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

Algoritmos de Reconhecimento de Dígitos para Integração de Equações Manuscritas em Sistemas Tutores Inteligentes

Sergio Chevtchenko[1], Luiz Rodrigues[2], Daniel Rosa[3], Filipe Cordeiro[3], Ruan Carvalho[3], Everton Souza[3], Thales Vieira[2], Diego Dermeval[2], Ig Ibert Bittencourt[2,4], Seiji Isotani[4], Marcelo Marinho[3], Valmir Macario[3]

[1]Universidade Federal de Pernanbuco - Brasil

[2]Núcleo de Excelência em Tecnologias Sociais (NEES), Universidade Federal de Alagoas - Brasil

[3]Universidade Federal Rural de Pernambuco - Brasil

[4]Escola de Educação de Harvard - EUA

sfc@cin.ufpe.br

Resumo. *Sistemas Tutores Inteligente (STIs) têm sido amplamente utilizados para auxiliar no aprendizado de matemática. No entanto, a diferença na forma de inserção de soluções nos STIs, que requer o uso de um teclado, em comparação com a prática padrão de escrever à mão, pode levar a problemas de usabilidade e prejudicar a aprendizagem. Para superar essa limitação, pesquisas recentes têm explorado o reconhecimento de caracteres escritos à mão em papel como entrada para os STIs. Porém, existe uma lacuna de conhecimento em relação ao desempenho dos algoritmos de reconhecimento de dígitos avançados no contexto de operações matemáticas básicas. Este artigo compara quatro algoritmos de última geração para o reconhecimento de dígitos em problemas matemáticos de adição e subtração. Os resultados revelam que o algoritmo BTTR obteve o melhor desempenho em termos de acurácia, enquanto o algoritmo SAN apresentou um bom equilíbrio entre acurácia e velocidade de reconhecimento. Essas descobertas são relevantes para pesquisadores e desenvolvedores ao selecionar os algoritmos mais adequados para o desenvolvimento de STIs baseados em entrada escrita à mão.*

Abstract. *Intelligent Tutoring Systems (ITSs) have been widely used to assist in mathematics learning. However, the difference in input methods between ITSs, which require keyboard usage, and the standard practice of handwriting, can lead to usability issues and hinder learning. To overcome this limitation, recent research has explored the recognition of handwritten characters on paper as input for ITSs. However, there is a knowledge gap regarding the performance of advanced digit recognition algorithms in the context of basic mathematical operations. This article compares four state-of-the-art algorithms for digit recognition in addition and subtraction math problems. The results reveal that the BTTR algorithm achieved the best performance in terms of accuracy, while the SAN algorithm demonstrated a good balance between accuracy and recognition speed. These findings are relevant for researchers and developers in selecting the most suitable algorithms for the development of ITSs based on handwritten input.*

1. Introdução

Sistemas de Tutoria Inteligente (STIs) são tecnologias educacionais que exploram a inteligência artificial para fornecer feedback e instruções personalizadas [Nkambou et al. 2010]. Os STIs têm sido amplamente utilizados para auxiliar no aprendizado de matemática, um contexto em que evidências empíricas demonstram seu potencial em comparação com uma variedade de métodos instrucionais [Hillmayr et al. 2020, Steenbergen-Hu and Cooper 2013]. No entanto, como uma tecnologia baseada em computador, os STIs possuem uma característica que pode mitigar suas contribuições para a educação matemática: enquanto estudantes aprendem e estão acostumados a resolver problemas matemáticos à mão (ou seja, escrevendo números e equações no papel), eles precisam inserir suas soluções nos STIs usando um teclado [Soofi and Ahmed 2019]. Consequentemente, essa diferença em relação à prática padrão dos estudantes pode levar a problemas de usabilidade e prejudicar experiências de aprendizagem [Anthony et al. 2005, Anthony et al. 2007, Morais and Jaques 2022].

Diante das limitações de digitar ao invés de escrever, pesquisas sobre STIs para matemática têm explorado interfaces para entrada de dados a partir da escrita à mão. Por exemplo, [Morais and Jaques 2022] apresentam um estudo experimental que, no contexto de um STI para álgebra, compara dois métodos de entrada: digitando em um teclado e escrevendo em uma interface de *touchscreen*. Os resultados demonstram que os estudantes que usaram o método de escrita manual resolveram mais perguntas e foram mais rápidos do que aqueles que usaram o outro método, embora não tenham encontrado diferenças significativas em termos de aprendizado dos estudantes. É importante destacar que esse, bem como estudos similares (por exemplo, [Anthony et al. 2012, Wang et al. 2016]), consideram a entrada por escrita à mão através de uma interface de touchscreen (e.g., usando um tablet ou o touchpad de um notebook). Diferentemente, pesquisas recentes têm defendido a importância de STIs reconhecerem como entrada soluções escritas à mão em papel [Patel et al. 2022]. A ideia é permitir que os estudantes trabalhem como estão acostumados (ou seja, resolvendo problemas matemáticos escrevendo em uma folha de papel) e ainda se beneficiem do feedback e das instruções personalizadas que os STIs fornecem.

Para implementar essa ideia, pode-se explorar o reconhecimento de caracteres/dígitos: uma técnica de visão computacional que reconhece dígitos e caracteres escritos à mão, entre outros símbolos matemáticos [Mouchère et al. 2014]. No entanto, apesar de muitas pesquisas nessa área [Davis et al. 2020, Vu et al. 2021, Patel et al. 2022, Zhao et al. 2021, Yuan et al. 2022a, Li et al. 2022, Yuan et al. 2022b, Zhao and Gao 2022], estudos anteriores sobre STIs para educação matemática com base em entrada escrita à mão não experimentaram com algoritmos de última geração para reconhecimento de dígitos [Davis et al. 2020], ou são limitados a problemas de álgebra [Patel et al. 2022]. Para preencher essa lacuna, este artigo apresenta um estudo experimental comparando quatro algoritmos de última geração para reconhecimento de dígitos no contexto de problemas matemáticos de adição e subtração. Portanto, expandimos a literatura ao: i) considerar algoritmos de última geração, em vez de desenvolver reconhecedores do zero [Davis et al. 2020], e ii) focar em um assunto matemático elementar (ou seja, adição e subtração), em vez de tópicos mais avançados [Patel et al. 2022].

Com base nesse contexto, as contribuições deste artigo são as seguintes. Primeiro, este artigo revela como a precisão de diferentes algoritmos de última geração se compara

para o contexto de equações de soma e subtração. Essa visão é valiosa para praticantes, indicando quais algoritmos têm mais chances de alcançar as melhores taxas de reconhecimento. Segundo, o estudo também revela como o desempenho dos algoritmos se compara em termos de tempo de inferência. Esse resultado ajuda praticantes tomarem decisões alinhadas com os requisitos não funcionais de seus STIs (e.g., capacidade de processamento e conectividade com servidores externos). Por fim, este artigo contribui com o conjunto de dados usado em nossos experimentos, que pode ser usado em pesquisas futuras para testar e expandir nossos resultados. Assim, este artigo contribui evidências empíricas que informam o projeto, bem como pesquisas futuras, sobre STIs baseados em entradas escritas à mão.

2. Revisão da Literatura

Existem bases de dados de expressões matemáticas com consideráveis desafios em termos de complexidade de expressões, tais como CHROME [Mouchère et al. 2014], HandMath [Le and Nakagawa 2016] e HME100K [Yuan et al. 2022b]. No entanto, poucos trabalhos focam especificamente em desenvolvimento de bases de dados para expressões matemáticas de ensino primário.

Davis et al. [Davis et al. 2020] propuseram um sistema de reconhecimento de equações em estágios, utilizando combinação de algoritmos de deep learning. O primeiro estágio consiste na localização dos dígitos e símbolos por uma rede SSD Mobilenet [Howard et al. 2017]. O treinamento desta etapa envolve anotações de localização dos dígitos e símbolos individuais na base de dados. A etapa seguinte emprega uma rede convolutiva para reconhecimento dos símbolos previamente recordados. Por fim, uma heurística é proposta para resolução e identificação de erros nas equações. As equações verticais são previamente impressas em um papel e o aluno precisa escrever apenas a resposta abaixo da linha da equação.

Com objetivo de facilitar correção de equações, Vu et al. [Vu et al. 2021] propuseram uma solução de agrupamento de equações escritas à mão. Argumentando que reconhecimento de equações escritas à mão é uma tarefa complexa e com precisão ainda baixa, os autores propõem um sistema semi-automático de correção. O sistema agrupa equações semelhantes, evitando que o professor tenha que corrigir múltiplas soluções parecidas.

A proposta de Patel et al. [Patel et al. 2022] consiste em um sistema de assistência para estudantes com difícil acesso a formas online ou primariamente digitais de aprendizagem, com objetivo principal de oferecer recomendações e comentários sobre erros comumente cometidos. Devido ao foco em estudos offline, um aspecto essencial do sistema é a captura de atividades manuscritas em papel e a sua digitalização e processamento, permitindo, assim, uma avaliação adaptativa. Imagens coletadas de resolução de equações por um grupo pequeno de estudantes foram manualmente segmentadas e uma ferramenta comercial ficou responsável pelo reconhecimento de escrita. Cada saída digitalizada está associada a um nível de confiança para determinar se deve ser analisada ou descartada. Os autores testaram a distância de edição para auxiliar no agrupamento de respostas parecidas, porém perceberam que se dois estudantes chegassem à mesma resposta com diferença grande na quantidade de passos, a distância seria indesejavelmente elevada. Os passos finais de resolução foram agrupados para a identificação de padrões

revelando procedimentos incorretos. Foram encontrados padrões de erros que podem ser generalizados com uma base de dados maior e mais diversificada.

Algoritmos avaliados neste trabalho são do tipo end-to-end, produzindo uma equação em LaTeX a partir de uma imagem em escala de cinza. Consequentemente, estes algoritmos podem ser facilmente retreinados para inclusão de novos símbolos ou estruturas de equação e não necessitam de anotações de localização dos símbolos individuais. Além disso, a base proposta envolve equações escritas livremente, proporcionando um desafio adicional para a etapa de reconhecimento.

3. Método

3.1. Base de dados

A base de dados utilizada neste trabalho é composta por 232 imagens escritas à mão por 4 pessoas, sendo 140 utilizadas para treinamento, 46 para validação e 46 para teste [Rosa et al. 2023]. Cada anotador capturou as imagens por um modelo distinto de celular e há algumas escritas a lápis, e outras à caneta. Todas elas envolvem apenas operações de soma ou subtração cujos operandos apresentam no máximo 3 dígitos: o resultado da soma eventualmente tem um algarismo extra, enquanto o da subtração é garantidamente não-negativo. Em ambas as operações pode ocorrer o "vai 1", porém na subtração há também um estilo adicional com "vai -1". A Figura 1 exibe alguns arquivos usados neste trabalho.

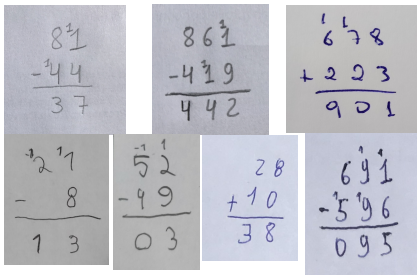

Figura 1. Exemplos de imagens da base de dados utilizada.

3.2. Algoritmos

Esta seção apresenta uma breve descrição dos algoritmos de aprendizagem profunda avaliados na base de dados proposta.

3.2.1. SAN

Reconhecer uma expressão matemática manuscrita é uma tarefa desafiadora: além da potencial ambiguidade da escrita humana, algumas estruturas apresentam uma sintaxe difícil, como somatório, fração e radiciação. *Syntax-Aware Network* (SAN) propõe usar uma rede neural profunda que incorpore os relacionamentos sintáticos gramaticais das expressões [Yuan et al. 2022a]. Além de validar o desempenho do modelo na base CHROME [Mouchère et al. 2014], os autores também apresentaram um novo conjunto de imagens: HME100K contém cerca de 10 vezes mais expressões que o CHROME e é composta por milhares de anotadores [Yuan et al. 2022b].

3.2.2. CAN

Counting Aware Network (CAN) parte da hipótese de que HMER e contagem de símbolos são tarefas complementares. Um módulo de contagem computa a quantidade de cada símbolo presente na expressão. Então essa informação é repassada para outro módulo, o qual é responsável por decodificar cada símbolo presente conforme a posição em que ocorre na expressão matemática [Li et al. 2022]. A performance do CAN foi avaliada nas bases CHROME [Mouchère et al. 2014] e HME100K [Yuan et al. 2022b].

3.2.3. BTTR e CoMER

Um desafio comum a ser superado em algoritmos para HMER é o problema da cobertura: algumas regiões da expressão podem ser traduzidas múltiplas vezes (*over-parsing*), e outras podem ficar sem tradução (*under-parsing*). Longas sequências de caracteres em Latex também tornam difícil para modelos de redes neurais recorrentes reconhecerem o relacionamento entre símbolos distantes, como um "{" inicial com um correspondente "}" apenas no final. Uma vez que modelos tradicionais realizam a predição de símbolos da esquerda para a direita, os prefixos da saída apresentam maior precisão que os sufixos. O algoritmo *Bidirectionally Trained TRansformer* (BTTR) aborda essas questões propondo um *transformer decoder* (uma arquitetura específica de redes neurais) com uma estratégia de treinamento bidirecional (da esquerda para a direita e da direita para a esquerda) [Zhao et al. 2021]. O desempenho do BTTR é validado pelos autores nas bases CHROME [Mouchère et al. 2014].

Zhao e Gao ainda propuseram dois mecanismos de cobertura adicionais ao BTTR, além de um método de aumento de dados, a fim de melhorar a performance e definiram um novo algoritmo: *Coverage information in the transforMER decoder* (CoMER) [Zhao and Gao 2022].

3.3. Otimização de parâmetros

Algoritmos de *deep learning* apresentam hiperparâmetros que definem as configurações do modelo a ser treinado e são determinantes para o desempenho. Uma vez que o espaço de busca é muito grande, um ajuste manual torna-se impraticável. Neste trabalho adotamos a biblioteca *Optuna*: um programa de otimização de código aberto [Akiba et al. 2019]. Assim é possível definir quais parâmetros devem ser otimizados e os respectivos espaços de busca.

A Tabela 1 apresenta os conjuntos de hipeparâmetros utilizados para ajuste dos algoritmos. Nota-se que pelas semelhanças na implementação, algoritmos SAN e CAN compartilham do mesmo conjunto de parâmetros, e um outro conjunto foi utilizado para BTTR e CoMER. Os termos destacados em cada linha correspondem aos valores padrão adotados pelos algoritmos.

Conforme discutido na Seção 3.2, os algoritmos avaliados neste trabalho foram inicialmente propostos para bases de dados com equações complexas e com muitas imagens: HME100K [Yuan et al. 2022b] e CHROME [Mouchère et al. 2014], com aproximadamente 100 mil e 10 mil imagens respectivamente. De modo geral, algoritmos de aprendizagem de máquina para HMER precisam de grande quantidade de dados para

SAN e CAN	BTTR e CoMER
nDenseBlocks: [4, 8, **16**]	d_model: [64, 128, **256**]
growthRate: [6, 12, **24**]	growthRate: [8, 16, **24**]
reduction: [0.1, 0.2, **0.5**]	num_layers: [4, 8, **16**]
bottleneck: [False, **True**]	nhead: [2, 4, **8**]
use_dropout: [False, **True**]	num_decoder_layers: [1, 2, **3**]
decoder_size: [64, 128, **256**]	dim_feedforward: [256, 512, **1024**]
attention_dim: [128, 256, **512**]	dropout: [0.0, **0.3**]
attention_ch: [8, 16, **32**]	beam_size: [5, **10**]
-	dc: [8, 16, **32**]

Tabela 1. Espaço de busca dos hiperparâmetros correspondentes aos modelos avaliados neste trabalho. Valores destacados fazem parte da configuração padrão, proposta originalmente para cada algoritmo.

treinamento. Uma vez que a base de dados proposta neste trabalho é significativamente menor, com 232 imagens, faz-se necessário adotar uma estratégia para evitar *overfitting* (memorização do conjunto de treinamento). Dessa forma foi utilizada a biblioteca *Tourchvision*[1] para aumento de dados. As transformações utilizadas são: rotação aleatória para esquerda e direita entre -30° e +30°, ajuste aleatório de *sharpness* com fator de 2 e distorções elásticas com parâmetros *alpha* e *sigma* de 40.0 e 3.0, respectivamente. Os outros parâmetros das transformações mantiveram seus valores padrão. Assim, além dos hiperparâmetros definidos na Tabela 1, cada configuração do algoritmo pode fazer uso do aumento de dados descrito acima. Caso utilizado, o aumento pode ser de 10x ou 100x. Conforme será visto na Seção 4, todas as configurações dos algoritmos com melhor desempenho fazem uso do aumento de dados em 100x.

Definidos todos os hiperparâmetros, a ferramenta de otimização *Optuna* executa 100 avaliações sequenciais para cada um dos quatro algoritmos (SAN, CAN, BTTR e CoMER), com objetivo de encontrar configurações com melhor desempenho em termos de taxa de acerto das expressões matemáticas (*ExpRate*) no conjunto de validação. O treinamento de cada configuração é limitado a 500 épocas, com interrupção automática caso não haja evolução no desempenho por 30 épocas seguidas. Ao final, selecionamos as 5 configurações que alcançaram o melhor desempenho em cada algoritmo [2]. Importante notar que apesar das configurações originalmente propostas para os quatro algoritmos estarem presentes no espaço de busca, conforme Tabela 1, os 20 algoritmos com melhor desempenho possuem configurações distintas e com menor quantidade de parâmetros do que as configurações originais. Este resultado sugere que para obter um melhor desempenho na base de dados proposta, algoritmos de *deep learning* devem ser ajustados e ter o seu número de parâmetros reduzido.

4. Resultados

Conforme descrito na Seção 3.3, um total de 20 algoritmos foram selecionados como resultado da etapa de otimização, sendo 5 para cada um dos modelos avaliados. Para garantir uma comparação mais justa, os algoritmos foram treinados novamente no conjunto de treinamento com um critério de parada (*early stopping*) de 100 épocas. Além disso, todos os experimentos descritos nesta seção utilizaram o mesmo computador. Em seguida,

[1] `https://pytorch.org/vision/stable/transforms.html`

[2] `https://osf.io/u9t5e/?view_only=d4533cd8d4e94f3a8203d18725c32753`

estes algoritmos foram avaliados no conjunto de teste, até então mantido separado dos experimentos anteriores. Os algoritmos foram avaliados considerando as seguintes métricas: *ExpRate*: taxa de acerto das expressões matemáticas; *WordRate*: taxa média de símbolos classificados corretamente em todo o conjunto de teste; *e1*, *e2*, *e3*: taxa de expressões acertadas com tolerância de no máximo 1 símbolo errado, ou 2, ou 3, respectivamente; e tempo de inferência médio, medido em segundos.

A Tabela 2 compila os resultados dos algoritmos. SAN obteve melhor desempenho no conjunto de validação, mas isso não necessariamente se reflete no conjunto de teste, indicando ocorrência de sobreajuste (*overfitting*). Nota-se que as configurações do algoritmo BTTR obtiveram os melhores valores para as 5 métricas de reconhecimento descritas acima. No entanto, como será analisado a seguir, associado à um custo computacional maior do que os outros algoritmos avaliados.

Algorithm	**#**	**ExpRate Validação**	**Teste**				
			ExpRate	**e1**	**e2**	**e3**	**WordRate**
SAN	1	0.7609	0.5435	0.8043	0.8913	0.9348	0.9612
	2	0.8261	0.5435	0.7174	0.8478	0.8696	0.9429
	3	**0.9347**	0.6522	0.7826	0.9130	0.9130	0.9670
	4	0.8043	0.5217	0.7609	0.7826	0.8261	0.9373
	5	0.8696	0.6087	0.7826	0.8478	0.8478	0.9147
CAN	1	0.7174	0.4348	0.6522	0.7174	0.7609	0.9187
	2	0.8913	0.5870	0.6739	0.7609	0.8261	0.9216
	3	0.3478	0.3043	0.5217	0.6521	0.8478	0.9384
	4	0.5434	0.4565	0.6304	0.7609	0.8261	0.9192
	5	0.7174	0.4783	0.6304	0.6739	0.7174	0.8934
BTTR	1	0.6087	0.5217	0.7174	0.7609	0.8696	0.9421
	2	0.8696	0.6739	0.8261	0.9130	**0.9565**	**0.9790**
	3	0.7174	0.6522	**0.8478**	**0.9565**	**0.9565**	0.9774
	4	0.6739	0.6522	**0.8478**	0.9130	0.9348	0.9723
	5	0.5435	**0.7174**	0.8043	0.8696	0.8913	0.9654
CoMER	1	0.7174	0.3478	0.6522	0.6739	0.6739	0.8778
	2	0.3043	0.0870	0.1957	0.2609	0.3043	0.7764
	3	0.2174	0.0652	0.1739	0.1957	0.3043	0.6970
	4	0.3913	0.2391	0.3913	0.4565	0.4783	0.7689
	5	0.5870	0.2609	0.3913	0.4565	0.5000	0.8100

Tabela 2. Comparação entre algoritmos

Outra métrica analisada foi o tempo de inferência: o intervalo médio necessário para o algoritmo obter o texto em Latex da expressão matemática a partir da imagem. Este tempo é medido utilizando computação paralela da placa gráfica (CUDA) ou apenas a CPU do computador. A Figura 2 permite avaliar o *trade-off* entre tempo de inferência e *ExpRate* para os 20 modelos testados. Nota-se que embora os modelos BTTR obtêm as melhores taxas de reconhecimento, algoritmo SAN oferece uma alternativa interessante quando se considera a velocidade de reconhecimento.

Para cada um dos quatro algoritmos a configuração com o melhor *ExpRate* no conjunto de testes é indicada com símbolo de estrela na Figura 2. Estas configurações foram avaliadas em termos de tempo médio de inferência utilizando tanto placa de vídeo (CUDA) RTX3060, quanto apenas o processador do computador (CPU) Intel® i5. Os resultados estão apresentados na Figura 3.

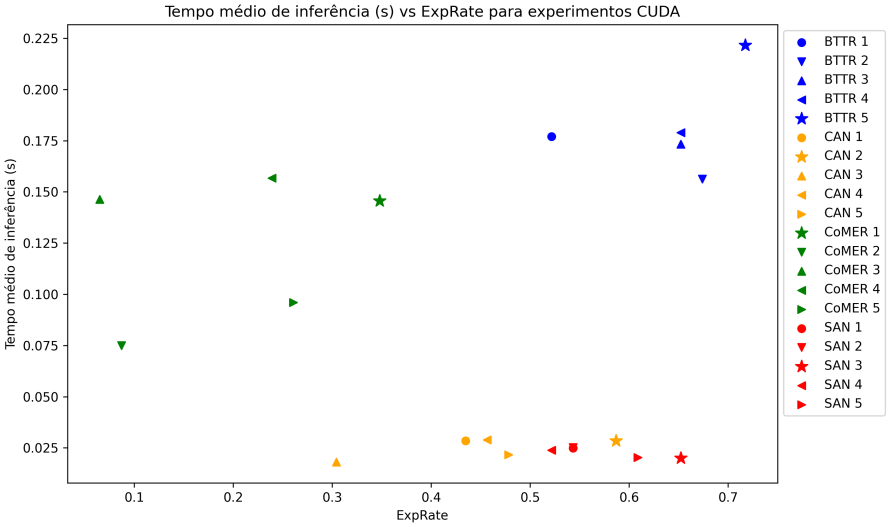

Figura 2. Comparação dos algoritmos no conjunto de teste em termos de *ExpRate* e tempo médio de inferência utilizando placa de vídeo.

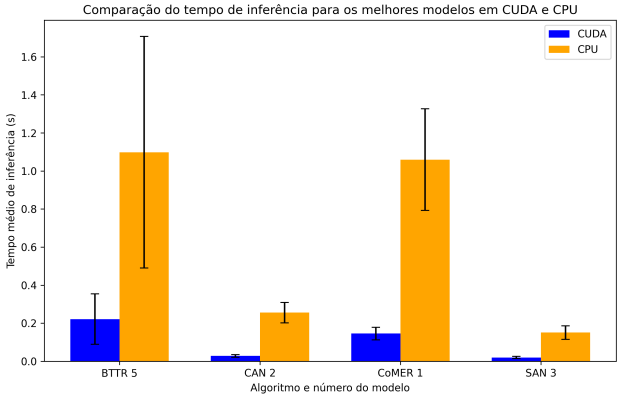

Figura 3. Comparação do tempo de inferência médio das melhores configurações, utilizando placa de vídeo (CUDA) e CPU. Desvio-padrão é indicado pelas linhas verticais.

5. Discussão

Neste estudo, observou-se que o algoritmo SAN apresentou melhor desempenho no conjunto de validação, porém, essa superioridade não se refletiu no conjunto de teste, sugerindo a ocorrência de sobreajuste (veja a Tabela 2). Os algoritmos BTTR demonstraram os melhores valores para todas as métricas de reconhecimento. No entanto, uma análise adicional revelou um custo computacional mais elevado em comparação aos outros algoritmos avaliados (veja a Figura 2). Embora os modelos BTTR tenham obtido as melhores taxas de reconhecimento, o algoritmo SAN oferece uma alternativa interessante em termos de velocidade de reconhecimento.

Em relação à literatura, embora existam bases de dados de expressões matemáticas com desafios complexos, poucos trabalhos focaram especificamente em bases de dados para expressões matemáticas de ensino primário (veja a Seção 2). Essa lacuna é signi-

ficativa no contexto dos STIs que buscam reconhecer equações manuscritas, que é fundamental para uma interação aluno-STI fluida. Além disso, a base de dados proposta, que envolve equações escritas livremente, proporciona um desafio adicional para a etapa de reconhecimento, o que reflete mais realisticamente a diversidade de respostas dos alunos. Assim, esse estudo contribui para a melhoria dos STIs com insumos sobre o potencial de diferentes algoritmos do estado da arte de visão computacional. Com base nesse contexto, as seções a seguir discutem as implicações de nossos achados, assim como recomendações para pesquisas futuras e limitações deste estudo.

5.1. Implicações

Com base em nossos resultados, esse artigo fornece três implicações principais. Primeiro, nossos resultados revelam quais modelos têm maior potencial para reconhecer equações pertinentes a STIs focados no auxílio do ensino de matemática a partir de entradas manuscritas. Nossos resultados mostram que apesar do SAN alcançar o melhor desempenho no conjunto de validação, o melhor desempenho no conjunto de teste foi do BTTR, seguido pela SAN em segundo lugar (veja a Tabela 2). Esses achados são valiosos para a literatura de STIs, uma vez que pesquisas semelhantes revelaram limitações na capacidade preditiva do reconhecimento de dígitos e são restritas a outros tipos de equações [Davis et al. 2020, Patel et al. 2022]. Assim, esse artigo demonstra o potencial do BTTR para realizar a predição de equações de soma/subtração na vertical.

Segundo, nossos resultados revelam os modelos de maior potencial em termos da velocidade de predição. Nesse quesito, SAN e CAN são os algoritmos com os menores tempo de predição. Entre estes, vale ressaltar que o SAN alcançou o segundo melhor desempenho em termos de acurácia (veja a Figura 2). Esse resultado é importante porque revela qual algoritmo tem o maior potencial para mitigar gargalos no uso de STIs baseados em entradas manuscritas (e.g., aguardar o reconhecimento de dígitos). Logo, esse artigo contribui evidência empírica de que o SAN fornece um dos melhores custo-benefício quando consideradas suas capacidades preditivas e de desempenho computacional.

Por fim, nossos resultados demonstram o papel da capacidade de processamento do dispositivo usado para o reconhecimento de tais equações. Nossos resultados demonstraram que existe uma diferença substancial entre o tempo de inferência dos algoritmos, quando esta é realizada em uma GPU, em comparação ao uso de uma CPU para o mesmo procedimento. Esse resultado é importante para o desenvolvimento de STIs porque demonstra que pesquisas sobre STIs com reconhecimento de equações manuscritas consideram o uso de dispositivos de baixo custo, como smartphones mais simples [Davis et al. 2020]. Sendo assim, esse artigo contribui ao relevar o papel de uma GPU para otimização do uso prático de tal STIs, bem como revela a importância de pesquisas que otimizem tais modelos para melhorar seus desempenhos na ausência de uma GPU.

Assim, esse estudo contribui insumos sobre quais algoritmos usar ao desenvolver STIs que suportam entradas manuscritas e quais limitações devem ser enfrentadas em pesquisas futuras, bem como revela a necessidade de pesquisas que otimizem tais modelos para dispositivos que não contam com uma GPU.

5.2. Limitações e Trabalhos Futuros

As principais limitações desse artigo estão relacionadas à base de dados. O tamanho da base de dados usado nos experimentos (232 imagens) pode ter influenciado nos resultados

encontrados, embora tenhamos usado *data augmentation* para mitigar essa questão. Além disso, apesar de restrita em comparação à literatura de visão computacional como um todo [Mouchère et al. 2014, Yuan et al. 2022b], nossa base tem tamanho semelhante às usadas em pesquisas relacionadas [Patel et al. 2022, Davis et al. 2020].

Outras considerações importantes são que i) a base não foi criada por crianças, o público que irá desenvolver as equações em um contexto de uso real, e ii) a base é limitada a equações de soma e subtração escritas na vertical. Embora essa situação possa restringir a aplicabilidade dos modelos desenvolvidos, essa abordagem nos permitiu alcançar resultados interessantes para uma configuração específica, de forma similar aos trabalhos relacionados [Patel et al. 2022, Davis et al. 2020], assim gerando insumos interessantes para pesquisas futuras e o desenvolvimento de STIs. Nesse contexto, recomendamos que pesquisas futuras busquem expandir nossa base de dados, incluindo outros formatos de equações (e.g., na horizontal), novas operações (e.g., multiplicação e divisão) e dados de outros públicos (e.g., crianças) a fim de expandir e validar nossos achados.

Além disso, deve-se considerar limitações relacionadas a nossos experimentos. Devido à complexidade dos modelos de visão computacional, o tempo para treinamento e validação dos algoritmos impõe restrições no número de algoritmos a ser considerado, bem como o quanto seus parâmetros podem ser otimizados. Para mitigar tal restrição, optamos por considerar apenas os principais algoritmos do estado da arte e usamos uma biblioteca para viabilizar a otimização de parâmetros, dado o tamanho do espaço de busca, conforme recomendado na literatura [Akiba et al. 2019]. Entretanto, reconhecemos que existe espaço para maiores explorações, logo, recomendamos que pesquisas futuras expandam nossos experimentos com estudos envolvendo outros algoritmos, *transfer learning*, parâmetros de otimização, e configurações de *data augmentation*.

6. Considerações Finais

Embora os STIs tenham sido amplamente utilizados para aprimorar o aprendizado de matemática, sua dependência de entrada por teclado apresenta problemas de usabilidade e dificulta a transferência de habilidades do ambiente papel para o digital. Para superar essa lacuna, pesquisas recentes têm explorado o reconhecimento de caracteres escritos no papel, à mão como entrada para os STIs. No entanto, ainda falta conhecimento sobre o desempenho de algoritmos avançados de reconhecimento de dígitos em operações matemáticas básicas.

Assim, este estudo compara quatro algoritmos de última geração para o reconhecimento de dígitos em problemas de adição e subtração. Os resultados indicam que o algoritmo BTTR supera os demais em termos de acurácia, enquanto o algoritmo SAN alcança um equilíbrio favorável entre acurácia e velocidade de reconhecimento. Essas descobertas têm implicações significativas para pesquisadores e desenvolvedores que buscam projetar STIs que aproveitem efetivamente a entrada escrita à mão. Ao selecionar algoritmos adequados, educadores podem aprimorar a usabilidade e os resultados de aprendizado dos STIs, capacitando os alunos a dominar conceitos matemáticos de forma mais eficiente.

Agradecimentos

Gostaríamos de expressar a nossa gratidão a todos os participantes deste projeto nacional. Este trabalho foi apoiado pelo Ministério da Educação (MEC).

Referências

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Anthony, L., Yang, J., and Koedinger, K. R. (2005). Evaluation of multimodal input for entering mathematical equations on the computer. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*, pages 1184–1187.

Anthony, L., Yang, J., and Koedinger, K. R. (2007). Benefits of handwritten input for students learning algebra equation solving. *Frontiers in Artificial Intelligence and Applications*, 158:521.

Anthony, L., Yang, J., and Koedinger, K. R. (2012). A paradigm for handwriting-based intelligent tutors. *International Journal of Human-Computer Studies*, 70(11):866–887.

Davis, S. R., DeCapito, C., Nelson, E., Sharma, K., and Hand, E. M. (2020). Homework helper: Providing valuable feedback on math mistakes. In *Advances in Visual Computing: 15th International Symposium, ISVC 2020, San Diego, CA, USA, October 5–7, 2020, Proceedings, Part II 15*, pages 533–544. Springer.

Hillmayr, D., Ziernwald, L., Reinhold, F., Hofer, S. I., and Reiss, K. M. (2020). The potential of digital tools to enhance mathematics and science learning in secondary schools: A context-specific meta-analysis. *Computers & Education*, 153:103897.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Le, A. D. and Nakagawa, M. (2016). A system for recognizing online handwritten mathematical expressions by using improved structural analysis. *International Journal on Document Analysis and Recognition (IJDAR)*, 19:305–319.

Li, B., Yuan, Y., Liang, D., Liu, X., Ji, Z., Bai, J., Liu, W., and Bai, X. (2022). When counting meets hmer: Counting-aware network for handwritten mathematical expression recognition. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 197–214. Springer.

Morais, F. and Jaques, P. A. (2022). Does handwriting impact learning on math tutoring systems? *Informatics in Education*, 21(1):55–90.

Mouchère, H., Viard-Gaudin, C., Zanibbi, R., and Garain, U. (2014). Icfhr 2014 competition on recognition of on-line handwritten mathematical expressions (crohme 2014). In *2014 14th International Conference on Frontiers in Handwriting Recognition*, pages 791–796.

Nkambou, R., Mizoguchi, R., and Bourdeau, J. (2010). *Advances in intelligent tutoring systems*, volume 308. Springer Science & Business Media.

Patel, N., Thakkar, M., Rabadiya, B., Patel, D., Malvi, S., Sharma, A., and Lomas, D. (2022). Equitable access to intelligent tutoring systems through paper-digital integration. In *Intelligent Tutoring Systems: 18th International Conference, ITS 2022, Bucharest, Romania, June 29–July 1, 2022, Proceedings*, pages 255–263. Springer.

Rosa, D., Cordeiro, F. R., Carvalho, R., Souza, E., Chevtchenko, S., Rodrigues, L., Marinho, M., Vieira, T., and Macario, V. (2023). Recognizing handwritten mathematical expressions of vertical addition and subtraction. *arXiv preprint arXiv:2308.05820*.

Soofi, A. A. and Ahmed, M. U. (2019). A systematic review of domains, techniques, delivery modes and validation methods for intelligent tutoring systems. *International Journal of Advanced Computer Science and Applications*, 10(3).

Steenbergen-Hu, S. and Cooper, H. (2013). A meta-analysis of the effectiveness of intelligent tutoring systems on k–12 students' mathematical learning. *Journal of educational psychology*, 105(4):970.

Vu, T.-M., Phan, K.-M., Ung, H.-Q., Nguyen, C.-T., and Nakagawa, M. (2021). Clustering of handwritten mathematical expressions for computer-assisted marking. *IEICE TRANSACTIONS on Information and Systems*, 104(2):275–284.

Wang, G., Bowditch, N., Zeleznik, R., Kwon, M., and LaViola, J. J. (2016). A tablet-based math tutor for beginning algebra. *Revolutionizing Education with Digital Ink: The Impact of Pen and Touch Technology on Education*, pages 91–102.

Yuan, Y., Liu, X., Dikubab, W., Liu, H., Ji, Z., Wu, Z., and Bai, X. (2022a). Syntax-aware network for handwritten mathematical expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4553–4562.

Yuan, Y., Liu, X., Dikubab, W., Liu, H., Ji, Z., Wu, Z., and Bai, X. (2022b). Syntax-aware network for handwritten mathematical expression recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4553–4562.

Zhao, W. and Gao, L. (2022). Comer: Modeling coverage for transformer-based handwritten mathematical expression recognition. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Computer Vision – ECCV 2022*, pages 392–408, Cham. Springer Nature Switzerland.

Zhao, W., Gao, L., Yan, Z., Peng, S., Du, L., and Zhang, Z. (2021). Handwritten mathematical expression recognition with bidirectionally trained transformer. In Lladós, J., Lopresti, D., and Uchida, S., editors, *Document Analysis and Recognition – ICDAR 2021*, pages 570–584, Cham. Springer International Publishing.