Análise do nível de confusão de estudantes com base no grau de dificuldade de questões de programação

Nathália R. M. dos Santos[1], Elaine H. T. Oliveira[1], David B. F. de Oliveira[1], Leandro S. G. Carvalho[1], Tanara Lauschner[1], Marcos A. P. de Lima[1], Tiago R. Kautzmann[2], Patricia A. Jaques[3]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
[2]Universidade do Vale do Rio dos Sinos (UNISINOS)
[3]Programa de Pós-Graduação em Informática – Universidade Federal do Paraná (PPGInf-UFPR)

{nathalia.santos,elaine,david,galvao,tanara,marcos.lima}@icomp.ufam.edu.br,
tkautzmann@gmail.com, patricia@inf.ufpr.br

Resumo. *Neste artigo, é descrito um estudo exploratório sobre o nível de confusão de alunos durante a aprendizagem de programação em ambientes computacionais de aprendizagem. Para tal, os dados de autorrelato de confusão e os logs das interações dos alunos com a plataforma foram coletados durante um semestre de aula. Ademais, foi estudada a relação do nível de confusão do aluno com o nível de dificuldade da tarefa, expresso pela taxa de acerto. Ao todo, foram analisados os relatos de 197 alunos e os logs de 185 questões. Nos resultados, foram identificados níveis mais altos de confusão em questões que continham estruturas condicionais e de repetição, assuntos que exigem uma lógica de programação mais aprimorada. Os resultados desse estudo são uma primeira etapa de uma pesquisa que visa a detecção automática do estado de confusão a partir das suas ações no ambiente computacional de aprendizagem, ou seja, de maneira independente dos relatos dos estudantes.*

Abstract. *In this paper, an exploratory study of students' level of confusion during programming learning in computational learning environments is described. To this end, self-reported confusion data and logs of student interactions with the platform were collected during a semester of class. Furthermore, the relationship between the student's level of confusion and the level of difficulty of the task, expressed by the hit rate, was studied. In all, the reports of 197 students and the logs of 185 questions were analyzed. In the results, higher levels of confusion were identified in questions that contained conditional and repeating structures, subjects that require more refined programming logic. The results of this study are a first stage of a research that aims at the automatic detection of the state of confusion from their actions in the computational learning environment, that is, independently of the students' reports.*

1. Introdução

Os altos índices de evasão e repetição em disciplinas de programação são um problema que atinge a maioria dos cursos de Ensino Superior em Computação [Barros et al. 2020]. Muitos podem ser os fatores que influenciam essa realidade: os estudantes podem possuir um ritmo de aprendizagem diferente, se deparar com muitas adversidades, ou, em grande

DOI: 10.5753/sbie.2022.225007

parte, se sentirem perdidos e confusos durante o aprendizado. Dessa forma, a detecção automática desses estados, e em específico do estado de confusão, por ambientes computacionais de aprendizagem, permitiria prover assistência adequada aos alunos que tenham dificuldade com disciplinas dessa área.

Na perspectiva de teorias cognitivas de emoções, como a teoria multicomponencial de emoções [Scherer 2019, Moors et al. 2013], estudos apresentam evidências de que a confusão é uma emoção que surge a partir de uma avaliação cognitiva (*appraisal*) de uma falta de correspondência entre a informação recebida pelo indivíduo e o conhecimento existente neste [Silvia 2010, Arguel et al. 2019, D'Mello and Graesser 2014].

Pensando em disciplinas introdutórias de programação, mais especificamente em atividades de codificação, um estado de confusão pode ocorrer por um ou mais motivos, como, por exemplo, quando o aluno recebe um *feedback* inesperado, quando encontra ambiguidades ou contradições no enunciado, ou até mesmo quando não tem certeza de que ações realizar para resolver um problema de codificação. A confusão é uma emoção propensa de ocorrer em contextos complexos de aprendizagem, como em programação de computadores, pois os alunos precisam entender o problema apresentado pela tarefa e mobilizar esforços cognitivos para identificar e relacionar estratégias de resolução de problemas e conhecimentos de programação [Lehman et al. 2013, Chi and Ohlsson 2005].

Este estudo também faz-se útil para docentes e monitores que podem desenvolver didáticas mais eficazes capazes de sanar o estado de confusão de seus alunos que, consequentemente, também se beneficiarão. Considerando turmas grandes, um acompanhamento pessoal é impraticável por um único professor e exigiria muito esforço seu. Logo, o uso de ferramentas que identifiquem alunos apresentando estados de confusão é de suma importância para uma abordagem imediata que vise orientar o discente quanto às ações necessárias para sair do seu estado de confusão. Além disso, quando confuso, o aluno emprega maiores esforços para resolver o desequilíbrio cognitivo em que se encontra, o que pode levar a mais aprendizagem. No entanto, se essa confusão não é resolvida, o aluno pode experimentar outros estados afetivos negativos que estão mais correlacionados com baixo desempenho e desistência, tais como frustração e tédio [Arguel et al. 2019, D'Mello and Graesser 2014]. Dessa forma, a confusão, quando bem regulada, teria um efeito positivo, visto que o aluno tiraria proveito do estado de dúvida e aprenderia muito mais com a solução da mesma.

Esta pesquisa foi realizada em um ambiente de aprendizagem que emprega um sistema de juiz on-line utilizado para aplicação e correção automática de tarefas e avaliações de codificação. Ambas são compostas por um número predefinido de questões, sendo que as avaliações ocorrem em ambiente controlado, sob supervisão e sem autorização para consultar outras fontes e suas questões são sorteadas aleatoriamente de um conjunto predeterminado pelos professores. Durante o processo de resolução, o ambiente apresenta um conjunto de questões de codificação, cada uma composta por um enunciado e alguns casos de teste. Para cada questão, o estudante deve desenvolver uma solução (código) na IDE (*Integrated Development Environment*) disponibilizada pelo juiz on-line. Cada solução elaborada por um estudante deve obrigatoriamente passar em todos os casos de testes definidos pelo professor para a questão.

Para conduzir o estudo proposto neste trabalho, foi adicionado na IDE do juiz on-line um botão que deve ser clicado pelos alunos sempre que se sentirem confusos durante a resolução das questões. Quando acionado, o botão de relato permitia a escolha do tipo de relato de confusão a ser registrado (enunciado ou código). Ademais, foram considerados nesta pesquisa dados extraídos da base de dados do juiz on-line e dados coletados a partir dos relatos de confusão feitos por 197 estudantes. A coleta de dados foi aplicada no segundo semestre de 2021 e foram coletados, ao todo, 177.134 registros gerais de interação com o sistema em 185 questões de codificação. Da mesma forma, 594 relatos de confusão foram gerados, sendo 409 relatados durante a codificação e 185 durante a compreensão do enunciado.

2. Trabalhos Relacionados

Um estado de confusão acontece quando um aluno recebe um feedback inesperado, se depara com contradições, ou quando está inseguro sobre qual ação tomar em seguida [Yang et al. 2016]. Um dos métodos utilizados para identificar e catalogar essa confusão em ambientes acadêmicos é o monitoramento dos alunos por meio de sensores, que, por exemplo, detectam oscilações na frequência cardíaca e, assim, identificam possíveis situações de estresse, tal como realizado por [Silva et al. 2019]. Tal método possibilita que análises do nível de confusão sejam feitas de acordo com as indicações de estresse dos alunos. Entretanto, a viabilidade desse método é limitada por alguns fatores, dentre eles, o custo com a aquisição de equipamentos para monitorar cada aluno individualmente. Além disso, o uso de equipamentos para esse fim tende a gerar o efeito Hawthorne [Sedgwick and Greenwood 2015], onde os indivíduos mudam ativamente seu comportamento por saberem que estão sendo monitorados, o que gera estresse e altera os parâmetros mensurados pelos equipamentos.

Métodos similares estão diretamente relacionados com o campo da Computação Afetiva (CA), a qual tem por objetivo o desenvolvimento de interfaces que automaticamente detectam e respondem às emoções dos usuários. Tais emoções podem ser divididas em emoções básicas (raiva, surpresa, felicidade, nojo, tristeza e medo) e emoções não básicas (**confusão**, tédio, frustração, engajamento e curiosidade). Entretanto, [D'Mello and Calvo 2013] citam e questionam a ênfase com que a CA computa mais as emoções básicas em detrimento das não básicas. Nessa pesquisa, foram realizados cinco estudos que apresentaram resultados nos quais as emoções não básicas foram as que, na verdade, tiveram uma maior taxa de ocorrência, com uma proporção de 5:1 em relação às emoções básicas. Com isso, a área de emoções não básicas carece de mais atenção e de pesquisas que a tenham como tema base.

Ainda acerca da pesquisa conduzida em [D'Mello and Calvo 2013], dos cinco estudos realizados, quatro utilizaram métodos que consistiam em protocolos de interação com o usuário. Por exemplo, o primeiro estudo foi feito por meio de um protocolo de emoção em voz alta, no qual sete participantes de graduação verbalizaram seus estados afetivos enquanto interagiam com um Sistema Tutor Inteligente (STI) por 90 minutos. Os três últimos estudos foram feitos usando protocolos similares entre si, onde as emoções eram monitoradas por uma tela que capturava os rostos dos participantes e avaliava suas emoções a partir de um conjunto de 14 estados. Análogo a essa pesquisa, [Calvo and D'Mello 2010] descreveu métodos semelhantes de detecção das emo-

ções, dentre eles o *Facial Action Coding System* (FACS), usado também para monitorar a movimentação facial e, assim, identificar a emoção passada pelo usuário.

Distintamente dos estudos apresentados até então, o presente artigo realiza a análise correlacional do nível de dificuldade das questões com o índice de confusão dos alunos. Tal índice deriva de análises feitas após a coleta das interações dos alunos com um STI, assim como de seus autorrelatos de confusão. Essas interações são coletadas como registros ou *logs*. Os passos até a obtenção desses *logs* começam quando o aluno interage de alguma forma com a interface gráfica ao resolver exercícios e questões, seja por cliques de mouse, submissões de códigos, cliques em uma área específica da interface, etc. Deste modo, cada interação é considerada um *log* e indica uma compreensão sobre a resolução do aluno de acordo com os eventos ocorrentes na interface gráfica. Essa forma de coleta acaba sendo mais viável, pois, além de gerar muitos dados de interação, o registro é feito somente quando o aluno interage com o sistema, adotando assim uma forma de coleta mais reservada.

A segunda abordagem usada neste trabalho foi utilizada por [Lima et al. 2020] para classificar a dificuldade de questões de programação. Para realizar essa classificação, a abordagem teve como base o "índice de facilidade" adotado pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (Inep) no Enade [INEP 2017]. Tal índice classifica as questões em cinco níveis de dificuldade com base na taxa de acerto, que são abordados mais detalhadamente na Seção 3. Ademais, outros trabalhos usam uma escala de dificuldade para a classificação de questões, como relatado em [Sheard et al. 2011], que classifica a dificuldade em três níveis (baixo, médio ou alto). Outro exemplo é o trabalho de [Santos et al. 2019], que sugeriu um método para correlacionar atributos de legibilidade extraídos do enunciado de questões com a dificuldade apresentada nelas.

3. Metodologia

Esta seção descreve como foram coletados os registros de confusão e a ferramenta utilizada para tal. Além disso, apresenta o cálculo realizado para obter a taxa de acerto das questões de programação e como isso implica na classificação de dificuldade.

3.1. Ferramenta de coleta dos dados

A coleta de dados utilizada neste trabalho foi feita utilizando o ambiente juiz on-line CodeBench[1] desenvolvido pela Universidade Federal do Amazonas (UFAM). Nesse sistema, os alunos podem se inscrever em suas respectivas turmas, para então terem acesso às listas de exercícios de programação e provas avaliativas previamente elaboradas por seus professores e disponibilizadas no ambiente. O sistema realiza a correção automática baseada em casos de testes previamente cadastrados, que são usados para testar os códigos submetidos pelos alunos e, somente caso o código passe em todos os casos, a questão é considerada correta. Durante a correção automática é feita uma comparação textual entre a resposta esperada no caso de teste e o retorno do código do estudante. Caso a solução esteja errada, é exibida uma mensagem de erro padrão. Em casos de erro de sintaxe, é exibida a mensagem do compilador indicando o tipo de erro cometido (*Syntax Error*).

[1] https://codebench.icomp.ufam.edu.br/

Para autorrelato pelos alunos da sua emoção, foi implementado um botão onde o aluno pode relatar um momento de confusão (Figura 1), que, ao ser apertado, exibe duas opções: "Durante codificação", momento de confusão durante a elaboração e construção do código de resolução, ou "Durante leitura do enunciado", momento de confusão durante a leitura e o entendimento do enunciado da questão. O aluno também tinha acesso ao horário do último relato feito em cada questão e a opção de cancelá-lo.

Além disso, após gerar um novo relato de confusão através do botão, o aluno pode cancelar o seu relato através de uma opção que é apresentada logo após o clique. Essa opção de cancelamento foi desenvolvida para os casos em que o aluno está simplesmente testando o botão, ou para os casos em que o botão é clicado sem querer.

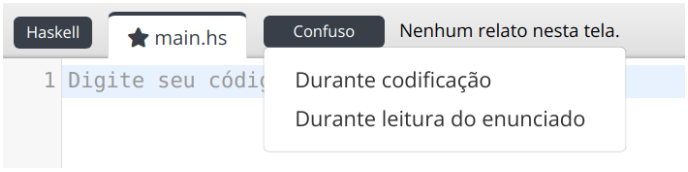

Figura 1. Botão de confusão

Além dos *logs*[2] de relatos de confusão, foram usados também *logs*[3] do *dataset*[4] do juiz on-line, como número de tentativas, testes, submissões, erros e acertos que cada aluno atingiu, juntamente com a nota obtida em cada uma das questões realizadas.

A ferramenta foi inicialmente testada em um estudo piloto com alguns alunos de cursos de exatas e, somente após ter obtido sucesso quanto à funcionalidade nos testes realizados nesse estudo, a ferramenta foi então disponibilizada para todas as disciplinas cadastradas no ambiente de aprendizagem. Além disso, devido ao modelo de ensino remoto, a coleta de dados foi feita de maneira totalmente on-line.

Ademais, duas considerações devem ser feitas. Em primeiro lugar, apesar de o botão de confusão ter sido aplicado em outras disciplinas cadastradas na plataforma, os dados usados nesta pesquisa foram coletados de alunos da disciplina de Introdução à Programação de Computadores (IPC), ministrada para cursos que não são da área de Computação (*non-majors*). Como segunda consideração, a identificação dos usuários no *dataset* é anonimizada, logo, sua identidade é preservada.

3.2. Taxa de acerto

Ao resolver um exercício, o aluno pode testar seu código antes de submetê-lo para a correção automática, a fim de se certificar se as saídas do código correspondem aos casos de exemplo fornecidos no enunciado da questão. Isso gera dois tipos de eventos nos *logs* do *dataset*: "TEST" e "SUBMISSION". Os eventos do tipo "TEST" são gerados quando o estudante executa seu código de solução no console da IDE, para que ele próprio faça alguns testes do tipo entrada-saída. Os eventos do tipo "SUBMISSION", por sua vez, são gerados quando o estudante, enfim, envia sua solução para a correção automática pelo juiz on-line. No entanto, há casos em que o usuário, mesmo quando seu código de

[2]https://codebench.icomp.ufam.edu.br/dataset/relatos_confusao.tar.gz
[3]https://codebench.icomp.ufam.edu.br/dataset/files/cb_dataset_2021_1_v1.50.tar.gz
[4]https://codebench.icomp.ufam.edu.br/dataset/

solução passa em todos os casos de teste, continua submetendo o código outras vezes. Assim, esse comportamento gera códigos de solução duplicados e também vários eventos de submissão adicionais nos *logs* do *dataset*. Se esse comportamento não fosse tratado, poderia levar a erros no cálculo da taxa de acerto das questões, contabilizando inúmeras submissões adicionais.

Logo, para evitar resultados errôneos, de x submissões feitas em uma questão só foi considerada aquela que obteve a maior porcentagem de acerto, chamada neste artigo de submissão única. Por exemplo, um aluno que fez três submissões em uma mesma questão e obteve as porcentagens 30%, 100% e 67%, respectivamente, terá como submissão única a segunda submissão feita. Dessa forma, foi considerada apenas uma submissão por aluno para cada questão.

Outra consideração relevante é o conceito de "acerto" da questão, usado somente no contexto em que a porcentagem de acerto for igual a 100%. Caso contrário, a questão é classificada como errada. Sendo assim, o número de acertos por questão foi definido como a soma das submissões únicas que obtiveram 100% de acerto. Com isso, o cálculo da taxa de acerto foi definido como a divisão do número de acertos da questão pelo total de submissões únicas nessa mesma questão.

3.3. Classificação de dificuldade

Neste trabalho, optou-se por utilizar o modelo de classificação de dificuldade adotado pelo Inep, que é dividido em cinco níveis e mapeado de acordo com a taxa de acerto das questões, conforme apresentado na Tabela 1.

Tabela 1. Discretização da taxa de acerto.

Taxa de Acerto	Classificação Inep
>0,86	Muito Fácil
0,61 a 0,85	Fácil
0,41 a 0,60	Média
0,16 a 0,40	Difícil
<0,15	Muito Difícil

Fonte: [Lima et al. 2020]

4. Resultados e Análise

4.1. Análises gerais do dataset

Como já mencionado na Seção 3.3, foram coletadas 185 questões que obtiveram pelo menos uma tentativa de resolução registrada. Nessas questões, foram observados 177.134 registros gerais de interação com a IDE, feitos por 197 estudantes. A maioria desses registros é relativa a testes de códigos, correspondendo a um total de 72%. Consequentemente, o restante corresponde às submissões registradas.

Da mesma forma, observou-se que aproximadamente 38% dos testes e submissões feitos pelos alunos apresentaram erros durante a execução do código. Como esperado, a ocorrência desses erros foi proporcionalmente menor nas submissões do que nos testes, visto que os alunos usam os testes de execução para ter certeza do funcionamento do código. Em relação às notas dos alunos, a maioria dos registros ou obteve nota zero, ou nota cem, como demonstrado no gráfico da Figura 2.

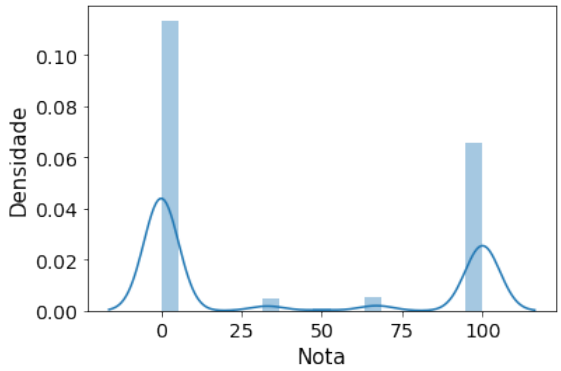

Figura 2. Distribuição das notas registradas

4.2. Nível de confusão e discussões

Na coleta dos *logs* de confusão, foram obtidos 594 relatos, sendo 409 (68,9%) do tipo codificação e 185 (31,1%) do tipo enunciado. Assim, como mais de dois terços dos relatos foram do tipo codificação, constata-se que os alunos, em sua maioria, se sentiram mais confusos durante o desenvolvimento do código de resposta. Da mesma forma, observou-se que houve também uma diferença relevante na distribuição dos relatos cancelados para os que não foram, gerando um percentual de 15,7% e 84,3%, respectivamente.

Para demonstrar visualmente a diferença entre relatos cancelados e não cancelados, foi gerado o gráfico da Figura 3. Ele representa a quantidade de relatos registrados para cada tipo (codificação ou enunciado), bem como os que foram cancelados e aqueles que não foram.

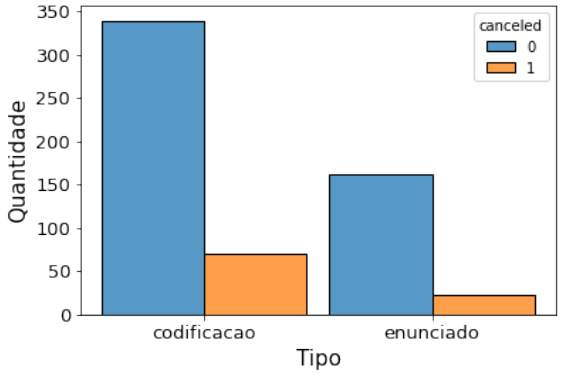

Figura 3. Distribuição dos tipos de relato de confusão

Visando justamente o objetivo do cancelamento de relato, foi preciso fazer um tratamento nos *logs* de confusão para que os relatos cancelados não estivessem inclusos no cálculo, restando somente os não cancelados. Logo, as análises a seguir foram feitas com este fator sendo tratado. Do contrário, muitos relatos adicionais iriam ser contabilizados e a funcionalidade de cancelamento perderia seu propósito.

Além disso, foi gerado um índice de confusão, calculado pela razão entre o número absoluto de relatos e o número de alunos que tentaram resolver determinada questão. Tal índice possibilita apontar as questões com maior taxa de relatos por tentativas regis-

tradas, apresentadas na Tabela 2. Assim, também foi feita a classificação das dez questões que obtiveram mais relatos de confusão por parte dos alunos (Figura 4).

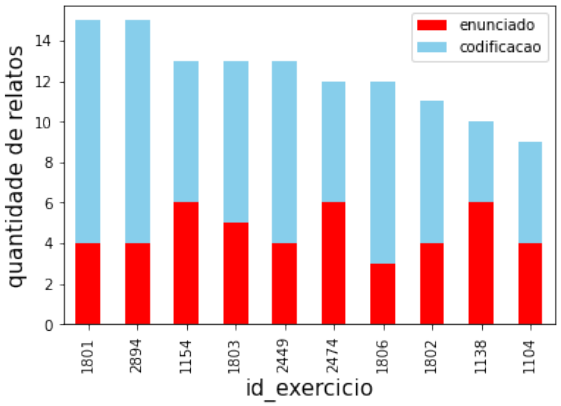

Figura 4. Dez questões com mais relatos de confusão

Tabela 2. Discretização das dez questões com maiores índices de confusão

Questão	Taxa de Acerto	Nº Tentativas	Nº Acertos	Nº Erros	Índice de Confusão(%)
2474	0,57	115	66	49	10.4
1117	0,49	96	47	49	8.3
602	0,63	101	64	37	7.9
1138	0,65	127	82	45	7.8
1154	0,64	176	113	63	7.4
1806	0,72	170	123	47	7.1
1803	0,64	188	121	67	6.9
1801	0,62	219	135	84	6.8
2894	0,71	221	158	63	6.7
3189	0,67	46	31	15	6.5

4.3. Taxa de acerto e dificuldade

Foi calculada a taxa de acerto de todas as questões presentes no *dataset* com pelo menos um registro de tentativa. Tais taxas foram definidas com base na escala descrita na Tabela 1, presente na seção 3.3 deste artigo. Assim, foi estruturado um gráfico com o objetivo de demonstrar as taxas de acerto mais recorrentes dentre as questões do *dataset*.

Como visto na Figura 5, a faixa entre 0,50 e 0,80 foi a que concentrou a maioria das taxas de acerto, com uma média de aproximadamente 0,66. A média das taxas de acerto de todas as questões foi comparada com a média das taxas de acerto das dez questões que obtiveram mais relatos de confusão. Enquanto a média geral resultou em um valor aproximado de 0,66, a média das questões com maior relato de confusão resultou em aproximadamente 0,64. Logo, por conta dessa baixa discrepância, não é possível afirmar que as questões envolvendo mais confusão serão sempre as mais difíceis.

A Tabela 3 aponta a quantidade de questões de acordo com a classificação do nível de dificuldade. Como pode ser visto, as questões classificadas como fáceis e medianas ocupam grande parte das 185 questões presentes no *dataset*, sendo cerca de 70% delas

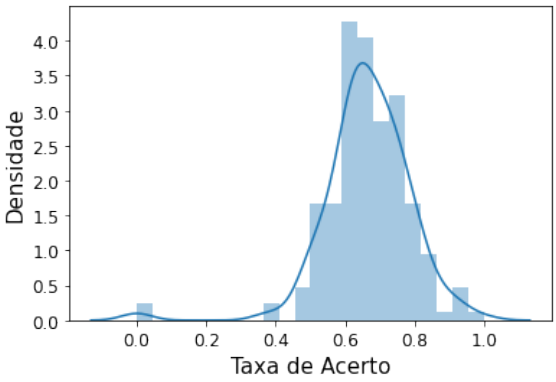

Figura 5. Taxa de acerto das questões do dataset

questões fáceis. Tal resultado já era esperado, visto que são necessárias mais questões fáceis para que o índice de reprovação na matéria não seja tão alto, principalmente em disciplinas de programação básica. A própria taxa de acerto se mostra aplicável justamente ao reforçar esse ponto. Analogamente, foi observado que apenas uma questão do *dataset* obteve 100% da taxa de acerto. Vale ressaltar que, apesar de tal questão ter sido classificada como fácil, ela obteve somente três tentativas no total.

Tabela 3. Número de questões por classificação de dificuldade.

Classificação Inep	Nº de questões	Porcentagem
Fácil	129	70%
Média	46	25%
Muito Fácil	6	03%
Muito Difícil	2	01%
Difícil	2	01%

4.4. Correlação da confusão com a dificuldade

Inicialmente, foi feito um estudo de correlação visando a análise do conjunto de questões como um todo. Tal correlação foi feita utilizado o coeficiente de correlação de Spearman e teve como variáveis de análise o índice de confusão e a taxa de acerto. O resultado foi uma correlação de -0.40, ou seja, uma associação média entre as duas variáveis. Dessa forma, uma alta taxa de acerto tende a indicar um baixo índice de confusão na questão, enquanto o inverso também é válido.

Em seguida, foram observados os assuntos mais abordados nas 185 questões presentes no *dataset*. Dentre elas, observou-se que os assuntos eram primariamente voltados a operações matemáticas, estruturas sequenciais e estruturas condicionais (*if-elif-else*). Assim, a maioria eram questões mais voltadas para lógica e cálculos matemáticos.

Outra análise realizada foi acerca das dez questões com maiores índices de confusão (Tabela 2) e das dez com mais relatos de confusão (Figura 4). Dentre elas, ambas apresentaram somente questões classificadas como fáceis e medianas. Logo, apesar de aparentarem ser questões de fácil solução e rápido entendimento, foram as que mais geraram relatos de confusão durante a resolução dos exercícios. Por fim, foi organizado um

arquivo[5] com os enunciados das dez questões com mais relatos de confusão e das dez com os maiores índices de confusão.

4.5. Premissas e Limitações

O botão de confusão possui uma opção para que os alunos possam cancelar o último relato feito, seja por terem apertado acidentalmente ou por terem conseguido solucionar o estado de confusão naquele momento. Entretanto, nesses casos, ocorre a possibilidade de o usuário esquecer de cancelar o relato, deixando assim o relato permanente de um estado de confusão equivocado. Em trabalhos futuros, será estudada a viabilidade de exibição de um lembrete temporário, momentos após o aluno ter apertado o botão, com duração de alguns segundos na tela.

Além disso, de acordo com [Lima et al. 2020], a variável taxa de acerto dificilmente conseguirá refletir a real dificuldade vivenciada pelos estudantes ao resolverem os exercícios de codificação. Assim, por mais que se possa gerar um classificador quantitativo capaz de indicar o nível de dificuldade, a abordagem utilizada para a classificação desse nível não é capaz de generalizar os casos tidos pelos estudantes, visto que fatores externos também podem influenciar como empecilhos para a resolução das questões.

5. Conclusão e trabalhos futuros

Este trabalho usou uma ferramenta interativa para coletar *logs* dos relatos de confusão feitos por alunos *non-majors* em Computação. Ao todo, 197 estudantes geraram os 177.134 registros coletados e resolveram as 185 questões utilizadas nesta pesquisa.

Como resultado desta análise exploratória, foi verificado um alto índice de confusão em questões que apresentam lógica e operações matemáticas como requisitos de resolução. Tal fato justifica-se por se tratar de estudantes de disciplinas introdutórias de programação que, muitas vezes, enfrentam dificuldades para entender a lógica de programação e o processo de resolução de problemas. Dessa forma, esta pesquisa se torna de relevante importância, visto que, os professores têm a necessidade de um melhor acompanhamento das experiências que os alunos estão tendo em suas disciplinas, principalmente em turmas iniciantes.

Como trabalho futuro, pretende-se realizar análises mais aprofundadas dos relatos de confusão e do *dataset*, assim como, prever com uma maior segurança o nível de confusão gerado em questões de codificação.

Agradecimentos

O presente trabalho é decorrente do projeto de Pesquisa e Desenvolvimento (PD) 001/2020, firmado entre a Fundação da Universidade do Amazonas e FAEPI, que conta com financiamento da Samsung, usando recursos da Lei de Informática para a Amazônia Ocidental (Lei Federal nº 8.387/1991), estando sua divulgação de acordo com o previsto no artigo 39.º do Decreto nº 10.521/2020. Elaine Oliveira e Patricia A. Jaques também receberam apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (Processos 308513/2020-7 e 306005/2020-4).

[5] https://drive.google.com/drive/folders/13loNbGbqoxNMnvc1Z8_RRuV5HDdhqo6i

Referências

Arguel, A., Lockyer, L., Kennedy, G., Lodge, J. M., and Pachman, M. (2019). Seeking optimal confusion: a review on epistemic emotion management in interactive digital learning environments. *Interactive Learning Environments*, 27(2):200–210.

Barros, R. P., Santana Junior, O. V., Medeiros Silva, I. R., Santos, L. F., and Neto, V. R. C. (2020). Predição do rendimento dos alunos em lógica de programação com base no desempenho das disciplinas do primeiro período do curso de ciências e tecnologia utilizando técnicas de mineração de dados. *Brazilian Journal of Development*, 6(1):2523–2534.

Calvo, R. A. and D'Mello, S. (2010). Affect detection: An interdisciplinary review of models, methods, and their applications. *IEEE Transactions on affective computing*, 1(1):18–37.

Chi, M. T. and Ohlsson, S. (2005). *Complex Declarative Learning.*, pages 371–399. Cambridge University Press.

D'Mello, S. and Calvo, R. A. (2013). Beyond the basic emotions: what should affective computing compute? In *CHI'13 extended abstracts on human factors in computing systems*, pages 2287–2294. ACM.

D'Mello, S. K. and Graesser, A. C. (2014). Confusion. In *International handbook of emotions in education*, pages 299–320. Routledge.

INEP (2017). Relatório síntese de área – Ciência da Computação. Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira.

Lehman, B., D'Mello, S., Strain, A., Mills, C., Gross, M., Dobbins, A., Wallace, P., Millis, K., and Graesser, A. (2013). Inducing and tracking confusion with contradictions during complex learning. *International Journal of Artificial Intelligence in Education*, 22:85–105.

Lima, M., Carvalho, L. S. G., Oliveira, E. H. T., Oliveira, D. B. F., and Pereira, F. D. (2020). Classificação de dificuldade de questões de programação com base em métricas de código. In *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, pages 1323–1332. SBC.

Moors, A., Ellsworth, P. C., Scherer, K. R., and Frijda, N. H. (2013). Appraisal theories of emotion: State of the art and future development. *Emotion Review*, 5:119–124.

Santos, P., Carvalho, L. S. G., Oliveira, E., and Fernandes, D. (2019). Classificação de dificuldade de questões de programação com base na inteligibilidade do enunciado. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 1886.

Scherer, K. R. (2019). Studying appraisal-driven emotion processes: Taking stock and moving to the future. *Cognition and Emotion*, 33(1):31–40.

Sedgwick, P. and Greenwood, N. (2015). Understanding the hawthorne effect. *Bmj*, 351.

Sheard, J., Carbone, A., Chinn, D., Laakso, M.-J., Clear, T., De Raadt, M., D'Souza, D., Harland, J., Lister, R., Philpott, A., et al. (2011). Exploring programming assessment instruments: a classification scheme for examination questions. In *Proceedings of the seventh international workshop on Computing education research*, pages 33–38.

Silva, G., Stroele, V., Dantas, M., and Campos, F. (2019). Hold up: Modelo de detecção e controle de emoções em ambientes acadêmicos. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 139.

Silvia, P. J. (2010). Confusion and interest: The role of knowledge emotions in aesthetic experience. *Psychology of Aesthetics, Creativity, and the Arts*, 4:75–80.

Yang, D., Kraut, R. E., and Rose, C. P. (2016). Exploring the effect of student confusion in massive open online courses. *Journal of Educational Data Mining*, 8(1):52–83.