a utilização do peso 2 para os atributos tipo, tecnologia e título e peso 1 aos atributos plataforma, data de postagem, tempo de duração e preço da tarefa.

A maior relevância atribuída ao tipo, tecnologia e título foi proposto pelo fato de possuírem maior relevância em contraste ao perfil do usuário, uma vez que o usuário possui tecnologias preferenciais associadas ao seu perfil, bem como geralmente costuma escolher tarefas de mesmo tipo para registrar-se. No caso do título, o peso maior foi atribuído, pois supõe-se que o objetivo da tarefa deve estar explícito no título, mesmo que resumidamente, portanto, a hipótese é que o usuário tenha preferência por tarefas com objetivos semelhantes a tarefas que ele participou no passado.

O resultado deste somatório é um número entre 0 e 10, sendo que quanto mais próximo a 10, maior a similaridade entre as tarefas, consequentemente quanto mais próximo a 0, menor a similaridade.

$$\sum_{ty}^{tx} sim = \mathrm{w}.\left(sim\_tech_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_platform_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_type_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_prize_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_post\_date_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_duration_{(tx,ty)}\right) + \mathrm{w}.\left(sim\_title_{(tx,ty)}\right) \quad (13)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário;
- ty: nova tarefa disponível para registro;
- $sim\_tech_{(tx,ty)}$: medida de similaridade entre tecnologias da tarefa tx com tarefa ty;

- $sim\_platform_{(tx,ty)}$: medida de similaridade entre plataformas da tarefa tx com tarefa ty;
- $sim\_type_{(tx,ty)}$: medida de similaridade entre tipos da tarefa tx com tarefa ty;
- $sim\_prize_{(tx,ty)}$: medida de similaridade entre preços da tarefa tx com tarefa ty;
- $sim\_post\_date_{(tx,ty)}$: medida de similaridade entre datas de postagem da tarefa tx com tarefa ty;
- $sim\_duration_{(tx,ty)}$: medida de similaridade entre tempo de duração da tarefa tx com tarefa ty;
- $sim\_title_{(tx,ty)}$: medida de similaridade entre títulos da tarefa tx com tarefa ty;
- w: peso atribuído a medida de similaridade.

Na Tabela 10 é demonstrado um exemplo do somatório das similaridades entre duas tarefas (tx, ty). Como pode ser observado, os valores obtidos nas medidas de similaridade entre tx e ty para $sim\_tech_{(tx,ty)}$, $sim\_platform_{(tx,ty)}$, $sim\_type_{(tx,ty)}$, $sim\_prize_{(tx,ty)}$, $sim\_post\_date_{(tx,ty)}$, $sim\_duration_{(tx,ty)}$, $sim\_title_{(tx,ty)}$ são respectivamente 0,33, 0,50, 1,00, 0,71, 0,50, 0,37, e 0,73. Após aplicar o cálculo do somatório ponderado das medidas de similaridades o resultado encontrado é **6,2**.

Tabela 10. Exemplo do somatório entre similaridades de tarefas

Similaridade	Total	Somatório Similaridades
$sim\_tech_{(tx,ty)}$	0,33	
$sim\_platform_{(tx,ty)}$	0,50	
$sim\_type_{(tx,ty)}$	1,00	
$sim\_prize_{(tx,ty)}$	0,71	**6,2**
$sim\_post\_date_{(tx,ty)}$	0,50	
$sim\_duration_{(tx,ty)}$	0,37	
$sim\_title_{(tx,ty)}$	0,73	
$\sum_{ty}^{tx} \boldsymbol{sim} = \mathbf{2} \times (\mathbf{0.33}) + \mathbf{1} \times (\mathbf{0.50}) + \mathbf{2} \times (\mathbf{1.00}) + \mathbf{1} \times (\mathbf{0.71}) + \mathbf{1} \times (\mathbf{0.50}) + \mathbf{1} \times (\mathbf{0.37}) + \mathbf{2} \times (\mathbf{0,73}) = \mathbf{6,2}$		

No modelo SIM-Crowd, para considerar que uma tarefa é similar a outra, esta tarefa deve possuir o somatório ponderado de similaridades superior a **7**, valor sugerido inicialmente que depois será avaliado nos experimentos realizados, portanto, considerando aplicação do SIM-Crowd neste exemplo, considera-se que a tarefa tx ***não é similar*** a tarefa ty, já que a medida de similaridade calculada foi igual a **6,2**.

O modelo de similaridade entre tarefas SIM-Crowd, têm um papel essencial no sistema de recomendação proposto, uma vez que é responsável pela identificação de quão similar é uma nova tarefa disponibilizada na plataforma em relação a tarefas que o usuário já interagiu no passado, seguindo os conceitos que norteiam a filtragem baseada em conteúdo, uma das principais técnicas utilizadas em sistemas de recomendação [30] detalhada na seção 2.2.1, e utilizada na abordagem de recomendação utilizada no RS-Crowd.

4.3. URH-CROWD - MODELO DE AVALIAÇÃO DE HISTÓRICO DE USUÁRIO

O URH-Crowd (*URH – User Rating History*) é um modelo que avalia o histórico de participação do usuário pela sua interação com as tarefas no TopCoder. As métricas utilizadas no URH-Crowd foram extraídas do trabalho de Yang et al [6] mais especificamente no modelo de decisão dinâmico proposto pelos autores onde características específicas do contexto de cada usuário são levados em consideração. Segundo os autores, o objetivo é identificar os principais fatores que influenciam no resultado na execução das tarefas, ou seja, se o usuário venceu, submeteu ou apenas registrou-se em determinada tarefa.

Para o desenvolvimento do URH-Crowd, dentre todas as métricas propostas pelos autores, foram selecionadas as métricas a seguir:

- ***"Submission Rate (SR)"***: Segundo Yang *et al* [6] o usuário tende a registrar-se em um número maior de tarefas do que é possível resolvê-las, principalmente quanto a restrição de tempo. Desta forma, é importante ter uma métrica que avalie o perfil do usuário por este viés, ou seja, quanto maior a taxa de envio, maior é a confiança que o usuário vai submeter uma tarefa;

- ***"Effort Concentration (EC)"**:* Neste caso a proposta é medir o esforço e foco do usuário em suas submissões, por meio da média de submissão em tarefas similares;
- ***"Submission Quality (SQ)"**:* A cada submissão realizada pelo usuário é atribuída uma pontuação após a revisão da mesma. Desta forma, o cálculo desta métrica é realizado pela média de pontuação do usuário em suas submissões em tarefas similares.

Na Figura 6 é apresentada uma representação do URH-Crowd, onde observa-se que a partir de um usuário e seu histórico de participação através de tarefas são geradas as métricas de avaliação propostas.

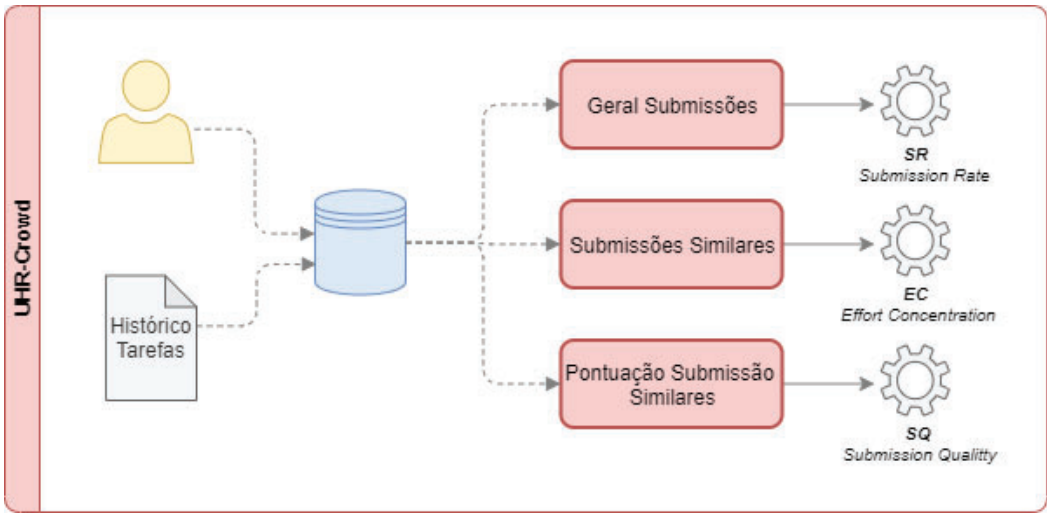

Figura 6. Representação do URH-Crowd

A seleção destas métricas dentre as demais, deve-se a um dos resultados apresentado no trabalho de Yang [6], onde são elencados os 10 principais fatores de impacto nas predições relacionadas ao histórico de participação do usuário. As métricas "Submission Quality (SQ)", "Submission Rate (SR)" e "Effort Concentration (EC)" estão colocadas em 1º, 2º e 3º, respectivamente, neste estudo, conforme pode ser observado na Figura 7.

Rank	Attribute Name	Type	Info Gain Rank	Chi Squared Rank	Average of ranks
1	SQ (average submission quality on similar tasks)	Dynamic	1	1	1
2	SR (Overall submission rate)	Static	2	2	2
3	EC (average submission rate on similar tasks)	Dynamic	3	3	3
4	CompetitorFactors-SubRate (average of average submission rate For Top Y Competitors)	Dynamic	4	4	4
5	NumWinTasksTDays (number of won tasks in last *T* days)	Dynamic	5	5	5
6	NumSubTasksTDays (number of submitted tasks in last *T* days)	Dynamic	6	6	6
7	AvgPrice (average price of registered tasks in last *T* days)	Dynamic	9	7	8
8	TotalPrize (task total prize won)	Static	8	8	8
9	NumRegTasksTDays (number of registered tasks in last *T* days)	Dynamic	7	10	8.5
10	Task Duration	Static	10	9	9.5

Figura 7. Ranking 10 fatores de impacto em predições [6].

Nas próximas seções, são detalhados os cálculos para obter-se as métricas selecionadas no URH-Crowd.

4.3.1. Taxa média geral de submissões

A taxa média geral de submissões (*Submission Rate – SR*) é calculada a partir do histórico de participação de um usuário u na plataforma TopCoder. Segundo Yang et al [6], este atributo é considerado estático, pois seu resultado é independente da análise de similaridade entre tarefas dos histórico do usuário e novas tarefas disponíveis para registro. O cálculo é realizado dividindo-se o somatório de tarefas submetidas pelo usuário u pelo total de tarefas em que o usuário u registrou-se, conforme observa-se na Equação (14).

$$SR_{(u)} = \frac{\sum_{\mathrm{u}} \mathrm{sub}}{\sum_{\mathrm{u}} \mathrm{reg}} \tag{14}$$

Sendo,

- u: usuário em que se deseja obter o cálculo;
- $\sum_{\mathrm{u}} \mathrm{sub}$: somatório de todas as tarefas submetidas pelo usuário u;
- $\sum_{\mathrm{u}} \mathrm{reg}$: somatório de todas as tarefas em que usuário u registrou-se.

Na Tabela 11 é demonstrado um exemplo deste cálculo considerando um usuário u. Como pode ser observado, o usuário deste exemplo tem um total de submissões de tarefas igual a 5 e um total de tarefas em que se registrou igual a 42. A partir do cálculo do $SR_{(u)}$ é obtida a taxa média de submissões de tarefas igual a **0,12**.

Tabela 11. Exemplo de cálculo da taxa média de submissões

Total de Submissões	Total de Registros	*Submission Rate*
5	42	**0,12**
$SR_{(u)} = \frac{5}{42} = 0{,}12$		

Neste exemplo, a taxa média de submissão de tarefas obtida para este usuário é igual a **0,12**, em termos percentuais pode-se dizer que este usuário têm uma taxa de submissão de 12% no período em que foi avaliado. Com base no estudo de Yang *et al* [6] onde os autores relatam uma taxa de abandono de tarefas no TopCoder igual 82,9% (entende-se abandono por tarefas que o usuário registrou-se e não submeteu), considera-se o usuário utilizado neste exemplo com uma taxa de submissão muito próximo da média registrada na plataforma.

4.3.2. Taxa média de submissões em tarefas similares

A taxa média de submissões em tarefas similares (*Effort Concentration - EC*) é calculada a partir do histórico de participação de um usuário u na plataforma TopCoder. Assim, tx é uma tarefa do histórico de participação do usuário u já previamente selecionada como uma tarefa similar a tarefa ty, usando o modelo de similaridade de tarefas SIM-Crowd, e ty uma nova tarefa disponível para registro. Com isso, é avaliada a taxa média de submissão do usuário u a partir do total de

submissões de tarefas similares a tarefa tx, pelo total de registros também em tarefas similares a tarefa tx, conforme Equação (15).

$$EC_{(tx,ty)} = \frac{\sum_{\text{sim\_tx}} \text{sub}}{\sum_{\text{sim\_tx}} \text{reg}} \quad (15)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário e similar a tarefa ty;
- ty: nova tarefa disponível para registro;
- $\sum_{\text{sim\_tx}} \text{sub}$: somatório de tarefas similares submetidas pelo usuário u;
- $\sum_{\text{sim\_tx}} \text{reg}$: somatório de tarefas similares em que o usuário u registrou-se.

Na Tabela 12 é demonstrado um exemplo deste cálculo considerando duas tarefas (tx, ty). Como pode ser observado, o usuário deste exemplo têm um total de submissões de tarefas similares tx igual a 3 e um total de tarefas em que registrou também similares a tx igual a 17. A partir do cálculo do $EC_{(tx,ty)}$ é obtida a taxa média de submissões de tarefas igual a **0,17**.

Tabela 12. Exemplo de cálculo da taxa média de submissão em tarefas similares

Total de submissões tarefas similares a tx	Total de registros tarefas similares a tx	Submission Rate
3	17	**0,17**
$EC_{(tx,ty)} = \frac{3}{17} = 0{,}17$		

Neste exemplo, a taxa média de submissão de tarefas similares obtida para este usuário é igual a **0,17**, em termos percentuais pode-se dizer que este usuário têm uma taxa de submissão em tarefas similares de 17% no período em que foi avaliado. Seguindo a mesma linha do estudo de Yang et al [6] sobre altas taxa de abandono de tarefas no TopCoder, desta vez considera-se este índice de 17% significativo, pois ao contrário da taxa obtida na seção 4.3.1 onde são avaliadas todas as tarefas que o usuário interagiu, nesta métrica a avaliação é realizada com um escopo mais restrito, sendo tarefas consideradas similares entre si.

4.3.3. Taxa média da pontuação de submissões em tarefas similares

A taxa média da pontuação de submissões em tarefas similares (*Submission Quality - SQ*) é calculada a partir do histórico de participação de um usuário u na plataforma TopCoder. Dessa forma, tx é uma tarefa do histórico de participação do usuário u já previamente selecionada como uma tarefa similar a tarefa ty usando o modelo de similaridade de tarefas SIM-Crowd, e ty uma nova tarefa disponível para registro. Com isso, é verificado se o usuário u realizou ao menos uma submissão em tarefas similares a tarefa tx. Em caso afirmativo, é recuperada a média de pontuação adquirida em todas as submissões em tarefas similares a tx e dividida pelo número de submissões realizadas também considerando tarefas similares a tx, conforme Equação (16).

$$SQ_{(tx,ty)} = \begin{cases} \text{num\_sub}_{\text{tx}} > 0, & \text{score\_total}_{\text{tx}} / \text{num\_sub}_{\text{tx}} \\ \text{num\_sub}_{\text{tx}} = 0, & 0 \end{cases} \quad (16)$$

Sendo,

- tx: tarefa existente no histórico de participação do usuário e similar a tarefa ty;
- ty: nova tarefa disponível para registro;
- $\text{score\_total}_{\text{tx}}$: pontuação total obtida em submissões de tarefas similares a tarefa tx pelo usuário;
- $\text{num\_sub}_{\text{tx}}$: número de submissões realizadas pelo o usuário em tarefas similares a tx.

Na

Tabela 13 é demonstrado um exemplo deste cálculo considerando duas tarefas (tx, ty). Como podem ser observados, existem três tarefas similares a tarefa tx: tarefa tv, tw e tz com pontuação adquirida na submissão pelo usuário u, sendo 98, 95 e 75, respectivamente. A partir do cálculo do $SQ_{(tx,ty)}$ é obtida a taxa média de pontuação na submissão de tarefas similares, que neste exemplo é igual a **89,33**.

Tabela 13. Exemplo de cálculo da taxa média de pontuação em tarefas similares

Tarefa Similar a tx	Score Total	Submission Quality
Tarefa tv	98	
Tarefa tw	95	**89,33**
Tarefa tz	75	
$SQ_{(tx,ty)} = \frac{98 + 95 + 75}{3} = 89{,}33$		

Neste exemplo, a taxa média da pontuação de submissão em tarefas similares obtida para este usuário é igual a **89,33**, uma taxa considerada significativa, uma vez que segundo o estudo de Yang *et al* [6] a qualidade de submissão geralmente é fraca constatada em seu trabalho pelo índice de 55,8% obtido quanto a reprovação de tarefas submetidas em determinado período de tempo, possuindo baixos índices de pontuação. Esta taxa obtida de **89,33** torna-se mais significativa, se novamente avaliarmos o escopo da métrica aplicada mais restrito, sendo tarefas consideradas similares.

4.4. RA-CROWD - ALGORITMO PARA RECOMENDAÇÕES

O RA-Crowd (*Recommender Algorithms for Crowd*) é o algoritmo responsável por gerar as recomendações tendo como objetivo realizar a busca do histórico de participação do usuário na plataforma TopCoder, buscar novas tarefas disponíveis também na plataforma e a partir disto aplicar o modelo de similaridade de tarefas SIM-Crowd bem como o modelo de avaliação do histórico de participante URH-Crowd. Com o resultado deste processamento, obtêm-se uma matriz de tarefas similares com suas respectivas métricas de similaridades. A última função do algoritmo é ordenar as tarefas a serem recomendadas conforme os critérios selecionados, tendo, por fim, a lista de tarefas a serem recomendadas. Na Figura 8 pode ser observado o algoritmo simplificado para gerar a lista de recomendação de tarefas.

```
function getRecommendations(userid, currentDateTime){

  const userHistory = getListHistory(userid);

  const newTasks = getListNewTasks(currentDateTime);

  const tasksSimilaritys = calcSimilarityTasks(userHistory, newTasks);

  const tasksRecommendations = orderTasksSimilaritys(tasksSimilaritys, [SIM, SQ, EC, SR], 10);

  return tasksRecommendations;
}
```

Figura 8. Algoritmo para Recomendação

O algoritmo proposto tem como parâmetros de entrada: *userid* sendo identificador do usuário para gerar as recomendações e *currentDateTime* que é a data e hora atual para pesquisa das tarefas disponíveis para registro na plataforma. O algoritmo em questão utiliza algumas funções auxiliares que são detalhadas a seguir:

- **getListHistory**: têm como parâmetro de entrada o identificador do usuário (*userid*), onde realiza uma busca na plataforma TopCoder via API/REST, recuperando o histórico de participação do usuário na plataforma conforme detalhado na Tabela 15;
- **getListNewTasks**: têm como parâmetro de entrada a data e hora atual (*currentDateTime*), tendo como objetivo recuperar todas as tarefas que estão disponíveis para registro na plataforma. Mais detalhes das informações recuperadas, podem ser observadas na Tabela 16.
- **calcSimilarityTasks:** têm como parâmetro de entrada o histórico de participação do usuário obtido pela função *getListHistory* detalhada acima, como também a lista de tarefas disponíveis para registro obtida na função *getListNewTasks*. A partir disto, esta função aplica as métricas detalhadas no modelo SIM-Crowd (similaridade de tarefas) bem como do modelo URH-Crowd (avaliação histórico usuário). O retorno desta função é uma matriz de tarefas contendo as métricas de similaridade entre as tarefas;
- **orderTasksSimilaritys:** o parâmetro de entrada desta função é a matriz de similaridades obtido pela função *calSimilarityTasks*, um array contendo os critérios de ordenação na respectiva ordem desejada e a quantidade de tarefas a serem recomendadas. Na Tabela 14, como exemplo, foram utilizados os seguintes critérios de ordenação:
    - 1º) SIM: Medida de similaridade entre os atributos das tarefas;

- 2º) SQ: Média de pontuação em submissão de tarefas similares;
- 3º) EC: Média de submissão de tarefas similares;
- 4º) SR: Média geral de submissão de tarefas.

O objeto *tasksRecommendations* contêm uma lista com as tarefas a serem recomendadas para o usuário em questão, ordenadas pela relevância da recomendação conforme os critérios descritos e também levando em consideração a quantidade parametrizada, um exemplo desta lista pode ser observado na Tabela 14.

Tabela 14. Matriz contendo métricas de similaridade e histórico

ID do Usuário	ID Tarefa Histórico	ID Nova Tarefa	Similaridade (SIM)	Média Pontuação (SQ)	Média Submissão (EC)	Média Geral Submissão (SR)
269983	30077774	30093216	6,85	10,00	0,125	0,025
269983	30077774	30087903	6,73	10,00	0,125	0,025
269983	30079854	30090895	6,40	10,00	0,111	0,025
269983	30077774	30090895	6,35	10,00	0,125	0,025
269983	30085723	30086476	9,27	10,00	0,100	0,025

Na Tabela 15. Informações sobre histórico de participação do usuárioTabela 15, pode ser observado as informações coletadas através da função *getListHistory* contendo informações sobre o histórico de participação do usuário e armazenado no objeto *userHistory*.

Tabela 15. Informações sobre histórico de participação do usuário

URL API/REST: https://api.topcoder.com/v3/members/$userid$/challenges							
Taskid	Título	Preço	Data de Registro	Data de Submissão	Status	Categoria	Tipo
30067224	Titan Eye - Proximity Hazard Detection Code Sprint	4300	2018-07-02	2018-07-16	COMPLETED	DEVELOP	CODE
30067344	Merlin Query Builder - Call API to get Input Names	150	2018-07-02	2018-07-12	COMPLETED	DEVELOP	FIRST_2_FINISH
30090373	[72 hrs] Quartz Energy - Core Trip Scheduler	900	2019-05-10	2019-05-13	COMPLETED	DEVELOP	CODE
30068528	Ideation Challenge - Build a	1500	2018-07-31	2018-08-08	COMPLETED	DEVELOP	CODE

	Better Tracker						
30069223	Karma VSCode extension Exploratory challenge	2200	2018-08-16	2018-08-21	COMPLETED	DEVELOP	CODE

Na Tabela 16, pode ser observado as informações coletadas através da função *getListNewTasks* contendo informações sobre novas tarefas disponíveis e armazenadas no objeto *newTasks*.

Tabela 16. Informações sobre novas tarefas

URL API/REST: https://api.topcoder.com/v3/challenges/?filter=status%3DACTIVE							
Taskid	Título	Preço	Data de Registro	Data de Submissão	Tecnologias	Plataforma	Tipo
30078144	Doppler App - Mobile App - Android - Fixes Part 1	400	2019-01-02	2019-03-24	React Native, NodeJS	MOBILE	CODE
30085985	Desktop Deployment Management Tool - API Implementation	1800	2019-04-10	2019-04-15	NodeJS, Express	DEVOPS	CODE
30086241	FTA - Fix Flex application	600	2019-03-13	2019-03-23	Java	DESKTOP	FIRST_2_FINISH

Neste capítulo foi apresentado uma visão geral sobre o sistema de recomendação proposto detalhando o modelo de similaridade de tarefas SIM-Crowd, modelo de avaliação do histórico do usuário URH-Crowd e por fim o algoritmo utilizado para gerar as recomendações RA-Crowd. No próximo capitulo serão detalhados os experimentos utilizados para avaliação deste sistema.

5. EXPERIMENTOS E RESULTADOS

O presente capítulo tem por objetivo apresentar os experimentos realizados com o intuito de avaliar o sistema de recomendação proposto denominado RS-Crowd (*Recommender System for Crowdsourcing*). Para cada experimento é apresentada a metodologia, os resultados e a discussão.

Os dados utilizados como base para os experimentos foram extraídos do TopCoder em um determinado período de tempo, utilizando uma metodologia de avaliação em Sistemas de Recomendação conceituada na literatura como abordagem offline [46], porém assumindo que os resultados estejam correlacionados com o comportamento real dos mesmos. Apesar da utilização da abordagem de avaliação offline, todas as etapas propostas no RS-Crowd foram utilizadas com a única diferença: ao invés de realizar a busca de tarefas novas para recomendação, buscou-se tarefas já realizadas comparando os resultados obtidos com as prováveis recomendações que seriam realizadas para cada usuário.

O processo realizado para coleta dos dados utilizados nos experimentos é detalhado na seção 5.1. Já na seção 5.2 são demonstrados os experimentos realizados organizados da seguinte forma: (i) Avaliação das recomendações propostas pelo RS-Crowd com as parametrizações iniciais sugeridas no experimento 5.2.1; (ii) Avaliação dos critérios de ordenação aplicados na lista de recomendações pelo experimento 5.2.2; (iii) Avaliação da variação da medida de similaridade associada entre as tarefas proposta no modelo SIM-Crowd.

5.1. DADOS UTILIZADOS NOS EXPERIMENTOS

Segundo Herlocker [46], uma avaliação offline objetiva e repetível de um Sistema de Recomendação requer a existência de uma base histórica de avaliações de itens por usuários. Ao se verificar a qualidade de uma recomendação, os itens recomendados são comparados com as avaliações contidas na base histórica. Herlocker [46] sugere que a qualidade é medida pela semelhança do conjunto de itens recomendados para um determinado usuário em relação ao conjunto dos itens melhor avaliados pelo mesmo usuário. No contexto do TopCoder os usuários não avaliam as tarefas, eles interagem por meio de registros e submissões, podendo vencer

determinado desafio. Neste caso o objetivo é utilizar os dados históricos extraídos do TopCoder para simular recomendações a usuários em um determinado período de tempo e verificar se as tarefas que o usuário interagiu no período posterior fazem parte do conjunto de tarefas recomendadas pelo RS-Crowd, ou seja, as recomendações seriam efetivas.

Em um sentido mais amplo, o que se espera da realização destes experimentos é a filtragem de um conjunto de recomendações potencialmente candidatas a um bom desempenho, para um conjunto menor, que possa ter seu processo de avaliação continuado e testado com o objetivo de avaliação real após esta filtragem.

Atualmente o TopCoder conta com uma base de dados com mais de 1,5 milhões de usuários cadastrados e mais de 55 mil tarefas completadas, dados estes extraídos do próprio site[12] Em virtude do seu tamanho e por questões de performance não serem objetos deste estudo, optou-se por selecionar parte destes dados para realização do experimento, seguindo os critérios detalhados abaixo:

- **Período de um ano:** O período determinado para coleta de dados foi entre 20/06/2018 à 20/06/2019;
- **Categoria "*Developer*":** Existem três principais tipos de categorias que as tarefas estão vinculadas: *Design*, *Data Science* e *Developer*. Como o objetivo deste trabalho é recomendar tarefas para desenvolvimento de software por meio do TopCoder, a categoria Developer foi escolhida;
- **Status "*Completed*":** Existem diversos tipos de status associados as tarefas, como cancelado, incompleto, etc... Para garantir que a tarefa teve seu ciclo completo dentro do TopCoder, ou seja, passou por todas as fases inerentes ao processo, optou-se por a realização deste filtro selecionando somente tarefas com status igual a "Completed";
- **Usuários Ativos:** Foram selecionados os usuários que já se registraram ao menos em uma tarefa no período selecionado. No estudo de Yang *et al* [6] é mencionado o alto índice de usuários considerados inativos na plataforma, ou seja, usuário que nunca ao menos registrou-se em alguma tarefa na plataforma, portanto, como o sistema recomendação proposto depende de

[12] https://www.topcoder.com/community/statistics/

histórico de usuários para gerar recomendações, os usuários sem nenhum tipo de interação foram ignorados.

Para coleta dos dados, utilizou o módulo “Buscar Histórico” e “Buscar Tarefas” existentes o RS-Crowd com a adaptação em relação a busca de tarefas e histórico para um período fixo, utilizando como parâmetro o período descrito acima.

Na utilização de experimentos com avaliação offline, existe a necessidade da divisão da base de dados [41], neste caso o histórico de participação dos usuários, em dois conjuntos: um de treinamento e outro de teste. O conjunto de treinamento será utilizado como base para originar-se as recomendações ao usuário baseado em seu histórico, já o conjunto de testes é utilizado para a avaliação das recomendações geradas aplicando métricas comuns em sistemas de recomendação. Segundo Marques [41] o processo de divisão destes conjuntos pode ser aleatório, ou manipulado de modo a garantir algumas propriedades em ambos subconjuntos. Ainda segundo Marques [41], o processo pode ser exclusivo ou não exclusivo, ou seja, os itens selecionados para um conjunto podem ou não estar presentes no outro conjunto.

Após definir o método de separação, é necessário definir a proporção de divisão entre os conjuntos de treinamento e testes. Neste caso, segundo Amatrian [32] uma configuração geralmente utilizada é a divisão aleatória exclusiva na proporção 80 por 20, ou seja, o conjunto original será dividido aleatoriamente em dois novos conjuntos sendo: 80% o conjunto de treinamento e 20% conjunto de teste, garantindo que itens presentes no conjunto de treinamento, não devem estar presentes no conjunto de testes.

Com base nestas definições, no presente trabalho optou-se pela utilização do processo divisão aleatória exclusiva na proporção de 75 por 25, sendo o critério de filtro a data de encerramento de registro na tarefa. Selecionou-se o processo de divisão aleatório exclusivo como forma de garantir a integridade das recomendações, uma vez que não faria sentido recomendar tarefas que o usuário já está interagindo. Quanto a proporção, optou-se pelo 75 por 25 pois como o critério de filtro são datas, para fins de arredondamento utilizou-se para o conjunto de treinamento os primeiros 9 meses de dados históricos, ou seja, o período de 20/06/2018 à 19/03/2019 e para o conjunto de testes os 3 meses seguintes, 20/03/2019 à 20/06/2019.

Na Tabela 17 pode ser observado o detalhamento dos conjuntos de treinamento e testes em relação ao conjunto original de dados.

Tabela 17. Divisão conjunto de treinamento e testes

Tipo Informação	Original	Treinamento	Testes
Usuários	352	331	206
Tarefas	1459	1022	437
Histórico (usuário x tarefa)	18678	14020	4658

Por se tratar de um experimento com abordagem de avaliação offline, com as tarefas já definidas, optou-se por realizar previamente o cálculo de similaridade entre as tarefas usando o modelo de similaridade de tarefas SIM-Crowd, bem como a aplicação do modelo de avaliação de histórico do usuário URH-Crowd, ação executada no módulo “Calcular Similaridades” proposto no RS-Crowd. Desta forma, ao final deste processo, obtêm-se uma matriz de tarefas com a medida de similaridade entre as mesmas e também os indicadores sobre o histórico de participação em tarefas semelhantes. Na Tabela 14 apresentada na seção 4.4 pode ser observado um exemplo da matriz retornada.

Os experimentos detalhados a seguir, utilizam como base os dados mencionados nesta seção, as variações em cada experimento se dão nos parâmetros utilizados como entrada no módulo “Ordenar Recomendações” e no caso do Experimento 2 a variação no índice para considerar uma tarefa similar, utilizado no módulo “Calcular Similaridades”.

5.2. EXPERIMENTOS REALIZADOS

A seguir são detalhados os experimentos realizados seguido da análise sobre os resultados obtidos.

5.2.1. Experimento 1 – Avaliar recomendações do RS-Crowd com as parametrizações iniciais sugeridas

Este experimento tem como objetivo verificar a capacidade do RS-Crowd gerar recomendações relevantes ao usuário, ou seja, indicar tarefas que possam ser de seu interesse com base em tarefas do passado. Para isto, foram utilizadas algumas métricas normalmente utilizadas para análise da qualidade da recomendação [33], conforme mencionadas na seção 2.2.4, sendo elas: Precisão (*Precision*), Revocação (*Recall*) e F-Measure.

Como no algoritmo proposto para o RS-Crowd o número de tarefas a serem recomendadas é parametrizável, neste mesmo experimento procurou-se avaliar também as métricas mencionadas acima variando o número de tarefas recomendadas ao usuário nas seguintes quantidades: 5, 10 e 20 tarefas, escolhidas por agrupamento, representados respectivamente pelas siglas TOP-5, TOP-10 e TOP-20.

Quanto ao critério de ordenação, foi utilizado neste experimento o critério sugerido inicialmente, ou seja, que as tarefas a serem recomendadas sigam a seguinte ordenação: SIM (Similaridade entre os atributos das tarefas), SQ (Média de pontuação em submissão de tarefas similares) e EC (Média de submissão de tarefas similares).

Para a realização do experimento, foi utilizada a abordagem de avaliação offline, usando o conjunto de dados coletado conforme mencionado na seção 5.1. Para cada usuário foi verificado seu histórico de participação obtido da base de treinamento com 1022 tarefas e comparado com as 437 tarefas existente na base de testes, a partir disto será gerado a matriz de similaridade, considerando os parâmetros quanto a quantidade e ordenação das recomendações conforme mencionado acima. Neste experimento, o índice utilizado no modelo SIM-Crowd para verificar se uma tarefa é similar a outra é igual ou superior a 7, índice este escolhido inicialmente que será reavaliado nos experimentos posteriores.

Na Tabela 18 são apresentados os resultados médios obtidos após o cálculo das métricas de recomendação e a variação quanto ao número de recomendações para todos os usuários.

Tabela 18. Cálculo de Precisão, Revocação e F-Measure

Recomendações	Precisão	Revocação	F-Measure
TOP-5	0,2057	0,0022	0,0044
TOP-10	0,1956	0,0041	0,0079
TOP-20	0,1792	0,0064	0,0123

A representação visual do relacionamento entre as métricas obtidas e sua variação entre o número de recomendações geradas pode ser observado no gráfico apresentado na Figura 9.

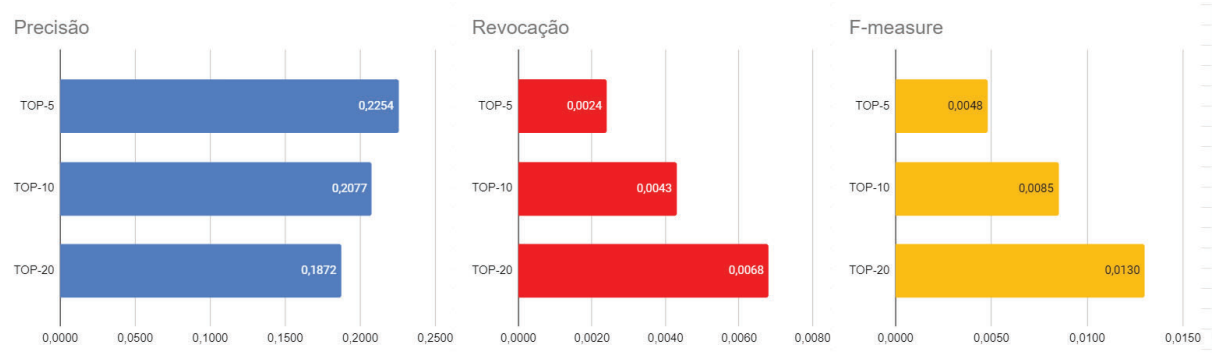

Figura 9. Gráfico representando o cálculo da precisão, revocação e f-measure.

Analisando a Tabela 18 e a Figura 9, observa-se que apesar da pequena variação de desempenho, a lista contendo 5 recomendações (TOP-5) obtêm um melhor resultado perante as demais em relação a precisão apresentada obtendo um valor médio de 0,2057. Já a menor precisão verificada é para a lista contendo 20 recomendações (TOP-20), sendo um valor médio de 0,1792. Isto deve-se a uma característica da métrica Precisão, onde à medida que o tamanho da lista de recomendações aumenta, a tendência é que a proporção de recomendações corretas diminua. O baixo valor de revocação medido principalmente na lista contendo 5 tarefas recomendadas (TOP-5) no valor de 0,0022 já era esperado devido a pequena quantidade de tarefas recomendadas em relação ao total de tarefas analisadas, conforme detalhado anteriormente, totalizando 437 tarefas. Pelo mesmo motivo, o valor mais expressivo é encontrado na lista contendo 20 tarefas (TOP-20), com um valor de revocação médio encontrado igual a 0,0064.

Segundo Herlocker [33], a medida mais apropriada para sistemas de recomendação que não propõe-se a retornar todos as recomendações geradas é a precisão, porém no presente trabalhou optou-se pela utilização da revocação para que sejam obtidos mais indicadores sobre as variações do número de tarefas a serem recomendadas. Outro ponto é que os valores de precisão e revocação possuem tendência de crescimento inversas, ou seja, à medida que o tamanho da lista de itens recomendados aumenta, a expectativa é que a precisão diminua e a revocação aumente. Desta forma também houve necessidade de utilização da métrica f-measure como maneira de obter uma média harmônica entre precisão e revocação.

5.2.2. Experimento 2 – Avaliar critérios de ordenação aplicados a lista de recomendações

Para gerar a recomendação de tarefas ao usuário proposta no RS-Crowd, além da necessidade de verificar a similaridade entre as tarefas existentes no histórico do usuário e as novas tarefas disponíveis na plataforma, a abordagem sugere que seja realizada uma avaliação no histórico do usuário por meio de sua interação com as tarefas do TopCoder usando o URH-Crowd. As métricas utilizadas neste modelo são a média geral de submissão de tarefas do usuário (*Submission Rate – SR*), média de submissão em tarefas similares (*Effort Concentration – EC*) e a média de pontuação em tarefas similares (*Submission Quality – SQ*).

Como no algoritmo proposto para o RS-Crowd o critério de ordenação das tarefas a serem recomendadas é parametrizável, procurou-se avaliar qual a melhor combinação de ordenação entre os parâmetros permitidos, sendo as métricas obtidas após aplicação do URH-Crowd (SR, SQ e EC) bem como a medida de similaridade obtida na aplicação do modelo SIM-Crowd (SIM). Neste experimento, não utilizou-se como critério de ordenação o atributo SR (*Submission Rate*) uma vez que trata-se de um atributo estático e não influencia nas métricas utilizadas considerando o contexto de um usuário em específico. O índice utilizado no modelo SIM-Crowd para verificar se uma tarefa é similar a outra é igual ou superior a 7, que foi indicado inicialmente e será avaliado no experimento 03 descrito na seção 5.2.3.

Na Tabela 19 são apresentados os resultados médios obtidos após o cálculo das métricas de recomendação, variação quanto ao número de recomendações e variação no tipo de ordenação utilizada.

Tabela 19. Cálculo de Precisão, Revocação e F-Measure para diferentes tipos de ordenação

Ordenação	Recomendações	Precisão	Revocação	F-Measure
sim_sq_ec	TOP-5	0,2057	0,0022	0,0044
sim_sq_ec	TOP-10	0,1956	0,0041	0,0079
sim_sq_ec	TOP-20	0,1792	0,0064	0,0123
sim_ec_sq	TOP-5	0,2057	0,0022	0,0044
sim_ec_sq	TOP-10	0,1956	0,0041	0,0079
sim_ec_sq	TOP-20	0,1796	0,0064	0,0124
sq_sim_ec	TOP-5	0,2245	0,0024	0,0048
sq_sim_ec	TOP-10	0,2064	0,0043	0,0084

sq_sim_ec	TOP-20	0,1870	0,0068	0,0130
sq_ec_sim	TOP-5	0,2254	0,0024	0,0048
sq_ec_sim	TOP-10	0,2077	0,0043	0,0085
sq_ec_sim	TOP-20	0,1872	0,0068	0,0130
ec_sim_sq	TOP-5	0,2156	0,0023	0,0046
ec_sim_sq	TOP-10	0,2059	0,0043	0,0084
ec_sim_sq	TOP-20	0,1870	0,0068	0,0130
ec_sq_sim	TOP-5	0,2164	0,0023	0,0046
ec_sq_sim	TOP-10	0,2059	0,0043	0,0084
ec_sq_sim	TOP-20	0,1866	0,0068	0,0130

A representação visual do relacionamento entre as métricas obtidas, sua variação entre o número de recomendações geradas e a variação entre a ordenação, pode ser observado no gráfico apresentado na Figura 10.

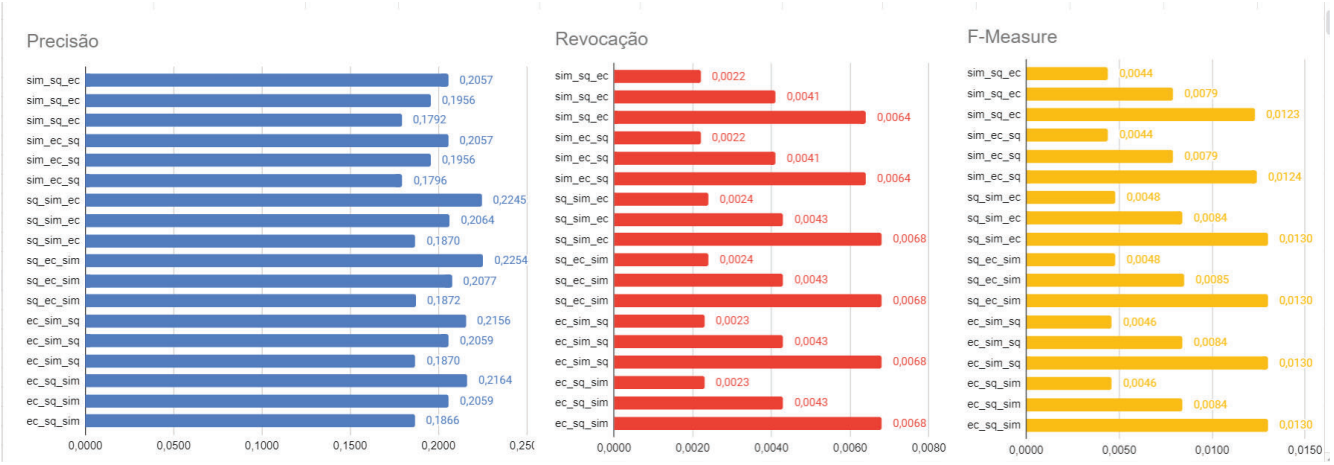

Figura 10. Cálculo precisão, revocação e f-measure com variação de ordenação.

Analisando a Tabela 19 e Figura 10, observa-se que a ordenação com melhor desempenho em relação a precisão é a lista contendo 5 recomendações (TOP-5) utilizando a ordenação SQ_ EC_ SIM obtendo um valor de 0,2254 seguida da lista de contendo 5 recomendações (TOP-5) com ordenação SQ_EC_SIM obtendo um valor de 0,2245. Como neste tipo de ordenação é levado em consideração, primeiramente, a taxa média de pontuação do usuário em tarefas similares (SQ), faz sentido este tipo de ordenação ter um resultado melhor perante os demais, pois a tendência é o usuário registrar-se/submeter tarefas em que teve um bom desempenho no passado.

Outro ponto que chama atenção, é que a ordenação SIM_SR_EC e SIM_EC_SR obtêm praticamente o mesmo desempenho, mesmo variando o número de tarefas recomendadas. Utilizando este tipo de ordenação nota-se uma pequena diferença nas medidas de precisão, revocação e f-measure para a lista de 20 tarefas para recomendação (TOP-20). Isto deve-se pela prevalência na ordenação do índice de similaridade entre as tarefas (SIM), como os índices de SQ e EC são interligados, pois tratam do histórico de submissão em tarefas similares. Em casos que o usuário não tiver histórico de submissão ou os índices forem muito similares entre todas as tarefas a serem comparadas, o índice de similaridade vai sempre prevalecer aos demais.

A principal relevância deste experimento foi identificar que o melhor critério de ordenação a ser utilizado é o SQ_SR_SIM, ao invés do critério SIM_SQ_SR, como proposto inicialmente. A diferença entre estas duas formas de ordenação pode ser melhor observada na Figura 11.

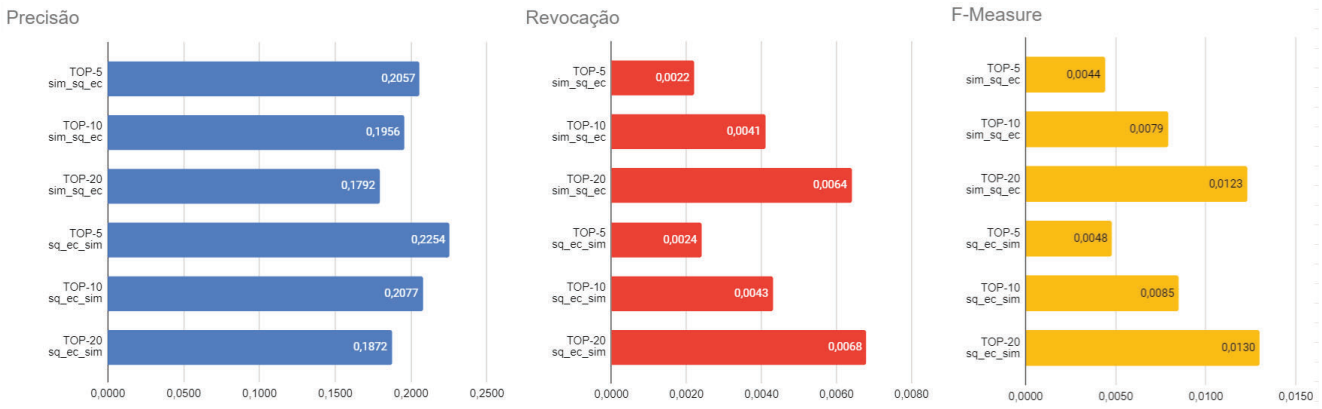

Figura 11. Cálculo precisão, revocação e f-measure com variação de ordenação entre SIM_SQ_SR e SQ_SR_SIM.

Como observa-se na Figura 11, a precisão encontrada utilizando a ordenação SQ_EC_SIM na lista contendo 5 recomendações (TOP-5) com valor de 0,2254 é significativamente superior a ordenação SQ_EC_SIM, também com uma lista de 5 recomendações (TOP-5) de medida 0,2057. As outras métricas como revocação e f-measure também são superiores utilizando a ordenação SQ_EC_SIM. Desta forma, sugere-se utilizar este tipo de ordenação como parâmetro inicial na utilização do sistema de recomendação RS-Crowd.

5.2.3. Experimento 3 – Avaliar variação da medida de similaridade associada entre tarefas

Conforme visto na seção 4.2, para identificar a similaridade entre as tarefas associadas ao histórico de participação do usuário e as novas tarefas disponíveis no TopCoder, é aplicado o SIM-Crowd, modelo de similaridade de tarefas. Este experimento objetiva avaliar, com base nas métricas de recomendação vistas nos experimentos anteriores (precisão, revocação e f-measure), quais os resultados obtidos variando-se o indicador de similaridade proposto com os valores 5, 6, 7 e 8.

Para uma análise mais ampla, optou-se por manter a variação da quantidade de itens a serem recomendados conforme proposto também nos experimentos anteriores, ou seja, foram calculadas as métricas para as recomendações TOP-5, TOP-10 e TOP-20 da lista de recomendações.

Quanto a ordenação, foi utilizado neste experimento o critério identificado com melhor desempenho no Experimento 2, ou seja, que as tarefas a serem recomendadas sigam a seguinte ordenação: SQ (Média de pontuação em submissão de tarefas similares), EC (Média de submissão de tarefas similares) e SIM (Similaridade entre os atributos das tarefas), representado pela sigla SQ_EC_SIM.

Na Tabela 20 são apresentados os resultados obtidos após o cálculo das métricas de recomendação, a variação quanto ao número de recomendações e a variação da medida aplicada no modelo de similaridade de tarefas.

Tabela 20. Cálculo de Precisão, Revocação e F-Measure para diferentes índices de similaridade.

Similaridade	Recomendações	Precisão	Revocação	F-Measure
5	TOP-5	0,1377	0,0015	0,0031
5	TOP-10	0,1224	0,0027	0,0053
5	TOP-20	0,1045	0,0046	0,0089
6	TOP-5	0,1472	0,0016	0,0032
6	TOP-10	0,1350	0,0029	0,0057
6	TOP-20	0,1267	0,0053	0,0102
7	TOP-5	0,2254	0,0024	0,0048
7	TOP-10	0,2077	0,0043	0,0085
7	TOP-20	0,1872	0,0068	0,0130
8	TOP-5	0,3537	0,0035	0,0068
8	TOP-10	0,3436	0,0057	0,0112
8	TOP-20	0,3398	0,0073	0,0141

A representação visual do relacionamento entre as métricas obtidas, sua variação entre o número de recomendações geradas e a variação entre as medidas de similaridade, pode ser observado no gráfico apresentado na Figura 12.

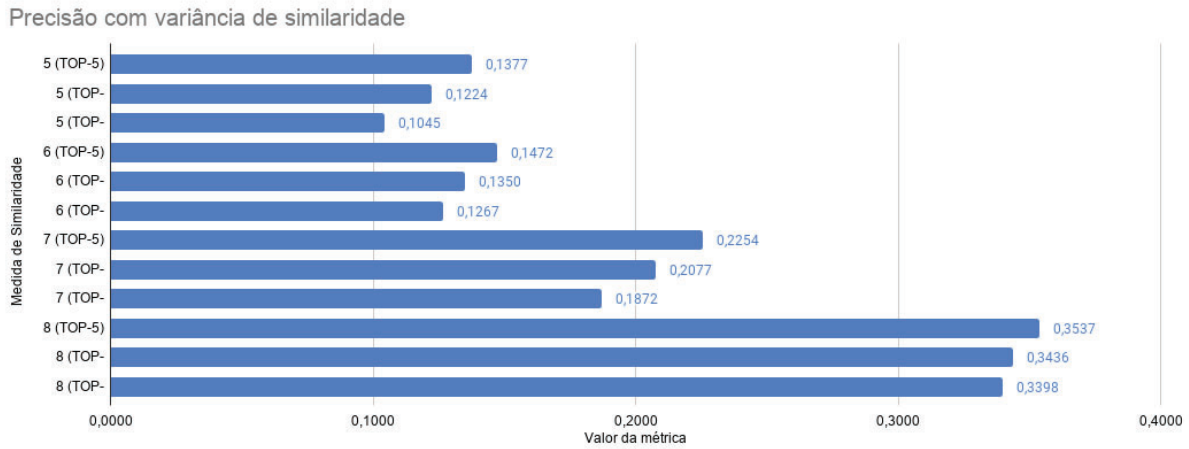

Figura 12. Comparação das métricas com variação da medida de similaridade

Analisando a Tabela 20 e a Figura 12, observa-se que utilizando a medida de similaridade maior ou igual a 8 no cálculo de similaridade de tarefas proposto no SIM-Crowd, o desempenho da métrica de precisão é significativamente superior as demais medidas. Isso ocorre em todos os tamanhos de listas de recomendações (TOP-5, TOP-10 e TOP-20) apresentando o valor 0,3537, 0,3436 e 0,3398, respectivamente ao tamanho das listas de recomendações. Já para as métricas de revocação e f-measure os valores permanecem semelhantes as demais medidas de similaridade. Na Figura 12, observa-se também que a precisão têm um melhor desempenho a medida que o índice utilizado para o cálculo de similaridade aumenta, isto deve-se que ao ser recuperado tarefas mais similares a tarefas que o usuário interagiu anteriormente, maior a chance do usuário ter interesse nestas tarefas. Porém, ao utilizar um índice muito alto de similaridade na comparação entre tarefas, o número de tarefas a serem recomendadas tende a diminuir em virtude da especialização na busca de tarefas, em alguns casos pode ocorrer que, para alguns usuários, o sistema não consiga recomendar tarefas, pois com índice muito alto pode não encontrar tarefas similares entre novas tarefas e o histórico do usuário.

Como forma de validar a proporção das recomendações gerada para os índices de similaridades propostos, pode se utilizar uma métrica também relacionada a sistemas de recomendação chamada cobertura (*coverage*), detalhada na seção 2.2.4.

Na Tabela 21, é apresentado a relação entre o índice de similaridade utilizado e a abrangência de recomendações representada pelo número de usuários em que o sistema pode recomendar tarefas, seguido da medida de cobertura considerando o base de treinamento com 331 usuários conforme definido anteriormente.

Tabela 21. Cálculo de cobertura variando índice de similaridade.

Similaridade	Abrangência Recomendações em usuários	Cobertura
5	308	0,9305
6	283	0,8550
7	224	0,6767
8	114	0,3444

Na Figura 13, é representado o relacionamento disposto na tabela acima.

Figura 13. Cobertura de recomendações com variação do índice de similaridade.

Como pode ser observado, utilizando o índice de similaridade maior ou igual a 8, foi possível recomendar tarefas para apenas 114 usuários, de um total 331 existentes na base de treinamento, ou seja, uma cobertura igual a 0,3444, a menor entre a cobertura calculadas para todas as medidas de similaridade. Utilizando o

índice de similaridade maior ou igual a 5, observa-se que possui uma cobertura mais abrangente entre todas as medidas de similaridades apresentadas, com um valor de 0,9305, atingindo em 308 usuários de um total de 331 existentes.

Apesar de apresentar o melhor índice de cobertura, o índice para cálculo de similaridade maior ou igual a 5 não demonstra um bom desempenho na precisão das recomendações, portanto sugere-se utilizar além do cálculo de cobertura, também as métricas de precisão, revocação e f-measure para que seja possível encontrar um equilíbrio na parametrização a ser utilizada no RS-Crowd. Desta forma, com base neste experimento sugere-se manter o índice proposto inicialmente, com índice de similaridade maior ou igual a 7, pois apesar de não possuir a melhor taxa de cobertura, considera-se que possui uma taxa de cobertura significativa com um valor de 0,6767, contemplando 224 usuários de um total de 331 na base de treinamento. Em experimentos anteriores, o índice maior ou igual a 7 também possui índices significativos de desempenho quanto a precisão nas recomendações geradas.

Uma restrição identificada na realização dos experimentos deste trabalho, foi a avaliação do Sistema de Recomendação em tempo real, pois seria necessária uma quantidade considerável de usuários interagindo no TopCoder com algum histórico de participação associado ao seu perfil, bem como a necessidade de uma integração direta com a plataforma para disponibilizar as recomendações, algo muito difícil de ocorrer devido ao reconhecimento da ferramenta a nível mundial e também por este trabalho ainda em encontrar-se em fase experimental. De qualquer forma entende-se que apesar da utilização de uma abordagem de avaliação offline, devido ao detalhamento do processo de recomendação seja possível realizar o mesmo em tempo real, pois a plataforma disponibiliza as informações necessárias em API aberta e também o processamento pode ser realizado conforme os modelos apresentados.

Também foi observado algumas limitações referentes a técnica para recomendação utilizada neste trabalho, neste caso a Filtragem Baseada em Conteúdo. A superespecialização é uma delas, resultando na dificuldade de surpreender o usuário através de recomendações diferentes da habitual. A partida fria também foi observada nos casos em que o usuário não possui histórico. Ambas limitações não forma mitigadas por não ser objeto deste trabalho.

Percebe-se alguns fatores limitantes neste trabalho, contudo estas restrições podem representar a oportunidade de novas investigações em trabalhos futuros.

6. CONCLUSÃO

Neste capítulo são apresentadas as conclusões, destacando-se os objetivos, as contribuições e os resultados obtidos no trabalho proposto. Por fim, são discutidas algumas perspectivas sobre trabalhos futuros identificados no decorrer do presente trabalho.

A crescente expansão das comunidades online surge como fontes intermináveis de criatividade e novos talentos, sendo que as tarefas que eram tradicionalmente realizadas por algum recurso contratado, estão sendo realizadas pelas comunidades de crowdsourcing. Este modelo pode ser utilizado em diversas áreas, entre elas o desenvolvimento de software, tendo inúmeras vantagens como uma melhor qualidade do software, redução de custo, solução diversificadas entre outros. Porém o sucesso na utilização do crowdsourcing para o desenvolvimento de software, depende de um grande número de desenvolvedores estarem engajados que registrem e submetam tarefas de forma recorrente. Um dos principais desafios encontrados pelos usuários de plataformas de crowdsourcing é a seleção de tarefas, devido ao grande número de tarefas disponíveis simultaneamente e a dificuldade de encontrar tarefas de acordo com seu perfil. Os sistemas de recomendação surgem como uma alternativa para amenizar este tipo de desafio, utilizados em situações quando o usuário não possui conhecimento suficiente para tomar uma decisão sobre um domínio específico ou quando ele não é capaz de avaliar todas as opções disponíveis.

Com objetivo de minimizar o desafio quanto a dificuldade na seleção de tarefas, neste trabalho foi proposto o RS-Crowd (*Recommender System for Crowdsourcing)* um sistema de recomendação de tarefas para usuários de plataformas de crowdsourcing, mais especificamente do TopCoder, plataforma amplamente utilizada no desenvolvimento de software crowdsourcing. A técnica de recomendação utilizada foi a filtragem baseada em conteúdo, que consiste em recomendar itens similares aos que o usuário interessou-se no passado. No RS-Crowd, foi proposto um algoritmo denominado RA-Crowd (*Recommender Algorithms for Crowd*) que sugere recomendações em tempo real com base no histórico de iteração do usuário na plataforma. Para isto trabalhou-se em uma métrica para calcular a similaridade entre novas tarefas disponíveis na plataforma com as tarefas

que o usuário interagiu no passado, por meio de um modelo de similaridade de tarefas intitulado SIM-Crowd (*Similaritys in Crowdsourcing*). Usando este modelo foi possível obter indicadores de similaridade quantitativas entre as tarefas. Ainda no RA-Crowd, foi proposto um modelo para avaliação do histórico de participação do usuário em tarefas similares, o URH-Crowd (*URH – User Rating History)*, tendo como resultado algumas métricas também quantitativas em relação a submissão e pontuação obtida pelo usuário na plataforma.

A avaliação do sistema de recomendação foi realizada por meio de experimentos, usando uma abordagem de avaliação offline, com dados extraídos do TopCoder em um determinado período de tempo. Aos experimentos foram aplicadas métricas comumente utilizadas na avaliação de sistemas de recomendação, como precisão (precision), revocação (recall), f-measure e cobertura (coverage).

Com a realização dos experimentos pode observar-se que o melhor critério a ser utilizado quanto a ordenação de tarefas a serem recomendadas deve considerar, primeiramente, atributos dinâmicos associados ao perfil do usuário que representam o seu desempenho na plataforma, que por sinal, provou-se diferente do critério sugerido inicialmente onde o grau de similaridade entre as novas tarefas e tarefas do seu histórico teriam preferência nesta ordenação. Outro fato importante revelado pelos experimentos foi que ao aumentar o indicador da medida de similaridade, a precisão das recomendações aumenta, porém, a cobertura destas recomendações diminui, não sendo consistente utilizar indicadores muito altos para esta medida.

Por fim, conclui-se que os objetivos deste trabalho foram atendidos, uma vez que foi possível gerar recomendações personalizadas para usuários da plataforma TopCoder, com uma boa taxa de precisão e cobertura, utilizando modelos de análise de similaridade de tarefas e modelo de avaliação do histórico do usuário conforme proposto inicialmente. Entende-se que mesmo com a limitação de não ser possível realizar as recomendações em tempo real devido a integração necessária com o TopCoder, a avaliação sendo realizada com dados históricos extraídos da plataforma através do detalhamento dos modelos e do algoritmo utilizado, é demonstrada a possibilidade de gerar recomendações em tempo real para estes usuários. A escolha da filtragem baseada por conteúdo, mostrou-se adequada pois dado o contexto, pode observar-se que as vantagens citadas na literatura foram refletidas nas recomendações geradas pelo RS-Crowd, pois elas são independentes por não ser comparados perfis de usuários, bem como a existência da possibilidade

de recomendar novos itens sem a necessidade de ter uma avaliação do usuário sobre estes itens, uma vez que as recomendações são geradas a partir do histórico do usuário. Algumas limitações desta técnica como a superespecialização que trata da dificuldade em recomendar itens com serendipidade, ou seja, que surpreenda o usuário e a partida fria que trata da dificuldade de gerar recomendações para usuários sem histórico, também foram identificadas na abordagem proposta porém a mitigação destas limitações não foram objeto deste trabalho.

Desta forma, este trabalho têm as seguintes contribuições constatadas: (i) um modelo de similaridade entre tarefas, com medidas individuais por atributo, finalizando com somatório ponderado entre todas métricas; (ii) um modelo de avaliação de histórico do usuário com métricas relacionadas ao seu desempenho quanto a submissões em tarefas; (iii) um algoritmo para buscar tarefas a serem recomendadas aplicando os modelos mencionados; (iv) o sistema de recomendação de tarefas com a sugestão de parametrização inicial quanto a número de tarefas recomendadas e tipo de ordenação a ser utilizado.

Uma outra contribuição que está relacionada a este trabalho, foi o artigo *CrowdRec: Desenvolvimento de um protótipo de sistema de recomendação para plataformas de crowdsourcing utilizando Google Venture Design Sprint* publicado na SBSI em 2018, durante o mestrado. Este artigo propôs um protótipo de interface inicial para o Sistema de Recomendação e seus resultados podem servir como base para a construção de uma aplicação voltada para recomendação de tarefas para os usuários das plataformas de crowdsourcing.

No decorrer do trabalho, foram identificadas algumas possibilidades de aprimoramento e maior cobertura dos estudos para trabalhos futuros, sendo alguns deles elencados a seguir:

- Avaliar a possibilidade de adição de mais atributos relacionados ao histórico de participação do usuário, como número de tarefas em que o usuário está registrado simultaneamente. Isto pode afetar seu desempenho, pois dependendo do número de tarefas inviabiliza uma submissão qualificada;
- Utilizar o atributo SR (*Submission Rate*) para recomendar tarefas diferenciadas a usuários que possuam uma boa taxa geral de submissão, ou seja, usuários que estejam submetendo tarefas com frequência na plataforma;

- Diferenciar recomendações para usuários que somente registraram e não submeteram tarefas, realizando ainda novos experimentos com este critério e avaliando o comportamento de diferentes parametrizações quanto a número de tarefas recomendadas, bem como os critérios de ordenação a serem utilizados;
- Realizar os experimentos de forma online e mais abrangente, como incluir as demais categorias existentes no TopCoder como “Design” e “Data Science”;
- Avaliar aderência da abordagem de recomendação proposta em outras plataformas de crowdsourcing.

Diante disso, por ser um desafio inerente ao contexto de software crowdsourcing e com objetivo de propor ferramentas que auxiliem o usuário para tomada de decisão, vislumbra-se a possibilidade de evolução e aprimoramento deste estudo com objetivo de aumentar o engajamento e participação de usuários que utilizam as plataformas de desenvolvimento de software crowdsourcing.

REFERÊNCIAS

[1] BRANQUINHO, C. L. Da S. Crowdsourcing: uma forma de inovação aberta. CETEM/MCTI. 2016.

[2] BEGEL, A.; HERBSLEB, J. D.; STOREY, M.-A. The Future of Collaborative Software Development. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work Companion*. 2012. p. 17–18.

[3] MAO, K. *et al.* A survey of the use of crowdsourcing in software engineering. *RN*. vol. 15, no. 01. 2015.

[4] HOWE, J. The rise of crowdsourcing. *Wired Mag.* vol. 14, no. 6. p. 1–4. 2006.

[5] MAO, K. *et al.* Developer recommendation for crowdsourced software development tasks. *Proc. - 9th IEEE Int. Symp. Serv. Syst. Eng. IEEE SOSE 2015*. vol. 30. p. 347–356. 2015.

[6] YANG, Y. *et al.* Who Should Take This Task ? – Dynamic Decision Support for Crowd Workers. *Esem 2016*. vol. 5777. 2016.

[7] ARCHAK, N. Money, glory and cheap talk: analyzing strategic behavior of contestants in simultaneous crowdsourcing contests on TopCoder. com. *Proc. 19th Int. Conf. World Wide Web*. p. 21–30. 2010.

[8] YUEN, M.-C.; KING, I.; LEUNG, K.-S. Task Recommendation in Crowdsourcing Systems. *Proceedings of the First International Workshop on Crowdsourcing and Data Mining*. 2012. p. 22–26.

[9] CHILTON, L. B. *et al.* Task Search in a Human Computation Market. *Proceedings of the ACM SIGKDD Workshop on Human Computation*. 2010. p. 1–9.

[10] YUEN, H.; KING, I.; DEPARTAMENTO, K. L. Recomendação tarefa em Sistemas de Crowdsourcing. 2012.

[11] DAVIS, F. D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* p. 319–340. 1989.

[12] ZANATTA, A. L.; OTHERS. Recomendações para trabalhadores na multidão superarem barreiras em projetos de software crowdsourcing. 2018.

[13] HOWE, J.; ARAUJO, A. M. *Poder Das Multidoes, O*. Elsevier Brasil, 2009.

[14] BRABHAM, D. C. Crowdsourcing. *Int. Encycl. Organ. Commun.* p. 1–6. 2017.

[15] LI, Z.; HONGJUAN, Z. Research of crowdsourcing model based on case study. *Icsssm11*. 2011. p. 1–5.

[16] NOVECK, B. S. *Wiki Government*. Brookings Institution Press, 2009.

[17] KATMADA, A.; SATSIOU, A.; KOMPATSIARIS, I. Incentive mechanisms for crowdsourcing platforms. *International Conference on Internet Science*. 2016. p. 3–18.

[18] WU, W.; TSAI, W.-T.; LI, W. An evaluation framework for software crowdsourcing. *Front. Comput. Sci.* vol. 7, no. 5. p. 694–709. 2013.

[19] TSAI, W.-T.; WU, W.; HUHNS, M. N. Cloud-based software crowdsourcing. *IEEE Internet Comput.* vol. 18, no. 3. p. 78–83. 2014.

[20] SCHENK, E.; GUITTARD, C. Towards a characterization of crowdsourcing practices. *J. Innov. Econ. Manag.* no. 1. p. 93–107. 2011.

[21] LAKHANI, K.; GARVIN, D.; LONSTEIN, E. TopCoder (A): Developing Software through Crowdsourcing. jan. 2010.

[22] REATEGUI, E. B.; CÉSAR, S.; SANTOS, F. Personalização de Páginas Web através dos Sistemas de Recomendação. no. December 2014.

[23] ALVAREZ, E. B. *et al.* Os Sistemas de Recomendação, Arquitetura da Informação e a Encontrabilidade da Informação TT - Recommendation Systems, Information Architecture, and Findability. *Transinformação*. vol. 28, no. 3. p. 275–286. 2016.

[24] RASSWEILER FILHO, J. R. Aprendizado Neural De Representação De Conteúdo Para Sistema De Recomendação De Filmes. p. 65. 2017.

[25] RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. in *Recommender systems handbook*. Springer. 2011. p. 1–35.

[26] GOLDBERG, D. *et al.* Using collaborative filtering to weave an information TAPESTRY. *Commun. ACM*. vol. 35. p. 61–70. 1992.

[27] MANBER, U.; PATEL, A.; ROBISON, J. Experience with personalization of Yahoo! *Commun. ACM*. vol. 43, no. 8. p. 35–39. 2000.

[28] BURKE, R. Hybrid recommender systems: Survey and experiments. *User Model. User-adapt. Interact.* vol. 12, no. 4. p. 331–370. 2002.

[29] REATEGUI, E. B.; CAZELLA, S. C. Sistemas de recomendaçao. *XXV Congresso da Sociedade Brasileira de Computação*. 2005. p. 306–348.

[30] ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible

extensions. *Knowl. Data Eng. IEEE Trans.* vol. 17. p. 734–749. 2005.

[31] BOBADILLA, J. *et al.* Recommender systems survey. *Knowledge-based Syst.* vol. 46. p. 109–132. 2013.

[32] AMATRIAIN, X. *et al.* Data Mining Methods for Recommender Systems BT - Recommender Systems Handbook. F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston, MA: Springer US. 2011. p. 39–71.

[33] HERLOCKER, J. Understanding and improving automated collaborative filtering systems /. 2018.

[34] CAZELLA, S. C. *et al.* Desenvolvendo um Sistema de Recomendação de Objetos de Aprendizagem baseado em Competências para a Educação: relato de experiências. *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. 2012. vol. 23, no. 1.

[35] CAZELLA, S. C.; DRUMM, J. V.; BARBOSA, J. L. V. Um serviço para recomendação de artigos científicos baseado em filtragem de conteúdo aplicado a dispositivos móveis. *RENOTE - Novas Tecnol. na Educ.* vol. 8, no. 3. p. 1–11. 2010.

[36] LOPS, P.; DE GEMMIS, M.; SEMERARO, G. Content-based Recommender Systems: State of the Art and Trends. in *Recommender Systems Handbook*. 2011. p. 73–105.

[37] BARTH, I. J. Modelando o perfil do usuário para a construção de sistemas de recomendação: um estudo teórico e estado da arte. *Rev. Sist. Informação da FSMA*. vol. 6. p. 59–71. 2010.

[38] ROLIM, V. *et al. Um Estudo Sobre Sistemas de Recomendação de Recursos Educacionais*. 2017.

[39] VIVIAN, G. R. Recomendação de carreira de pesquisadores :uma abordagem baseada em personalização, similaridade de perfil e reputação. p. 98. 2017.

[40] BARBOSA, C. E. M. Estudo de técnicas de filtragem h{\'\i}brida em sistemas de recomendação de produtos. *Monogr. Cent. Informática, Ciência da Comput. UFPE*. 2014.

[41] JÚNIOR, S. M. Da S. Recomendação de Conteúdo Baseada em Informações Semânticas Extraídas de Bases de Conhecimento. p. 1–102. 2017.

[42] GUNAWARDANA, A.; SHANI, G. A survey of accuracy evaluation metrics of recommendation tasks. *J. Mach. Learn. Res.* vol. 10, no. Dec. p. 2935–2962. 2009.

[43] HUANG, W. *et al.* Recommending Citations: Translating Papers into References. *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. 2012. p. 1910–1914.

[44] GE, M.; DELGADO-BATTENFELD, C.; JANNACH, D. Beyond accuracy: evaluating recommender systems by coverage and serendipity. *Proceedings of the fourth ACM conference on Recommender systems*. 2010. p. 257–260.

[45] STASIU, R. K. Avaliação da Qualidade de Funções de Similaridade no Contexto de Consultas por Abrangência. p. 115. 2007.

[46] SCHAFER, J. Ben *et al.* Collaborative filtering recommender systems. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. vol. 4321 LNCS, no. 1. p. 291–324. 2007.