CBIE 2023
CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO
Uma escola para o futuro: Tecnologia e conectividade a serviço da educação

Aprimorando a experiência de aprendizado em ambientes *online* massivos: o papel dos sistemas de recomendação

Wilson M. Sanches[1], Fabiana Z. Ferreira[2], Paulo J. D. O. Evald[3], André P. Vargas[1], Jean L. Bez[4], Silvia S. da C. Botelho[1]

[1]Programa de Pós-Graduação em Modelagem Computacional
Universidade Federal do Rio Grande – Rio Grande – RS – Brasil

[2]Programa de Pós-Graduação em Educação em Ciências
Universidade Federal do Rio Grande – Rio Grande – RS – Brasil

[3]Centro de Engenharias
Universidade Federal de Pelotas – Pelotas – RS – Brasil

[4]Universidade Regional Integrada do Alto Uruguai e das Missões
Erechim – RS – Brasil

{sancheswm78,fabinhazaffalon, paulo.evald}@gmail.com,

{prisco.c3,silviacb.botelho}@gmail.com,

bez@urionlinejudge.com.br

Abstract. *Massive environments have been used in education because they provide a large collection of exercises for students. In this paper, a recommendation system is proposed for use as an online judge in massive learning environments. The proposed method recommends problems considering the user's skills and motivation; that is, it recommends exercises solved by other users with similar skills and motivation. For this purpose, the traditional collaborative filtering method with a similarity measure adapted to the current domain was adopted. The effects of matrix settings using accuracy and recall metrics are analyzed.*

Resumo. *O uso de ambiente massivo tem sido utilizado na educação, pois neste ambiente provê uma grande coleção de exercícios aos estudantes. Neste trabalho é proposto um sistema de recomendação para uso em juiz online nos ambientes de ensino massivo. O método proposto recomenda problemas considerando as habilidades e motivação do usuário, ou seja, recomenda exercícios resolvidos por outros usuários com habilidades e motivação semelhantes. Para tal, o método tradicional de filtragem colaborativa com uma medida de similaridade adaptada ao domínio atual foi adotado. Os efeitos das configurações da matriz usando métricas de Precisão e Recall são analisados.*

1. Introdução

No contexto do ensino de programação, o uso de juiz online também conhecido como ambiente massivo online para avaliar e pontuar o desempenho dos estudantes em exercícios de programação ou lógica, traz várias vantagens no contexto educacional [Xiao et al. 2018]. tais como a avaliação objetiva, *feedback* instantâneo e prática repetitiva. No

entanto, como há uma grande quantidade de questões disponíveis nos juízes online, os estudantes podem ter dificuldade de escolher problemas mais adequados às suas habilidades [de Oliveira et al. 2022]. Para contornar esse problema, muitos desses juízes online incorporam sistemas de recomendação (SR), que sugerem exercícios aos usuários com base em seu desempenho e perfil [de Oliveira et al. 2022].

Resnick e Varian (1997) ressaltam a importância dos SRs em ambientes de aprendizagem. Eles enfatizam que esses sistemas podem desempenhar um papel fundamental ao recomendar recursos, materiais de estudo, atividades e exercícios personalizados para os alunos, levando em consideração suas preferências individuais e necessidades específicas. Além disso, Regueras e colaboradores (2009) conduziram uma pesquisa que explorou a relação entre o desempenho dos usuários em ambientes online e sua motivação. Já Nadolski et al. (2009) discutiu a regra pedagógica que sustenta que "o esforço dos alunos aumenta quando eles estão mais motivados", explorando como os SRs podem ser aplicados de maneira estratégica para aumentar a motivação dos estudantes. Yera e colaboradores (2017) exploraram a aplicação de SRs em ambientes de aprendizagem adaptativos, nos quais o sistema de recomendação é integrado a um ambiente que se ajusta automaticamente às características e desempenho do aluno. Ramirez et al. apresentaram um modelo de personalização inteligente para um ambiente de aprendizado massivo online que utiliza técnicas de análise de comportamento do aluno para construir um modelo individualizado do aluno, permitindo a adaptação do conteúdo, atividades e recomendações com base em seu progresso, preferências e estilo de aprendizagem.

A ausência de um sistema de recomendação em um juiz online pode trazer algumas dificuldades no progresso e engajamento dos estudantes nos ambientes virtuais com juiz online por diversos motivos, dentre eles: dificuldade em encontrar exercícios adequados ao seu nível de habilidade e conhecimento, desmotivação causada pela falta de desafio ou excesso de dificuldade dos exercícios escolhidos, perda de tempo filtrando exercícios disponíveis na plataforma e estagnação no aprendizado pela escolha de exercícios incompatíveis com seu conhecimento atual [Hew and Cheung 2014, Kop 2011].

O presente estudo tem como objetivo responder à seguinte indagação de pesquisa: "É viável recomendar problemas em um sistema Juiz Online visando o aprimoramento das habilidades de programação dos estudantes utilizando exclusivamente informações de similaridade entre pares de estudantes?". Para alcançar esse objetivo, os pesquisadores exploraram dados disponíveis no Juiz Online denominado beecrowd, com a finalidade de aplicar um sistema de recomendação colaborativo e contribuir para o aprimoramento das competências dos usuários em ambientes online de larga escala. De acordo com Souza et al. (2016), a criação de um ambiente pedagógico que permita aos alunos resolver problemas reais adaptados ao seu nível de habilidade é uma das possíveis soluções para mitigar o elevado índice de reprovações em disciplinas de programação. Nesse sentido, este trabalho busca melhorar a experiência de aprendizagem dos alunos por meio de um recomendador que propõe listas personalizadas de exercícios de programação, levando em consideração o nível de habilidade e motivação de cada usuário, contribuindo para a minimização das taxas de reprovação em disciplinas de programação, oferecendo aos estudantes uma abordagem personalizada e adaptativa para o desenvolvimento de suas competências em programação em ambientes online massivos.

Ao adotar uma abordagem com filtragem colaborativa baseada em informações

de similaridade entre pares de estudantes, o sistema de recomendação busca explorar o conhecimento coletivo gerado pelas interações e soluções apresentadas pelos estudantes no Juiz Online *beecrowd*. O modelo proposto foi testado na base dados do *beecrowd*, considerando as submissões históricas, que contém os seguintes dados: *data* e *hora*, *identificação do usuário, identificação do problema* e a *resposta*. Assim, um conjunto de dados com 19.695 de submissões realizadas na plataforma *beecrowd* por 155 usuários a 205 problemas foi construído. Um conjunto de 205 problemas foi analisado por um especialista, que identificou os caminhos de solução, associando valores de dificuldade e relevância a cada um. Em seguida, um estudo de caso foi realizado com 54 estudantes matriculados em disciplinas de programação. Os estudantes submeteram suas soluções aos problemas da plataforma *beecrowd* ao longo de 30 dias. Os dados das submissões foram tabulados e o modelo proposto foi aplicado para análise.

O sistema proposto considera tanto os problemas resolvidos quanto os não resolvidos pelos usuários. O sistema utiliza matrizes de habilidade e motivação para identificar os problemas já resolvidos e analisar o desempenho e interesse individual em cada questão. Além disso, o sistema utiliza uma matriz de similaridade específica para construir a vizinhança do usuário e atribuir pontuações aos problemas não resolvidos. Essa matriz é baseada em uma medida de similaridade de Pearson, ponderada pelo número de submissões do usuário em cada problema. Essa abordagem é diferente das abordagens tradicionais de sistemas de recomendação, que geralmente utilizam apenas a matriz de habilidade para construir a vizinhança do usuário. A inclusão da matriz de motivação permite que o sistema considere o nível de interesse dos usuários nos problemas, além do seu desempenho. Essa diferença é importante, pois pode melhorar a qualidade das recomendações, pois os usuários podem não estar interessados em resolver problemas que já resolveram, mesmo que tenham um desempenho alto nesses problemas. Além disso, os usuários podem não estar interessados em resolver problemas que sejam muito difíceis para o seu nível de habilidade.

O restante deste trabalho é organizado da seguinte forma: a Seção 2 apresenta os conceitos teóricos que fundamentam o modelo proposto, enquanto a Seção 3 esclarece a metodologia. Em seguida, a Seção 4 discute os resultados. Finalmente, a Seção 5 encerra este trabalho fundamento as conclusões deste estudo.

2. Referencial teórico

Nesta seção serão discutidos os conceitos básicos que fundamentam o método proposto.

2.1. Sistema de Recomendação

Os sistemas de recomendação são ferramentas para filtrar informações que melhor se encaixem às necessidades e preferências de um usuários [Yera and Martínez 2017]. Neste estudo, será adotado o conceito de filtragem colaborativa (FC) como uma abordagem central para aprimorar a experiência dos usuários em ambientes online. A FC é uma técnica que se baseia na troca de informações e experiências entre os usuários que possuem interesses semelhantes. Essa abordagem busca utilizar o conhecimento coletivo dos usuários para recomendar objetos que sejam relevantes e personalizados.

O sistema de FC, busca em primeiro momento por usuários com conhecimentos semelhantes ao usuário alvo que vai receber esta lista de problemas recomendados, calculando as similaridades entre ambos [Da Silva et al. 2016]. Em seguida, seleciona o grupo

de usuários de maior similaridade a um usuário alvo. Em nosso estudo adotamos o grau de similaridade de 60%, criando uma matriz de informações dos problemas dos usuários. Através dessa matriz é criada uma "vizinhança" para cada usuário. Assim, um usuário poderá receber problemas que não foram resolvidos por ele, mas que foram resolvidos por seus vizinhos [Isinkaye et al. 2015]. Porém, os sistemas de FC possuem algumas limitações no aspecto de sua aplicação [Reategui and Cazella 2005, Su and Khoshgoftaar 2009], que devem ser devidamente tratados, tais como:

- *Cold start*: pequeno ou inexistente número de informações iniciais, o que dificulta a aplicação do FC para novos usuários ou novos itens;
- Esparsialidade: falta de avaliações dos itens pelos usuários;
- Escalabilidade: à medida que o número de usuários e problemas cresce, os algoritmos de FC podem enfrentar desafios em termos de tempo de processamento e uso de recursos computacionais.

2.2. Juízes *Online*

Juízes *online* são plataformas voltadas para a avaliação automática de códigos-fontes de linguagens de programação. Basicamente, os juízes *online* possuem um padrão de entrada de dados que são processados e formatados em uma saída padronizada, permitindo a avaliação automática [Chaves et al. 2013]. No contexto brasileiro, o *beecrowd* é uma plataforma amplamente utilizada que oferece uma extensa coleção de problemas de programação para os usuários. Nessa plataforma, os usuários têm a liberdade de selecionar os problemas que desejam resolver, bem como a linguagem de programação com a qual desejam abordar a solução. Entretanto os *feedbacks* providos por esta plataforma referem-se apenas a corretude do problema ou problemas de formatação do código-fonte em relação à linguagem de programação adotada.

2.3. *Logistic regression*

Os modelos preditivos, ou classificadores, inserem um vetor de valores de recursos discretos ou contínuos, cujo objetivo é identificar a classe de dados [Domingos 2012]. Nessa abordagem de aprendizagem supervisionada, um conjunto de dados previamente rotulados, como um conjunto de treinamento com N pares de entrada e saídas $(x_1, y_1), (x_2, y_2)...(x_n, y_n)$, onde y_j é gerado por uma função desconhecida $y = f(x)$ é utilizado. Com isso, o algoritmo busca descobrir uma função h que aproxime da função f [Russell and Norvig 2004]. Nesse trabalho, será utilizado o aprendizado supervisionado *Logistic regression*, que é um algoritmo capaz de realizar tarefas de classificação binária, prevendo a probabilidade de um resultado. Tal técnica expressa a probabilidade como

$$P = \frac{e^{a+bX}}{1 + e^{a+bX}}, \tag{1}$$

onde P é a probabilidade e e é a base do logaritmo natural (cerca de 2,718). Além disso, os parâmetros a e b são o intercepto (termo de polarização) e um coeficiente ajustado de acordo com os dados de entrada, respectivamente, que são aprendidos continuamente.

2.4. *Singular value decomposition*

Um dos desafios nos SRs é a escalabilidade dos algoritmos, o que pode ser mitigado com a redução da dimensionalidade dos dados [Bobadilla et al. 2013]. A *Singular value decomposition* (SVD) é uma técnica que realiza a fatoração de matrizes, o que é adequado

para processar grandes quantidades de dados esparsos. Além disso, a SVD possui a capacidade de pesquisar dezenas de milhares de potenciais vizinhos em tempo real e produzir recomendações de alta qualidade.

O modelo matemático para a SVD pode ser descrito da seguinte forma: Dada uma matriz A de dimensão m x n, onde m representa o número de linhas e n o número de colunas, a SVD busca fatorar A como $A = U\Sigma V^T$, onde U é uma matriz m x m unitária, Σ é uma matriz m x n diagonal com elementos não negativos (valores singulares) na diagonal principal e V é uma matriz n x n unitária. Os valores singulares de A são calculados a partir dos autovalores da matriz $A^T * A$, que é uma matriz simétrica e semi-definida positiva. A partir desses autovalores, é possível obter a matriz Σ. As matrizes U e V podem ser obtidas a partir das colunas dos autovalores e autovetores de AA^T e A^TA, respectivamente.

2.5. Similaridade de Cosseno

A similaridade entre dois problemas p_1 e p_2, considerados vetores em um espaço de dimensão n, pode ser computada através do cosseno do ângulo que é formado entre eles [Ricci et al. 2011]. Em nosso sistemas de recomendação em ambientes massivos, os usuários e os problemas são os vetores em um espaço de dimensão n. A similaridade entre eles é calculada utilizando a seguinte expressão matemática

$$\text{similaridade}(\vec{u}, \vec{p}) = \cos(\vec{u}, \vec{p}) = \frac{\vec{u} \cdot \vec{p}}{\|\vec{u}\| * \|\vec{p}\|}, \tag{2}$$

onde $\vec{u}$ e $\vec{p}$ são os vetores que representam, o usuário e o problema, respectivamente.

3. Metodologia

Nesta seção, descrevemos a metodologia do sistema recomendador proposto para ambientes massivos online. O sistema é baseado no comportamento dos usuários ao resolver ou não problemas recomendados. A entrada do sistema é uma quádrupla (*t,u,p,r*), composta pela data e hora da submissão da solução do problema (*t*), o usuário (*u*), o problema (*p*) e a resposta da solução submetida (*r*). A abordagem consiste em recomendar ao usuário x os problemas resolvidos pelos outros usuários com comportamentos semelhantes, considerando tanto os problemas resolvidos como os não resolvidos. Para isso, utiliza-se uma medida de similaridade específica para construir uma vizinhança do usuário x e atribuir pontuações aos problemas não resolvidos, com base nos usuários vizinhos que conseguiram resolvê-los.

A matriz de habilidade, representada por H, é uma matriz binária que indica se um usuário resolveu ou não um problema, onde $H_{ij} = 1$ se o usuário i resolveu o problema j e $H_{ij} = 0$ caso contrário. A matriz de motivação, representada por M, contém a quantidade de submissões do usuário em cada problema, onde M_{ij} é o número de submissões do usuário i no problema j. Essas matrizes permitem a identificação dos problemas já resolvidos pelos usuários e a análise do desempenho individual em cada questão.

A medida de similaridade de cosseno, representada por S, é calculada entre dois usuários x e y com base em suas respectivas habilidades, denotadas por H_x e H_y. Essa medida de similaridade é obtida pela divisão do produto escalar entre H_x e H_y pela multiplicação das normas Euclidianas dessas linhas, conforme a fórmula:

$$S(x, y) = \frac{H_x \cdot H_y}{|H_x| \cdot |H_y|}. \quad (3)$$

Após calcular a similaridade entre usuários com base em suas habilidades, a similaridade entre os usuários também é calculada considerando a matriz de motivação, representada por M. Essa matriz contém informações sobre a quantidade de submissões de cada usuário em relação a cada problema. Utiliza-se a mesma fórmula de similaridade de cosseno para calcular a similaridade entre os usuários com base em suas motivações.

Além disso, é realizada uma análise de similaridade entre problemas e usuários, considerando a matriz de habilidade H. Essa análise visa identificar a compatibilidade entre os problemas não resolvidos pelos usuários e os problemas resolvidos pelos usuários vizinhos com comportamentos semelhantes. Novamente, a fórmula de similaridade de cosseno é utilizada para calcular a similaridade entre os problemas e os usuários, levando em consideração as colunas correspondentes na matriz de habilidade.

Dessa forma, a medida de similaridade de cosseno desempenha um papel fundamental na identificação das relações de similaridade entre usuários e problemas, permitindo a recomendação de problemas adequados com base nos comportamentos e nas habilidades dos usuários..

A utilização de sistemas de recomendação é de suma importância em ambientes de aprendizado online, onde a quantidade de informações disponíveis pode ser avassaladora. As matrizes de habilidade e motivação proporcionam uma base sólida para a construção desses sistemas, permitindo uma seleção mais precisa de atividades personalizadas para cada usuário, melhorando assim a experiência de aprendizado.

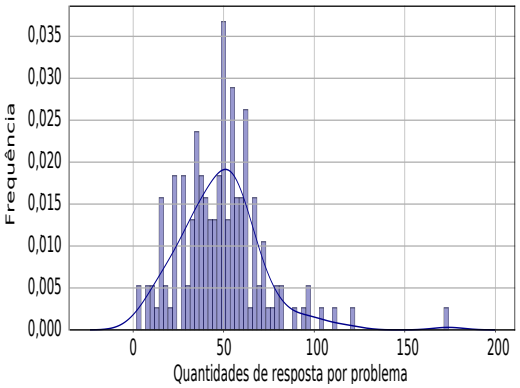
4. Resultados e Discussões

Um experimento foi conduzido utilizando os dados do *beecrowd*, considerando as submissões históricas. Os dados contidos na base são: *data* e *hora*, *identificação do usuário, identificação do problema* e a *resposta*. Assim, um conjunto de dados com 19.695 de submissões realizadas na plataforma *beecrowd* por 155 usuários a 205 problemas foi construído. Os problemas do *beecrowd* tem seus identificadores etiquetados da seguinte maneira: primeiro problema (id_1001), segundo problema (id_1002), e assim consecutivamente. Os problemas selecionados não seguem uma ordem contínua. Do total de problemas na base *beecrowd* foram extraídos apenas 205 de forma aleatória. A similaridade entre os problemas selecionados foi de 60%, com base na similaridade de cosseno. Ressalta-se que a a similaridade leva em conta as resoluções submetidas por todos os usuários para os problemas em questão. Com essa informação, o recomendador sugeriu 50 problemas similares com base na matriz de habilidades e mais 50 problemas com base na matriz de motivação. A quantidade de problemas sugeridos está relacionada com a similaridade calculada, ou seja, se a similaridade for superior a 0,6, então a quantidade de problemas sugerida pelo recomendador seria menor que 50.

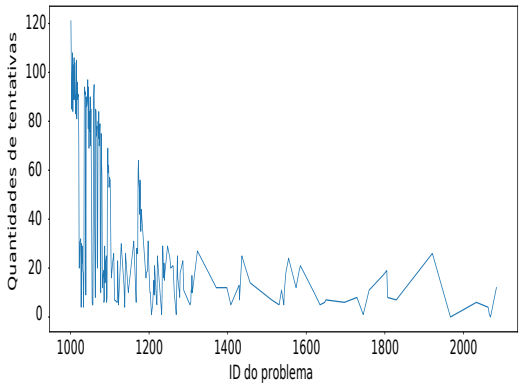
As matrizes de habilidade e de motivação são esparsas. Portanto, foi necessário fazer uma redução da dimensão. Para tal, aplicou-se a técnica SVD, que reduziu a matriz de 205x155 para 205x10, não havendo prejuízo de informação, uma vez que a matriz é decomposta em autovalores e autovetores. Para realizar a predição dos problemas a se-

rem recomendados, e assim obter a matriz confusão, foi implementada a técnica *Logistic regression* [Kleinbaum et al. 2002].

A Figura 1(a) mostra que aproximadamente 3% dos usuários responderam corretamente 48 exercícios (em média), com um desvio padrão de 23,40. Este desvio padrão elevado, tanto para o número de acertos quanto para o número de tentativas, indica uma grande variação nos resultados obtidos pelos alunos, o que pode sugerir que a plataforma ou a auto-recomendação não está oferecendo um nível adequado de suporte e orientação para os estudantes, o que indica a necessidade do recomendador. A análise do desempenho dos usuários em relação aos exercícios pode fornecer informações valiosas sobre a eficácia do sistema de ensino e aprendizagem. Nesse sentido, a Figura 1(b) revela que a quantidade de tentativas realizadas pelos usuários para cada problema pode ser um indicativo importante da dificuldade e do interesse dos alunos. O estudo revelou que, em média, cada exercício teve 36,7 tentativas de resolução. O alto número de tentativas por exercício indica que os alunos podem estar gastando muito tempo tentando resolver cada questão, o que pode ser frustrante e desmotivador, com desvio padrão de 33,0.



((a)) Distribuição dos usuários por resposta certa aos problemas.



((b)) Distribuição por tentativas dos problemas por usuário.

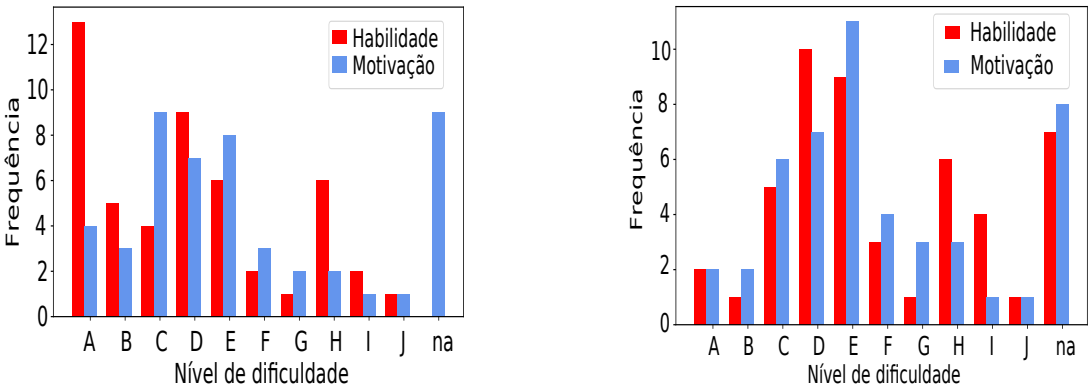
Figura 1. Distribuição de usuários por resposta certa aos problemas e quantidade de tentativas por usuário.

Para compreender melhor os padrões de habilidade e motivação dos usuários foram construídas matrizes que indicam se cada aluno conseguiu ou não resolver corretamente cada problema, bem como o número de tentativas realizadas até a solução de cada questão. Essas informações permitem que um sistema de recomendação possa identificar quais são as habilidades e dificuldades de cada usuário e fornecer sugestões de exercícios que possam melhorar seu desempenho e engajamento no aprendizado. Dois problemas específicos foram selecionados de forma totalmente aleatória para uma análise mais detalhada: o problema 1110 e o problema 1921. Com esses dados, é possível criar estratégias mais personalizadas de ensino e aprendizagem, visando uma melhor adaptação às necessidades individuais dos alunos.

4.1. Compatibilidade entre as matrizes de habilidade e motivação

A complexidade desse processo se intensifica quando consideram-se os problemas que apresentam pouca informação disponível, ou seja, problemas que tiveram baixa tentativa de resolução e poucos alunos escolheram esse problema, como é o caso do problema

1921, localizado na cauda do gráfico da Figura 1(b). Isso dificulta o cálculo de similaridade entre os pares, o que pode resultar em uma baixa compatibilidade entre as listas de recomendação geradas pela matriz de habilidade e pela matriz de motivação. Por outro lado, para problemas com maior número de informações disponíveis, como o problema 1110, localizado em uma região mais populosa de informações na Figura 1(b), ocorrência oposta do problema 1921, a eficiência da FC se mostra mais alta, resultando em um aumento na similaridade entre as listas recomendadas pelas matrizes de habilidade e motivação.



((a)) Problema 1921. **((b)) Problema 1110.**

Figura 2. Distribuição dos problemas sugeridos pelo recomendar em em relação ao grau de dificuldade.

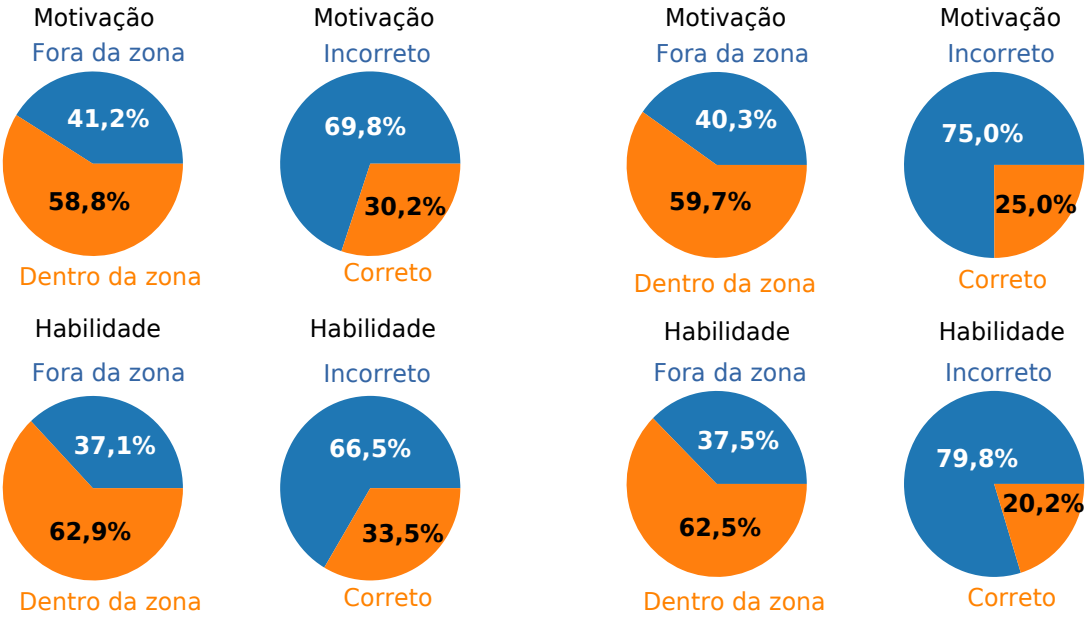
4.2. Nível de dificuldade das recomendações

Os problemas considerados nesse trabalho estão classificados no *beecrowd* de acordo com seu grau de dificuldade, sendo o nível *A* o mais fácil e o nível *J* como o mais difícil. O nível *na* representa os problemas que não foram classificados. Os problemas 1110 e 1921 são ambos da categoria *C*. Nas Figuras 2(b) e 2(a) são mostradas as distribuições dos problemas recomendados, considerando habilidade e motivação, em relação ao grau de dificuldade, quando o recomendador considera os problemas 1110 e 1921 como problema-base para sugerir novos problemas, respectivamente. Desses gráficos, é notável que os problemas indicados pela matriz de habilidades e pela matriz de motivação tem uma distribuição de dificuldade similar.

4.3. Nível de similaridade

Na base de dados extraída do *beecrowd* havia um grupo de 19 usuários que tinham se auto-recomendado o problema 1110. Para verificar a similaridade dos problemas sugeridos pelo recomendador com os problemas auto-sugeridos pelos usuários, uma comparação das listas de exercícios foi realizada. A Figura 3(a) apresenta os resultados da comparação para o problema 1110, onde observa-se que a lista de recomendação oriunda da motivação tem 58,8% dos problemas recomendados dentro da zona de recomendação criada pelos próprios usuários. Além disso, dos problemas que estavam dentro da zona de recomendação, apenas 30,2% dos problemas foram resolvidos de forma correta. Por outro lado, a lista de exercícios sugeridos pela matriz de habilidade, obteve 62,9% de similaridade com os problemas recomendados pelo próprio usuário, cuja a taxa de acertos foi de 33,5%. De modo similar, a Figura 3(b) apresenta a comparação quando considerado o

problema 1921 como base de recomendação. Note que dos problemas sugeridos pelo matriz de motivação, 62,5% dos problemas também constam na zona de recomendação dos usuários, sendo a taxa de acerto de apenas 25%. Por outro lado, quando os problemas sugeridos pela matriz de habilidade tem uma similaridade com as listas auto-recomendadas dos usuários de 62,5%, com uma taxa de acerto de 20,2%.



((a)) Problema 1110. ((b)) Problema 1921.

Figura 3. Similaridade entre listas de exercícios sugeridas pelo recomendador e auto-recomendada.

5. Conclusão

Neste trabalho buscou apresentar um sistema recomendação com aplicação para ambientes de aprendizado massivo online. Através da aplicação desse sistema na plataforma beecrowd, foram obtidos resultados que indicam que um sistema de recomendação para problemas de programação pode ser uma ferramenta valiosa nesses contextos educacionais. O sistema proposto utiliza uma abordagem baseada em filtragem colaborativa e demonstrou a capacidade de sugerir problemas que estejam alinhados com o nível de conhecimento do estudante, mesmo quando há pouca informação disponível sobre o perfil do usuário.

A análise dos resultados revelou algumas falhas recorrentes por parte dos alunos, como a falta de clareza sobre seus próprios conhecimentos, dificuldades na resolução de problemas semelhantes e uma seleção limitada e pouco diversificada de problemas. Diante dessas constatações, é sugerido que outras variáveis, como os tipos de erros cometidos pelos estudantes e problemas de conectividade, sejam incorporadas à matriz de habilidade e motivação do sistema de recomendação. Essa ampliação visa aperfeiçoar o desempenho do sistema, e será abordada em trabalhos futuros.

Essas descobertas evidenciam a importância contínua do desenvolvimento e aprimoramento de sistemas de recomendação para maximizar a eficácia do processo de aprendizagem em ambientes de educação massiva online. Ao considerar fatores adicionais e

ajustar o sistema de recomendação de acordo, espera-se proporcionar aos alunos uma experiência de aprendizagem mais personalizada e eficiente.

Referências

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, 46:109–132.

Chaves, J. O., Castro, A., Lima, R., Lima, M. V., and Ferreira, K. H. A. (2013). Uma ferramenta baseada em juízes online para apoio às atividades de programação de computadores no moodle. *Revista Novas Tecnologias na Educação*, 11(3):1–10.

Da Silva, E. Q., Camilo-Junior, C. G., Pascoal, L. M. L., and Rosa, T. C. (2016). An evolutionary approach for combining results of recommender systems techniques based on collaborative filtering. *Expert Systems with Applications*, 53:204–218.

de Oliveira, M. G., da Silva, M. F., and Rodrigues, C. B. (2022). Curso híbrido baseado em moocs de lovelace e oficinas presenciais para aprendizagem ativa e nobre de pensamento computacional e programação. In *Anais do XXVIII Workshop de Informática na Escola*, pages 179–188. SBC.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

Hew, K. F. and Cheung, W. S. (2014). Students' and instructors' use of massive open online courses (moocs): Motivations and challenges. *Educational research review*, 12:45–58.

Isinkaye, F. O., Folajimi, Y. O., and Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, 16(3):261–273.

Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., and Klein, M. (2002). *Logistic regression*. Springer.

Kop, R. (2011). The challenges to connectivist learning on open online networks: Learning experiences during a massive open online course. *International Review of Research in Open and Distributed Learning*, 12(3):19–38.

Nadolski, R. J., van den Berg, B., Berlanga, A. J., Drachsler, H., Hummel, H. G., Koper, R., and Sloep, P. B. (2009). Simulating light-weight personalised recommender systems in learning networks: A case for pedagogy-oriented and rating-based hybrid recommendation strategies. *Journal of Artificial Societies and Social Simulation*, 12(1):1–4.

Ramírez Luelmo, S. I., El Mawas, N., and Heutte, J. (2021). Learner models for mooc in a lifelong learning context: A systematic literature review. In *Computer Supported Education: 12th International Conference, CSEDU 2020, Virtual Event, May 2–4, 2020, Revised Selected Papers 12*, pages 392–415. Springer.

Reategui, E. B. and Cazella, S. C. (2005). Sistemas de recomendação. In *XXV Congresso da Sociedade Brasileira de Computação*, pages 306–348. SBC.

Regueras, L. M., Verdu, E., Munoz, M. F., Perez, M. A., De Castro, J. P., and Verdu, M. J. (2009). Effects of competitive e-learning tools on higher education students: A case study. *IEEE Transactions on Education*, 52(2):279–285.

Resnick, P. and Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, 40(3):56–58.

Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender Systems Handbook*, pages 1–35. Springer.

Russell, S. J. and Norvig, P. (2004). *Inteligencia Artificial: un enfoque moderno*. Pearson Hall.

Souza, D. M., da Silva Batista, M. H., and Barbosa, E. F. (2016). Problemas e dificuldades no ensino de programação: Um mapeamento sistemático. *Revista Brasileira de Informática na Educação*, 24(1):39.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–20.

Xiao, J., Wang, M., Jiang, B., and Li, J. (2018). A personalized recommendation system with combinational algorithm for online learning. *Journal of ambient intelligence and humanized computing*, 9:667–677.

Yera, R. and Martínez, L. (2017). A recommendation approach for programming online judges supported by data preprocessing techniques. *Applied Intelligence*, 47(2):277–290.