Classificação de Caracteres Manuscritos para Correção Automática do Sistema Multiprova

Darlan de Castro Silva Filho[1], **Rex Antonio da Costa Medeiros**[2], **Helton Maia** [2]

[1]Departamento de Engenharia de Computação e Automação
UFRN, Natal-RN, Brazil

[2]Escola de Ciências e Tecnologia
UFRN, Natal-RN, Brazil

nalrad.filho@gmail.com, {rex.medeiros,helton.maia}@ufrn.br

Abstract. *Multiprova is a system developed by professors from the School of Science and Technology of the Federal University of Rio Grande do Norte (ECT/UFRN), which allows the creation of questions and tests. The problem to be solved consists of developing an application to improve the Multiprova, bringing the ability to recognize handwritten characters from each student's response card and, thus, automatically, with the help of a camera, carry out the correction of the test. For this, computer vision techniques and convolutional neural networks were used. A series of configurations for different architectures were tested until reaching high accuracy and reliability in character classification.*

Resumo. *O Multiprova é um sistema desenvolvido por professores da Escola de Ciências e Tecnologia da Universidade Federal do Rio Grande do Norte (ECT/UFRN), que permite a criação de questões e avaliações. O problema a ser solucionado consiste no desenvolvimento de uma aplicação para auxiliar o Multiprova, trazendo a capacidade de reconhecer os caracteres manuscritos provenientes do cartão resposta de cada aluno, e assim, automaticamente, apenas com ajuda de uma câmera, realizar a correção da prova. Para isso, foram utilizadas, técnicas de visão computacional e redes neurais convolucionais. Uma série de configurações para diferentes arquiteturas foram testadas, até que alcançasse uma alta acurácia e confiabilidade na classificação dos caracteres.*

1. Introdução

De acordo com John McCarthy a Inteligência Artificial (IA), pode ser definida como a ciência e engenharia de fazer máquinas inteligentes, especialmente programas de computador. Esta inteligência está relacionada a uma tarefa que é semelhante ao uso de computadores para entender a capacidade humana, no entanto, sem a necessidade de se limitar aos métodos biologicamente observáveis [McCarthy 2007]. A IA é subdividida em diversas áreas, entre elas, pode-se citar o Aprendizado de Máquina (*Machine Learning*), em que as máquinas utilizam algoritmos para tomada de decisões e interpretação de dados, executando tarefas automaticamente, ou seja, as máquinas não serão programadas para realizar ações específicas [Sebe et al. 2005]. O Aprendizado de Máquina possui uma abordagem para solução de problemas diferente da programação tradicional. Nos sistemas comuns, são escritas as regras e os fluxos no qual o software deverá seguir. Já para o Aprendizado

de Máquina, espera-se que o sistema utilize os dados para criar as suas próprias regras de funcionamento, invertendo o processo de construção usual.

A técnica conhecida como Aprendizagem Profunda (*Deep Learning*) que é uma subdivisão do *Machine Learning* tem sido amplamente utilizada, principalmente devido ao avanço da computação paralela e facilidade de utilização de GPUs (*Graphics Processing Units*) para o processamento dos dados de uma rede neural artificial. O *Deep Learning* tem como função, adquirir abstrações através do processo de aprendizagem, permitindo generalizar padrões e fazer previsões confiáveis [Li et al. 2019]. Dentro do paradigma da Aprendizagem Supervisionada Profunda, a utilização de redes neurais convolucionais (CNN's) para o reconhecimento de caracteres manuscritos é um tópico que tem chamado a atenção dos pesquisadores. Este conceito de CNN foi apresentado por [LeCun et al. 1998] no artigo *Gradient-Based Learning Applied to Document Recognition* em 1998.

O reconhecimento de caracteres manuscritos é um problema conhecido da visão computacional e nas redes neurais convolucionais. Porém, desenvolver uma aplicação para dispositivos móveis, inclui elementos que tornam o problema a ser resolvido neste trabalho ainda mais desafiador. Inicialmente, a imagem é obtida por meio de câmera de celular em um dado ângulo em relação ao cartão resposta, em seguida é realizada uma transformação matemática para o ajuste de perspectiva dessa imagem. Este processo inclui distorções na imagem final, como linhas que eram originalmente retas, se tornando curvas na imagem ajustada, impactando na etapa de segmentação da imagem para seleção dos caracteres. Em razão dessa especificidade do *dataset*, não existem muitos trabalhos expressivos para referência, já que se trata de uma ramificação do problema de reconhecimento de caracteres (reconhecer caracteres manuscritos em um cartão-resposta). No entanto, já existiam métodos para a utilização de IA no reconhecimento óptico de caracteres (OCR - *Optical Character Recognition*), ainda que não se utilizassem redes neurais convolucionais (CNN's).

Trabalhos relacionados na área de reconhecimento de padrões utilizando *machine learning* estão em pleno desenvolvimento [De Menezes et al. 2019]. Por exemplo, o trabalho intitulado de *Uso de aprendizado de máquinas para reconhecimento de padrões* [Tácora Amasifuen et al.], sobre Máquinas de Vetor de Suporte (SVM) tanto para a classificação binária quanto para a classificações múltiplas. Já o *MathReader: API for Handwritten Mathematical Expressions Recognition* [dos Reis and Lorenzi 2020], apresenta o MathReader, um software para identificação e leitura de expressões matemáticas manuscritas através do uso de CNN's. Outro trabalho importante para o desenvolvimento de *Machine Learning* nesta área, apresenta o *dataset* EMNIST [Cohen et al. 2017]. Esse conjunto de imagens deriva do *NIST Special Database 19*, que é um super conjunto de imagens de caracteres manuscritos. Por fim o [Altwaijry and Al-Turaiki 2021] utiliza redes neurais convolucionais para fazer o reconhecimento de escrita em Árabe, mostrando a robustez da técnica.

A motivação para a realização deste trabalho, foi a iminente volta às aulas presenciais na Universidade Federal do Rio Grande do Norte (UFRN), depois do período da pandemia do COVID-19. Além da necessidade do software Multiprova, utilizado para realizar atividades educacionais com os alunos, precisar se adaptar às provas em papel. O Multiprova (*https://site.multiprova.ufrn.br*) nasceu como um sistema para avaliação

dos estudantes, ganhou um módulo de resolução de provas online totalmente remoto, sendo essencial para a comunidade acadêmica da UFRN durante os dois anos de ensino à distância, necessários para a amenização dos efeitos da pandemia. Com a volta às salas de aula, o Multiprova precisou se ajustar ao ensino presencial e com isso surgiu um problema: Como manter o sistema benéfico à comunidade acadêmica, sabendo que presencialmente este método para avaliação de provas escritas com correção manual do professor já funciona adequadamente. Para solução desse problema, foi pensado e construído pela equipe do Multiprova, um cartão resposta para provas em papel que pode ser observado na Figura 1. Este cartão resposta vai além dos cartões respostas comuns - que só permitem a correção automática de questões de múltipla escolha. Este permite a utilização de questões do tipo múltipla escolha, associação de colunas e verdadeiro ou falso.

A utilização de caracteres manuscritos no cartão resposta simplifica seu desenvolvimento, pois utilizando uma mesma estrutura (coluna de caixas em branco para escrita dos caracteres) é possível realizar provas com questões de múltipla escolha, verdadeiro ou falso e associação de colunas, mantendo a possibilidade de novos tipos de questões serem adicionados no futuro dada a natureza genérica do cartão resposta. A caligrafia das letras dos alunos pode gerar problemas no reconhecimento, no entanto o sistema será treinado para reconhecer números e letras maiúsculas. A escolha por letras somente maiúsculas se deu pela maior padronização de escrita com esse formato de letra, com poucas variações na escrita dos caracteres, independente da caligrafia do aluno.

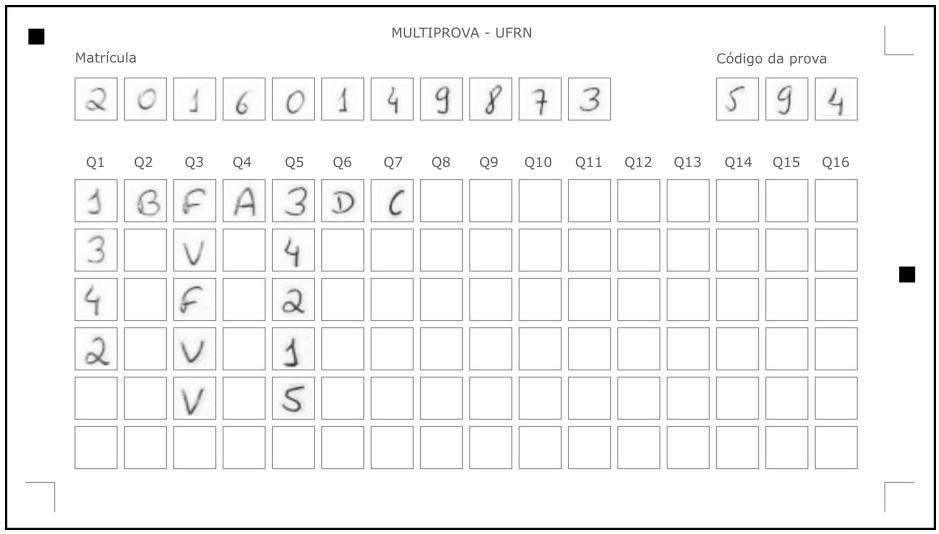


Figura 1. Cartão resposta desenvolvido pela equipe do Multiprova.
[MOREIRA et al. 2020]

Desta forma, o problema a ser resolvido se resume ao desenvolvimento de uma ferramenta computacional, que tem como objetivo automatizar o processo de correção dos cartões respostas provenientes do sistema Multiprova. Neste tipo de cartão resposta, os alunos escrevem manualmente suas respostas. O sistema desenvolvido neste trabalho difere dos demais já existentes, por ser um instrumento de correção de provas através do reconhecimento de caracteres e não um identificador de marcações em gabaritos. Essa diferença expande as possibilidades de criação das questões que podem ser aceitas pelo software e simplifica o padrão dos cartões. Além disso, a solução desenvolvida é totalmente integrada com a plataforma Multiprova, o que permite maior facilidade no desen-

volvimento de novas funcionalidades. Este é um sistema gratuito para instituições de ensino públicas brasileiras mediante contato com a equipe Multiprova.

2. Metodologia

O trabalho desenvolvido realizou testes empíricos para diferentes configurações de redes neurais artificiais. Foi analisado o desempenho destas redes com auxílio de métricas como a acurácia e precisão. Os testes envolveram modificações fundamentais nas configurações das redes, tais como: quantidades de camadas, número de neurônios, quantidade e tamanho dos filtros convolucionais, tipos de funções de ativação, *optimizers* e funções de perda. Isto permitiu selecionar a melhor arquitetura de rede dentre as configurações analisadas, antes do software entrar em produção.

2.1. Linguagens de Programação e Softwares Auxiliares

Neste trabalho foram utilizadas duas linguagens de programação. O Python [Van Rossum and Drake Jr 1995] para o processamento das imagens dos cartões resposta e na criação e treinamento da rede neural. Já a segunda, a linguagem Javascript [Flanagan 2006], na realização do porte das configurações da rede treinada para o ambiente *mobile*, com o objetivo de predizer as respostas dos alunos nos cartões.

Além disso, criado e mantido pela Google, o TensorFlow [Abadi et al. 2015] que é uma biblioteca de código aberto para computação numérica e *Machine Learning*. Essa biblioteca agrupa vários algoritmos e modelos de *Deep Learning*, simplificando seu uso para o usuário final. Devido a sua facilidade de uso e integração com a linguagem Python, o TensorFlow é a ferramenta para o desenvolvimento de *Machine Learning* mais utilizada na atualidade.

Por fim, também foi utilizado o OpenCV, ou *Open Source Computer Vision* [Bradski 2000], uma biblioteca de código aberto que foi inicialmente desenvolvida pela Intel no ano de 2000, visando facilitar o desenvolvimento de projetos com o foco em visão computacional. Está implementada em diversas linguagens de programação e é primariamente utilizada para análises e manipulações em imagens e vídeos.

2.2. Sistema Multiprova e Planejamento do Fluxo de Correção

O software Multiprova é uma aplicação constituída de duas partes principais:

- Lado do cliente;
- Lado do servidor.

O lado do cliente - ou *front-end* - (interface em que alunos e professores interagem) é composta por duas frentes. A aplicação feita para navegadores - tanto *desktop* quanto *mobile* - e o aplicativo *mobile* nativo. A aplicação para navegadores é a parte principal onde os discentes e docentes podem executar por completo o fluxo de provas virtuais, desde a criação das provas por parte dos professores até a execução delas pelos alunos. Essa aplicação é feita com React. Já a aplicação nativa para *mobile* é feita com React Native e tem como função ser uma interface entre as correções de provas presenciais e o lado do servidor do sistema Multiprova.

O lado do servidor (ou *back-end*) é feito com JavaScript, utilizando o modelo de API REST e o banco de dados NoSQL MongoDB. O servidor é a parte principal do

sistema, guarda todos os dados de questões e provas desde a sua criação até a sua execução e correção. É importante destacar que os lados do cliente e do servidor se comunicam através do protocolo HTTP.

Para o processo de correção automática é necessário fazer um tratamento inicial na imagem do cartão resposta, obtendo assim um formato padrão. O tratamento tem como objetivo normalizar a perspectiva da imagem, de modo que ela aparente ter sido fotografada paralelamente à câmera. Além disso, deve-se ajustar a sua rotação. Para esses ajustes, podem ser utilizadas as referências de posição indicadas no cartão.

Depois da obtenção da imagem do cartão, agora formatada, são feitos cortes predefinidos em todas as 110 caixas de texto, onde 11 são correspondentes à matrícula do aluno, 3 ao código da prova e 96 são distribuídas igualmente entre as 16 questões (6 para cada questão). Tendo em vista que o problema apresentado tem como base fundamental o reconhecimento de caracteres escritos a mão, a utilização de Inteligência Artificial é a escolha mais adequada no desenvolvimento de uma solução para um problema desta complexidade. Para este caso, foi necessário o desenvolvimento de CNN's que consigam identificar com boa taxa de precisão, letras e números. O paradigma da aprendizagem supervisionada se adapta melhor ao problema apresentado.

2.3. Obtenção e Organização dos Dados

Para a criação do *dataset* foram solicitados que aproximadamente 400 alunos da Escola de Ciências e Tecnologia (ECT), da UFRN, completassem um cartão com exemplos dos caracteres requeridos. Para todos o *dataset* (letras, dígitos e V ou F), as quantidades de amostras por classe estavam inicialmente desbalanceadas.

Desta forma, para realizar a distribuição adequada na quantidade de amostras por classe, é recomendável aplicar técnicas de balanceamento de dados, e assim, melhorar a qualidade do *dataset*. Para isso, foram aplicadas técnicas de *oversampling* para aumentar a diversidade e a qualidade das amostras utilizadas no treinamento, tais como: *zoom in*, *zoom out*, espelhamento vertical e/ou horizontal, deslocamento e rotação da imagem. A quantidade de amostras por classe obtidas através dos alunos da ECT se mantiveram em torno de 1700 amostras por classe. Ao final do processo de aumento de dados, as quantidades de amostras utilizadas para o treinamento das redes neurais variaram entre 5700 e 7000 amostras por classe.

2.4. Arquitetura e Treinamento das CNN's

A estrutura base das CNN's foi a seguinte:

- Camada convolucional com 32 filtros (64 para as camadas além da primeira) e *kernel* (3 x 3) com função de ativação ReLU. Dados de entrada no formato (32 x 32 x 3) (imagem 32 x 32 *pixels* com 3 canais de cor);
- Camada de *pooling* máximo com filtro (2 x 2);
- Camada *flatten*;

O *optimizer* utilizado foi o Adam (ou *Adaptive Moment Estimation Algorithm*), proposto em 2014 por Diederik P. Kingma e Jimmy Ba. De acordo com seus criadores, esse *optimizer* é computacionalmente eficiente, requer pouca memória, é invariante ao redimensionamento diagonal dos gradientes e é adequado para problemas que são grandes em dados e/ou parâmetros [Kingma and Ba 2014].

O *dataset* construído foi subdividido utilizando as seguintes quantidades, 70% das amostras para treinamento, 15% das amostras para validação, e 15% das amostras para os testes. Todas as imagens obtidas foram formatadas para o tamanho de entrada da CNN com dimensão (32 x 32 x 3) , não sedo necessário um tratamento prévio nesse quesito. O treinamento ocorre até que a CNN não apresente uma melhora na precisão ou na perda dentro de uma margem de 0,1% (técnica conhecida como *early stopping*).

3. Resultados

De modo geral, os treinamentos das CNN's atingiram os resultados esperados, alcançando valores de precisão acima dos 98%, e perda, abaixo dos 6% na sua melhor configuração. Para o reconhecimento das letras e dígitos, estruturas com 4 camadas convolucionais, seguidas de *max pooling* se mostraram mais eficientes. Para o caso do reconhecimento de caracteres que podem ser "V"e "F", uma rede especializada e estruturada com 3 camadas convolucionais, seguidas de *max pooling*, apresentaram os melhores resultados.

A estrutura base das CNN's (estrutura 1 ou primeira estrutura) é composta por uma camada convolucional, seguida por uma camada de *pooling* máximo, uma camada *flatten* e uma camada totalmente conectada. A medida que as estrutura vão avançando, as configurações das redes também incrementam. Para cada estrutura além da primeira, são adicionadas, nessa ordem, uma camada convolucional e uma camada de *max pooling* entre a última camada de *pooling* máximo e a camada *flatten*. Essa adição, melhora a habilidade da rede em identificar e priorizar as informações mais importantes presentes na imagem.

Os treinamentos das redes especializadas no reconhecimento de letras e dígitos, foram executados em uma média de 150 segundos (24 épocas), enquanto que para as redes especializadas em V ou F, em média, duraram apenas 12 segundos (7 épocas).

O desenvolvimento computacional das CNN's foi realizado utilizando Python, TensorFlow e toda a sua extensa API para a facilitação do processo na plataforma do Google Colaboratory.

As especificações da máquina virtual provida pelo Google são:

- CPU: Intel(R) Xeon(R) @ 2.20GHz, 1 *core*;
- Memória RAM: 12,68 GB;
- Disco: 78,19 GB;
- GPU: Nvidia Tesla K80.

3.1. Reconhecimento de Letras

A CNN especializada no reconhecimento das letras, possui o objetivo de reconhecer os caracteres de A a J, além da classe Branco - que não possui caracteres.

A estrutura 1 apresentou resultados satisfatórios a partir da configuração mais simples. A classe Branco apresentou quase 100% de precisão, tendo a pior classe com 90% de acerto (classe E) e as melhores (exceção da classe Branco) com 95% de acerto (classes F, G e H). A estrutura apresentou precisão média de 93,26%. As taxas de confusão entre uma classe e outra, no entanto, estão presentes, sendo o pior deles de 3,7% entre as classes H e A e entre as classes E e C.

A segunda estrutura traz melhores resultados, se comparados com a primeira estrutura. Com uma taxa de acerto de 97,52% - 4,16% superior à estrutura 1 - a pior taxa de confusão foi de 2,7% entre as classes G e B, a pior classe com 94% de precisão (classe G) e as melhores - com exceção da classe Branco - chegando à 99% de acerto (classes I e J).

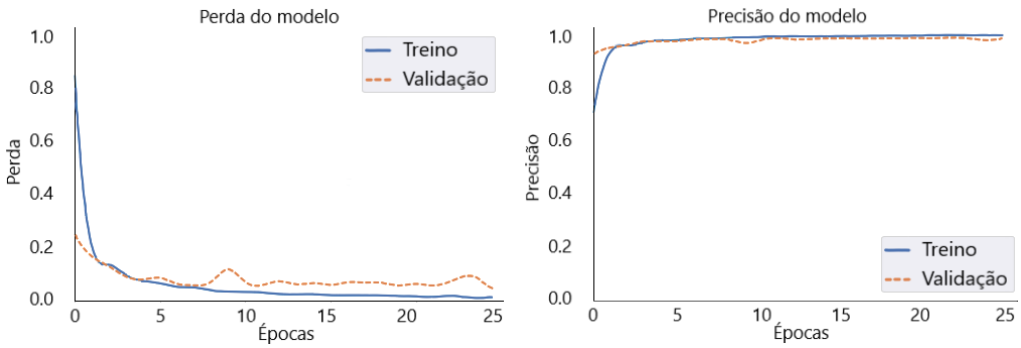
A estrutura 3 apresentou erros relativamente menores, porém com a pior taxa de confusão de 3,2% entre as classes A e B, maior do que o erro apresentado na estrutura 2 (2,7% entre as classes G e B). A classe com mais confusão, também se manteve em 94% - agora na classe A - e as melhores classes, com exceção da classe Branco, são as classes B, F, H e J. A estrutura obteve uma média de acerto geral de 98,1%.

Por fim, a estrutura 4 apresentou uma taxa de precisão média de 98,38%, a taxa de confusão máxima foi de 2,3%, sendo perceptível a boa taxa de acerto das classes na diagonal principal - que não foi menor que 96% na classe D. A quarta estrutura apresentou o melhor resultado entre as 4 configurações analisadas. A Tabela 1 expõe o progresso dos resultados referentes a precisão e perda, obtidos através das quatro configurações testadas.

Estrutura	Precisão	Perda
Estrutura 1	93,36%	0,245
Estrutura 2	97,52%	0,116
Estrutura 3	98,10%	0,068
Estrutura 4	98,38%	0,067

Tabela 1. Evolução da CNN de letras através das estruturas

Os gráficos da Figura 2, detalham a evolução do treinamento e da validação com o passar das épocas. Com uma boa convergência para os valores ideais - 0 para perda e 1 para precisão.



(a) Gráfico de evolução do fator de perda durante o treinamento.

(b) Gráfico de evolução do fator de precisão durante o treinamento.

Figura 2. Gráficos de perda e precisão (letras).

A Figura 3 apresenta a matriz de confusão da melhor estrutura analisada (estrutura 4).

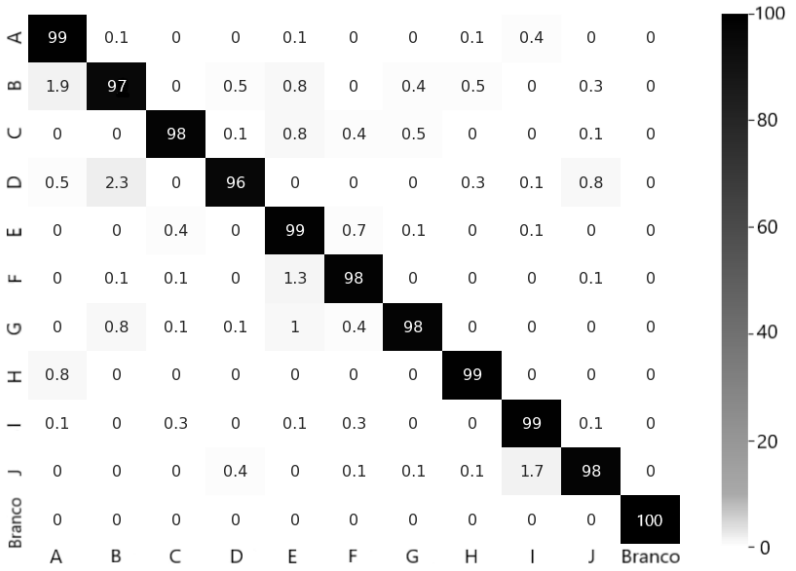


Figura 3. Matriz de confusão (letras).

Os pontos de destaque presentes na matriz são poucos erros acima de 1% (somente 4 ocorrências) e as classes A, E, H, I e Branco que atingiram 99% de precisão ou mais.

3.2. Reconhecimento de Dígitos

A CNN de dígitos têm o desafio de reconhecer os caracteres de 0 a 9 e a classe Branco.

O resultados da primeira estrutura mostram que com apenas uma camada convolucional e de *max pooling* a CNN tem dificuldade em reconhecer os detalhes de cada classe, com destaque para as classes 2, 3 e 8 que não atingiram os 90% de precisão e para a classe 9 que atingiu somente 72% de precisão com uma porcentagem de erro de 5,8% - um valor alto - sendo confundida com a classe 7. Um ponto de destaque foi a classe Branco que já atingiu os 100% de precisão. Para fim de comparação com as outras estruturas a porcentagem de acerto geral foi de 89,41% com a pior taxa de acerto por classe de 72% na classe 9.

Já na estrutura 2, os resultados apontam para uma melhora na capacidade de identificar as diferentes classes. No entanto, os erros são menos visíveis, com o pior deles de 4,4% na classe 8 sendo confundida com a classe 9. A única classe abaixo dos 95% de precisão foi a classe 7, que atingiu somente 90% de acerto. A taxa de acerto geral foi de 95,87%, e a pior classe chegou a 90% de acurácia - classe 8.

Para a terceira estrutura é perceptível mais uma melhora - ainda que não tenha sido tão grande quanto da estrutura 1 para a estrutura 2. A configuração obteve uma média geral de acerto de 97,8% e a pior classe com 92% de precisão (classe 7), com destaque para as classes 0, 2 e 4 que atingiram 99% de acerto.

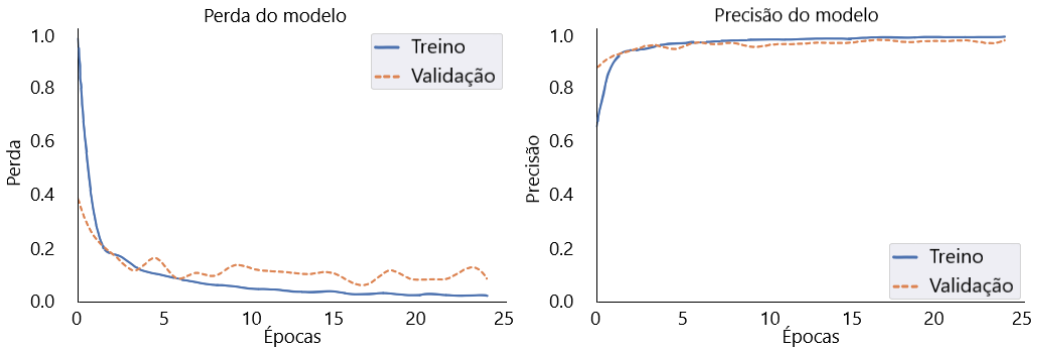
Por fim, a estrutura 4 indica mais uma melhora, pequena em relação à estrutura 3. Com uma média geral de acerto de 98,84% e a pior classe com 98% de acerto. A quarta estrutura apresentou o melhor resultado entre as 4 configurações analisadas.

A Tabela 2 apresenta a evolução dos resultados de precisão e perda ao longo dos testes para as quatro estruturas analisadas.

Estrutura	Precisão	Perda
Estrutura 1	89,41%	0,351
Estrutura 2	95,87%	0,115
Estrutura 3	97,80%	0,098
Estrutura 4	98,84%	0,064

Tabela 2. Evolução da CNN de dígitos através das estruturas

A Figura 4 apresenta as métricas de precisão e perda ao longo do treinamento da CNN. Mais uma vez as curvas apontam para um treinamento bem sucedido, com uma convergência bastante rápida no caso da precisão e, embora um pouco mais lenta, no caso da perda têm-se bons valores finais para as curvas de treino e validação.



(a) Gráfico de evolução do fator de perda durante o treinamento.

(b) Gráfico de evolução do fator de precisão durante o treinamento.

Figura 4. Gráficos de perda e precisão (dígitos).

A Figura 5 apresenta a matriz de confusão da estrutura 4 analisada com os melhores resultados.

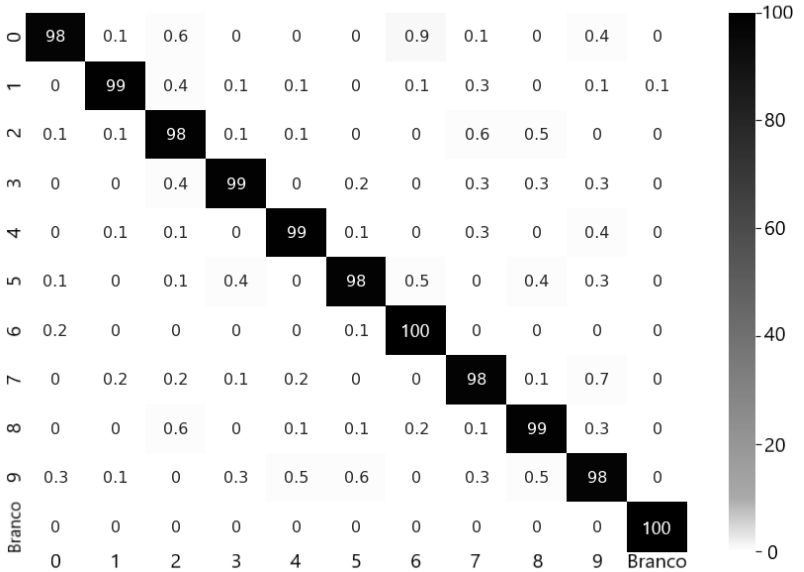


Figura 5. Matriz de confusão (dígitos).

Ainda na Figura 5, é possível verificar um erro máximo de 0,9% para classificação.

A matriz apresenta poucos erros acima de 0,5% e reforça a ótima taxa de acerto das classes na diagonal principal (que não foi menor que 98%), com destaque para a classe 6, que atingiram 100% de acertos.

3.3. Reconhecimento dos caracteres: V e F

Neste caso, o modelo teve como objetivo classificar apenas para 3 classes. Para a estrutura 1 - que é a mais básica, foi alcançada uma precisão média de 99,02%, com a pior percentual de acerto por classe chegando a 98% de precisão - classe V - com o pior erro de 1,8% (entre as classes V e F). Comparando com as estruturas analisadas anteriormente, a mesma configuração de rede da estrutura 1, apresentou melhores e mais refinados resultados, principalmente em razão da reduzida quantidade de classes.

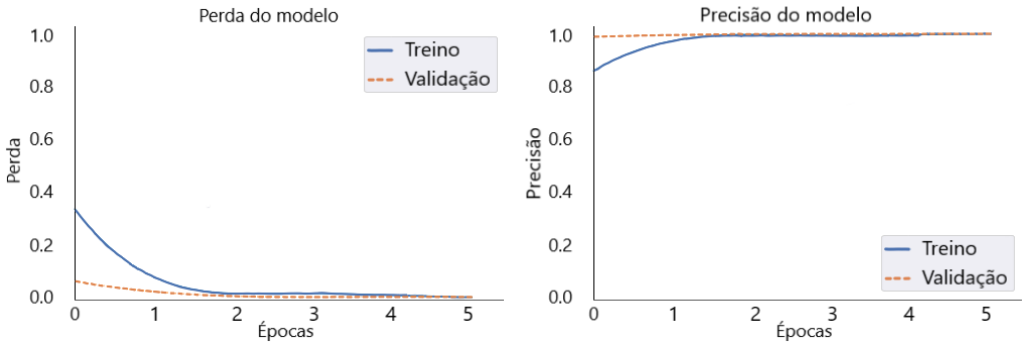
Já a estrutura 2 chegou aos 99,68% de precisão média geral - uma melhora de 0,66% comparado com a estrutura 1. A pior classe foi a V, porém com uma alta precisão de 99%, e o pior erro em 0,7% entre as classes V e F. Destaque para a classe F, que junto da classe Branco, atingiram 100% de acerto. Por fim, a terceira estrutura apresentou resultados ainda melhores que a estrutura 2, chegando à taxa média de precisão de 99,89%. Somente uma célula de erro é vista na matriz de confusão - erro de 0,6% entre as classes V e F.

Ao contrário das classificações de dígitos e letras, foram analisadas somente 3 estruturas de CNN para a classificação de verdadeiro ou falso. Isso se deve ao fato de que a terceira estrutura, já apresentou taxa de 99,89% de precisão. A Tabela 3 sintetiza o processo de otimização da CNN de verdadeiro ou falso com a evolução das métricas de precisão e perda.

Estrutura	Precisão	Perda
Estrutura 1	99,02%	0,039
Estrutura 2	99,68%	0,015
Estrutura 3	99,89%	0,008

Tabela 3. Evolução da CNN de V ou F através das estruturas

A Figura 6 exibe a evolução das métricas de perda e precisão ao longo do treinamento.



(a) Gráfico de evolução do fator de perda durante o treinamento.

(b) Gráfico de evolução do fator de precisão durante o treinamento.

Figura 6. Gráficos de perda e precisão (V ou F).

Os gráficos da Figura 6 apresentam uma boa convergência das curvas, tanto para o conjunto de treino, quanto para o conjunto de validação. Uma pequena quantidade de épocas, apenas cinco, foi necessária para atingir bons resultados.

A Figura 7 apresenta a matriz de confusão da melhor estrutura analisada para a CNN de V ou F. A matriz apresentada exibe uma alta acurácia, com uma pequena taxa de 0,6% de confusão entre as classes V e F. A pior classe é a V, que possui 99% de precisão, enquanto as classes F e Branco alcançaram 100% de acertos.

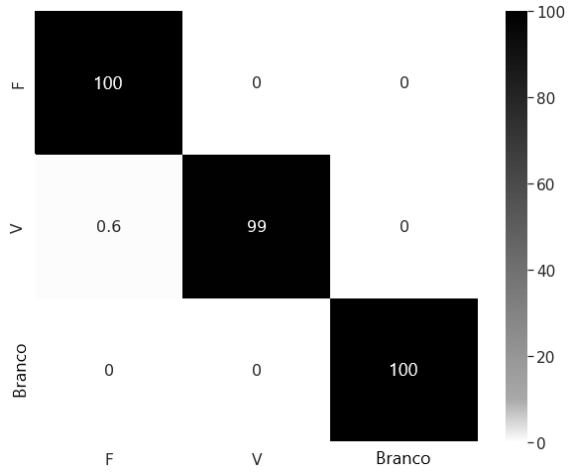


Figura 7. Matriz de confusão (V ou F).

4. Conclusões

O Multiprova ganhou notoriedade em razão dos problemas acarretados pela pandemia do Coronavírus, principalmente, devido à necessidade de se realizar avaliações remotamente. Considerando à volta das aulas presenciais, este trabalho visou o desenvolvimento de uma nova funcionalidade para o Multiprova, permitindo que os professores mantivessem a possibilidade de correção rápida das provas, ainda que fossem realizadas em papel.

Com os resultados obtidos: 98,84% de precisão para CNN de dígitos, 98,38% de precisão para CNN de letras, e 99,89% de precisão para CNN de verdadeiro ou falso, estes valores apontam uma ótima taxa média de acerto, ainda que melhorias sejam possíveis. De forma prática, utilizando estes resultados, foi possível exportar os arquivos de pesos das CNN's já treinadas e inserir no aplicativo do Multiprova (Multiprova Corretor). O aplicativo foi desenvolvido utilizando a tecnologia *React Native* e possui suporte para o *framework* de *machine learning* do Google, conhecido como Tensorflow e suas respectivas bibliotecas.

O aplicativo Multiprova Corretor, está disponível na loja virtual *Play Store*. Isto permite que os professores façam *download* e utilizem a aplicação nos seus próprios aparelhos e não dependam de meios de terceiros (máquina de correção de cartões como era feito na versão anterior do Multiprova) para efetuar a correção automática de suas provas. A aplicação está disponibilizada através do link (`https://play.google.com/store/apps/details?id=br.ufrn.multiprova`).

Referências

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Altwaijry, N. and Al-Turaiki, I. (2021). Arabic handwriting recognition system using convolutional neural network. *Neural Computing and Applications*, 33(7):2249–2261.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Cohen, G., Afshar, S., Tapson, J., and van Schaik, A. (2017). Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*.

De Menezes, R. S. T., Magalhaes, R. M., and Maia, H. (2019). Object recognition using convolutional neural networks. In *Recent Trends in Artificial Neural Networks-from Training to Prediction*. IntechOpen.

dos Reis, C. S. and Lorenzi, F. (2020). Mathreader: Api for handwritten mathematical expressions recognition. In *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1282–1289. IEEE.

Flanagan, D. (2006). *JavaScript: the definitive guide*. "O'Reilly Media, Inc.".

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Li, S., Song, W., Fang, L., Chen, Y., Ghamisi, P., and Benediktsson, J. A. (2019). Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):6690–6709.

McCarthy, J. (2007). What is artificial intelligence?

MOREIRA, A. B., MEDEIROS, R. A., LEITÃO, G. B., and SILVA, D. R. (2020). Multiprova: Software educacional.

Sebe, N., Cohen, I., Garg, A., and Huang, T. S. (2005). *Machine learning in computer vision*, volume 29. Springer Science & Business Media.

Tácora Amasifuen, F. S. et al. Uso de aprendizado de máquinas para reconhecimento de padrões.

Van Rossum, G. and Drake Jr, F. L. (1995). *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam.