Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: 31/May/2023; 1st round notif.: 27/Nov/2023; New version: 11/Jan/2024; 2nd round notif.: 27/Apr/2024;
Camera ready: 27/Oct/2024; Edition review: 31/Oct/2024; Available online: 31/Oct/2024; Published: 31/Oct/2024;

Detecção de Emoções na Aprendizagem de Programação: Os Efeitos de Usar Estimativas de Conhecimento em Modelos Livres de Sensores que Detectam a Confusão do Aluno

Title: Emotion Detection in Programming Learning: The Effects of Using Knowledge Estimates in Sensor-Free Models that Detect Student Confusion

Título: Detección de Emociones en el Aprendizaje de Programación: Los Efectos de Usar Estimaciones de Conocimiento en Modelos Libres de Sensores que Detectan la Confusión del Estudiante

Tiago R. Kautzmann
PPGCA, UNISINOS
ORCID: 0000-0002-6017-8340
tkautzmann@gmail.com

Gabriel de O. Ramos
PPGCA, UNISINOS
ORCID: 0000-0002-6488-7654
gdoramos@unisinos.br

Patrícia A. Jaques
PPGC, UFPEL
PPGInf, UFPR
ORCID: 0000-0002-2933-1052
patricia.jaques@inf.ufpel.edu.br
patricia@inf.ufpr.br

Resumo

A confusão é uma emoção provável de ocorrer em tarefas de aprendizagem de conteúdos complexos, como na aprendizagem de programação de computadores. Quando não regulada pelo aluno, a confusão pode afetar negativamente o aprendizado. Quando regulada, pode levar a aprendizagem a níveis mais profundos. O estudo descrito neste artigo buscou melhorar o desempenho de modelos livres de sensores que detectam a confusão do aluno enquanto envolvido em tarefas de aprendizagem de programação. Estes modelos são interessantes quando integrados a ferramentas de programação porque, ao detectar a confusão do aluno durante a aprendizagem, a ferramenta poderia intervir e auxiliar o aluno na regulação dessa emoção. Trabalhos relacionados treinaram modelos de detecção de confusão usando dados de interação do aluno com o ambiente de programação, como dados sobre movimentos de teclado e mouse. Nosso estudo levantou a hipótese que incorporar dados sobre estimativas de conhecimento do aluno aos dados de interação poderia melhorar o desempenho dos modelos. Nós comparamos o desempenho de modelos de aprendizado de máquina treinados com a abordagem da hipótese com modelos treinados com a abordagem dos trabalhos relacionados. Os modelos foram treinados com dados coletados de 62 alunos em aulas de programação ao longo de cinco meses. Os resultados apresentaram evidências positivas que apoiam nossa hipótese. Também discutimos cenários onde nossa abordagem é vantajosa, como o tamanho adequado dos segmentos de dados, os algoritmos com melhor desempenho e o poder de generalização dos modelos para alunos de diferentes níveis de ensino.

Palavras-chave: *Detecção de confusão; Modelos livres de sensores; Aprendizado de máquina; Estimativas de conhecimento do aluno; Aprendizagem de programação de computadores; Regulação emocional na aprendizagem; Emoções na aprendizagem.*

Abstract

Confusion is an emotion likely to occur in learning tasks involving complex content, such as in computer programming learning. When not regulated by the student, confusion can negatively affect learning. When regulated, it can lead to deeper levels of learning. The study described in this article sought to improve the performance of sensor-free

Cite as: Kautzmann, T. R., Ramos, G. O. & Jaques, P. A. (2024). Detecção de Emoções na Aprendizagem de Programação: Os Efeitos de Usar Estimativas de Conhecimento em Modelos Livres de Sensores que Detectam a Confusão do Aluno. Revista Brasileira de Informática na Educação, 32, 642-678. https://doi.org/10.5753/rbie.2024.3437.

models that detect student confusion while engaged in programming learning tasks. These models are interesting when integrated into programming tools because, by detecting student confusion during learning, the tool could intervene and assist the student in regulating his/her emotion. Related work trained confusion detection models using data from student interactions with the programming environment, such as data on keyboard and mouse movements. Our study hypothesized that incorporating data on student knowledge estimates into interaction data could improve the models' performance. We compared the performance of machine learning models trained with the hypothesis approach to models trained with the approach of related work. The models were trained with data collected from 62 students in programming classes over five months. The results presented positive evidence supporting our hypothesis. We also discussed scenarios where our approach is advantageous, such as the appropriate size of data segments, the best-performing algorithms, and the generalization power of the models for students of different educational levels.
Keywords: *Confusion detection; Sensor-free models; Machine learning; Student knowledge estimates; Programming learning; Emotional regulation in learning; Emotions in learning.*

Resumen
La confusión es una emoción probable que ocurra en tareas de aprendizaje de contenidos complejos, como en el aprendizaje de programación de computadoras. Cuando no es regulada por el estudiante, la confusión puede afectar negativamente el aprendizaje. Cuando es regulada, puede llevar a niveles más profundos de aprendizaje. El estudio descrito en este artículo buscó mejorar el rendimiento de modelos libres de sensores que detectan la confusión del estudiante mientras está involucrado en tareas de aprendizaje de programación. Estos modelos son interesantes cuando se integran en herramientas de programación porque, al detectar la confusión del estudiante durante el aprendizaje, la herramienta podría intervenir y ayudar al estudiante a regular esta emoción. Trabajos relacionados entrenaron modelos de detección de confusión utilizando datos de interacción del estudiante con el entorno de programación, como datos sobre movimientos de teclado y ratón. Nuestro estudio planteó la hipótesis de que incorporar datos sobre estimaciones del conocimiento del estudiante a los datos de interacción podría mejorar el rendimiento de los modelos. Comparamos el rendimiento de modelos de aprendizaje automático entrenados con el enfoque de la hipótesis con modelos entrenados con el enfoque de trabajos relacionados. Los modelos fueron entrenados con datos recopilados de 62 estudiantes en clases de programación durante cinco meses. Los resultados presentaron evidencias positivas que apoyan nuestra hipótesis. También discutimos escenarios donde nuestro enfoque es ventajoso, como el tamaño adecuado de los segmentos de datos, los algoritmos con mejor rendimiento y el poder de generalización de los modelos para estudiantes de diferentes niveles educativos.
Palabras clave: *Detección de confusión; Modelos libres de sensores; Aprendizaje automático; Estimaciones del conocimiento del estudiante; Aprendizaje de programación; Regulación emocional en el aprendizaje; Emociones en el aprendizaje.*

1 Introdução

As emoções têm um papel determinante na aprendizagem, modulando processos cognitivos como memória e tomada de decisão, potencialmente facilitando ou obstaculizando a aquisição de conhecimentos (Pekrun, 2011). Em contextos de aprendizagem, são comumente observadas emoções classificadas como acadêmicas ou orientadas à aprendizagem, incluindo engajamento, confusão, frustração e tédio (D'Mello & Calvo, 2013; Pekrun, 2014).

D'Mello e Graesser (2012) apresentam um modelo que busca decifrar a dinâmica dessas emoções durante a aprendizagem profunda. Segundo o modelo, um aluno confuso experimenta um estado de desequilíbrio cognitivo, provocado por contradições, conflitos ou informações incorretas (D'Mello & Graesser, 2014). Esse estado pode desencadear emoções negativas relacionadas à aprendizagem, como frustração e tédio, possivelmente levando ao desengajamento (D'Mello et

al., 2014; Silvia, 2010). Por outro lado, a resolução da confusão pode propiciar uma aprendizagem mais profunda, visto que o desequilíbrio cognitivo pode incitar o aluno a mobilizar recursos cognitivos adicionais (D'Mello & Graesser, 2014).

Ambientes computacionais de aprendizagem que conseguem detectar a confusão do aluno têm a capacidade de tomar decisões instrucionais significativas, auxiliando o aluno a gerir sua confusão (Arguel et al., 2019). Isso é especialmente relevante em contextos de aprendizagem de programação, onde a confusão é uma emoção comumente reportada (Bosch & D'Mello, 2017; Bosch et al., 2013; Coto et al., 2021). Na aprendizagem de programação, o aluno precisa entender o problema subjacente à tarefa e mobilizar esforços cognitivos para identificar e conectar conhecimentos e estratégias de solução de problemas (Chi & Ohlsson, 2005; Lehman et al., 2013).

Em ambientes inteligentes de aprendizagem, as emoções dos estudantes podem ser detectadas através de várias modalidades de entrada, como imagens de vídeo, voz, sinais fisiológicos (ritmo cardíaco, respiração, etc), entre outros (Calvo & D'Mello, 2010). No entanto, essas abordagens necessitam a instalação e uso de sensores, implicando em custos adicionais. Também possuem desvantagens como invasão de privacidade, possíveis desconfortos causados pela utilização de certos sensores e dependência de hardware específico. Essas técnicas também podem ser afetadas por ruído ambiental e outras variáveis que não estão diretamente relacionadas à emoção que se deseja detectar. Por exemplo, a detecção de emoções através de imagens de vídeo pode ser afetada pela iluminação do ambiente, enquanto a detecção através de sinais fisiológicos pode ser influenciada por fatores como atividade física e condições de saúde do indivíduo.

Uma alternativa às abordagens baseadas em sensores são os modelos livres de sensores, que se baseiam unicamente em dados coletados durante a interação do aluno com o ambiente de aprendizagem (Botelho et al., 2017). Esses dados podem incluir registros de atividades realizadas pelo aluno, tempo gasto em cada atividade, progresso alcançado, dificuldades encontradas, entre outros. Essa abordagem não necessita da instalação e uso de sensores adicionais, além de preservar a privacidade do aluno, uma vez que não são coletados dados pessoais sensíveis. Além disso, é menos susceptível a ruído e outras variáveis não relacionadas à emoção que se deseja detectar. Esses modelos são menos intrusivos e mais aplicáveis em larga escala, sobretudo em escolas com restrições financeiras (Arroyo et al., 2009; Yang et al., 2019).

Pesquisas têm utilizado modelos de aprendizado de máquina supervisionado para detectar a emoção de confusão dos alunos. As amostras de treinamento desses modelos consistem em dados sobre a interação do aluno com o ambiente de programação, como movimentos do *mouse* e pressionamentos de teclas. No entanto, esses dados não possuem uma relação causal conhecida com a emoção de confusão. Teorias sugerem que os modelos de detecção emocional podem se beneficiar de uma abordagem mista, que utiliza tanto dados sem relação causal conhecida com a emoção quanto dados com relação causal conhecida (D'Mello, 2020; D'Mello & Graesser, 2014).

Alguns estudos têm focado especialmente na detecção da emoção de confusão durante tarefas de programação (Felipe et al., 2012; Lee et al., 2011; Vea & Rodrigo, 2017), mas estes apresentaram limitações significativas. Felipe et al. (2012) desenvolveu um modelo de detecção de confusão utilizando dados de digitação, mas utilizou uma quantidade limitada de dados de alunos para treinamento dos modelos. O estudo de Vea e Rodrigo (2017) também se baseou em dados de digitação, além de informações derivadas das ações do *mouse*, e obteve desempenho moderado. O estudo de Lee et al. (2011) tentou detectar confusão utilizando dados relacionados à

compilação do código, mas esse modelo não detectou a confusão durante a fase de codificação.

No presente estudo, levantamos a hipótese que ***uma abordagem que incorpora dados sobre estimativas de conhecimento do aluno ao conjunto de dados de interação do aluno com o ambiente pode melhorar o desempenho dos modelos livres de sensores na detecção da confusão do aluno em tarefas de programação, em comparação com a abordagem dos trabalhos relacionados, que usaram somente dados de interação***.

Para levantar nossa hipótese, nós assumimos que as estimativas do conhecimento dos alunos possuem uma relação causal com a confusão. Teorias cognitivas de emoções apoiam esta premissa. O modelo multicomponencial de Scherer, uma das teorias que modelam o aspecto cognitivo das emoções (Moors et al., 2013; Scherer, 2005), relaciona as emoções com aspectos cognitivos. O modelo considera o componente de *appraisal*, o processo interno de um indivíduo que avalia a importância do ambiente para o bem-estar pessoal. A importância para o bem-estar pode ser entendida como a satisfação ou a não satisfação de necessidades pessoais, objetivos, crenças ou tudo o que interessa ao indivíduo (Moors et al., 2013). A confusão surge de um processo de *appraisal* quando o indivíduo é confrontado com uma contradição, tal como quando a informação recebida não pode ser integrada ao mapa mental existente ou quando o indivíduo não tem certeza do que fazer a seguir em uma tarefa (D'Mello & Graesser, 2014). O trabalho de Silvia (2010) descreve as emoções relacionadas ao conhecimento, como a confusão, como emoções causadas pelas crenças dos indivíduos sobre seus pensamentos e conhecimentos e decorrentes de objetivos associados à aprendizagem. Silvia encontrou evidências positivas para a hipótese de que a confusão está associada a avaliações de novidade (algo desconhecido) e baixa compreensão no contexto da informação que chega ao indivíduo (Silvia, 2010). Outros estudos também encontraram evidências de uma correlação entre a confusão e a falta de conhecimento (D'Mello et al., 2009; Lee et al., 2011; Rodrigo et al., 2010) e que a confusão é um bom preditor de resultados de testes de conhecimento (Graesser et al., 2007). Outras discussões também sugerem os efeitos do conhecimento prévio dos alunos na aprendizagem e nas contradições que levam à confusão. Elas sugerem que níveis mais intermediários de conhecimento sobre um assunto podem ser melhores preditores da confusão (D'Mello et al., 2014). Um exemplo desse cenário é quando o aluno tem algum conhecimento para engajar em uma tarefa, mas ainda precisa de um entendimento completo. Caso ele enfrente alguma falha ou contradição na tarefa, a situação poderá deixá-lo em um estado de confusão, talvez precedido de um estado de surpresa (Silvia, 2010). No entanto, estudantes com baixo conhecimento prévio em um determinado assunto poderiam não conseguir identificar contradições e nem mesmo experimentar a confusão que surge das contradições (D'Mello et al., 2014; VanLehn et al., 2003).

O conhecimento prévio do aluno possui uma função reconhecida em métodos e estratégias de aprendizagem e em teorias da educação, como no Construtivismo, que enfatiza a aprendizagem como um processo ativo onde os alunos constroem novos conhecimentos com base em suas experiências e conhecimentos prévios (Jófili, 2002). O conceito de Zona de Desenvolvimento Proximal (ZDP), elaborado por Vygotsky, também se apropria do conhecimento prévio do aluno. A ZDP é a diferença entre o que um aluno pode fazer sem a ajuda de alguém mais experiente e o que ele pode fazer com a ajuda. O conhecimento prévio é crucial para identificar a ZDP de um aluno, permitindo que os educadores forneçam suporte adequado para impulsionar a aprendizagem (Fino, 2001). Ausubel, por sua vez, argumenta que a aprendizagem é mais eficaz quando o novo material de estudo é relacionado ao que o aluno já sabe, de forma que o conhecimento

prévio fornece uma "estrutura" para acomodar novas informações (Ausubel et al., 1978; Pelizzari et al., 2002). No método de Aprendizagem Baseada em Problemas (ABP), os alunos começam com um problema para resolver e usam seu conhecimento prévio como ponto de partida para a investigação. Isso ajuda a tornar a aprendizagem mais relevante e significativa, pois os alunos veem a aplicação direta de seu conhecimento prévio (de Oliveira Alves et al., 2020). A Teoria da Carga Cognitiva também reforça a importância do conhecimento prévio. A Teoria sugere que a capacidade de aprendizagem de um aluno é afetada pela quantidade de informação que sua memória de trabalho pode manipular de uma vez. O conhecimento prévio ajuda a reduzir a carga cognitiva, pois os alunos podem integrar novas informações com o que já sabem, tornando o aprendizado mais eficiente (Sweller et al., 1988, 2019). Estes são alguns exemplos da importante função do conhecimento prévio do aluno em teorias da educação e métodos de aprendizagem.

O ***primeiro objetivo*** do nosso trabalho foi buscar evidências que verificam nossa hipótese, ou seja, verificar se *incorporar estimativas do conhecimento do aluno aos dados de interação pode aprimorar a identificação da confusão do aluno por modelos livre de sensores*. Nosso ***segundo objetivo*** foi determinar a duração necessária de observação do comportamento do aluno no ambiente de programação para que o modelo de Inteligência Artificial (IA) possa inferir adequadamente se o aluno está confuso ou não. Estudos prévios utilizaram segmentos de dados de até 20 segundos (Felipe et al., 2012; Vea & Rodrigo, 2017). Um segmento de dado é um conjunto de dados sobre o comportamento do aluno no ambiente de programação em determinada janela (intervalo) de tempo, como a quantidade de teclas digitadas e a quantidade de movimentos de mouse registradas em 40 segundos. Nós testamos segmentos com janelas de 5, 10, 20, 40, 60, 90, 120, 180, 240 e 360 segundos. Queremos saber quanto tempo do comportamento do aluno o modelo precisa observar para inferir adequadamente que o aluno está confuso ou não. Nosso ***terceiro objetivo*** foi verificar se os modelos generalizam bem seus resultados para alunos com diferentes características heterogêneas, de diferentes níveis de ensino. Por fim, nosso ***quarto objetivo*** foi identificar os dados mais relevantes para os modelos aprenderem a identificar a confusão do aluno e os melhores algoritmos.

Este artigo estende nosso trabalho anterior (Kautzmann et al., 2022) nos seguintes pontos: a) no trabalho anterior nós não comparamos nossa abordagem com a dos trabalhos relacionados, mas fazemos isso aqui; b) acrescentamos evidências sobre o poder de generalização dos modelos para alunos com características heterogêneas; c) ampliamos a análise sobre a relevância dos diferentes tipos de dados no desempenho dos modelos; e d) ampliamos as discussões sobre a pesquisa.

O restante do artigo está assim organizado: a Seção 2 apresenta discussões sobre o que é a confusão e como ela se relaciona com aspectos cognitivos relacionados ao conhecimento. A Seção 3 situa o presente estudo em relação aos trabalhos relacionados. A Seção 4 apresenta uma visão geral sobre os métodos aplicados. A Seção 5 descreve o modelo de conhecimento implementado para gerar as estimativas de conhecimento do aluno. A Seção 6 descreve o ambiente de programação utilizado no estudo. As seções 7 e 8 descrevem os métodos utilizados pela pesquisa para alcançar seus objetivos e os resultados obtidos. Por fim, a Seção 9 conclui o estudo e apresenta discussões sobre suas limitações e possíveis direções para pesquisas futuras.

2 Confusão

Da mesma forma que não há consenso entre os pesquisadores sobre o que é uma emoção (Scherer, 2005), a confusão também tem recebido diferentes conceitualizações (D'Mello et al., 2014). A confusão já foi classificada como uma emoção genuína (Rozin & Cohen, 2003), uma emoção relacionada ao conhecimento (Silvia, 2010), uma emoção epistêmica (Pekrun & Stephens, 2012) e uma "não emoção" (Hess, 2003; Keltner & Shiota, 2003).

O trabalho de D'Mello et al. (2014) apresenta uma discussão interessante sobre o que é a confusão. Segundo os autores, a falta de uma clara definição sobre o que é uma emoção, suas múltiplas perspectivas e a escassez de pesquisa básica sobre a confusão, leva à falta de uma definição mais precisa sobre a confusão e se ela é uma emoção. Com o objetivo de encontrar uma correspondência entre o que se sabe sobre a confusão e as características de uma emoção, os autores sugerem iniciar a discussão a partir de um estudo de Izard (2010), que buscou identificar as características que definem uma emoção. Este estudo solicitou respostas a 37 pesquisadores de emoções para seis questões, entre elas, "o que é uma emoção". O resultado do estudo descreve a emoção como um estado afetivo que envolve seis características, a saber:

a) Circuitos neurais parcialmente dedicados ao processamento emocional;
b) Ativação de sistemas de respostas em preparação para alguma ação;
c) Ter estados de sentimentos distintos;
d) Desempenhar um papel no comportamento expressivo;
e) Surgir a partir de resultados de processos de *appraisal*;
f) Poder envolver interpretação cognitiva de sentimentos.

Segundo D'Mello et al. (2014), a confusão compartilha pelo menos quatro (*a*, *c*, *d* e *e*) das seis principais características de emoção identificadas pelos pesquisadores. Em relação ao item *a*, já foram encontradas evidências de que a confusão está ligada a atividades neurais, como no trabalho descrito por (Halgren & et al., 2002), que encontrou um alto valor para N400[1] em atividades de eletroencefalograma, quando um indivíduo foi confrontado com uma sentença que apresentou uma anomalia semântica, como "Maria estava com fome, então ela foi até uma loja comprar algumas cadeiras".

Em relação a *c*, já foi descrito um estado de sentimento distinto relacionado à confusão (Rozin & Cohen, 2003). Em relação a *d*, já foi encontrado um componente expressivo para a confusão, a sobrancelha franzida (Craig & et al., 2008a; Grafsgaard et al., 2011; McDaniel & et al., 2007). Em relação ao item *e*, foram encontradas evidências de que a confusão surge a partir de um processo de *appraisal* cognitivo relacionado a uma falta de correspondência entre a informação que chega ao indivíduo e o seu conhecimento anterior relativo à informação (D'Mello & Graesser, 2014; Silvia, 2010).

Sobre as demais características de emoções apontadas por Izard (2010), não se sabe ainda muito sobre mudanças corporais associadas com a confusão (característica *b*), mas o trabalho apresentado por Alzoubi et al. (2012) conseguiu algum sucesso em distinguir a confusão do estado neutro e outras emoções, a partir de dados fisiológicos. Assim, nosso trabalho qualifica a confusão como uma emoção, acompanhando o mesmo entendimento de D'Mello et al. (2014),

[1] Sinal eletrofisiológico relacionado a uma informação semanticamente anômala.

que consideraram as evidências científicas descritas anteriormente, relacionadas a características vinculadas a emoções, como ser um estado afetivo que surge de interações neurais, que tem um componente expressivo, que tem um estado de sentimento distinto e que está relacionado a um componente cognitivo de *appraisal*.

2.1 *Appraisal* da confusão

Ao ser caracterizada como uma emoção epistêmica (Pekrun & Stephens, 2012), por ser causada pelo processamento cognitivo de informação como parte de uma tarefa de aprendizagem (Arguel et al., 2019), e uma emoção relacionada ao conhecimento (Silvia, 2010), é importante para o presente trabalho destacar a relação desta emoção com o conhecimento do indivíduo, uma vez que buscamos melhorar a detecção da confusão por modelos de IA usando esse tipo de informação.

O *appraisal* é um processo interno de avaliação de um indivíduo sobre como um evento de estímulo se relaciona com as próprias metas, valores, conhecimentos e habilidades. O trabalho de Silvia (2010) descreve emoções relacionadas ao conhecimento como emoções causadas pelas crenças dos indivíduos sobre seus próprios pensamentos e conhecimentos e decorrem de objetivos associados ao aprendizado, como o conhecimento, o pensamento e a compreensão. Seu trabalho encontrou evidências positivas para a hipótese de que a confusão está associada com *appraisals* de novidade (familiar) e de baixa compreensibilidade, no contexto da informação que chega para o indivíduo. Neste sentido, a teoria de discrepância de MacDowell e Mandler (1989) descreve que os indivíduos estão constantemente assimilando novas informações em um modelo mental de conhecimento. Quando uma informação discrepante é identificada, como um conflito com o conhecimento prévio, ocorre uma excitação no sistema nervoso autônomo, e o indivíduo experimenta uma variedade de possíveis emoções. A emoção experimentada depende do contexto e do processo de *appraisal*.

O trabalho de Silvia (2010) também encontrou que a surpresa pode preceder a confusão quando um estímulo inesperado é avaliado no processo de *appraisal* como algo incompreensível. Com isso, os estudos que consideram o componente emocional de *appraisal* descrevem a confusão como uma emoção que ocorre quando há uma falta de correspondência entre a informação que chega e o conhecimento anterior, ou quando essa informação não pode ser integrada aos modelos mentais existentes, iniciando um desequilíbrio cognitivo (D'Mello & Graesser, 2014).

3 Trabalhos Relacionados

Nosso estudo conduziu uma pesquisa abrangente nas principais bases de dados acadêmicas, incluindo *Springer*, *Elsevier Science Direct*, *IEEE Xplore*, *ACM Digital Library*, *Web of Science*, *Scopus*, *Taylor & Francis* e *Google Scholar*. Foram utilizadas *strings* de busca contendo as palavras-chave "*predict*", "*detect*", "*confusion*" e "*programming*" adaptadas para cada base de dados. As palavras-chave utilizadas nas *strings* de busca foram cuidadosamente escolhidas para abranger o tema da pesquisa: detecção de confusão em tarefas de programação.

Os critérios de inclusão adotados envolveram a seleção de artigos que apresentassem mode-

los de detecção de confusão em ambientes computacionais de programação, como em IDEs[2] de desenvolvimento de software. Os critérios de exclusão incluíram artigos não escritos em inglês, duplicados, com os mesmos resultados de avaliação e publicados em formatos como pôsteres, capítulos de livros, resumos, teses e relatórios técnicos.

Além das buscas nas bases de dados, foi realizado um mapeamento recursivo dos trabalhos relacionados citados nos estudos selecionados, bem como dos principais grupos de pesquisa que se dedicam à afetividade em ambientes computacionais de aprendizagem, como os grupos dos pesquisadores Sidney D'Mello, Ryan Baker, Ivon Arroyo, Rosalind Picard, Jennifer Sabourin, Cristina Conati e Beverly Park Woolf e o programa de pesquisa sobre o Cognitive Tutor, da Universidade Carnegie Melon.

3.1 Trabalhos selecionados

Após a aplicação dos critérios de inclusão e exclusão, foram selecionados sete trabalhos relacionados que abordam a detecção de confusão e outras emoções durante a aprendizagem de programação. O trabalho realizado por Tiam-Lee e Sumi (2018) descreve um ambiente computacional que modifica suas ações com base na detecção da confusão dos alunos. Eles utilizaram um modelo *HMM* (*Hidden Markov Model*) e dados coletados de 11 alunos em uma sessão de aprendizagem de programação. O ambiente atribuiu exercícios com dificuldade crescente e, quando a confusão foi detectada, ofereceu guias e dicas para auxiliar os alunos. O trabalho alcançou resultados satisfatórios na detecção da confusão, com validação cruzada e avaliação de 35 alunos.

Em outro estudo de Tiam-Lee e Sumi (2019), foram exploradas diferentes emoções, como confusão, engajamento, frustração e tédio. Eles utilizaram atributos derivados de dados de digitação, compilação e rosto dos alunos. O trabalho contou com a participação de 73 estudantes, e diferentes conjuntos de atributos de dados foram utilizados para treinar modelos de classificação dos estados afetivos. Os resultados obtidos apresentaram acurácia e *kappa* variados para a detecção das diferentes emoções.

O trabalho de Vea e Rodrigo (2017) concentrou-se na detecção das emoções tédio, confusão e frustração em alunos novatos em programação. Eles utilizaram atributos de dados derivados de atividades de teclado e *mouse*. O estudo envolveu 60 alunos e três sessões de programação, utilizando o ambiente *Dev-C++*. Foram testados diferentes classificadores e conjuntos de atributos de dados, alcançando resultados moderados na detecção das emoções.

Grafsgaard et al. (2011) desenvolveram um modelo de detecção de confusão que utilizou o corpus do diálogo textual entre aluno e tutor, além de dados da tarefa e expressões faciais do aluno. O estudo envolveu 14 alunos que resolveram problemas introdutórios de programação. Os resultados mostraram que o modelo baseado em *HMM* superou outros modelos, alcançando alta acurácia na detecção da confusão.

Lee et al. (2011) investigaram a relação entre confusão e desempenho de alunos novatos em programação no ambiente *BlueJ*. Eles utilizaram anotações humanas para rotular amostras de *logs*

[2]Do inglês *Integrated Development Environment* (Ambiente de Desenvolvimento Integrado). É um software que fornece facilidades abrangentes para programadores durante o desenvolvimento de software, como um editor de texto avançado, um interpretador ou compilador de códigos de programação e demais funcionalidades que auxiliam o processo de desenvolvimento de software.

como confusas ou não confusas. O estudo envolveu 149 alunos e analisou dados relacionados a compilações, como tempo médio entre compilações e número de compilações com erros. Os resultados sugerem uma associação entre confusão prolongada e baixo desempenho dos alunos.

Felipe et al. (2012) apresentaram um modelo de detecção de confusão e tédio em um ambiente de programação em linguagem *C++*. O estudo, embora com uma quantidade limitada de alunos, alcançou resultados promissores na detecção da confusão usando dados de pressionamento de teclas. O trabalho também foi avaliado posteriormente por Vea e Rodrigo (2017), que trouxe críticas sobre a pequena quantidade de alunos envolvidos e sugeriu que os dados de pressionamento de teclas por si só podem não ser suficientes para a detecção de estados afetivos.

Por fim, o trabalho de Bosch et al. (2014) apresentou um modelo de detecção de emoções em alunos novatos aprendendo programação, utilizando dados derivados de vídeos do rosto dos alunos. Os dados foram coletados de 99 alunos em uma sessão de resolução de exercícios de programação. O estudo explorou diferentes tamanhos de segmentos de vídeo e classificadores, alcançando resultados significativamente melhores com o uso de um modelo baseado em *HMM*.

3.2 Discussão

Dos trabalhos selecionados na revisão da literatura, apenas três trabalhos implementaram modelos livres de sensores Felipe et al. (2012), Lee et al. (2011) e Vea e Rodrigo (2017). O principal diferencial do nosso trabalho em relação a estes modelos livres de sensores encontra-se no tipo de informação do aluno que foi considerada para detectar a confusão do estudante. Enquanto aqueles trabalhos se concentraram em dados de interação do aluno com o ambiente, não justificados em teorias de emoções, como dados de pressionamento de teclas e uso do *mouse*, nós buscamos utilizar dados com forte aderência às teorias cognitivas de *appraisal* de emoções. Isso implica em utilizar dados que estejam mais diretamente relacionados aos estados afetivos do aluno durante a codificação, como o seu conhecimento prévio. Além disso, diferente daqueles trabalhos, nós utilizamos segmentos de dados obtidos em momentos não fixos, baseados em decisões arbitrárias de juízes humanos (os próprios alunos), possibilitando uma vinculação mais precisa dos dados aos rótulos emocionais. A forma como esses segmentos foram obtidos são detalhados mais adiante.

Em suma, nosso trabalho busca superar as limitações dos trabalhos relacionados ao utilizar dados com embasamento teórico mais consistente, estudar o tamanho mais adequado dos segmentos de dados no desempenho dos classificadores de confusão, verificar o poder de generalização dos resultados para estudantes de diferentes níveis de ensino e identificar os dados mais relevantes para a detecção da confusão e os algoritmos com melhor desempenho.

4 Visão geral dos métodos

Como explicado na Introdução, nosso trabalho tem quatro objetivos. Nosso *primeiro objetivo* é verificar a hipótese do estudo, ou seja, pretendemos verificar se o desempenho de modelos de detecção de confusão livres de sensores que consideram dados relativos à interação do aluno com o ambiente juntamente de dados que estimam o conhecimento do aluno em programação podem superar o desempenho dos modelos dos trabalhos relacionados, que consideram apenas os dados de interação. Para atingir este primeiro objetivo, foram treinados e validados diversos modelos

de aprendizado de máquina supervisionado que visam identificar a confusão e a não confusão do aluno com base em amostras de dados que refletem o comportamento do estudante em um ambiente de programação durante a realização de tarefas de programação.

Os modelos gerados foram divididos em duas abordagens: a **abordagem da hipótese** e a **abordagem *baseline***. Os **modelos da hipótese** representam a abordagem da hipótese, treinados e validados com amostras compostas por dados de interação do aluno com o ambiente de programação juntamente de dados sobre estimativas de conhecimento do aluno. Os **modelos *baseline*** representam a abordagem dos trabalhos relacionados, treinados somente com dados de interação do aluno com o ambiente de programação. A única diferença entre os modelos é que nos modelos da hipótese foram utilizados dados sobre estimativas de conhecimento do aluno, enquanto no modelos *baseline* estes dados foram omitidos. Nas duas abordagens, para cada amostra, foi atribuído um rótulo emocional ("confusão" ou "não confusão"), de modo que cada amostra representa um momento da interação do aluno com o ambiente em que ele estava confuso ou não estava confuso. O objetivo de gerar modelos nas duas abordagens é poder comparar as métricas de avaliação. Espera-se que os modelos que seguem a abordagem da hipótese apresentem melhor desempenho que os modelos que representam a abordagem dos trabalhos relacionados.

O *segundo objetivo* do estudo é determinar a duração necessária de observação do comportamento do aluno para inferir adequadamente se o aluno está confuso ou não. Para atingir este objetivo, foram gerados modelos que consideram amostras que refletem diferentes tamanhos de segmentos de dados.

O *terceiro objetivo* do estudo foi verificar o poder de generalização dos modelos para estudantes de diferentes níveis de ensino, com diferentes idades e diferentes experiências anteriores em programação. Para atingir este objetivo, todos os modelos de ambas abordagens foram gerados novamente, com os treinamentos realizados com amostras de estudantes de um nível de ensino e testados com amostras de alunos de outro nível de ensino. Este método também traz evidências para o primeiro objetivo da pesquisa.

Por fim, o *quarto objetivo* do estudo é identificar os tipos de dados mais relevantes nos melhores modelos da hipótese. Em outras palavras, tentamos identificar quais dados mais contribuíram para os algoritmos diferenciarem melhor o estado de confusão e de não confusão dos alunos. Nossa pesquisa espera que os dados sobre estimativas de conhecimento do aluno estejam entre os mais relevantes. Também buscamos identificar os algoritmos que apresentaram os desempenhos mais interessantes.

As próximas Seções detalham os métodos da pesquisa e seus resultados. Mas, primeiramente, a Seção 5 apresenta o modelo utilizado para gerar as estimativas de conhecimento do aluno. A Seção 6 descreve o ambiente de programação adaptado para esta pesquisa.

5 Modelo de conhecimento do aluno

Com o objetivo de obter as estimativas do conhecimento do aluno nas tarefas de programação de computadores para serem utilizadas nas amostras de treinamento dos modelos da hipótese, implementamos um modelo de conhecimento do aluno baseado no método *Bayesian Knowledge Tracing* (BKT). No entanto, antes de detalhar o BKT, precisamos apresentar o conceito de **com-**

ponente de conhecimento (CC).

5.1 O componente de conhecimento

O modelo BKT representa o conhecimento do aluno a partir de um conjunto de componentes de conhecimento (CC). Um *componente de conhecimento* é uma unidade adquirida de função ou estrutura cognitiva que pode ser inferida através do desempenho do indivíduo que é observado em um conjunto de tarefas (Koedinger et al., 2012). Na literatura, trabalhos que modelam componentes de conhecimento no domínio de programação de computadores geralmente definem CCs que representam estruturas, conceitos ou regras de programação, como estruturas de seleção, estruturas de repetição e atribuição de valores a variáveis (Kasurinen & Nikula, 2009; Penmetsa, 2021; Raposo et al., 2019; S. Wang et al., 2017). No presente trabalho, definimos um CC como uma estrutura ou um conceito de programação. Os CCs foram identificados nos exercícios de programação atribuídos pelos professores das turmas participantes da pesquisa. As turmas participantes foram turmas de introdução à programação. A Tabela 1 mostra os 29 componentes de conhecimento modelados. Por exemplo, o CC 6 representa o conhecimento do aluno sobre codificação de métodos construtores padrão. Outro exemplo é o CC 26, que representa o conhecimento do aluno sobre estruturas de repetição (*loops*).

Uma das nossas preocupações ao gerar o modelo de conhecimento do aluno foi aumentar a granularidade da representação dos componentes do conhecimento. Por exemplo, ao invés de criar um único CC para "Método de classe", foi possível fragmentar este CC em quatro CCs: "Método de classe com parâmetro e com retorno"; "Método de classe com parâmetro e sem retorno"; "Método de classe sem parâmetro e com retorno"; e "Método de classe sem parâmetro e sem retorno". Quanto maior a granularidade dos componentes do conhecimento, mais precisa será a representação do conhecimento do aluno, especialmente em domínios de conhecimento complexos, como em programação de computadores (Pelánek, 2017). No entanto, aumentar a granularidade de todos os CCs exigiria coletar mais dados da aluno em tarefas envolvendo estes CCs mais específicos e isso não foi possível para todos os CCs mais específicos que gostaríamos de gerar. Por esse motivo, alguns CCs possuem maior granularidade que outros.

5.2 Modelo BKT

Implementamos um modelo de conhecimento do aluno baseado no método *Bayesian Knowledge Tracing* (BKT), um tradicional método usado para modelar o conhecimento do aluno em sistemas tutores inteligentes nos últimos 30 anos (**Koedinger2006**; Agarwal et al., 2020; Corbett & Anderson, 1994; Slater & Baker, 2018; Yudelson et al., 2013) em domínios como matemática (**Koedinger2006**), leitura (Beck & Chang, 2007), genética (Corbett et al., 2010) e programação de computadores (Corbett & Anderson, 1994). O modelo correlaciona-se bem com resultados de testes de desempenho de alunos (Baker et al., 2010; Pardos & Heffernan, 2011). Sua capacidade preditiva é comparável a modelos recentes e complexos de inferência de conhecimento (Baker et al., 2011; Gong et al., 2010; Gowda et al., 2011; Khajah et al., 2016; Mao, 2018), como o *Deep Knowledge Tracing* (DKT), um modelo baseado em redes neurais (Khajah et al., 2016). Um estudo mostrou que quando comparado ao BKT clássico, o DKT tende a ter melhor desempenho. No entanto, quando extensões são adicionadas ao BKT clássico para tratar as suas limitações, o desempenho do BKT pode ser semelhante e, em alguns casos, até superior ao do DKT (Khajah

Tabela 1: Componentes de conhecimento.

#	**Componente de conhecimento (CC)**
1	Atributo de classe
2	Classe
3	Método construtor com inicialização de atributos
4	Método construtor com inicialização de atributos do tipo *array*
5	Método construtor com inicialização de atributos herdados
6	Método construtor padrão
7	Método de classe com parâmetros e com retorno
8	Método de classe com parâmetros e sem retorno
9	Método de classe sem parâmetros e com retorno
10	Método de classe sem parâmetros e sem retorno
11	Método principal (*main*)
12	Métodos de modificação e de acesso
13	Subclasse
14	Método *toString*
15	Entrada de *array*, instanciação, invocação de método e saída de *array*
16	Entrada, instanciação, invocação de método e saída
17	Entrada, invocação de método e saída
18	Escrita de *array*
19	Algoritmo formal
20	Incremento e decremento de atributos
21	Invocação de métodos
22	Leitura de *array* de tipo não primitivo
23	Leitura de *array* de tipo primitivo
24	Leitura de atributos
25	Operação aritmética
26	Repetição (*loop*)
27	Seleção de caso (*if*. . . *else*, *case*)
28	Seleção de caso com operação aritmética
29	Verificação de tipo de objeto

et al., 2016).

Uma vantagem do BKT sobre os modelos DKT é que ele fornece uma melhor interpretação do processo de inferência de conhecimento do que os modelos baseados em redes neurais, que funcionam como caixas pretas (Mao, 2018; Slater & Baker, 2018). Todas as etapas de inferência do modelo BKT são transparentes. O modelo BKT é tão simples de implementar que suas inferências podem ser obtidas através de fórmulas simples em uma planilha Excel. Além disso, os parâmetros e estimativas do BKT são fáceis de interpretar. Por se tratar de um modelo probabilístico, seus parâmetros e estimativas recebem valores no intervalo entre 0,0 e 1,0.

O modelo BKT clássico possui quatro parâmetros: 1) a probabilidade de o aluno dominar o componente do conhecimento antes da primeira oportunidade de aplicá-lo, ou seja, a probabilidade de ter aprendido no tempo zero (*p(L0)*); 2) a probabilidade de o aluno passar do estado de não domínio para o estado de domínio após uma oportunidade de aplicar o componente de

conhecimento, ou seja, uma probabilidade de transição (*p(T)*); 3) a probabilidade de o aluno aplicar incorretamente o componente de conhecimento quando está no estado de domínio, ou seja, uma probabilidade de deslize (do inglês *slip*) (*p(S)*); e 4) a probabilidade de o aluno aplicar corretamente o componente de conhecimento quando está no estado de não domínio, ou seja, uma probabilidade de adivinhação (do inglês *guess*) (*p(G)*). Esses parâmetros são ajustados (estimados) para cada componente do conhecimento. Para ajustar os parâmetros, nós utilizamos o algoritmo *Expectation Maximization* (EM) (Moon, 1996), um algoritmo eficiente e amplamente utilizado para ajustar os parâmetros dos modelos BKT (Pardos & Heffernan, 2010; Pelánek, 2017).

O modelo BKT também mantém as seguintes estimativas: 1) a probabilidade de o aluno dominar o componente de conhecimento após aplicá-lo, ou seja, a probabilidade de ter aprendido (*p(L)*); e 2) a probabilidade de o aluno aplicar corretamente o componente de conhecimento na próxima oportunidade de aplicá-lo (*p(Correto)*). O modelo mantém essas estimativas para cada componente de conhecimento e as atualiza a cada oportunidade de o aluno aplicá-lo.

Uma limitação do modelo BKT clássico é que ele assume que o aluno não esquece o componente de conhecimento. Quando um aluno domina um componente de conhecimento específico, temos *p(L)* = 1,0. Esse valor não regride, mesmo que o aluno apresente futuramente soluções incorretas para uma tarefa envolvendo o componente conhecimento. Para resolver esta limitação do modelo BKT clássico, usamos uma extensão do modelo BKT para considerar o esquecimento do aluno nos CCs, chamada *BKT+forget* (Badrinath et al., 2021). Essa versão inclui um quinto parâmetro, que corresponde à probabilidade de o aluno esquecer o componente do conhecimento (*p(F)*). O parâmetro *p(F)* foi modelado para ter comportamento oposto ao parâmetro *p(T)*. Enquanto *p(T)* indica a probabilidade de o aluno passar do estado de não domínio para o estado de domínio, *p(F)* indica a probabilidade de o aluno passar do estado de domínio para o estado de não domínio. No modelo BKT clássico, este parâmetro não existe, ou seja, *p(F)* = 0 (zero) (Khajah et al., 2016). Embora já tenha sido demonstrado o bom desempenho do modelo *BKT+forget* em comparação com os modelos clássicos BKT e DKT (Khajah et al. 2016), nossa pesquisa realizou testes com ambas as versões do BKT (BKT clássico e BKT+forget). O modelo *BKT+forget* superou o modelo clássico. Para avaliar o desempenho do modelo *BKT+forget* em inferir estimativas de conhecimento dos alunos, realizamos uma validação cruzada *k-fold* (k = 5). Obtivemos uma precisão de 0,831 e um valor RMSE (raiz do erro quadrático médio[3]) de 0,299. Esses resultados são semelhantes aos melhores valores encontrados na literatura para modelos BKT (Lin & Chi, 2016; Nagatani et al., 2019; S. Wang et al., 2017; Yudelson et al., 2013).

Na presente pesquisa, utilizamos os valores das estimativas e dos parâmetros do modelo *BKT+forget* para gerar atributos de dados para os modelos de aprendizado de máquina da abordagem da hipótese, os quais serão detalhados mais adiante no artigo.

6 BlueJ Afetivo

O ambiente BlueJ Afetivo é uma versão modificada da IDE BlueJ, projetado para o ensino de programação em Java. O BlueJ Afetivo foi desenvolvido especialmente para esta pesquisa. Além de coletar os dados de interação do aluno durante a codificação dos exercícios para futuramente

[3]Do inglês *root mean-square error*.

usarmos no treinamento dos modelos de IA, ele apresenta ao aluno na sua interface gráfica os exercícios atribuídos pelo professor. Ele também utiliza um Juiz Online [4] para avaliar as soluções do estudante para os exercícios e dar *feedbacks*.

A interface gráfica do BlueJ Afetivo oferece uma opção para o estudante visualizar a lista de exercícios atribuídos pelo professor e abrir a lista de objetivos dos exercícios (ver Figura 1). Cada exercício é organizado como uma lista de objetivos. Durante a codificação, o aluno pode indicar qual objetivo do exercício vai codificar e, ao finalizar a codificação, pode enviá-la para a avaliação do Juiz Online. A tela do editor de código (Ver Figura 2) coleta informações sobre cada tecla digitada e ações do mouse. Além disso, a qualquer momento da codificação o aluno pode relatar sua confusão apertando o botão "CONFUSO". Utilizamos estes relatos para gerarmos rótulos de emoções que são vinculados às amostras de treino e validação dos modelos de IA.

Figura 1: BlueJ Afetivo: Tela de objetivos do exercício.

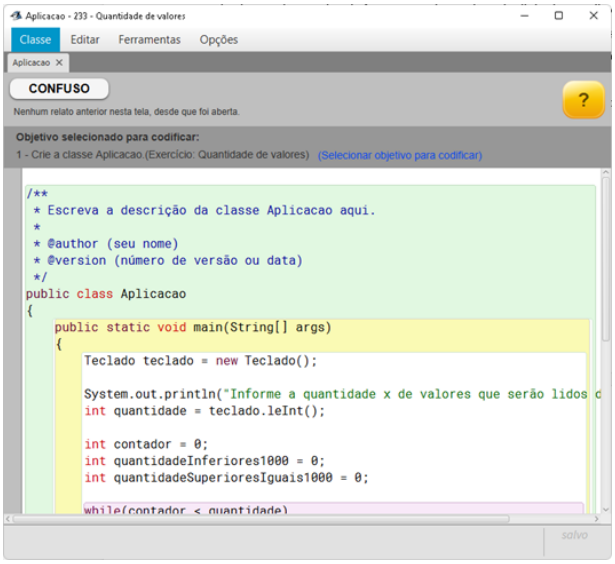

Figura 2: BlueJ Afetivo: Tela do editor de código.

7 Treinando modelos de detecção de confusão

A partir desta Seção, discutimos os métodos e os resultados da pesquisa. Com o objetivo de verificar se a abordagem da nossa hipótese supera a abordagem dos trabalhos relacionados (**Objetivo 1**) e identificar os melhores tamanhos de segmentos de dados sobre o comportamento dos alunos e os melhores algoritmos (**Objetivo 2**), geramos diversos modelos de aprendizado de máquina supervisionado para representar a abordagem do estudo (chamaremos de **modelos da hipótese**) e modelos que representam a abordagem dos trabalhos relacionados (chamaremos de **modelos *baseline***). Os modelos da hipótese foram validados[5] com amostras compostas por atributos de dados sobre estimativas de conhecimento dos alunos juntamente de atributos de dados sobre a interação dos alunos com o ambiente de programação (BlueJ Afetivo). Os modelos *baseline*, por

[4]Nós adaptamos para o nosso estudo a implementação do Juiz Online apresentado por **alves2014ambiente<empty citation>**.

[5]Treinados com amostras de treinamento e suas previsões validadas com amostras de teste.

sua vez, foram validados com amostras compostas apenas por atributos de dados sobre a interação do aluno com o ambiente. A única diferença é que os modelos da hipótese utilizaram amostras com estimativas do conhecimento dos alunos, enquanto os modelos *baseline* não o fizeram.

7.1 Coleta de Dados

Os modelos foram treinados com amostras compostas por segmentos de dados gerados a partir de *logs* coletados durante a interação dos alunos participantes da pesquisa com o ambiente de programação BlueJ Afetivo enquanto realizavam exercícios de programação. Participaram da coleta 62 alunos de três turmas de ensino superior e de duas turmas de ensino técnico, ambas turmas relacionadas a disciplinas de introdução à programação de computadores. Os alunos forneceram consentimento e assentimento para participar do estudo. O BlueJ Afetivo foi a principal ferramenta de programação utilizada pelos alunos para realizar os exercícios atribuídos pelos professores ao longo de cinco meses, de fevereiro a julho de 2021. Os alunos tiveram liberdade para decidir a ordem dos exercícios a resolver. A coleta de dados ocorreu durante a pandemia de COVID-19, quando os alunos participavam de aulas remotas em suas residências. Os alunos tiveram que instalar o BlueJ Afetivo em seus computadores para resolver os exercícios de programação.

No BlueJ Afetivo, os alunos visualizavam os enunciados dos exercícios atribuídos pelos professores. Cada enunciado era apresentado como uma lista de objetivos. Cada objetivo estava associado a um ou dois componentes de conhecimento de programação (CC primário e/ou secundário). Isso permitiu acompanhar quais componentes de conhecimento o aluno estava envolvido durante a codificação. Para cada objetivo de exercício, o aluno podia enviar sua solução para ser avaliada pelo Juiz Online integrado ao BlueJ Afetivo. O BlueJ afetivo é descrito na Seção 6.

Os *feedbacks* do Juiz Online indicam se a solução do aluno para o objetivo de exercício é correta ou incorreta. Estes *feedbacks* foram utilizados pelo modelo *BKT+forget* (ver Seção 5) para gerar as estimativas sobre o conhecimento do aluno nos componentes de conhecimento associados aos objetivos dos exercícios. As estimativas incluem a probabilidade do aluno dominar o CC antes da primeira oportunidade de aplicá-lo (*p(L0)*), a probabilidade do aluno dominar o CC após a última oportunidade de aplicá-lo (*p(L)*), a probabilidade do aluno aplicar corretamente o CC na próxima oportunidade (*p(Correto)*), a probabilidade do aluno aprender o CC (*p(T)*), a probabilidade do aluno esquecer o CC (*p(F)*), a probabilidade do aluno cometer um deslize no CC (*p(S)*) e a probabilidade do aluno de adivinhar o CC (*p(G)*). Essas estimativas foram utilizadas na geração dos atributos relacionados às estimativas de conhecimento do aluno.

Cada registro de *log* foi gerado a partir de eventos no BlueJ Afetivo, incluindo teclas pressionadas ou liberadas pelo aluno, cliques do *mouse*, movimento do *mouse* (início ou parada), erros de sintaxe durante a codificação, submissão de soluções de objetivo para avaliação do Juiz Online e os *feedbacks* da avaliação do Juiz Online. No total, foram coletados 13,526664 milhões de registros de *log*.

Os rótulos emocionais (“CONFUSÃO” ou “NÃO CONFUSÃO”) atribuídos às amostras foram obtidos por meio de autorrelatos. Sempre que o aluno se sentia confuso durante as tarefas, ele apertava um botão na interface do BlueJ Afetivo (ver Figura 2). Ao todo, foram coletados 1.147 relatos de confusão. Com o objetivo de lembrar os alunos a relatarem sua confusão, foi orientado aos professores das turmas participantes que a cada início de aula os alunos fossem lembrados sobre o relato da confusão durante as tarefas de aprendizagem. Além disso, no início

da coleta de dados, nós participamos de uma aula em cada turma e promovemos uma discussão com os alunos sobre o que é a confusão e sobre como reconhecê-la no contexto de tarefas de programação de computadores. Também disponibilizamos um vídeo aos alunos, antes do início da participação deles (também disponível na interface do BlueJ Afetivo), com orientações sobre o que é a confusão e sobre como identificá-la. Também foi implementado um sinal piscante no botão de relato da confusão, a cada 10 minutos, para lembrar os alunos para relatarem a confusão, quando a sentissem.

Uma característica interessante da abordagem adotada é que os relatos de confusão são obtidos de maneira espontânea. Já foi verificado que quando a coleta de rótulos *ground-truth* é realizada em momentos fixos, de forma forçada, apesar de obter mais rótulos de emoção, o desempenho dos modelos classificadores tende a piorar em comparação quando são treinados com rótulos obtidos espontaneamente (Bosch et al., 2014).

7.2 Amostras

Foram gerados 10 conjuntos de amostras de segmentos de dados. Cada conjunto de amostras representa uma configuração de segmento de dados de tempo fixo (5, 10, 20, 40, 60, 90, 120, 180, 240 e 360 segundos). As amostras utilizadas nos modelos da hipótese são compostas por 65 atributos de dados que sintetizam comportamentos observados durante o intervalo de tempo, relacionados a interações do aluno com o ambiente e com informações sobre o conhecimento do aluno. Os atributos gerados foram classificados em diferentes tipos: a) teclado, derivados dos registros de *log* de pressionamento de tecla (8 atributos); b) *mouse*, derivados dos registros de *log* de cliques e movimentos do *mouse* (6 atributos); c) erros de sintaxe, derivados dos registros de *log* de erros de sintaxe na codificação (1 atributo); d) submissão de solução, derivados dos registros de *log* de submissões ao Juiz Online (5 atributos); e) interações gerais com a interface gráfica (3 atributos); e f) conhecimento do aluno, derivados das estimativas de conhecimento inferidas pelo modelo de conhecimento do aluno *BKT+forget* (42 atributos). As amostras utilizadas para treinar os modelos *baseline* são as mesmas, porém sem os atributos de dados relacionados ao conhecimento do aluno. A Tabela 2 apresenta alguns dos 65 atributos de dados gerados.

Para facilitar a compreensão sobre as características das amostras, apresentamos alguns exemplos. Um segmento de dados de 10 segundos vinculado ao rótulo de "CONFUSÃO" para um modelo da hipótese reflete dados de interação e de conhecimento do aluno relativos aos 10 segundos anteriores a um relato de confusão do aluno. Outro exemplo é o seguinte: um segmento de dados de 20 segundos vinculado ao rótulo de "NÃO CONFUSÃO" para um modelo *baseline* reflete somente dados de interação do aluno relativos a 20 segundos em que o aluno não relatou estar confuso. A Figura 3 mostra exemplos de registros do banco de dados com fragmentos de amostras. A primeira linha representa um registro de amostra referente a uma janela de observação de 20 segundos, no qual o aluno relatou estar confuso. O registro mostra que, nos 20 segundos anteriores ao relato de confusão, foram pressionadas 14 teclas (*quant_teclas*) e realizados cinco movimentos de *mouse* (*quant_movimentos_mouse*). Além disso, a média das probabilidades de o aluno acertar os componentes de conhecimento do objetivo de exercício em andamento (*media_pcorreto_ccs*) era de 82%, e a probabilidade de o aluno dominar os componentes de conhecimento desse objetivo (*media_pln_ccs*) era de 71%.

A Tabela 3 apresenta a quantidade de amostras geradas para cada janela de observação.

Tabela 2: Exemplos de atributos de dados gerados para as amostras.

Atributo de dados	**Tipo**
Quantidade de teclas pressionadas	Teclado
Quantidade de teclas de *backspace* pressionadas	Teclado
Quantidade de teclas *delete* pressionadas	Teclado
Velocidade de digitação	Teclado
Quantidade de movimentos do mouse	Mouse
Quantidade de cliques do mouse	Mouse
Tempo total dos movimentos do mouse	Mouse
Quantidade de erros de sintaxe	Erro de sintaxe
Quantidade de submissões de soluções do objetivo do exercício	Submissões
Quantidade de submissões de soluções incorretas do objetivo do exercício	Submissões
Quantidade de submissões de soluções corretas do objetivo do exercício	Submissões
Tempo total ocioso	Outras interações
Quantidade de intervalos ociosos	Outras interações
Média de *p(Correto)* nos CCs do objetivo do exercício	**Conhecimento**
Média de *p(L)* nos CCs do objetivo do exercício	**Conhecimento**
Média de *p(T)* nos CCs do objetivo do exercício	**Conhecimento**

aluno	janela	confuso	nao_confuso	quant_teclas	quant_movimentos_mouse	media_pcorreto_ccs	media_pln_ccs
495	20.000	1	0	14.000	5.000	0.8201698320	0.7175748379
495	20.000	1	0	0.000	15.000	0.8201698320	0.7175748379
495	20.000	1	0	0.000	19.000	0.6786934689	0.5287970711
495	20.000	0	1	11.000	20.000	0.7806274778	0.8552131155
495	20.000	0	1	32.000	8.000	0.8981124742	0.8848881743
495	20.000	0	1	2.000	18.000	0.9139540613	0.9675229613

Figura 3: Fragmento de amostras para um segmento de dados de 20 segundos.

A tabela mostra um balanceamento na quantidade de amostras para os rótulos “CONFUSÃO” e “NÃO CONFUSÃO”, uma característica importante para o treinamento adequado dos modelos de aprendizado de máquina em cada classe de saída (Raschka, 2018).

7.3 Treinamento dos modelos

A Figura 4 mostra o esquema dos modelos gerados. Cada modelo foi desenvolvido para um algoritmo específico e para um conjunto específico de amostras. Foram selecionados algoritmos de aprendizado de máquina baseados em árvores (*Decision Tree*), seguindo a abordagem dos trabalhos relacionados, e algoritmos que utilizam conjuntos de árvores (*Random Forest* e *XGBoost*), recomendados para dados tabulares, como os utilizados nesta pesquisa. Os demais algoritmos são algoritmos tradicionais de aprendizado de máquina (*Logistic Regression*, *Support Vector Machines*, *Naive Bayes*, *k-Nearest Neighbors*, *Stochastic Gradient Descent*, *AdaBoost*) (Kubat, 2017) e um algoritmo de rede neural (*Multilayer Perceptron*). Também usamos dez conjuntos de amostras para os modelos do estudo (listados acima à esquerda) e dez conjuntos para os modelos *baseline* (listados abaixo à esquerda), totalizando 100 modelos da hipótese e 100 modelos *baseline*.

Tabela 3: Quantidade de amostras geradas para cada configuração de janela nos segmentos de dados.

Segmentos (segundos)	Total CONFUSÃO	Total NÃO CONFUSÃO	Total Geral
5	963	963	1926
10	949	949	1898
20	944	944	1888
40	933	933	1866
60	924	924	1848
90	902	902	1804
120	883	883	1766
180	828	828	1656
240	771	771	1542
360	689	689	1378

A Figura 5 mostra o *pipeline* que usamos para gerar e validar cada modelo. A entrada do *pipeline* é composta por amostras não normalizadas e um algoritmo não otimizado. Na primeira etapa, normalizamos as amostras, que consistiu em ajustar os valores individuais dos atributos de dados para favorecer o comportamento dos algoritmos e melhorar seu desempenho (Kubat, 2017). O valor de cada atributo em uma amostra foi modificado para ter a propriedade de uma distribuição normal padrão com $\mu = 0$ (o valor médio do atributo em todas as amostras) e $\sigma = 1$ (o desvio padrão). Na segunda etapa, fizemos a otimização dos hiperparâmetros, método utilizado para otimizar o desempenho de algoritmos (Raschka, 2018). Utilizamos o método *grid search* com validação cruzada *k-fold* (k = 10) e *folds* estratificados.

Na terceira etapa, de validação dos modelos, executamos dois métodos de validação. Em ambos os métodos, consideramos as seguintes métricas: acurácia, *kappa*, *AUC ROC*, *precisão*, *recall* e *F1*. No **primeiro método de validação**, utilizou-se validação cruzada *k-fold* com k = 10 e amostras estratificadas nos *folds*. O objetivo foi verificar o poder de generalização dos modelos para diferentes alunos. O método divide aleatoriamente as amostras em *k* conjuntos (*folds*), com quantidades de amostras iguais em cada conjunto, e realiza *k* iterações de treinamento e teste. Em cada iteração, *k-1* (9) conjuntos de amostras são usados no treinamento do modelo, enquanto o conjunto restante de amostras é usado para testar o modelo. Assim, é calculada a média dos desempenhos para cada métrica nas *k* iterações (Wong, 2015). O método de estratificação garante que na alocação aleatória de amostras aos *folds*, cada *fold* contenha um conjunto balanceado de amostras para cada classe de saída (Raschka, 2018), ou seja, um conjunto balanceado de amostras rotuladas como amostras de confusão e de não confusão.

No **segundo método de validação**, os modelos foram treinados com amostras de alunos de um determinado nível de ensino e testados com amostras de alunos de outro nível de ensino. O objetivo foi verificar o poder de generalização dos modelos para alunos com diferentes níveis de escolaridade, com diferentes idades e com diferentes conhecimentos prévios e experiências anteriores em programação de computadores. Os participantes da pesquisa foram estudantes do ensino técnico e superior. A aplicação de um teste *Mann-Whitnney U* encontrou uma diferença significativa de idade ($p < 0{,}05$) entre os alunos dos níveis de ensino. Além disso, através da aplicação de um questionário criado por nós, foi encontrada diferença significativa (*Mann-Whitney U*, $p < 0{,}05$) entre estudantes dos dois níveis de ensino em relação ao conhecimento prévio nos

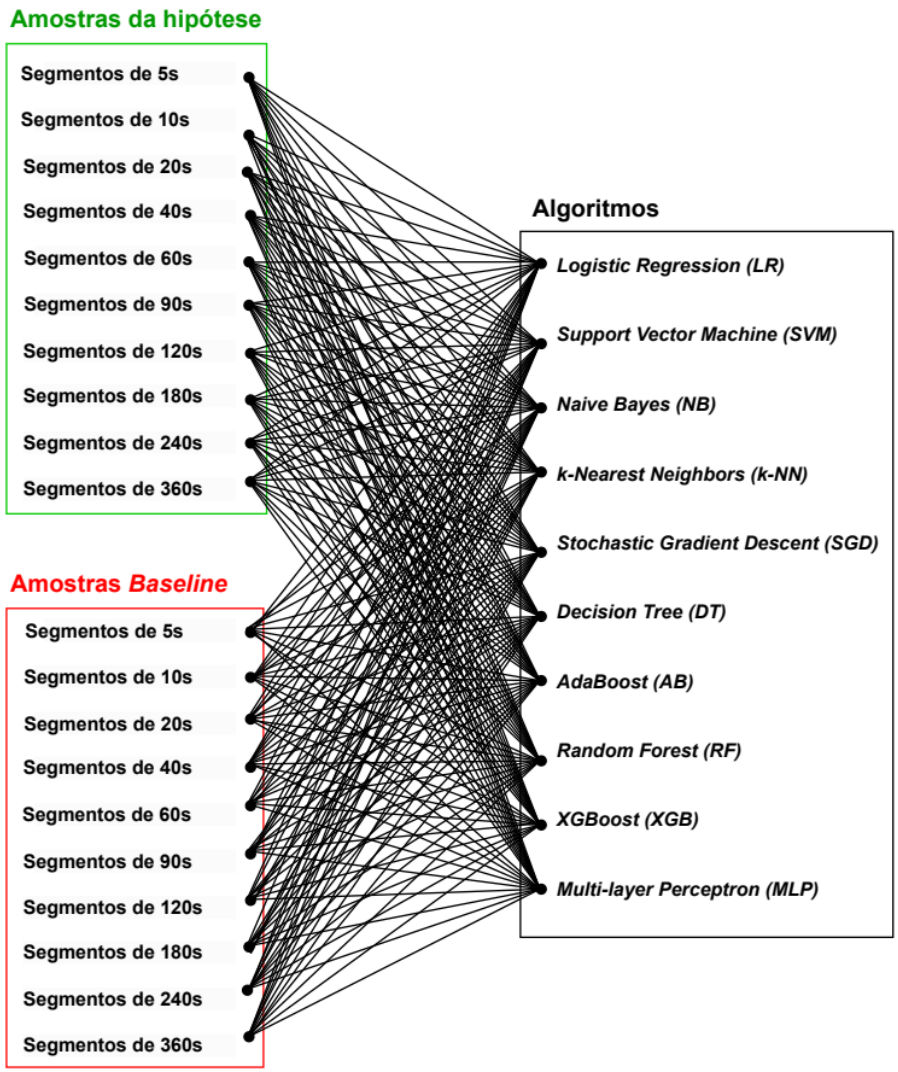

Figura 4: Esquema dos modelos gerados na pesquisa.

componentes do conhecimento e no tempo de experiência profissional com programação. Os alunos das turmas de ensino técnico são mais jovens, têm menos conhecimentos prévios e menos experiência profissional em programação. Verificamos também se havia diferença em relação à competência emocional dos alunos. A competência emocional (ou inteligência emocional) é considerada a capacidade de um indivíduo perceber, expressar e regular emoções (Mayer et al., 2004). Após a aplicação de um questionário de habilidades e competências emocionais (Taksic, 2000), não foi encontrada diferença significativa entre os alunos dos dois níveis de ensino.

O Código 1 apresenta um pseudocódigo que descreve o fluxo de geração dos modelos. O pseudocódigo explicita o funcionamento da *pipeline* para cada modelo. Primeiramente, nas linhas 1 e 2, são carregados em memória os segmentos de cada janela (5, 10, 20, 40, 60, 90, 120, 180, 240 e 360) e os algoritmos de aprendizado de máquina. Para cada janela (linha 3), são executadas as operações descritas a seguir. São carregadas em memória as amostras relativas à janela (linha 4). Na linha 5, as amostras carregadas são atualizadas por uma operação de normalização. Para cada um dos algoritmos (linha 6), são executadas as seguintes operações. São obtidos os melhores

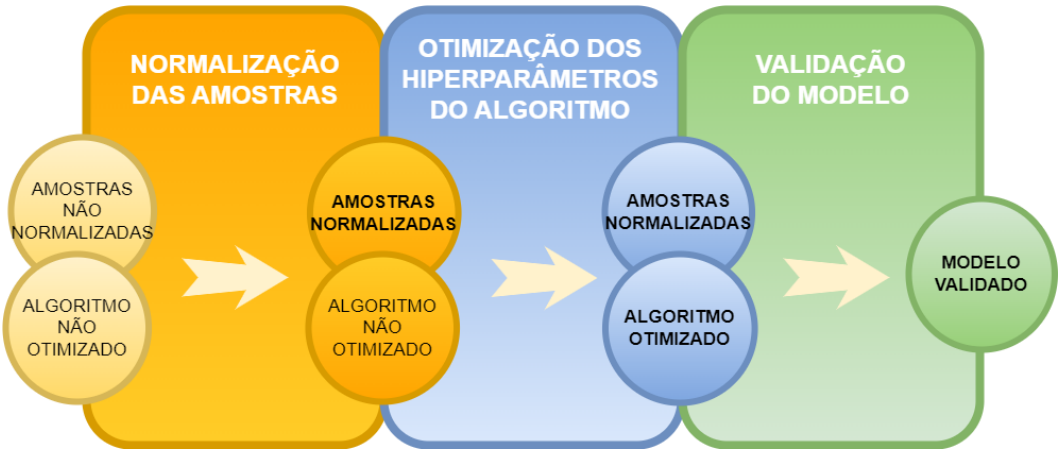

Figura 5: *Pipeline* de geração e validação dos modelos.

hiperparâmetros para o algoritmo e as amostras carregadas (linha 7). Na linha 8, é realizada a validação do modelo composto pelo algoritmo, seus hiperparâmetros e as amostras carregadas. Os resultados de desempenho na validação são guardados em memória (linha 9). Para cada janela, é guardado em memória o modelo que obteve o melhor desempenho na métrica de acurácia.

```
1. janelas <- (5,10,20,40,60,90,120,180,240,360]
2. algoritmos <- [LR, SVM, NB, KNN, SGD, DT, AB, RF, XGB, MLP]
3. para cada janela em janelas faça
4.    amostras <- obtemAmostras(janela)
5.    amostras <- normaliza(amostras)
6.    para cada algoritmo em algoritmos faça
7.        hiperParams <- obtemMelhoresHiperParams(algoritmo, amostras)
8.        resultados[janela, algoritmo] <- valida(algoritmo, hiperParams, amostras)
9.    melhorModelo[janela] <- obtemMelhorModelo(resultados[janela])
```

Código 1: Pseudocódigo de geração e validação dos modelos.

7.4 Comparação dos modelos da hipótese e dos modelos *baseline*

Para comparar a acurácia preditiva dos modelos da hipótese e *baseline* em cada janela de observação, foram aplicados testes de hipótese estatística. O método utilizado foi o teste McNemar (1947), um teste estatístico não paramétrico recomendado para comparar modelos de aprendizado de máquina (Dietterich, 1998; Raschka, 2018). A hipótese nula é que a acurácia preditiva dos dois modelos comparados é igual, e a hipótese alternativa é que é diferente. Considerando um nível de significância estatística de 0,05 ($\alpha = 0{,}05$), caso seja obtido um valor de p (*p-value*) inferior a α, é possível rejeitar a hipótese nula e aceitar a hipótese alternativa (Raschka, 2018).

7.5 Resultados e discussões

Antes de apresentar os resultados, consideramos importante descrever as métricas de desempenho dos modelos: acurácia, *AUC ROC*, *kappa*, precisão, *recall* e *F1*. A **acurácia** mede o desempenho geral do modelo, ou seja, o percentual de classificações corretas (“CONFUSÃO” ou “NÃO CONFUSÃO”) em relação à quantidade total de amostras (Kubat, 2017). A métrica ***AUC ROC***

mede o desempenho do modelo em identificar corretamente a confusão nas amostras de exemplos positivos (quando o aluno estava confuso) e a não confusão nas amostras de exemplos negativos (quando o aluno não estava confuso) (Kubat, 2017). A métrica ***kappa*** é uma medida baseada na acurácia que considera a possibilidade de a concordância do modelo ter ocorrido por acaso para fazer um ajuste na medida (Cohen, 1960). Diferente das demais métricas, que possuem valores entre 0,0 e 1,0, o *kappa* recebe valores entre -1,0 e 1,0. Valores iguais a 0,0 (zero) indicam que a concordância é a mesma que a esperada por acaso. Valores iguais a -1,0 indicam nenhuma concordância (Cohen, 1960). A faixa de valores de *kappa* entre 0,4 e 0,6 é considerada uma concordância justa (apertada) e moderada, enquanto a faixa de valores entre 0,6 e 0,75 é considerada boa. A faixa com valores maiores que 0,75 é considerada excelente (Knottnerus & Tugwell, 2010). A métrica de **precisão** mede o percentual de verdadeiros positivos entre todos os exemplos que o modelo rotulou como positivo. A precisão mede a probabilidade de o modelo estar correto quando classifica um exemplo como positivo (Kubat, 2017). No contexto do trabalho, essa métrica mede a precisão do modelo quando ele indica que o aluno está confuso. Se num contexto educacional for um problema o modelo indicar que o aluno está confuso quando não está, a precisão é uma métrica que deve ser levada em consideração. O ***recall*** mede o percentual de exemplos positivos que são corretamente identificados como positivos (Kubat, 2017). Se num contexto educacional for fundamental o modelo identificar a confusão quando o aluno está confuso, o *recall* é uma métrica que deve ser levada em consideração. Por fim, a métrica **F1** é uma média harmônica entre a precisão e o *recall* (Kubat, 2017).

A Tabela 4 mostra o desempenho dos modelos da hipótese (denotado por H) e dos modelos *baseline* (denotado por B) que apresentaram a melhor acurácia em cada janela de segmento de dados após a aplicação do primeiro método de validação. Os algoritmos utilizados nesses melhores modelos também são apresentados. A Tabela 4 mostra que para todas as métricas de desempenho geral (acurácia, *AUC ROC* e *kappa*), em todos os segmentos, os melhores modelos da hipótese superaram os modelos *baseline*. Os maiores valores em cada métrica são destacados. O melhor resultado de acurácia (0,897) foi encontrado em uma janela de 10 segundos, de *AUC ROC* (0,950) em uma janela de 10 segundos e de *kappa* (0,794) também em uma janela de 10 segundos.

Na métrica de precisão, nossa abordagem obteve valores superiores em todos os segmentos, com o melhor resultado encontrado em um segmento de 10 segundos (precisão = 0,879). A precisão mede a porcentagem de acertos quando o modelo indica que o aluno está confuso. Esta métrica é útil em ambientes educacionais que tomam medidas imediatas quando uma confusão é detectada. Caso o ambiente atue em função do detector, sinalizando que o aluno está confuso, essa detecção deve estar correta.

A diferença entre as abordagens na métrica de *recall* foi estreita. Em um dos segmentos (5 segundos), o melhor valor de *recall* foi maior no melhor modelo *baseline* (*recall* = 0,930). Nos segmentos de 10, 20, 40, 60, 180, 240 e 360 segundos, os melhores modelos do nosso estudo superaram os melhores modelos *baseline* no *recall*. A métrica de *recall* mede a porcentagem de exemplos positivos corretamente identificados como positivos pelo classificador, ou seja, se o aluno estiver confuso, o modelo identifica a confusão. Num contexto educacional onde é estritamente necessário que o ambiente identifique a confusão do aluno quando esta ocorre, os modelos do nosso estudo não oferecem uma vantagem tão clara.

Em relação à métrica *F1*, em todos os segmentos os melhores modelos da hipótese demonstraram valores superiores aos melhores modelos *baseline*. O melhor valor de *F1* encontrado

Tabela 4: Desempenho dos melhores modelos.

Segm.		Alg.	Acur.	AUC ROC	Kappa	Prec.	Recall	F1	Teste McNemar chi2	Teste McNemar p
5	H	XGB	0,891	0,948	0,783	0,872	0,925	0,894	39619,565	**<0,001**
	B	XGB	0,764	0,816	0,529	0,709	**0,930**	0,792		
10	H	RF	**0,897**	**0,950**	**0,794**	**0,879**	0,923	**0,899**	28013,880	**<0,001**
	B	XGB	0,766	0,832	0,532	0,731	0,910	0,788		
20	H	XGB	0,877	0,935	0,754	0,862	0,914	0,879	28408,503	**<0,001**
	B	XGB	0,779	0,844	0,559	0,743	0,894	0,795		
40	H	XGB	0,848	0,919	0,696	0,832	0,877	0,852	4222,340	**<0,001**
	B	RF	0,785	0,849	0,571	0,755	0,866	0,797		
60	H	XGB	0,843	0,911	0,686	0,824	0,874	0,848	17500,564	**<0,001**
	B	RF	0,787	0,848	0,575	0,757	0,853	0,800		
90	H	XGB	0,821	0,904	0,643	0,809	0,852	0,825	36800,270	**<0,001**
	B	MLP	0,775	0,834	0,551	0,748	0,862	0,788		
120	H	RF	0,822	0,901	0,645	0,813	0,838	0,825	3103,030	**<0,001**
	B	RF	0,766	0,843	0,532	0,748	0,839	0,779		
180	H	XGB	0,807	0,888	0,615	0,800	0,822	0,810	32314,540	**<0,001**
	B	MLP	0,759	0,820	0,519	0,744	0,804	0,765		
240	H	XGB	0,802	0,867	0,604	0,798	0,809	0,803	40200,247	**<0,001**
	B	LG	0,749	0,825	0,499	0,741	0,780	0,757		
360	H	RF	0,796	0,869	0,592	0,800	0,793	0,795	28708,417	**<0,001**
	B	MLP	0,739	0,811	0,479	0,733	0,775	0,747		

ocorreu em um segmento de 10 segundos (*F1* = 0,899).

Outra informação interessante apresentada na Tabela 4 é que o algoritmo com mais ocorrências entre os melhores modelos é o *XGBoost* (XGB) (10 ocorrências), seguido por *Random Forest* (RF) (6 ocorrências), *Multilayer Perceptron* (MLP) (3 ocorrências) e Regressão Logística (RL) (1 ocorrência). Os resultados apoiam discussões na literatura que recomendam o uso dos algoritmos *XGBoost* e *Random Forest* para dados tabulares (Shwartz-Ziv & Armon, 2022), como os utilizados nesta pesquisa.

A Tabela 4 também mostra que a aplicação do teste estatístico de McNemar para comparar os melhores modelos da hipótese e os melhores modelos *baseline* encontrou uma diferença significativa de acurácia preditiva em todas as configurações de janelas de segmentos de dados, com valor de *p* sempre inferior a 0,05. Esses resultados indicam que o desempenho superior encontrado nos melhores modelos da hipótese em todas as métricas de desempenho geral (acurácia, *AUC ROC* e *kappa*) é significativo.

A Figura 6 ilustra o desempenho dos melhores modelos da hipótese e os melhores modelos *baseline* em cada métrica e para cada tamanho de segmento. A figura deixa claro que, com exceção do *recall*, o desempenho superior do melhor modelo da hipótese foi mais evidente nos segmentos de 5, 10 e 20 segundos. Verificar o tamanho adequado dos segmentos de dados foi o segundo objetivo do nosso estudo. Os resultados encontrados nos mostram que janelas de observação maiores que 20 segundos podem reduzir o desempenho dos modelos.

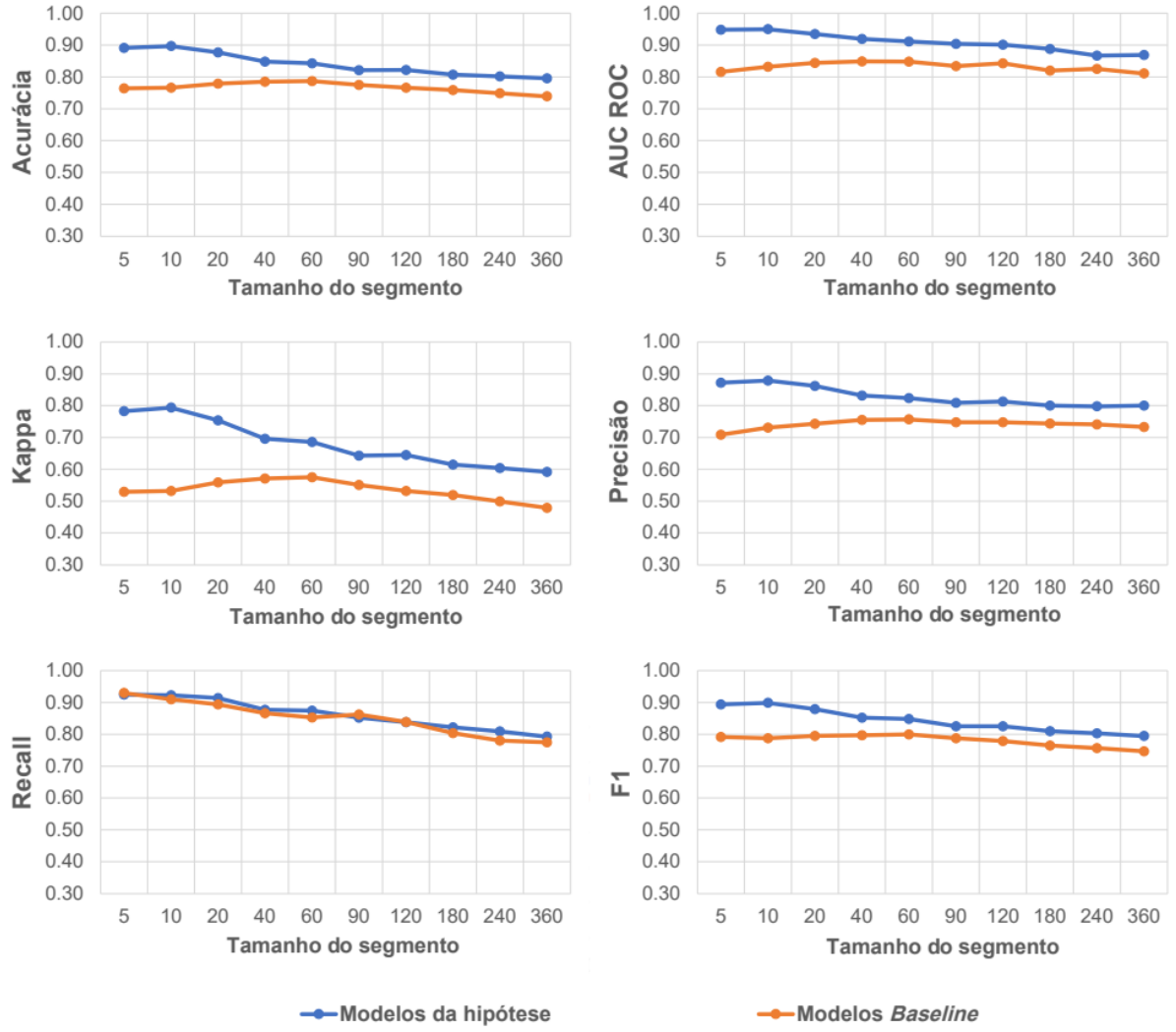

Figura 6: Resultados dos melhores modelos do estudo e *baseline*.

Não é nossa intenção comparar diretamente nossos resultados com aqueles apresentados nos trabalhos relacionados, pois as amostras utilizadas nesses trabalhos diferem das nossas. Nossa pesquisa gerou modelos que apenas representam a abordagem conceitual dos trabalhos relacionados (dados de interação) e a abordagem apresentada por nosso estudo (dados de interação + dados sobre estimativas de conhecimento do aluno). Contudo, é possível fazer uma comparação indireta dos resultados. Os três trabalhos relacionados limitaram seus resultados às métricas de acurácia, precisão e *kappa*. O melhor valor de *kappa* dos trabalhos relacionados foi 0,860, superior ao melhor valor encontrado em nossa pesquisa (0,794). A melhor precisão dos trabalhos relacionados foi de 0,74 (um único trabalho utilizou esta métrica). O melhor modelo de nosso estudo encontrou um valor superior (0,879) de precisão. O melhor resultado de acurácia dos trabalhos relacionados foi de 0,77 (um único estudo utilizou essa métrica), valor inferior ao melhor valor de acurácia encontrado em nossa pesquisa (0,897).

Como terceiro objetivo do nosso estudo, avaliamos o poder de generalização das abordagens para estudantes de diferentes níveis de ensino. Para atingir esse objetivo, foi utilizado o segundo método de validação. A Tabela 5 apresenta o desempenho dos modelos validados em

dois cenários. No primeiro cenário (Cenário 1), os modelos foram treinados com amostras de alunos matriculados no ensino técnico e testados com amostras de alunos matriculados no ensino superior. No segundo cenário (Cenário 2) , os modelos foram treinados com dados de alunos do ensino superior e testados com dados de alunos do ensino técnico.

Tabela 5: Desempenho dos melhores modelos nos cenários 1 e 2.

Segm.		**Alg.**	**Acur.**	***AUC ROC***	***Kappa***	**Prec.**	***Recall***	***F1***	**Teste McNemar** ***chi2***	***p***
Treinamentos com o ensino técnico e testes com o ensino superior										
5	H	SGD	**0,841**	**0,897**	**0,682**	**0,896**	**0,923**	**0,848**	9524497,00	**<0,001**
	B	ADB	0,685	0,730	0,370	0,630	**0,923**	0,739		
10	H	SGD	0,831	0,895	0,663	0,837	0,856	0,834	7924163,00	**<0,001**
	B	ADB	0,683	0,728	0,366	0,633	0,913	0,742		
20	H	SGD	0,791	0,853	0,582	0,774	0,856	0,799	2967804,00	**<0,001**
	B	LR	0,711	0,766	0,423	0,657	0,908	0,755		
Treinamentos com o ensino superior e testes com o ensino técnico										
5	H	LR	0,866	0,935	0,733	0,894	0,991	0,878	1314,935	**<0,001**
	B	NB	0,817	0,876	0,635	0,810	0,924	0,835		
10	H	SGD	**0,876**	**0,940**	**0,752**	**0,940**	**0,997**	**0,888**	3475,177	**<0,001**
	B	NB	0,798	0,867	0,596	0,800	0,911	0,818		
20	H	SGD	0,840	0,914	0,680	0,903	0,986	0,849	481,283	**0,028**
	B	LR	0,805	0,871	0,611	0,820	0,802	0,800		

Em ambos os cenários, os melhores valores de desempenho em todas as métricas foram encontrados nos melhores modelos da hipótese, com exceção do *recall*, no primeiro cenário. Os melhores resultados para cada métrica são destacados em negrito. A tabela 5 mostra apenas os resultados para os segmentos de 5, 10 e 20 segundos, pois foi apenas nestas janelas que encontramos uma diferença estatística significativa.

Tabela 6: Desempenho médio dos melhores modelos nos cenários 1 e 2.

Segmento (segundos)		**Algoritmo**	**Acurácia**	***AUC ROC***	***Kappa***	**Precisão**	***Recall***	**F1**
5	H	SGD	0,852	0,908	0,704	0,861	**0,925**	**0,862**
	B	LR	0,745	0,803	0,491	0,717	0,915	0,781
10	H	SGD	**0,854**	**0,917**	**0,708**	**0,889**	0,914	0,861
	B	ADB	0,733	0,793	0,466	0,716	0,886	0,763
20	H	SGD	0,815	0,883	0,631	0,825	0,865	0,824
	B	LR	0,758	0,816	0,517	0,739	0,855	0,777

A Tabela 6 mostra a média dos valores obtidos para as métricas nos dois cenários. Os maiores valores foram encontrados nos melhores modelos da hipótese para segmentos de 5 e 10 segundos. A Figura 7 apresenta os resultados da Tabela 6, mostrando que os melhores modelos do nosso estudo superam os modelos *baseline* nas métricas de acurácia, AUC ROC, kappa, precisão e F1 para tamanhos de segmento de até 20 segundos. A Figura 7 também mostra que em janelas

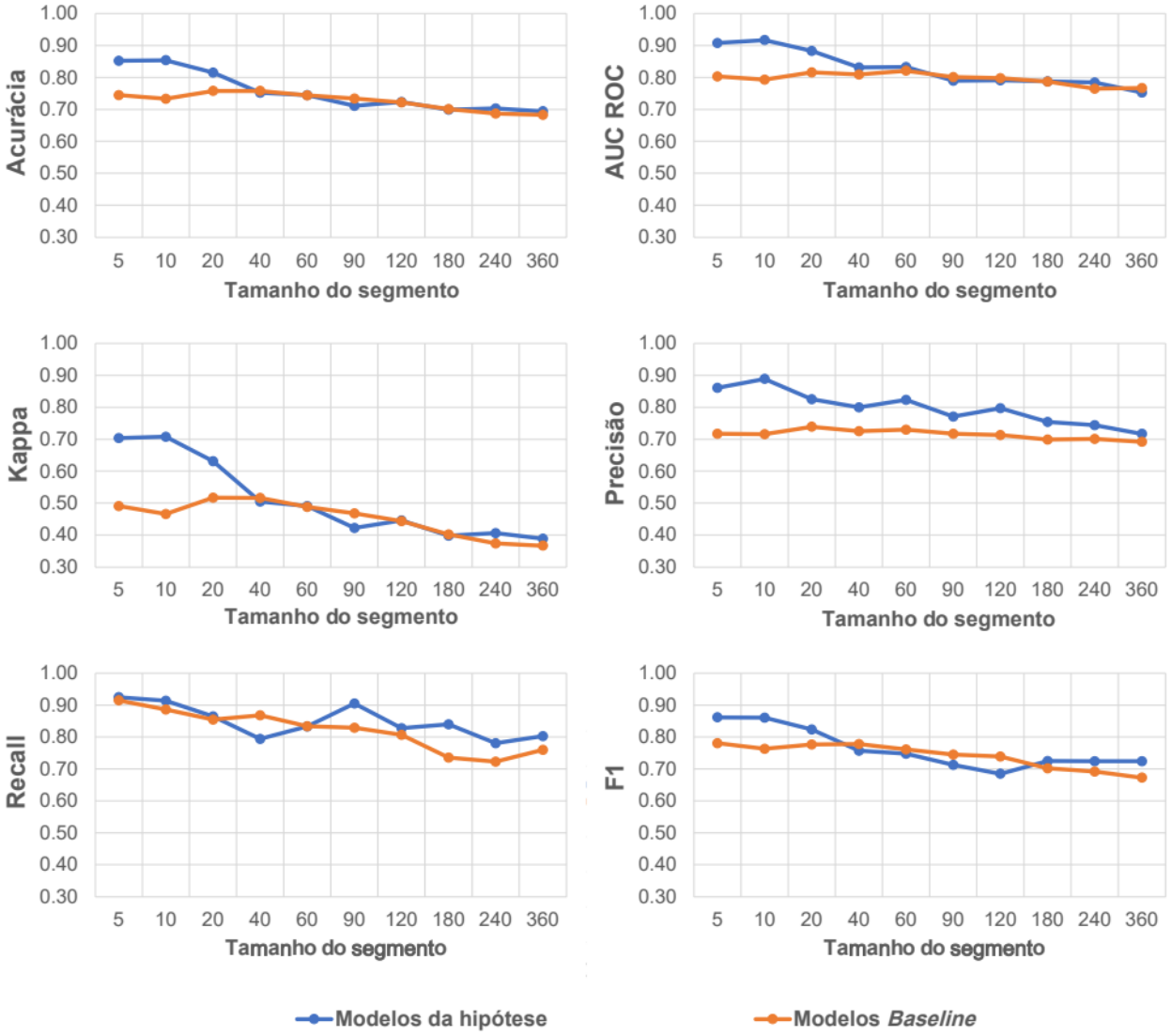

Figura 7: Resultados do desempenho médio das abordagens nos cenários 1 e 2.

superiores a 20 segundos as linhas ficam próximas, o que ajuda a explicar porque não conseguimos encontrar diferença estatística significativa entre as abordagens nestas janelas.

Os resultados sugerem que, quando generalizada para estudantes de diferentes níveis de ensino, nossa abordagem supera a dos trabalhos relacionados (*baseline*). Os resultados também reforçam as evidências de que os modelos funcionam melhor com janelas de até 20 segundos. Também é possível verificar que o desempenho global dos modelos foi inferior neste segundo método de validação (generalização para diferentes níveis de ensino) do que no primeiro método de validação. Este comportamento era esperado para validação cruzada com k = 2 (Raschka, 2018). Neste segundo método de validação, também descobrimos que outros algoritmos apresentaram os melhores resultados: Regressão Logística (LR), *Stochastic Gradient Descent* (SGD) e *Multilayer Perceptron* (MLP). No entanto, os melhores resultados da pesquisa vieram dos algoritmos *XGBoost* e *Random Forest* no primeiro método de validação.

8 Identificando atributos de dados relevantes

O quarto objetivo do estudo foi identificar os atributos de dados mais relevantes para os melhores modelos da hipótese. É interessante para o estudo verificar quais atributos de dados sobre estimativas de conhecimento ajudaram os modelos a detectar a confusão e a não confusão do estudante. Aplicamos o método *permutation feature importance* (Breiman, 2001; Molnar et al., 2023) nos melhores modelos da hipótese para cada janela de segmento. Neste método, a relevância de um atributo de dado é obtida ao verificar o desempenho do modelo quando os valores do atributo são embaralhados aleatoriamente. Quanto maior for a diminuição do desempenho (métrica de acurácia), maior será a relevância do atributo para o modelo (Breiman, 2001; Molnar et al., 2023). Cada atributo foi embaralhado 50 vezes em cada modelo. Também geramos informações estatísticas descritivas para os atributos de dados mais relevantes no melhor modelo da hipótese.

8.1 Resultados e discussões

A Figura 8 mostra os dez atributos de dados mais relevantes no melhor modelo da hipótese. Os valores mostrados no eixo *x* representam uma medida de relevância dos atributos de dados para o modelo. Quanto maior o valor, mais relevante é o atributo.

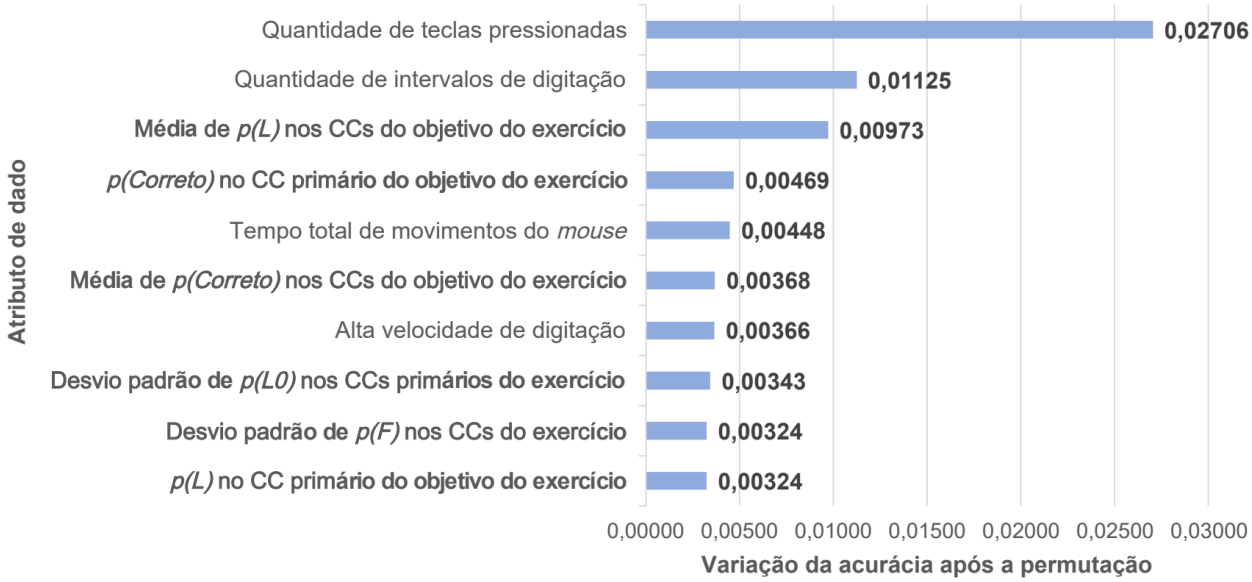

Figura 8: Os 10 atributos de dados mais relevantes para o melhor modelo do estudo.

Observa-se que entre os dez atributos mais relevantes, seis são atributos sobre estimativas do conhecimento do aluno (destacados em negrito) e quatro são atributos sobre a interação do aluno com o ambiente de programação. Os resultados mostram que as estimativas *p(L)* e *p(Correto)* do modelo *BKT+forget* geraram os atributos de dados mais relavantes sobre estimativas de conhecimento do aluno. Este resultado mostra que as principais inferências do modelo *BKT+forget* foram relevantes para os modelos de IA. *p(L)* descreve a probabilidade de o aluno dominar um componente de conhecimento. *p(Correto)* descreve a probabilidade de o aluno aplicar corretamente o componente de conhecimento na próxima oportunidade de aplicá-lo. Estas estimativas são atualizadas a cada oportunidade do aluno de aplicar a componente de conhecimento. Também foram

relevantes os atributos de dados gerados a partir de *p(L0)* e *p(F)*, respectivamente representando a probabilidade do aluno dominar o componente de conhecimento antes da primeira oportunidade de aplicá-lo e a probabilidade dele esquecer o componente de conhecimento. Essas estimativas (*p(L0)* e *p(F)*) são parâmetros do modelo *BKT+forget* que são gerados antes do aluno começar a se envolver nas tarefas de programação, o que indica que, mesmo sendo estimativas preliminares, elas permanecem relevantes para o modelo de detecção de confusão ao longo das interações do aluno com as tarefas de aprendizagem.

Nós também queríamos observar o comportamento dos atributos relevantes quando os alunos estavam confusos e quando não estavam confusos. Então nós geramos informações estatísticas descritivas sobre esses atributos no melhor modelo da hipótese. Nós calculamos valores de média e de desvio padrão para estes atributos quando o aluno estava confuso e quando não estava confuso. Esses dados descritivos estão disponíveis na Tabela 7.

Tabela 7: Estatística descritiva sobre os atributos mais relevantes no melhor modelo do estudo.

#	Atributo de dado	Confusão		Não confusão		*MW U*
		Média	Desvio	Média	Desvio	*p*
1	Quantidade de teclas pressionadas	1,405	3,713	10,394	12,576	<0,001
2	Quantidade de intervalos de digitação	0,370	0,803	1,121	1,337	<0,001
3	Média de *p(L)* nos CCs do objetivo do exercício	0,689	0,166	0,897	0,120	<0,001
4	*p(Correto)* no CC primário do objetivo de exercício	0,455	0,260	0,777	0,223	<0,001
5	Tempo total de movimento de mouse	2,213	1,484	1,569	1,469	<0,001
6	Média *p(Correto)* nos CCs do objetivo de exercício	0,713	0,134	0,873	0,106	<0,001
7	Alta velocidade de digitação	0,125	0,331	0,447	0,497	<0,001
8	Desvio padrão de *p(L0)* nos CCs primários do exercício	0,101	0,074	0,091	0,065	0,004
9	Desvio padrão de *p(F)* nos CCs do exercício	0,094	0,060	0,080	0,048	<0,001
10	*p(L)* no CC primário do objetivo de exercício	0,486	0,305	0,831	0,222	<0,001

A aplicação dos testes estatísticos não paramétricos *Mann-Whitney U (MW U)* mostrou que as diferenças nas médias de cada atributo de dado foram todas significativas, com valor de p abaixo do nível de significância ($\alpha = 0{,}05$). Os resultados mostram que quando confusos, os alunos apresentaram menos atividade de digitação (número de teclas pressionadas e número de intervalos de digitação) e gastaram mais tempo movimentando o mouse (tempo total de movimentos do mouse). Também foi encontrado um maior desvio nas probabilidades de conhecimento inicial nos componentes do conhecimento primário e um maior desvio nas probabilidades de esquecimento dos componentes do conhecimento. Quando não estavam confusos, eram mais propensos a dominar e aplicar corretamente os componentes do conhecimento. Eles também mostraram mais atividade

de digitação e maior velocidade de digitação.

Nós também classificamos os dez atributos de dados mais frequentes entre os dez atributos de dados mais relevantes nos melhores modelos da hipótese em todos os segmentos de dados. A Tabela 8 apresenta esse ranqueamento. Cinco atributos de dados foram gerados a partir das estimativas de conhecimento do aluno (destacados em negrito) e outros cinco atributos de dados foram gerados a partir de dados de interação do aluno com o BlueJ Afetivo. Os resultados mostram novamente a relevância dos atributos de dados gerados a partir das estimativas *p(L)* e *p(Correto)*. Também é possível observar a relevância do atributo de dado "Quantidade de teclas pressionadas" e atributos sobre movimentos do mouse ("Tempo total de movimentos do mouse" e "Quantidade de movimentos do mouse"). Os resultados reforçam as evidências de que a abordagem mista de utilizar atributos de interação do aluno juntamente de atributos sobre estimativas de conhecimento do aluno foi vantajosa para os modelos.

Tabela 8: Os 10 atributos de dados mais frequentes entre os 10 atributos de dados mais relevantes em todas as janelas.

#	Atributo de dado	Frequência
1	Quantidade de teclas pressionadas	10
2	**Média de *p(L)* nos CCs do objetivo do exercício**	9
3	Quantidade de intervalos de digitação	8
4	Tempo total de movimentos do mouse	8
5	***p(L)* no CC primário do objetivo do exercício**	7
6	**Média de *p(Correto)* nos CCs do exercício**	6
7	Quantidade de movimentos do mouse	6
8	Quantidade de erros de sintaxe	5
9	**Média de *p(L)* nos CCs do exercício**	4
10	***p(Correto)* no CC primário do objetivo do exercício**	4

9 Conclusão

O artigo apresenta um estudo que forneceu evidências que apoiam a hipótese que o uso de atributos de dados baseados em estimativas de conhecimento do aluno juntamente com dados de interação do aluno com o ambiente de programação, pode melhorar o desempenho de modelos livres de sensores na detecção da confusão do aluno durante tarefas de programação de computadores. Os resultados foram mais fortes quando os segmentos de dados que representam o comportamento do aluno quando ele estava e quando ele não estava confuso foram referentes a janelas de observação de até 20 segundos. Foi somente nestas janelas de até 20 segundos que foram obtidos resultados significativos quando verificamos o poder de generalização dos modelos para alunos de diferentes níveis de ensino.

Os resultados também sugerem a utilização dos algoritmos *XGBoost* e *Random Forest*, que corroboram com os resultados encontrados na literatura (Shwartz-Ziv & Armon, 2022) para dados tabulares como os utilizados na nossa pesquisa.

Também encontramos que nos melhores modelos da hipótese, os atributos de dados sobre estimativas de conhecimento do aluno que se destacaram foram aqueles derivados das principais

estimativas geradas pelo modelo de conhecimento do aluno, o *p(L)* e o *p(Correto)*. É importante destacar que apesar da relevância dos atributos sobre o conhecimento do aluno, o atributo de dado que mais se destacou foi um atributo de interação do aluno com o ambiente, o de "Quantidade de teclas pressionadas". Estes resultados corroboram com a hipótese do estudo de que a utilização conjunta desses diferentes tipos de dados sobre o aluno pode beneficiar os modelos livres de sensores na detecção da confusão dos alunos.

O presente artigo estendeu os resultados de nossa publicação anterior (Kautzmann et al., 2022). No presente estudo, nós levantamos a hipótese que nossa abordagem de usar o conhecimento do aluno juntamente de dados de interação poderia superar a abordagem dos trabalhos relacionados. Então nós geramos modelos que representam a abordagem dos trabalhos relacionados (*baseline*) e usamos testes estatísticos para comparar os resultados das abordagens. Para todos os tamanhos de janelas verificadas foram encontradas diferenças significativas e os melhores resultados em todas as métricas permaneceram sendo os da nossa abordagem (acurácia = 0,897, *AUC ROC = 0,950*, *kappa* = 0,794, precisão = 0,879 e F1 = 0,899) em segmentos de janelas de 10 segundos. Apenas na métrica de *recall* foi encontrado um valor superior para um melhor modelo *baseline* na janela de 5 segundos (*recall* = 0,930). Outra contribuição do presente trabalho em relação ao trabalho anterior (Kautzmann et al., 2022) foi que conseguimos verificar o poder de generalização dos modelos para estudantes de diferentes níveis de ensino (ensino técnico e ensino superior). Esta foi uma contribuição interessante porque as diferenças dos resultados entre as abordagens foram significativas apenas em janelas de até 20 segundos, com resultados de acurácia preditiva superiores nos melhores modelos da nossa abordagem. Além de reforçar os benefícios de nossa abordagem, esses resultados reforçam os achados do trabalho anterior sobre as recomendações de usar janelas de observação menores, de até 20 segundos. Por fim, nós também ampliamos a análise sobre os atributos de dados mais relevantes, que mostrou que as principais inferências do modelo de conhecimento do aluno geraram atributos de dados relevantes para o melhor modelo de nossa abordagem.

9.1 Limitações e trabalhos futuros

Nós avaliamos que os resultados dos modelos da hipótese poderiam ter sido ainda melhores com um melhor ajuste do modelo de conhecimento do aluno. O modelo *BKT+forget* que treinamos obteve um desempenho semelhante aos melhores resultados da literatura. Contudo, seu desempenho em alguns componentes do conhecimento, como "Repetição" e "Escrita de *Array*", ficou abaixo da média. Os modelos do estudo podem ter tido mais dificuldade em detectar o estado emocional do aluno para estes componentes do conhecimento. Um possível ajuste seria aumentar a granularidade desses CCs, ou seja, derivar componentes de conhecimento mais específicos, como "Repetição com número predefinido de iterações" e "Repetição com número indefinido de iterações". Porém, é conhecida a dificuldade de gerar modelos de conhecimento mais precisos em domínios de aprendizagem complexos, como o de programação de computadores. Quanto menor for a granularidade dos componentes do conhecimento, mais dados sobre a interação do aluno com tarefas envolvendo esses componentes específicos serão necessários. Mas nem sempre o professor conseguiria atribuir tantos exercícios sobre um componente de conhecimento tão específico (Pelánek, 2017).

Outra limitação do nosso trabalho é o método de validação utilizado para verificar a genera-

lização dos resultados para alunos de diferentes níveis de ensino. Este método teve como objetivo verificar se os modelos do estudo poderiam generalizar bem suas previsões para alunos com características heterogêneas. A pesquisa partiu do pressuposto que os estudantes do ensino técnico e do ensino superior apresentam características heterogêneas. Embora tenha havido diferença significativa na idade e experiência anterior em programação de computadores entre os grupos, não houve diferença significativa em relação à percepção emocional dos alunos. Além disso, embora, em média, os estudantes do ensino superior tenham mais experiência anterior do que os estudantes do ensino técnico, identificou-se que alguns estudantes do ensino superior tinham a mesma experiência de programação que os estudantes do ensino técnico. Poderia ser uma abordagem mais interessante agrupar os alunos por experiência e conhecimento anterior em programação e não por nível de ensino. Um melhor agrupamento de alunos com características heterogêneas poderia proporcionar uma melhor compreensão do poder de generalização dos modelos.

Apesar dos resultados interessantes encontrados pela pesquisa para apoiar sua hipótese utilizando algoritmos de aprendizado de máquina, uma sugestão para trabalhos futuros é verificar o desempenho de algoritmos de aprendizado profundo (*deep learning*). O trabalho de Botelho et al. (2017) utilizou diferentes redes neurais recorrentes (RNN), *Gated Recurrent Unit Neural Networks* (GRU) e *Long Short-Term Memory* (LSTM) e dados de interação com o sistema tutor de matemática *ASSISTments* para modelar um detector livre de sensores para quatro estados afetivos (concentração engajada, tédio, confusão e frustração). As redes neurais recorrentes possuem estruturas que capturam informações recorrentes em dados de séries temporais (Williams & Zipser, 1989). O melhor resultado dos autores foi um valor de 0,78 para *AUC ROC* com uma rede LSTM. Este resultado superou dois outros estudos que utilizaram a mesma base de dados do sistema *ASSISTments* (Y. Wang et al., 2015). A suposição dos autores para o desempenho superior encontrado com esses modelos na aprendizagem de matemática é que a detecção de estados afetivos poderia se beneficiar de modelos que observassem a estrutura temporal dos dados de entrada (Botelho et al., 2017). O melhor resultado de Botelho et al. (2017) foi inferior ao melhor resultado da nossa pesquisa, que obteve um valor de *AUC ROC* de 0,950 em um segmento de 10 segundos. Embora não haja tanto espaço para a evolução dos resultados, verificar o desempenho das redes neurais recorrentes no contexto da nossa pesquisa pode ser interessante.

Nós também pretendemos integrar o modelo de detecção de confusão em um ambiente computacional adaptativo de aprendizagem de programação, como um sistema tutor inteligente. Ao detectar a confusão do aluno, o ambiente poderia tomar decisões instrucionais, como fornecer *feedback* ao aluno sobre o conteúdo e preencher suas lacunas de conhecimento. O ambiente também poderia auxiliar o estudante na regulação da sua confusão (Arguel et al., 2019).

Agradecimentos

O presente trabalho foi realizado com o apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, da FAPERGS (Processo 17/2551-0001203-8) e do CNPq (processo 306005/2020-4).

Artigo Premiado Estendido

Esta publicação é uma versão estendida de trabalho premiado no Concurso Alexandre Direne de Teses, Dissertações e Trabalhos de Conclusão de Curso em Informática na Educação (CTD-IE 2022) do XI Congresso Brasileiro de Informática na Educação (CBIE 2022). O artigo é intitulado "Os efeitos de usar estimativas de conhecimento do aluno em programação de computadores em modelos livres de sensores de detecção da emoção confusão", DOI: 10.5753/cbie_estendido.2022.226387.

Referências

Agarwal, D., Baker, R., & Muraleedharan, A. (2020). Dynamic knowledge tracing through data driven recency weights. Em A. N. Rafferty, J. Whitehill, V. Cavalli-Sforza & C. Romero (Ed.), *Proceedings of The 13th International Conference on Educational Data Mining (EDM 2020)* (pp. 725–729). https://eric.ed.gov/?id=ED607821 [GS Search].

Alzoubi, O., D'Mello, S., & Calvo, R. A. (2012). Detecting Naturalistic Expressions of Nonbasic Affect Using Physiological Signals. *IEEE Transactions on Affective Computing*, *3*(3), 298–310. https://doi.org/10.1109/t-affc.2012.4 [GS Search].

Arguel, A., Lockyer, L., Kennedy, G., Lodge, J. M., & Pachman, M. (2019). Seeking optimal confusion: a review on epistemic emotion management in interactive digital learning environments. *Interactive Learning Environments*, *27*(2), 200–210. https://doi.org/10.1080/10494820.2018.1457544 [GS Search].

Arroyo, I., Cooper, D., Burleson, W., Woolf, B., Muldner, K., & Christopherson, R. (2009). Emotion sensors go to school. *Frontiers in Artificial Intelligence and Applications*, (1), 17–24. https://doi.org/10.3233/978-1-60750-028-5-17 [GS Search].

Ausubel, D. P., Novak, J. D., & Hanesian, H. (1978). *Educational Psychology: A Cognitive View*. Holt, Rinehart; Winston. [GS Search].

Badrinath, A., Wang, F., & Pardos, Z. (2021). pybkt: An accessible python library of bayesian knowledge tracing models. *Proceedings of the 14th International Conference on Educational Data Mining*. https://doi.org/10.48550/arXiv.2105.00385 [GS Search].

Baker, R. S. d., Corbett, A., Gowda, S. M., Wagner, A. Z., MacLaren, B. A., Kauffman, L. R., Mitchell, A. P., & Giguere, S. (2010). Contextual slip and prediction of student performance after use of an intelligent tutor. *User Modeling, Adaptation, and Personalization: 18th International Conference, UMAP 2010, Big Island, HI, USA, June 20-24, 2010. Proceedings 18*, 52–63. https://doi.org/10.1007/978-3-642-13470-8_7 [GS Search].

Baker, R. S. d., Pardos, Z., Gowda, S. M., Nooraei, B. B., & Heffernan, N. T. (2011). Ensembling predictions of student knowledge within intelligent tutoring systems. *User Modeling, Adaption and Personalization: 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings 19*, 13–24. https://doi.org/10.1007/978-3-642-22362-4_2 [GS Search].

Beck, J. E., & Chang, K.-m. (2007). Identifiability: A fundamental problem of student modeling. Em C. Conati, K. McCoy & G. Paliouras (Ed.), *International Conference on User Modeling* (pp. 137–146). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-73078-1_17 [GS Search].

Bosch, N., Chen, Y., & D'Mello, S. (2014). It's written on your face: Detecting affective states from facial expressions while learning computer programming. *Intelligent Tutoring Systems*, 39–44. https://doi.org/10.1007/978-3-319-07221-0_5 [GS Search].

Bosch, N., & D'Mello, S. (2017). The affective experience of novice computer programmers. *International Journal of Artificial Intelligence in Education*, *27*(1), 181–206. https://doi.org/10.1007/s40593-015-0069-5 [GS Search].

Bosch, N., D'Mello, S., & Mills, C. (2013). *What emotions do novices experience during their first computer programming learning session?* https://doi.org/10.1007/978-3-642-39112-5_2 [GS Search].

Botelho, A. F., Baker, R. S., & Heffernan, N. T. (2017). Improving sensor-free affect detection using deep learning. *Artificial Intelligence in Education*, 40–51. https://doi.org/10.1007/978-3-319-61425-0_4 [GS Search].

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32. https://doi.org/10.1023/A:1010933404324 [GS Search].

Calvo, R. A., & D'Mello, S. (2010). Affect detection: An interdisciplinary review of models, methods, and their applications. *IEEE Transactions on Affective Computing (TAC)*, *1*(1), 18–37. https://doi.org/10.1109/T-AFFC.2010.1 [GS Search].

Chi, M. T., & Ohlsson, S. (2005). *Complex Declarative Learning*. Cambridge University Press. https://doi.org/10.1007/978-1-4419-1428-6_295 [GS Search].

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, *20*(1), 37–46. https://doi.org/10.1177/001316446002000104 [GS Search].

Corbett, A., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, *4*, 253–278. https://doi.org/10.1007/bf01099821 [GS Search].

Corbett, A., Kauffman, L., Maclaren, B., Wagner, A., & Jones, E. (2010). A Cognitive Tutor for genetics problem solving: Learning gains and student modeling. *Journal of Educational Computing Research*, *42*(2), 219–239. https://doi.org/https://doi.org/10.2190/EC.42.2.e [GS Search].

Coto, M., Mora, S., Grass, B., & Murillo-Morera, J. (2021). Emotions and programming learning: systematic mapping. *Computer Science Education*, *32*(1), 1–36. https://doi.org/10.1080/08993408.2021.1920816 [GS Search].

Craig, S. D., & et al. (2008a). Emote aloud during learning with AutoTutor: Applying the Facial Action Coding System to cognitive - Affective states during learning. *Cognition and Emotion*, *22*(5), 777–788. https://doi.org/10.1080/02699930701516759 [GS Search].

de Oliveira Alves, M., Medeiros, F. P. A., & Melo, L. B. (2020). Levantamento do Estado da Arte sobre Aprendizagem baseada em Problemas na Educação a Distância e Híbrida. *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, 61–71. https://doi.org/10.5753/cbie.sbie.2020.61 [GS Search].

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, *10*(7), 1895–1923. https://doi.org/10.1162/089976698300017197 [GS Search].

D'Mello, S. (2020). Big data in the science of learning. Em *Big data in psychological research* (pp. 203–225). American Psychological Association. https://doi.org/10.1037/0000193-010 [GS Search].

D'Mello, S., & Calvo, R. A. (2013). Beyond the basic emotions: what should affective computing compute? *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, 2287–2294. https://doi.org/10.1145/2468356.2468751 [GS Search].

D'Mello, S., & Graesser, A. (2012). Dynamics of affective states during complex learning. *Learning and Instruction*, *22*(2), 145–157. https://doi.org/10.1016/j.learninstruc.2011.10.001 [GS Search].

D'Mello, S., & Graesser, A. C. (2014). Confusion. Em *International Handbook of Emotions in Education* (pp. 299–320). Routledge. https://doi.org/10.4324/9780203148211 [GS Search].

D'Mello, S., Lehman, B., Pekrun, R., & Graesser, A. (2014). Confusion can be beneficial for learning. *Learning and Instruction*, *29*, 153–170. https://doi.org/10.1016/j.learninstruc.2012.05.003 [GS Search].

D'Mello, S., Person, N., & Lehman, B. (2009). Antecedent-consequent relationships and cyclical patterns between affective states and problem solving outcomes. *Frontiers in Artificial Intelligence and Applications*, *200*, 57–64. https://doi.org/10.3233/978-1-60750-028-5-57 [GS Search].

Felipe, D. A. M., Gutierrez, K. I. N., Quiros, E. C. M., & Vea, L. A. (2012). Towards the development of intelligent agent for novice c/c++ programmers through affective analysis of event logs. *Proc. Int. MultiConference Eng. Comput. Sci*, *1*, 2012. https://www.iaeng.org/publication/IMECS2012/IMECS2012_pp511-518.pdf [GS Search].

Fino, C. N. (2001). Vygotsky e a Zona de Desenvolvimento Proximal (ZDP): três implicações pedagógicas. *Revista Portuguesa de educação*, *14*, 273–291. https://www.redalyc.org/pdf/374/37414212.pdf [GS Search].

Gong, Y., Beck, J. E., & Heffernan, N. T. (2010). Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. *Intelligent Tutoring Systems: 10th International Conference, ITS 2010, Pittsburgh, PA, USA, June 14-18, 2010, Proceedings, Part I 10*, 35–44. https://doi.org/10.1007/978-3-642-13388-6_8 [GS Search].

Gowda, S. M., Rowe, J. P., de Baker, R. S. J., Chi, M., & Koedinger, K. R. (2011). Improving Models of Slipping, Guessing, and Moment-By-Moment Learning with Estimates of Skill Difficulty. *EDM*, *2011*, 199–208. https://educationaldatamining.org/EDM2011/wp-content/uploads/proc/edm11_proceedings.pdf [GS Search].

Graesser, A., Chipman, P., King, B., McDaniel, B., & D'Mello, S. (2007). Emotions and learning with autotutor. *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, 569–571. https://dl.acm.org/doi/abs/10.5555/1563601.1563699 [GS Search].

Grafsgaard, J. F., Boyer, K. E., & Lester, J. C. (2011). Predicting facial indicators of confusion with hidden markov models. *Affective Computing and Intelligent Interaction*, *6974 LNCS*(PART 1), 97–106. https://doi.org/10.1007/978-3-642-24600-5_13 [GS Search].

Halgren, E., & et al. (2002). N400-like Magnetoencephalography Responses Modulated by Semantic Context, Word Frequency, and Lexical Class in Sentences. *NeuroImage*, *17*(3), 1101–1116. https://doi.org/10.1006/nimg.2002.1268 [GS Search].

Hess, U. (2003). Now you see it, now you don't–the confusing case of confusion as an emotion: Commentary on Rozin and Cohen (2003). *Emotion*, *3*(1), 76–80. https://doi.org/10.1037/1528-3542.3.1.76 [GS Search].

Izard, C. E. (2010). The many meanings/aspects of emotion: Definitions, functions, activation, and regulation. *Emotion Review*, *2*(4), 363–370. https://doi.org/10.1177/1754073910374661 [GS Search].

Jófili, Z. (2002). Piaget, Vygotsky, Freire e a construção do conhecimento na escola. *Educação: teorias e práticas*, *2*(2), 191–208. https://www.maxwell.vrac.puc-rio.br/7560/7560.PDF [GS Search].

Kasurinen, J., & Nikula, U. (2009). Estimating programming knowledge with Bayesian knowledge tracing. *ACM SIGCSE Bulletin*, *41*(3), 313–317. https://doi.org/10.1145/1595496.1562972 [GS Search].

Kautzmann, T. R., Ramos, G. d. O., & Jaques, P. A. (2022). O uso de estimativas de conhecimento do aluno em programação de computadores em modelos de detecção da emoção confusão livres de sensores. *Anais do XXXIII Simpósio Brasileiro de Informática na Educação*, 1196–1208. https://doi.org/10.5753/sbie.2022.225768 [GS Search].

Keltner, D., & Shiota, M. N. (2003). New displays and new emotions: A commentary on Rozin and Cohen (2003). *Emotion*, *3*(1), 86–91. https://doi.org/10.1037/1528-3542.3.1.86 [GS Search].

Khajah, M., Lindsey, R. V., & Mozer, M. C. (2016). How deep is knowledge tracing? *arXiv preprint arXiv:1604.02416*. https://doi.org/10.48550/arXiv.1604.02416 [GS Search].

Knottnerus, J. A., & Tugwell, P. (2010). Real world research. *Journal of clinical epidemiology*, *63*(10), 1051–1052. https://doi.org/10.1016/j.jclinepi.2010.08.001 [GS Search].

Koedinger, K. R., Corbett, A., & Perfetti, C. (2012). The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, *36*(5), 757–798. https://doi.org/10.1111/j.1551-6709.2012.01245.x [GS Search].

Kubat, M. (2017). *An introduction to machine learning* (Vol. 2). Springer. https://doi.org/10.1007/978-3-319-63913-0 [GS Search].

Lee, D. M. C., Rodrigo, M. M. T., Baker, R. S., Sugay, J. O., & Coronel, A. (2011). Exploring the relationship between novice programmer confusion and achievement. *International Conference on Affective Computing and Intelligent Interaction*, 175–184. https://doi.org/10.1007/978-3-642-24600-5_21 [GS Search].

Lehman, B., D'Mello, S., Strain, A., Mills, C., Gross, M., Dobbins, A., Wallace, P., Millis, K., & Graesser, A. (2013). Inducing and tracking confusion with contradictions during complex learning. *International Journal of Artificial Intelligence in Education*, *22*, 85–105. https://doi.org/10.3233/JAI-130025 [GS Search].

Lin, C., & Chi, M. (2016). Intervention-bkt: incorporating instructional interventions into bayesian knowledge tracing. *Intelligent Tutoring Systems: 13th International Conference, ITS 2016, Zagreb, Croatia, June 7-10, 2016. Proceedings 13*, 208–218. https://doi.org/10.1007/978-3-319-39583-8_20 [GS Search].

MacDowell, K. A., & Mandler, G. (1989). Constructions of emotion: Discrepancy, arousal, and mood. *Motivation and Emotion*, *13*(2), 105–124. https://doi.org/10.1007/bf00992957 [GS Search].

Mao, Y. (2018). Deep Learning vs. Bayesian Knowledge Tracing: Student Models for Interventions. *Journal of educational data mining*, *10*(2). https://doi.org/10.5281/zenodo.3554691 [GS Search].

Mayer, J. D., Salovey, P., & Caruso, D. R. (2004). Emotional intelligence: Theory, findings, and implications. *Psychological inquiry*, *15*(3), 197–215. http://www.jstor.org/stable/20447229 [GS Search].

McDaniel, B., & et al. (2007). Facial Features for Affective State Detection in Learning Environments. *Proceedings of the Annual Meeting of the Cognitive Science Society*. https://escholarship.org/uc/item/9w00945d [GS Search].

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, *12*(2), 153–157. https://doi.org/10.1007/bf02295996 [GS Search].

Molnar, C., Freiesleben, T., König, G., Herbinger, J., Reisinger, T., Casalicchio, G., Wright, M. N., & Bischl, B. (2023). Relating the partial dependence plot and permutation feature importance to the data generating process. *World Conference on Explainable Artificial Intelligence*, 456–479. https://doi.org/10.1007/978-3-031-44064-9_24 [GS Search].

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal processing magazine*, *13*(6), 47–60. https://doi.org/10.1109/79.543975 [GS Search].

Moors, A., Ellsworth, P. C., Scherer, K. R., & Frijda, N. H. (2013). Appraisal theories of emotion: State of the art and future development. *Emotion Review*, *5*(2), 119–124. https://doi.org/10.1177/1754073912468165 [GS Search].

Nagatani, K., Zhang, Q., Sato, M., Chen, Y.-Y., Chen, F., & Ohkuma, T. (2019). Augmenting knowledge tracing by considering forgetting behavior. *The world wide web conference*, 3101–3107. https://doi.org/10.1145/3308558.3313565 [GS Search].

Pardos, Z., & Heffernan, N. (2010). Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. *Educational Data Mining 2010*. https://files.eric.ed.gov/fulltext/ED538834.pdf [GS Search].

Pardos, Z., & Heffernan, N. T. (2011). KT-IDEM: Introducing item difficulty to the knowledge tracing model. *User Modeling, Adaption and Personalization: 19th International Conference, UMAP 2011, Girona, Spain, July 11-15, 2011. Proceedings 19*, 243–254. https://doi.org/10.1007/978-3-642-22362-4_21 [GS Search].

Pekrun, R. (2011). Emotions as drivers of learning and cognitive development. *New perspectives on affect and learning technologies*, *3*, 23–39. https://doi.org/10.1007/978-1-4419-9625-1_3 [GS Search].

Pekrun, R., & Stephens, E. J. (2012). Academic emotions. Em *APA Educational Psychology Handbook, Vol 2: Individual Differences and Cultural and Contextual Factors* (pp. 3–31, Vol. 2). American Psychological Association. https://doi.org/10.1037/13274-001 [GS Search].

Pekrun, R. (2014). Emotions and learning. https://eric.ed.gov/?id=ED560531 [GS Search].

Pelánek, R. (2017). Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Modeling and User-Adapted Interaction*, *27*, 313–350. https://doi.org/10.1007/s11257-017-9193-2 [GS Search].

Pelizzari, A., KriegL, M. d. L., Baron, M. P., Finck, N. T. L., & Dorocinski, S. I. (2002). Teoria da aprendizagem significativa segundo Ausubel. *revista PEC*, *2*(1), 37–42. http://portaldoprofessor.mec.gov.br/storage/materiais/0000012381.pdf [GS Search].

Penmetsa, P. (2021). *Investigate effectiveness of code features in knowledge tracing task on novice programming course*. North Carolina State University. https://ceur-ws.org/Vol-3051/CSEDM_12.pdf [GS Search].

Raposo, A. C., Maranhão, D., & Neto, C. S. (2019). Analise do modelo bkt na avaliacao da curva de aprendizagem de alunos de algoritmos. *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, *30*(1), 479. https://doi.org/10.5753/cbie.sbie.2019.479 [GS Search].

Raschka, S. (2018). Model evaluation, model selection, and algorithm selection in machine learning. https://doi.org/10.48550/arXiv.1811.12808 [GS Search].

Rodrigo, M. M. T., Baker, R. S. J., & Nabos, J. Q. (2010). The relationships between sequences of affective states and learner achievement. *Proceedings of the 18th International Conference on Computers in Education*, 56–60. https://learninganalytics.upenn.edu/ryanbaker/ICCE2010-RBN.pdf [GS Search].

Rozin, P., & Cohen, A. B. (2003). High frequency of facial expressions corresponding to confusion, concentration, and worry in an analysis of naturally occurring facial expressions of Americans. *Emotion*, *3*(1), 68–75. https://doi.org/10.1037/1528-3542.3.1.68 [GS Search].

Scherer, K. R. (2005). What are emotions? and how can they be measured? *Social science information*, *44*(4), 695–729. https://doi.org/10.1177/0539018405058216 [GS Search].

Shwartz-Ziv, R., & Armon, A. (2022). Tabular data: Deep learning is not all you need. *Information Fusion*, *81*, 84–90. https://doi.org/10.1016/j.inffus.2021.11.011 [GS Search].

Silvia, P. J. (2010). Confusion and interest: The role of knowledge emotions in aesthetic experience. *Psychology of Aesthetics, Creativity, and the Arts*, *4*(2), 75–80. https://doi.org/10.1037/a0017081 [GS Search].

Slater, S., & Baker, R. S. (2018). Degree of error in Bayesian knowledge tracing estimates from differences in sample sizes. *Behaviormetrika*, *45*(2), 475–493. https://doi.org/10.1007/s41237-018-0072-x [GS Search].

Sweller, J., van Merrienboer, J. J. G., & Paas, F. G. W. C. (1988). Cognitive Architecture and Instructional Design. *Educational Psychology Review*, *10*(3), 251–296. https://doi.org/10.1023/A:1022193728205 [GS Search].

Sweller, J., van Merriënboer, J. J. G., & Paas, F. (2019). Cognitive Architecture and Instructional Design: 20 Years Later. *Educational Psychology Review*, *10*, 251–296. https://doi.org/10.1007/s10648-019-09465-5 [GS Search].

Taksic, V. (2000). Convergent and divergent validity of the Emotional Skills and Competence Questionnaire. *Comunicación presentada en XII Days of Psychology, Zadar, Croacia*. [GS Search].

Tiam-Lee, T. J., & Sumi, K. (2018). Adaptive feedback based on student emotion in a system for programming practice. *Intelligent Tutoring Systems*, 243–255. https://doi.org/10.1007/978-3-319-91464-0_24 [GS Search].

Tiam-Lee, T. J., & Sumi, K. (2019). Analysis and prediction of student emotions while doing programming exercises. *International Conference on Intelligent Tutoring Systems*, 24–33. https://doi.org/10.1007/978-3-030-22244-4_4 [GS Search].

VanLehn, K., Siler, S., Murray, C., Yamauchi, T., & Baggett, W. B. (2003). Why do only some events cause learning during human tutoring? *Cognition and Instruction*, 209–249. https://doi.org/10.1207/s1532690xci2103_01 [GS Search].

Vea, L., & Rodrigo, M. M. (2017). Modeling negative affect detector of novice programming students using keyboard dynamics and mouse behavior. *Trends in Artificial Intelligence: PRICAI 2016 Workshops*, 127–138. https://doi.org/10.1007/978-3-319-60675-0_11 [GS Search].

Wang, S., Han, Y., Wu, W., & Hu, Z. (2017). Modeling student learning outcomes in studying programming language course. *2017 Seventh International Conference on Information Science and Technology (ICIST)*, 263–270. https://doi.org/10.1109/icist.2017.7926768 [GS Search].

Wang, Y., Heffernan, N. T., & Heffernan, C. (2015). Towards better affect detectors: effect of missing skills, class features and common wrong answers. *Proceedings of the fifth international conference on learning analytics and knowledge*, 31–35. https://doi.org/10.1145/2723576.2723618 [GS Search].

Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, *1*(2), 270–280. https://doi.org/10.1162/neco.1989.1.2.270 [GS Search].

Wong, T. T. (2015). Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. *Pattern Recognition*, *48*, 2839–2846. https://doi.org/10.1016/j.patcog.2015.03.009 [GS Search].

Yang, T.-Y., Baker, R. S., Studer, C., Heffernan, N., & Lan, A. S. (2019). Active learning for student affect detection. *Proceedings of the 12th International Conference on Educational Data Mining*, 208–217. https://eric.ed.gov/?id=ED599173 [GS Search].

Yudelson, M. V., Koedinger, K. R., & Gordon, G. J. (2013). Individualized bayesian knowledge tracing models. *Artificial Intelligence in Education: 16th International Conference, AIED 2013, Memphis, TN, USA, July 9-13, 2013. Proceedings 16*, 171–180. https://doi.org/10.1007/978-3-642-39112-5_18 [GS Search].