Análise da Capacidade Preditiva de Técnicas para Modelagem do Conhecimento Aplicadas ao Aprendizado de Algoritmos

Antonio Carlos Raposo[1], **Djefferson Maranhão**[1], **Carlos de Salles Soares Neto**[1]

[1]Departamento de Informática – Universidade Federal do Maranhão (UFMA)
São Luis – MA – Brazil

{antoniocarlosraposo93,djefferson.maranhao}@gmail.com, csalles@deinf.ufma.br

Abstract. *Learning algorithms is a complex topic and there are several didactic initiatives to improve the student experience, such as intelligent tutors. The predictive capacity of knowledge models is fundamental for the proper functioning of intelligent tutors. Additionally, there are few studies that indicate the effects that the domain has on the predictive capacity of models. This article presents a qualitative and comparative analysis between knowledge models such as Bayesian Knowledge Tracing and Performance Factor Analysis in order to identify possible peculiarities in the field of algorithms.The results show that the compilation errors directly influence the predictive capacity of the models and there are arguments to maintain and to remove them.*

Resumo. *O aprendizado de algoritmos é um tema complexo e existem diversas iniciativas didáticas para aprimorar a experiência do aluno, como tutores inteligentes. A capacidade preditiva de modelos de conhecimento é fundamental para o funcionamento apropriado de tutores inteligentes. Adicionalmente, existem poucos estudos, neste contexto, que indicam os efeitos que o domínio de ensino tem sobre a capacidade preditiva dos modelos. Este artigo apresenta uma análise qualitativa e comparativa entre modelos de conhecimento como Bayesian Knowledge Tracing e Performance Factor Analysis a fim de se identificar possíveis peculiaridades no domínio do ensino de algoritmos. Os resultados mostram que os erros de compilação influenciam diretamente na capacidade preditiva dos modelos e há argumentos para mantê-los e removê-los.*

1. Introdução

O ensino de algoritmos é um tópico que possui diversas iniciativas didáticas devido a sua complexidade. Aprender a programar é um exercício em diversos fundamentos básicos como abstração, decomposição de problemas, identificação de padrões e escrita de algoritmos. Tais habilidades normalmente não fazem parte da didática de uma disciplina introdutória de algoritmos.

Adicionalmente, o ensino de algoritmos é modular com etapas que são pré-requisitos uma das outras. O progresso do aluno em cada etapa é fundamental para o seu desenvolvimento na disciplina. Isso faz com que a análise do conhecimento dos alunos seja uma etapa fundamental no ensino. Entretanto, é difícil para o professor identificar as necessidades específicas de cada aluno. Uma possível solução para os problemas apresentados é o uso de sistemas tutores inteligentes.

DOI: 10.5753/cbie.sbie.2020.1653

Sistemas Tutores Inteligentes (STI) são sistemas educacionais que utilizam técnicas de inteligência artificial e aprendizagem de máquina para dar suporte ao processo de aprendizagem. Um STI tem como objetivo instruir e apresentar feedbacks de maneira espontânea para o aluno [Psotka et al. 1988]. Além disso, STI são capazes de se adaptar as individualidades dos alunos e prover conteúdo visando aprimorar a estratégia educacional [Ramesh et al. 2015].

A modelagem e predição do conhecimento de alunos é essencial para o desenvolvimento de um tutor inteligente [Käser et al. 2017]. É através da predição do conhecimento do aluno que o tutor baseia a sua tomada de decisão. Um tutor inteligente visa prover conteúdo que o aluno esteja preparado para assimilar uma vez que o erro nesta estimativa pode desestimulá-lo. Para isso, o tutor inteligente precisa possuir modelos complexos do conhecimento dos alunos que consigam prever com excelência o nível em que o aluno se encontra.

Adicionalmente, Existem poucos estudos que indicam qual o efeito que um domínio possui sobre a modelagem do conhecimento dos alunos. O ensino de algoritmos apresenta mais tentativas e erros para solução de problemas que outros domínios devido à sua natureza. Um típico erro neste domino são os erros de compilação. Os erros de compilação são apenas erros de sintaxe e não dizem nada quanto a lógica apresentada para a solução do problema.

Neste artigo apresenta-se uma análise da capacidade preditiva de modelos criados utilizando os dados coletados no Ambiente Virtual de aprendizagem (AVA) COSMO desenvolvido especificamente para o ensino de algoritmos, além de uma análise comparativa de tais modelos com modelos de outros domínios. Na presente proposta, empregou-se o *Bayesian Knowledge Tracing* (BKT) com algumas variações e o *Performance Factor Analysis* (PFA) para a avaliação da capacidade preditiva de modelos de conhecimento. Há uma discussão comparativa e qualitativa sobre a análise dos resultados obtidos durante o experimento.

O objetivo central deste trabalho é analisar qualitativa e comparativamente a capacidade preditiva entre modelos de conhecimento de alunos no domínio da programação e outros domínios. BKT-EM se mostrou promissor quanto a capacidade preditiva, sendo o mais estável dos modelos observados neste trabalho. Além disso, PFA se mostrou como uma alternativa ao BKT-EM, entretanto mais dependente da quantidade de dados presentes nos datasets. Por fim, os modelos criados voltados para o ensino de algoritmo apresentam uma variação significativa ao se observar os erros de compilação, há uma argumentação tanto para a remoção dos erros quanto para manter tais erros.

O artigo está organizado como se segue. A seção 2 apresenta a fundamentação teórica sobre as técnicas de modelagem de conhecimento usadas. A seção 3 explica como o experimento foi executado e a seção 4 apresenta uma discussão e os resultados obtidos.

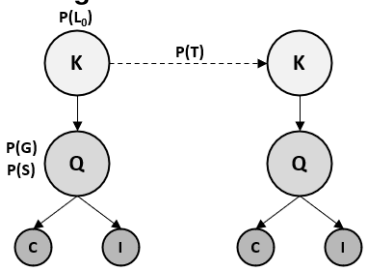
2. Fundamentação Teórica

Esta seção apresenta a fundamentação teórica necessária para o entendimento deste trabalho. A seção apresenta as técnicas de modelagem de conhecimento adotadas neste trabalho.

2.1. *Bayesian Knowledge Tracing*

De acordo com [Yudelson 2016], a técnica de modelagem de conhecimento denominada de *Bayesian Knowledge Tracing (BKT)* é uma das mais populares na literatura relacionada a tutores inteligentes. Os modelos BKT fundamentam-se em modelos ocultos de Markov (HMM, do inglês *Hidden Markov Models*) e são construídos com base em variáveis latentes e observáveis [Käser et al. 2017]. As variáveis latentes representam o conhecimento acerca de um determinado conteúdo, determinando se o aluno possui determinado conhecimento ou não. Por sua vez, as variáveis observáveis representam as respostas atribuídas pelos alunos a um determinado conteúdo, que podem ser classificadas como corretas ou incorretas.

Figura 1. Modelo BKT



Para [Yudelson et al. 2013], um modelo básico do BKT (Figura 1) é composto pelos seguintes parâmetros: uma probabilidade inicial de se possuir um determinado conhecimento K a priori, definida $p(L_0)$; um probabilidade de que a habilidade será aprendida em cada oportunidade de praticar o conhecimento, definida $p(T)$; uma probabilidade do estudante cometer um erro quando já detém um determinado conhecimento, definida $p(S)$; e uma probabilidade do estudante acertar uma questão ao aplicar uma habilidade ainda não conhecida, definida $p(G)$.

A variável Q consiste na resposta atribuída por um estudante a uma determinada atividade. Essas variáveis são binários, podendo assumir apenas os valores correto e incorreto. Com base nesses parâmetros, infere-se quanto ao conhecimento de um estudante na enésima oportunidade de exercitar um determinado conhecimento, que é definido por $P(L_n)$.

Em conformidade com [David et al. 2016], as premissas básicas do modelo são: 1) um aprendiz somente poderá incorrer em acerto ou erro, a cada oportunidade de aprendizado; e 2) um aprendiz somente poderá estar nos estados de conhecimento ”não aprendido”ou ”aprendido”para cada habilidade; e 3) uma vez aprendida, uma habilidade jamais é esquecida.

Com base nos parâmetros definidos acima, pode-se predizer quando a resposta de um aprendiz esta correta na oportunidade n por meio da Lei das Probabilidades Totais, expressa na Equação 1:

$$P(correto_n) = P(L_n) \times (1 - P(S)) + (1 - P(L_n)) \times P(G) \quad (1)$$

De forma similar, a probabilidade de que a resposta seja incorreta no tempo n pode ser computada pela Equação 2:

$$P(incorreto_n) = P(L_n) \times P(S) + (1 - P(L_n)) \times (1 - P(G)) \quad (2)$$

Conforme explica [David et al. 2016], existem dois estágios distintos ao se utilizar um modelo de conhecimento. O primeiro estágio consiste no ajuste dos parâmetros do modelo a partir dos dados, denominado de *fitting*. O segundo consiste em aplicar o modelo para inferir o conhecimento de um estudante ao longo do tempo, com base em suas respostas.

Desse modo, dada a observação de uma resposta de um estudante no tempo n (correta ou incorreta), a probabilidade $p(L_n)$ de que o estudante domine a habilidade pode ser calculada pelo Teorema de Bayes. Assim, ao observar uma resposta correta, a probabilidade pode ser obtida por meio da Equação 3, em que $P(correto_n)$ foi definida na Equação 1:

$$P(L_n \mid correto_n) = \frac{P(L_n) \times (1 - P(S))}{P(correto_n)} \tag{3}$$

Em contrapartida, ao observar uma resposta incorreta, a probabilidade pode ser obtida por meio da Equação 4, em que $P(incorreto_n)$ foi definida na Equação 2:

$$P(L_n \mid incorreto_n) = \frac{P(L_n) \times P(S)}{P(incorreto_n)} \tag{4}$$

Finalmente, demonstra-se como o conhecimento do estudante a respeito de um determinado assunto pode ser atualizado dadas suas interações com o sistema. O resultado consiste na soma de duas parcelas: a probabilidade de que o estudante já conheça a habilidade; e a probabilidade de que o estudante não detinha a habilidade, mas foi capaz de aprendê-la.

$$P(L_n) = P(L_{n-1} \mid obs_{n-1}) + (1 - P(L_{n-1} \mid obs_{n-1})) \times P(T) \tag{5}$$

2.1.1. Abordagens para ajuste dos parâmetros do modelo

O modelo tradicional do BKT é vulnerável ao problema estatístico da identificabilidade, isto é, uma mesma estimativa de desempenho pode ser resultado de várias combinações distintas de parâmetros do modelo [Beck and Chang 2007]. Dessa forma, múltiplas combinações de $P(L_0)$, $P(T)$, $P(G)$ e $P(S)$ podem resultar em uma mesma taxa de erro para $P(correto_n)$.

Para resolver esse problema, [Beck and Chang 2007] propuseram avaliar as diferentes soluções para encontrar as mais plausíveis, restringindo os parâmetros do modelo por meio de uma probabilidade prévia obtida de todos os componentes de conhecimento. Contudo, segundo [d Baker et al. 2008], essa solução pode levar a um novo problema, a degeneração do modelo, que é a obtenção de parâmetros que refletem um comportamento paradoxal.

Esse novo problema acontece quando $P(S)$ e $P(G)$ são maiores que 0.5. Assim, quando $P(S) > 0.5$, o estudante que sabe um conteúdo tem maior chance de errar do que de acertar a resposta. Por outro lado, quando $P(G) > 0.5$, o estudante que não sabe um conteúdo tem maior chance de acertar do que de errar a resposta.

Os problemas de degeneração e identificabilidade podem ser resolvidos de diversas maneiras, por meio do ajuste dos parâmetros do modelo. A forma mais simples consiste na atribuição de valores fixos aos parâmetros $P(L_0)$, $P(T)$, $P(G)$ e $P(S)$ para cada

componente de conhecimento, conforme [Corbett and Anderson 1994]. Para isso, $P(S)$ e $P(G)$ devem receber valores menores que 0.3, como forma de inibir a degeneração do modelo.

Por fim, há outras abordagens para ajuste dos parâmetros: *Expectation-Maximization* [Beck and Chang 2007], *Conjugate Gradient Search* [Corbett and Anderson 1994], *Discretized Brute-Force Search* [Yudelson et al. 2013], *Contextual Guess-and-Slip* [Beck and Chang 2007] e *Individual Difference Weights* [Corbett and Anderson 1994].

2.2. *Performance Factor Analysis*

A Análise de Fator de Desempenho (PFA, do inglês *Performance Factor Analysis*) é um modelo de regressão logística capaz de predizer a acurácia baseada no número de fracassos e acertos do aluno anteriormente [Kurup et al. 2016][Pavlik Jr et al. 2009]. O modelo se baseia em uma reconfiguração da Análise de Fator de Aprendizagem (LFA, *Learning Factor Analysis*) [Cen et al. 2006].

O formato tradicional da LFA é apresentado na Equação 6, em que m é um valor que representa o aprendizado acumulado pelo estudante i, usando um ou mais componentes de conhecimento (KC, do inglês *Knowledge Components*) j. As dificuldades de cada KC são capturadas pelo parâmetro β, e a importância da realização de práticas anteriores para cada KC é uma função de *n* observações prévias para cada estudante i com KC j (adicionado de γ para cada observação).

$$m(i, j \in KCs, n) = \alpha_i + \sum_{j \in KCs} (\beta_j + \gamma_j, n_i, j) \tag{6}$$

A função logística apresentada na Equação 7 é utilizada na conversão de m em valores de predição probabilística. Esse modelo é equivalente ao da Equação 6, com $\gamma = 0$ e apenas um valor para β [Pavlik Jr et al. 2009].

$$p(m) = \frac{1}{1 + e^{-m}} \tag{7}$$

A PFA é uma variação da LFA que utiliza a forma padrão do modelo de regressão logística, adicionando o desempenho do estudante como variável dependente. Os dois modelos são bastante similares, conforme se pode verificar na Equação 8. A diferença básica está na reconfiguração das variáveis independentes, vez que a PFA elimina a variável do estudante α e muda as variáveis KCs com a questão a ser respondida. Além disso, o parâmetro β não mais captura a dificuldade do KC e sim da questão a ser resolvida. A PFA também adiciona dois parâmetros γ e ρ para cada habilidade, refletindo os efeitos dos sucessos e falhas anteriores [Gong et al. 2011].

$$m(i, j \ \in \ KCs, f) = \sum_{j \in KCs} (\beta_j + \gamma_j S_{i,j} + \rho_j f_{i,j}) \tag{8}$$

Entretanto PFA não leva em consideração a ordem de acertos e erros do aluno ao fazer seu cálculo, o que pode vir a ser um problema. Um exemplo bem simples de se

entender é se o aluno responder 4 questões, 2 corretas primeiramente e duas incorretas, e a ordem for invertida, o resultado da curva de aprendizado do aluno deverá ser possivelmente diferente, entretanto isso não acontece. Trabalhos como [Gong et al. 2011] fazem uma proposta para adicionar um fator de envelhecimento para os dados, ou seja, quanto mais velho for o dado, menos impacto no cálculo ele deverá ter.

3. Experimento

Esta seção apresenta a passo a passo utilizado para o desenvolvimento do experimento proposto neste trabalho. O experimento consiste em uma análise qualitativa e comparativa da capacidade preditiva de modelos em relação a modelos no domínio da programação. Compreende, também, a utilização de técnicas de modelagem de conhecimento para criação de tais modelos.

O experimento pode ser dividido em 4 etapas : seleção de datasets, seleção de técnicas de modelagem, criação dos modelos e análise dos modelos.

Inicialmente foi feita uma seleção de possíveis datasets a serem utilizados no experimento. Foram selecionados os datasets KDD CUP 2010 , um dataset de álgebra com 9 milhões de entradas e *Spanish Vocabulary Spring 2006* , um dataset de espanhol com aproximadamente 17 mil entradas, devido à popularidade. A Etapa entretanto falhou em encontrar datasets abertos voltados para o ensino da programação.

Devido à falha em encontrar datasets voltados ao domínio da programação, foi necessário a compilação de um novo dataset. Para a criação do novo dataset foram usados os dados coletados anteriormente utilizando o AVA COSMO em [Raposo et al. 2019], sendo chamado de COSMODS para este artigo. COSMODS é composto por aproximadamente 7000 tentativas feitas por 90 alunos em uma turma inicial de algoritmos entre o ano de 2017 e 2019. O dataset possui 2 tipos de conteúdo: variáveis e condicionais. Além disso, durante o experimento foi identificado que considerar os erros de compilação dos alunos da mesma forma que erros de código pode afetar a capacidade preditiva dos modelos. Dada essa observação, foi criado um segundo dataset que consiste nos dados do COSMODS sem os erros de compilação dos alunos. Tal dataset é chamado de COSMO-F neste artigo. COSMO-F possui aproximadamente 4000 tentativas de 90 alunos em uma turma inicial de algoritmos.

A seleção das técnicas de modelagem utilizadas nesse trabalho se deu através da popularidade e relevância dentro da literatura. *Bayesian Knowledge Tracing* se mostrou a forma mais popular enquanto *Performance Factor Analysis* se mostrou como uma alternativa ao BKT, entretanto BKT possui diversas variações relacionadas a como o modelo faz o *fitting* de seus dados. Para este trabalho foram selecionados as versões do BKT que usam *expectation-maximization* [Beck and Chang 2007] e *Conjugate Gradient Search* [Corbett and Anderson 1994], além do PFA.

Os datasets foram então utilizados para alimentar os algoritmos de BKT e PFA e, com isso, criar os modelos de conhecimento capazes de apresentar predições sobre o conhecimento dos alunos. Os resultados serão apresentados e analisados por técnica de modelagem na seção seguinte.

4. Resultados

Esta seção apresenta os resultados principais do experimento, por meio dos dados obtidos e da análise da capacidade preditiva dos modelos de conhecimento gerados.

4.1. *BKT - Expectation-Maximization*

A Tabela 1 apresenta os resultados obtidos através do BKT-EM. Para o dataset COSMODS o modelo teve um RMSE de 0,4080 e uma Acurácia de 0,7698, resultados normais e dentro do esperado em comparação aos outros modelos. O modelo para o dataset COSMO-F, que não possui os erros de compilação dos alunos, apresentou uma queda na acurácia e um aumento do RMSE em relação a sua contraparte COSMODS. É interessante perceber que, apesar de ter os erros de compilação removidos do dataset, os resultados apresentados para o dataset COSMO-F tiveram uma acurácia pior que o modelo COSMODS e um RMSE maior. Isso possivelmente se dá devido ao fato de uma grande quantidade das entradas são erros de compilação, o que faz com que seja mais fácil prever o erro do aluno.

O modelo utilizando o dataset KDD CUP apresentou uma acurácia de 0,8571 e RMSE de 0,3390, o melhor resultado para os modelos gerados utilizando BKT-EM. Já o modelo criado utilizando o dataset *Spanish Vocabulary* apresentou uma acurácia de 0,8044 e um RMSE de 0,3823 mostrando que apesar de importante, é possível criar modelos válidos com um número de dados reduzidos.

Tabela 1. Resultado dos Modelos BKT-EM

	RMSE	Acurácia
COSMODS	0,4080	0,7698
COSMO-F	0,4248	0,7191
KDD CUP	0,3390	0,8571
Spanish Vocabulary	0,3823	0,8044

De maneira geral todos os valores apresentados pelos modelos gerados pelo BKT-EM são consistentes com o que aponta a literatura. Vale ressaltar que RMSE é uma métrica que diz por quanto o modelo erra e seus valores não são absolutos e variam de acordo com o que se tenta prever. Por exemplo, se o modelo estiver tentando prever a quantidade de cimento em um prédio e erra por alguns centímetros quadrados não é um problema, porém se a métrica observada for em metros ou quilômetro o mesmo valor não é mais aceitável. Com isso em mente os valores de RMSE apresentados para o dataset COSMODS e COSMO-F, apesar de estarem dentro do esperado, estão mais próximos de invalidarem o modelo.

4.2. *BKT - Conjugate Gradient Descent*

A Tabela 2 apresenta os resultados obtidos através do BKT-CGD. Para o dataset COSMODS o modelo teve um RMSE de 0,5494 e acurácia de 0,3280, mostrando um claro constaste em comparação com o modelo BKT-EM do mesmo dataset. Já para o dataset COSMO-F, onde os erros de compilação foram removidos, foram obtidos os valores 0,5104 para RMSE e 0,5061 para acurácia. BKT-CGD apresentou valores abaixo do esperado em ambos datasets, entretanto percebe-se que os resultados de COSMO-F foram

melhores que os resultados de COSMODS se diferenciando do que aconteceu com BKT-EM.

O modelo criado utilizando o dataset KDD CUP apresentou uma acurácia de 0,8547 e RMSE de 0,3566. Os valores apresentados são comparáveis com os valores vistos BKT-EM para o mesmo dataset, isso se dá provavelmente devido a grande quantidade de dados presentes no KDD CUP. Já o modelo criado utilizando o dataset *Spanish Vocabulary* apresentou uma acurácia de 0,1916 e RMSE de 0,7246, valores que invalidam completamente o modelo. Não foi possível identificar o motivo da discrepância de valores apresentados entre os modelos BKT-EM e BKT-CGD.

Tabela 2. Resultado dos Modelos BKT - CGD

	RMSE	Acurácia
COSMODS	0,5494	0,3280
COSMO-F	0,5104	0,5061
KDD CUP	0,3566	0,8547
Spanish Vocabulary	0,7246	0,1916

Os valores apresentados pelos modelos criados pelo *BKT Conjugate Gradient Descent* se mostraram em geral insatisfatórios e muito dependente de uma grande quantidade de dados para serem válidos, como no caso do dataset KDD CUP. Além disso, não existiu nenhuma melhora considerável ao BKT-EM o que faz com que o resultado apresentado pelo BKT-CGD seja o pior do experimento.

4.3. *Performance Factor Analysis*

A tabela 3 apresenta os resultados obtidos através do PFA. Para o dataset COSMODS o modelo teve um RMSE de 0,4235 e uma acurácia de 0,7161, números muitos próximos do modelo gerado pelo BKT-EM. O modelo para o dataset COSMO-F, que não possui erros de compilação, apresentou RMSE de 0,4065 e acurácia de 0,5016, acurácia bem abaixo do esperado em comparação com o outro modelo gerado pelo BKT-EM. Percebe-se que o PFA é capaz de produzir modelos com qualidade semelhante aos de BKT-EM, entretanto ele é mais dependente da quantidade de dados, outro fator que pode estar afetando o modelo é o fato da remoção de erros de compilação, uma vez que remove grande parte dos erros dos alunos, fazendo ficar mais difícil prever os resultados.

O modelo produzido pelo PFA para o dataset KDDCUP apresentou um RMSE de 0,3192 e uma acurácia de 0,8665, valores superiores aos apresentados pelo BKT-EM. Já o modelo criado utilizando o dataset *Spanish Vocabulary* apresentou RMSE de 0,3617 e acurácia de 0,8176, valores também superiores aos apresentados pelo BKT-EM.

Tabela 3. Resultado dos modelos PFA

	RMSE	Acurácia
COSMODS	0,4235	0,7161
COSMO-F	0,4065	0,5016
KDD CUP	0,3192	0,8665
Spanish Vocabulary	0,3618	0,8176

PFA se mostrou de maneira geral como uma alternativa viável para o BKT-EM para o dataset COSMODS, entretanto a técnica de modelagem se mostrou muito dependente da quantidade e dos dados presentes no dataset. Entretanto, é de interesse para a modelagem que os erros de compilação sejam removidos, visto que eles não fazem parte do processo de aprendizagem do conteúdo da disciplina, e PFA se mostrou como uma alternativa não viável uma vez que tais dados são removidos.

5. Conclusão

Este artigo descreve uma análise qualitativa e comparativa acerca do emprego de técnicas de modelagem para o ensino de algoritmos, visando identificar possíveis peculiaridades. Nesse âmbito, realizou-se um experimento em que foram criados diversos modelos utilizando técnicas de modelagem e *datasets* diversos. Foram criados modelos com as seguintes técnicas: *Bayesian Knowledge Tracing - Expectation-Maximization*, *Bayesian Knowledge Tracing - Conjugate Gradient Descent* e *Performance Factor Analysis*. Os *datasets* utilizados no experimento foram os seguintes: KDD CUP 2010, *Spanish Vocabulary*, COSMODS e COSMO-F.

De maneira geral, o BKT-EM mostrou-se a técnica de modelagem com maior estabilidade dentre as avaliadas. Os resultados mostraram que, embora uma gama de dados rica seja importante para a capacidade preditiva dos modelos, também é possível criar modelos válidos com uma quantidade de dados reduzida. Já o BKT-CGD apresentou o pior resultado dentre as técnicas, mostrando que essa variação é muito dependente da quantidade de dados disponíveis. O PFA mostrou-se ser uma alternativa viável. É importante salientar que o BKT-EM apresentado resultados superiores para alguns *datasets*. Contudo, ao remover os erros de compilação, a técnica apresentou resultados inferiores. Assim, a variação de BKT-EM mostrou-se superior no que diz respeito à criação de modelos preditivos para o ensino de algoritmos.

Um dos achados de pesquisa deste trabalho foi o impacto que se tem pela forma que se trata erros de compilação nos dados coletados. Considerar erros de compilação da mesma forma que erros de código tem muita influência sobre os resultados dos modelos de conhecimento gerados, especialmente quando se leva em conta a grande incidência de erros de compilação em uma turma de aprendizes de algoritmos. Durante este trabalho, identificou-se que isso é importante na capacidade preditiva de modelos, tornando até alguns modelos impraticáveis. De maneira geral, considerar ou não os erros de compilação podem melhorar ou piorar a capacidade preditiva dos modelos, de acordo com as técnicas utilizadas. Pode-se argumentar tanto pela remoção dos erros de compilação quanto pela sua admissão. Entretanto, existe o argumento de que erros de compilação não fazem parte do aprendizado do aluno, visto que são erros de sintaxe e não erros de lógica de programação.

Para trabalhos futuros, pretende-se extrair informações dos códigos-fonte apresentados pelos alunos para as diversas atividades, os quais foram armazenados durante a etapa de coleta de dados. Esses códigos-fonte possuem diversos padrões que podem ser usados para inferir informações sobre a compreensão dos alunos acerca do conteúdo ensinado. Além disso, é necessário ampliar a coleta de dados, a fim de diversificar as informações presentes no *dataset*.

Referências

Beck, J. E. and Chang, K.-m. (2007). Identifiability: A fundamental problem of student modeling. In *International Conference on User Modeling*, pages 137–146. Springer.

Cen, H., Koedinger, K., and Junker, B. (2006). Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, pages 164–175. Springer.

Corbett, A. T. and Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278.

d Baker, R. S., Corbett, A. T., and Aleven, V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *International conference on intelligent tutoring systems*, pages 406–415. Springer.

David, Y. B., Segal, A., and Gal, Y. K. (2016). Sequencing educational content in classrooms using bayesian knowledge tracing. In *Proceedings of the sixth international conference on Learning Analytics & Knowledge*, pages 354–363. ACM.

Gong, Y., Beck, J. E., and Heffernan, N. T. (2011). How to construct more accurate student models: Comparing and optimizing knowledge tracing and performance factor analysis. *International Journal of Artificial Intelligence in Education*, 21(1-2):27–46.

Kurup, L. D., Joshi, A., and Shekhokar, N. (2016). A review on student modeling approaches in its. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 2513–2517.

Käser, T., Klingler, S., Schwing, A. G., and Gross, M. (2017). Dynamic bayesian networks for student modeling. *IEEE Transactions on Learning Technologies*, 10(4):450–462.

Pavlik Jr, P. I., Cen, H., and Koedinger, K. R. (2009). Performance factors analysis–a new alternative to knowledge tracing. *Online Submission*.

Psotka, J., Massey, L. D., and Mutter, S. A. (1988). *Intelligent tutoring systems: Lessons learned*. Psychology Press.

Ramesh, V. M., Rao, N. J., and Ramanathan, C. (2015). Implementation of an intelligent tutoring system using moodle. In *2015 IEEE Frontiers in Education Conference (FIE)*, pages 1–9.

Raposo, A. C., Maranhão, D., and Neto, C. S. (2019). Analise do modelo bkt na avaliacão da curva de aprendizagem de alunos de algoritmos. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 479.

Yudelson, M. V. (2016). Individualizing bayesian knowledge tracing. are skill parameters more important than student parameters?. *International Educational Data Mining Society*.

Yudelson, M. V., Koedinger, K. R., and Gordon, G. J. (2013). Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer.