Reconhecimento de expressões matemáticas escritas a mão por meio de redes neurais convolucionais.

Leonardo Loureiro Costa
Instituto de Ciência e Tecnologia
Universidade Federal de São Paulo
São José dos Campos, Brasil
leonardo.costa@unifesp.br

***Resumo*—No ramo de tecnologias assistivas, uma sub-área de visão computacional, algoritmos de "Optical Character Recognition"(OCR) são amplamente usados para reconhecer e extrair informações textuais presentes em imagens. Neste artigo exploraremos como isso pode ser utilizado para Segmentar e classificar os caracteres de uma expressão matemática, possibilitando a sua resolução. Mostraremos que é possível obter um grau médio de semelhança entre os rótulos verdadeiros e os previstos superior a** 84.43%**.**

I. Introdução

A visão computacional é uma sub-área da computação gráfica que almeja estudar a ciência e a tecnologia por trás de como máquinas enxergam. [4]

Classificar imagens é uma tarefa desse campo de estudo desde a década de 1970, consiste em as processar digitalmente de maneira a segmentá-las e rotulá-las.

Esse processo é utilizado atualmente em diversas áreas, como no diagnóstico por imagens, na navegação de veículos autônomos, no controle e inspeção de processos industriais, e em tecnologias assistivas.

No escopo de tecnologias assistivas, a visão computacional desempenha um papel importantíssimo no auxílio a pessoas com deficiências visuais. Sua implementação correta permite ao usuário perceber o mundo a sua volta com melhor claridade, utilizando tecnologias que efetivamente transformam imagem em descrição.

Este artigo almeja apresentar uma solução de visão computacional para auxiliar no reconhecimento de expressões matemáticas — textos escritos a mão — e um algoritmo que, solucionando-as, retorna o valor desejado.

Com isso, duas tarefas: o reconhecimento dos caracteres desenhados e a solução matemática da expressão obtida.

II. Fundamentação teórica

A. Morfologia —Dilatação

A dilatação no processamento de imagens é uma técnica utilizada para aumentar o tamanho dos objetos de uma imagem. [11] Isto é feito adicionando píxeis ao limite dos objetos na imagem, resultando em uma versão maior da imagem original. A dilatação é frequentemente usada em aplicações de processamento de imagem, tais como detecção e segmentação de objetos, onde ajuda a tornar os objetos mais visíveis e fáceis de identificar.

B. Filtros passa-baixa

Um filtro passa-baixa de mediana é um tipo de filtro digital usado para remover ruídos de um sinal. [8] Ele funciona substituindo o valor de cada píxel de uma imagem pelo valor mediano de um conjunto de píxeis vizinhos. Isto tem o efeito de suavizar a imagem, ajudando a reduzir a aparência de ruídos do tipo impulsivo.

"passa-baixa"vem do fato de que este tipo de filtro só permite a passagem de sinais de baixa frequência, enquanto os sinais de alta frequência são atenuados.

C. Segmentação

Segmentação de imagem é o processo de divisão de uma imagem em múltiplos segmentos ou regiões, cada um dos quais corresponde a um objeto ou fundo diferente na imagem.

O objetivo da segmentação de imagem é facilitar a análise e a compreensão do conteúdo de uma imagem, dividindo-a em regiões distintas. Por exemplo, em uma aplicação de imagens médicas, a segmentação de imagem pode ser usada para identificar diferentes órgãos ou tecidos em uma ressonância magnética. Em geral, os algoritmos de segmentação de imagem funcionam identificando limites ou contornos em uma imagem que correspondem aos objetos ou regiões de interesse.

Estes limites são então usados para dividir a imagem em segmentos, que podem ser analisados separadamente.

D. Redes neurais e camadas convolucionais

Redes neurais convolucionais (CNNs) são um tipo de rede neural especialmente projetada para trabalhar com dados que possuem uma estrutura em grade, como uma imagem. Elas são compostas por múltiplas camadas de nós interconectados, assim como outras redes neurais. No entanto, as camadas em uma CNN são projetadas de uma maneira específica que as permite aprender automaticamente hierarquias espaciais de características dos dados.

Uma das principais características das CNNs é que elas usam camadas de convolução, que aplicam um filtro nos dados de entrada para extrair características locais dele. Esses filtros deslizam pelos dados de entrada, extraindo características de diferentes partes da entrada. Ao usar múltiplos filtros com características diferentes, uma CNN pode aprender a reconhecer uma ampla gama de características nos dados de entrada.

No geral, as CNNs são adequadas para tarefas que envolvem análise de dados visuais, como classificação de imagens, detecção de objetos e segmentação. Elas conseguem aprender automaticamente hierarquias espaciais de características dos dados e conseguem lidar com abundantes dados eficientemente. Isso as torna uma escolha popular para muitas aplicações em visão computacional.

E. Optical Character Recognition

Optical Character Recognition (OCR) é um conjunto de técnicas de visão computacional que visam reconhecer e extrair informação de texto presente em imagens ou vídeos. O OCR é amplamente utilizado para digitalizar documentos, como livros, jornais, revistas e formulários, e torná-los disponíveis em formato digital. Ele também é utilizado em sistemas de leitura de placas de veículos, em aplicativos de tradução de imagens e em muitas outras aplicações.

III. Trabalhos relacionados

Photomath [1] — Com a ajuda da câmera de um smartphone, os usuários podem utilizar o Photomath, um sistema de álgebra computacional móvel com um sistema de reconhecimento óptico aprimorado, para digitalizar e detectar equações matemáticas. O aplicativo então mostra explicações passo a passo na tela.

K. Gupta, [2] utilizou uma rede MLP com camadas convolucionais para classificar dígitos do dataset MNIST, obtendo uma acurácia de 99.15%.

IV. Metodologia

Para obter o resultado de uma expressão matemática desenhada a mão é necessário seccionar a tarefa "Captura-da-imagem $\implies$ Resultado"em duas partes; a segmentação dos objetos em cena, e o processamento da expressão matemática.

A imagem de uma determinada expressão matemática, assim que capturada, necessita passar por etapas de processamento, de modo a transformar essa foto em "frames-– imagens de menor tamanho — que contenham os significantes da expressão matemática.

Uma imagem a ser processada passa pelas etapas listadas na figura 1.

1) Captura da imagem: consiste na obtenção da imagem a ser utilizada.
2) Pré-processamento: nessa etapa serão realizados os procedimentos da tabela I
3) Segmentação:
4) Classificação: será feita utilizando uma rede neural convolucional treinada nos datasets MNIST.
5) Resolução: resolve a expressão utilizando um script em Python

A. Captura da imagem

As imagens utilizadas para teste são obtidas por meio de um algoritmo, que aleatoriamente simula expressões matemáticas escritas a mão [9], por meio da combinação de diferentes caracteres em diferentes fontes com fundos diversos, de modo a produzir uma imagem que se assemelha com a realidade.

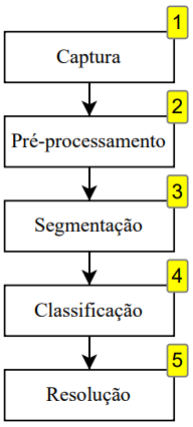

Figura 1: Etapas do algoritmo de resolução de equações.

As imagens produzidas possuem 3 números naturais, de 1 a 99, intercalados com 2 operadores, um sinal de igualdade e o resultado da expressão, que em 20% dos casos é um resultado aleatório, para simular possíveis erros de cálculo.

B. Pré-processamento

Etapa	Descrição
1º	Conversão de RGB para um único canal, tons de cinza
2º	Aplicação de threshold — binarização — com o algortimo de Otsu
3º	Inversão dos valores usando o operador lógico NOT.
4º	Operação morfológica de dilatação.
5º	Filtro passa-baixa de mediana.
6º	Filtro passa-baixa de gaussiano.

Tabela I: Etapas de pré-processamento da imagem

As etapas de pré-processamento visa transformar a imagem original em uma imagem pronta para a segmentação. a tabela I mostra as etapas utilizadas.

C. Segmentação

A segmentação será feita por meio do algorítmo cv2.findContours, uma função do módulo cv2 (openCV) [7] [10] que encontra os contornos de uma imagem binária.

Primeiro, ele recebe como entrada uma imagem binária e uma constante opcional que determina o método de aproximação dos contornos encontrados.

A função cria uma cópia da imagem e, em seguida, aplica o algoritmo de detecção de contornos em cima dela, procurando por bordas na imagem e criando uma linha ao redor delas.

Por fim, a função retorna uma lista de contornos encontrados. Para cada contorno, o algoritmo de segmentação obtém o retângulo que o envolve usando a função cv2.boundingRect, extrai o símbolo do contorno, redimensiona o símbolo para 20x20 pixels e adiciona um "padding"de 4 píxeis, tornando as imagens 28x28 píxeis. As imagens são então salvas em uma lista para a classificação.

D. Classificação

A classificação será feita por meio de uma rede neural convolucional.

A rede será treinada em cima da base de dados MNIST [5] dígitos de 0 a 9 e no dataset "Handwritten Math Symbols" [12].

A topologia da rede — número de camadas e de neurônios por camadas — está presente na figura 2. A primeira camada densa possui função de ativação "ReLu"; a última, "softmax"com 14 neurônios — 10 para os dígitos e 4 para os operadores e ponto decimal.

Após a classificação, a lista de caracteres obtidas da expressão passa por uma verificação sintática, que une, por exemplo, dois sinais de subtração — um ao lado do outro — como um sinal "=", ou uma combinação de dois pontos e um "−"como um "÷".

Essa lista processada segue para a etapa de resolução

E. Resolução

O algoritmo define uma função chamada "evaluate"que pode ser usada para avaliar expressões matemáticas simples consistindo em números e os operadores aritméticos básicos — $+, -, \cdot, \div$. As expressões contem apenas números racionais. A função recebe uma string como entrada e retorna o resultado da avaliação da expressão.

Para avaliar a expressão, a função primeiro divide a string de entrada em uma lista de tokens, os números e operadores individuais na expressão. Em seguida, usa duas pilhas para armazenar os valores e os operadores encontrados na expressão.

A função itera sobre a lista de tokens e processa cada um. Se o token é um operador, ele remove os últimos dois valores da pilha de valores e aplica o operador a eles. Em seguida, empilha o resultado de volta na pilha de valores. Se o token não é um operador, deve ser um número, então a função o empilha na pilha de valores.

Depois que todos os tokens forem processados, a função aplica quaisquer operadores restantes aos valores restantes na pilha. Em seguida, retorna o resultado, o único valor restante na pilha de valores.

O código também define dois dicionários que mapeiam as representações em string dos operadores para as funções correspondentes do módulo "operator"e definem a precedência dos operadores. Estes são usados para ajudar a função a avaliar a expressão corretamente.

F. Avaliação

Assumamos α como um índice de semelhança entre duas strings de texto, onde α pode ser obtido por meio de uma função que avalia qual a maior sequência idêntica de caracteres entre duas strings, e θ como um limiar de α, arbitrariamente escolhido.

Considerando X como o sinal de entrada do algoritmo — a imagem da expressão —, Y como o rótulo correto, e $\hat{Y}$ como o rótulo predito, temos que $\alpha(Y, \hat{Y})$ representa a similaridade, acurácia, entre o valor verdadeiro e o resultado do algoritmo.

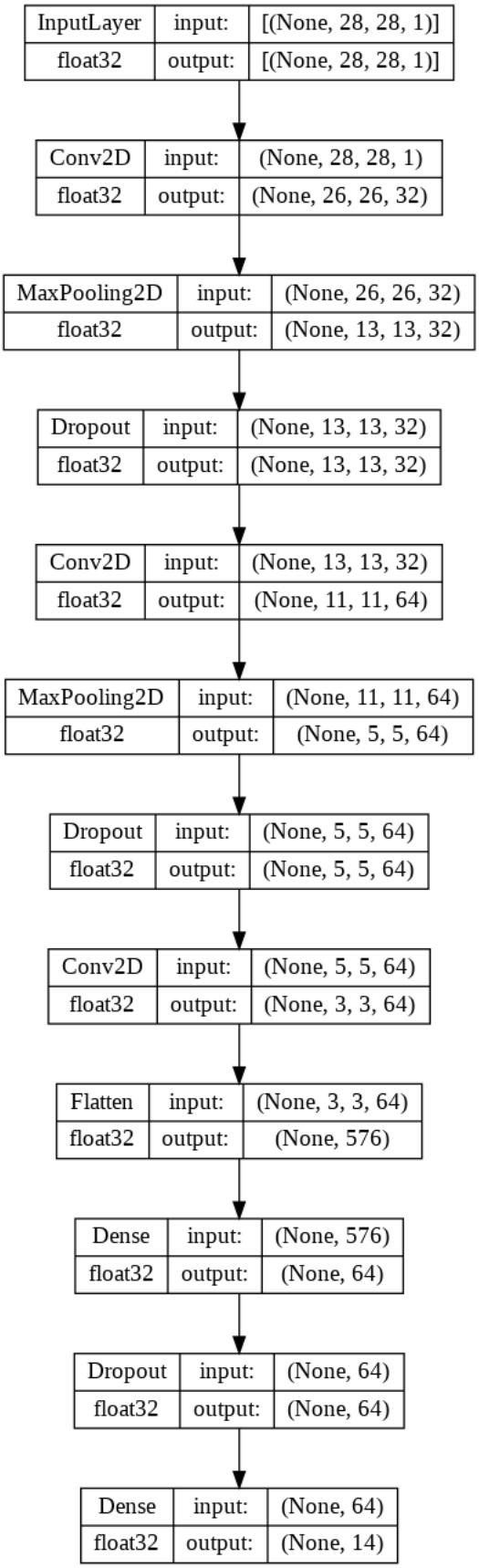

Figura 2: Rede neural convolucional utilizada para a classificação dos caracteres da expressão matemática

Dessa forma temos que $\bar{\alpha}$ representa a média da acurácia para um determinado conjunto de m elementos, como presente na equação 1.

$$\bar{\alpha} = \frac{1}{m} \sum_{i=1}^{m} \alpha(Y^{(i)}, \hat{Y}^{(i)}) \tag{1}$$

Onde:

m : número de imagens de teste
α : semelhança entre duas strings
Y : rótulo verdadeiro
$\hat{Y}$: rótulo previsto

V. Resultados e discussões

As 1000 imagens do conjunto de teste foram processadas pelo algoritmo proposto nesse relatório.

A média da acurácia, $\bar{\alpha}$, possui valor $84,43\%$, com um desvio padrão de 14.5%.

A figura 3 ilustra a distribuição da acurácia do algorítimo. Observa-se que a maioria dos valores se concentram ao redor da média, com a maior incidência sendo um α de 100%, cerca de 20.8% de m.

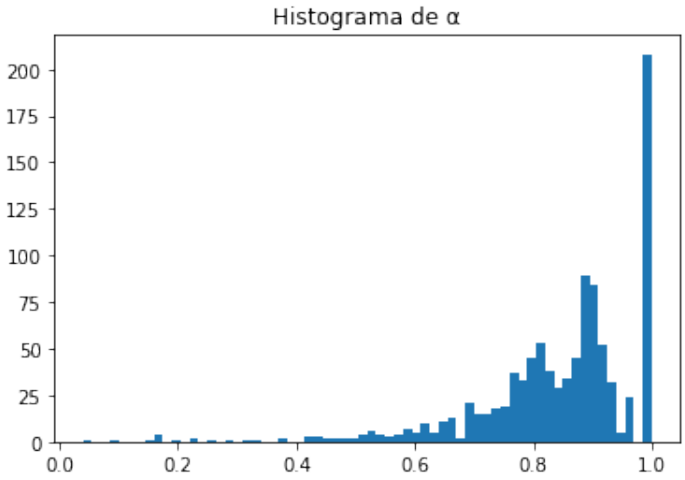

Figura 3: Histograma de α, um índice de semelhança entre duas strings.

Assumindo que a distribuição de α é similar à uma gaussiana, temos que a probabilidade de uma nova imagem obter um valor de α superior a um limiar θ pode ser ilustrada na figura 4.

Considerando somente os casos onde o algorítimo rotulou com 100% de acerto as equações, 208 de 1000 testes, obtemos uma taxa relativamente baixa de acertos.

É possível que essa taxa seja baixa pelos seguintes motivos:

1) A rede neural não consegue abstrair o conjunto do MNIST e do dataset de operadores para as imagens utilizadas no conjunto de teste.
2) A segmentação falha em filtrar somente os operadores e operandos, assumindo ruídos como caracteres.

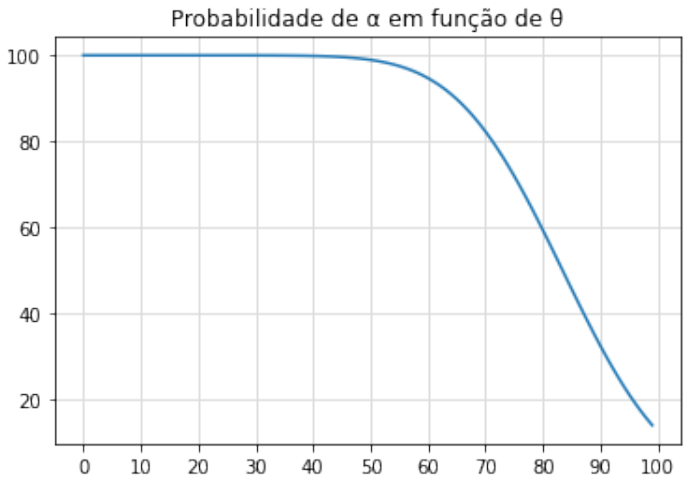

Figura 4: Probabilidade de uma nova imagem semelhante às imagens do conjunto de treino de obter um valor de α em função de θ — um limiar de similaridade arbitrário

VI. Conclusão

Este relatório apresentou uma solução de visão computacional para auxiliar no reconhecimento de expressões matemáticas escritas à mão. O algoritmo proposto segue as etapas de processamento de imagem, incluindo pré-processamento, segmentação, e classificação. Os resultados obtidos mostraram uma média de acurácia de 84.43%, com um desvio padrão de 14.5%.

Ainda assim, a taxa relativamente baixa de acertos pode ser explicada pela dificuldade da rede neural em abstrair o conjunto de dados e pela falha na segmentação em filtrar somente os operadores e operandos.

A análise dos resultados apresentou que a maioria dos valores se concentram ao redor da média, com a maior incidência sendo um índice de semelhança de 100%, cerca de 20.8% dos casos.

No entanto, é importante destacar que mesmo com uma média de acurácia elevada, ainda há espaço para melhorias no algoritmo.

Uma possível causa para a baixa taxa de acertos é a dificuldade da rede neural em abstrair o conjunto de dados utilizado para treinamento.

Além disso, a segmentação de imagem pode não estar filtrando somente os operadores e operandos, o que pode levar a erros na classificação.

Referências

[1] Photomath, 2022.
[2] Kajol Gupta. Digit recognition using convolution neural network. *arXiv preprint arXiv:2004.00331*, 2020.
[3] S. Haykin. *Neural Networks and Learning Machines*. Pearson Education, 2009.
[4] Thomas Huang. Computer vision: Evolution and promise. 1996.
[5] Kaggle. Mnist dataset. https://www.kaggle.com/competitions/digit-recognizer/data, n.d. Accessed: 22/12/2022.
[6] O. Kroeger. Tensorflow, mnist and your own handwritten digits, 2021.

[7] OpenCV. Opencv: Open source computer vision library. https://opencv.org/, n.d. Accessed: [insert date].
[8] Alan V Oppenheim, John Buck, Michael Daniel, Alan S Willsky, Syed Hamid Nawab, and Andrew Singer. *Signals & systems*. Pearson Educación, 1997.
[9] RobinXL. Handwritten math equation image generator. https://github.com/RobinXL/Handwritten-Math-Equation-Image-Generator, 2023.
[10] Raqueeb Shaikh. Opencv (findcontours) detailed guide. *Medium*, n.d. Accessed: 22/12/2022.
[11] Frank Y Shih. *Image processing and mathematical morphology: fundamentals and applications*. CRC press, 2017.
[12] xainano. Handwritten math symbols. https://www.kaggle.com/datasets/xainano/handwrittenmathsymbols, n.d. Accessed: 22/12/2022.