Coordenação do Curso Superior de Bacharelado em Engenharia de Computação

Cláudio Alberto de Sousa Bezerra
Harrison Wendel Rodrigues

Sistema de Correção Quantitativa para Teste de Atenção Concentrada através de Técnicas de Visão Computacional e Aprendizado de Máquina

Campina Grande, 2022

Cláudio Alberto de Sousa Bezerra
Harrison Wendel Rodrigues

Sistema de Correção Quantitativa para Teste de Atenção Concentrada através de Técnicas de Visão Computacional e Aprendizado de Máquina

Monografia apresentada à Coordenação do Curso Superior de Bacharelado em Engenharia de Computação – Campus Campina Grande como requisito para conclusão do curso de Bacharelado em Engenharia de Computação

Orientador: Prof. Dr. Katyusco de Farias Santos

B574s Bezerra, Cláudio Alberto de Sousa.

Sistema de correção quantitativa para teste de atenção concentrada através de técnicas de visão computacional e aprendizado de máquina / Claudio Alberto de Sousa Bezerra, Harrison Wendel Rodrigues. - Campina Grande, 2022.

52p.:il.

Trabalho de Conclusão de Curso - Monografia (Curso de Bacharelado em Engenharia da Computação) - Instituto Federal da Paraíba, 2022.

Orientador: Prof. Dr. Katyusco de Farias Santos.

1.Engenharia da computação. 2. Processamento digital de imagem. 3.Visão computacional. I. Rodrigues, Harrison Wendel. II. Título.

CDU 004.932

Sistema de Correção Quantitativa para Teste de Atenção Concentrada através de Técnicas de Visão Computacional e Aprendizado de Máquina

Cláudio Alberto de Sousa Bezerra
Harrison Wendel Rodrigues

_____

Katyusco de Farias Santos
Orientador

_____

Membro da Banca

_____

Membro da Banca

AGRADECIMENTOS

A minha trajetória até aqui foi marcada por fenômenos inexplicáveis, dos quais só consigo atribuir a causa principal sendo o Deus criador, então a Ele toda honra, glória e louvor. Gostaria de agradecer a todos que me apoiaram de alguma forma e a todos aqueles que não apoiaram também, vocês foram fundamentais no meu processo de amadurecimento e crescimento. Aos professores, meus agradecimentos por todo conhecimento transmitido durante a trajetória, que por sinal, foram além do conhecimento técnico e profissional. Por fim, de uma forma muito especial, agradeço aos meus pais, Cláudio Aleixo e Alba Cristina, por me apoiarem de seu jeito, mas sem vocês nada disso seria possível. A minha noiva, Amanda Moura, por sempre apoiar, acreditar e me fazer enxergar as coisas por outra perspectiva. Aos meus amigos de turma, obrigado por cada dia convivido, cada experiência compartilhada e cada desafio em que estivemos unidos e motivando um ao outro.

Cláudio Alberto de Sousa Bezerra

AGRADECIMENTOS

Aos meus pais, Adma Clécia e José Gilberto, por todo o esforço imensurável que fizeram para que eu pudesse chegar aonde cheguei, mesmo diante de inúmeras dificuldades que a vida impôs. Aos professores e mentores que encontrei durante minha trajetória até aqui, que sempre me incentivaram e acreditaram no meu sucesso, por todo o conhecimento transmitido e apoio para vencer tantos obstáculos. Aos meus colegas de turma, que estiveram ao meu lado durante todos esses anos, por terem caminhado ao meu lado nessa longa jornada. A minha esposa Michele de Lima que está todos os dias ao meu lado lutando pelo meu sucesso. Ao meu filho Dominic, por ter acendido uma chama de esperança e felicidade no meu coração. Gostaria de agradecer a todos que tiveram alguma participação para que eu chegasse até aqui.

Harrison Wendel Rodrigues Santos

RESUMO

O teste de setas é um dos exames que compõem o teste de psicotécnico. Sua aplicação é de fundamental importância para os psicólogos obterem uma análise do comportamento de um indivíduo. Todavia, os testes de setas são corrigidos de forma manual, gerando uma grande carga de trabalho para os respectivos profissionais. Pensando nisso, o objetivo deste projeto é desenvolver uma aplicação que seja capaz de automatizar o processo de correção do teste de setas através de técnicas de processamento de imagem para aquisição, melhoria das imagens, visão computacional para reconhecimento e aprendizagem de máquina supervisionada para classificação. Para isso, foram impressas e respondidas diversas cópias do teste de setas, para realizar o treinamento e validação da rede neural. A rede neural desenvolvida mostrou acurácia superior a 95% durante a fase de treinamento e validação do modelo. Dado o contexto da aplicação, notamos que o modelo pode ser ainda aperfeiçoado para obter um desempenho melhor em relação ao custo computacional. Além disso, foi de grande importância construir e testar com diferentes formatos de dados para possibilitar a maior generalização do modelo de treinada.

Palavras-chave: Processamento Digital de Imagem, Redes Neurais, Testes de Lógica, Visão Computacional, Aprendizado de Máquina.

ABSTRACT

The arrow test is one of the exams that make up the psychotechnical test. Its application is of fundamental importance for psychologists to obtain an analysis of an individual's behavior. However, the arrow tests are manually corrected, generating a large workload for the respective professionals. With that in mind, the objective of this project is to develop an application that is capable of automating the process of correction of the arrow test through image processing techniques for acquisition, image improvement, computer vision for recognition and supervised machine learning for classification. For this, several copies of the arrow test were printed and answered to perform the training and validation of the neural network. The developed neural network showed accuracy greater than 95% during the model training and validation phase. Given the context of the application, we note that the model can still be improved to obtain a better performance in relation to the computational cost. In addition, it was of great importance to build and test with different data formats to enable greater generalization of the trained model.

Keywords: Digital Image Processing, Neural Networks, Logic Tests, Computer Visio, Machine Learning.

LISTA DE FIGURAS

Figura 1 - Teste de setas aplicado pelo Detran 12
Figura 2 - Exemplo de Setas 15
Figura 3 - Imagem do Gabarito 15
Figura 4 - Etapas de um sistema de PDI 16
Figura 5 - Representação de uma imagem digital 17
Figura 6 - Imagem digital sob perspectiva de cada canal de cor 18
Figura 7 - Representação gráfica do processo de filtragem espacial 19
Figura 8 - Aplicação do *Mediam Blur* com diferentes kernels/máscaras 20
Figura 9 - Exemplo de aplicação de limiarização 21
Figura 10 - Representação de neurônio MCP da rede Perceptron 24
Figura 11 - Representação em formato de grafo orientado de uma RNA 24
Figura 12 - *Multilayer* Perceptron com multicamadas 25
Figura 13 - Representação da arquitetura de uma API 28
Figura 14 - Exemplos de testes respondidos 30
Figura 15 - Região do gabarito 31
Figura 16 - Resultado da etapa de segmentação das respostas 32
Figura 17 - Imagem original e com a aplicação de filtros, respectivamente 34
Figura 18 - Matriz de respostas com limites 35
Figura 19 - Extração do ROI para cada linha da matriz 36
Figura 20 - ROI da seta identificado e delimitado 36
Figura 21 - Seta cortada corretamente 37
Figura 22 - Arquitetura da rede neural desenvolvida 37
Figura 23 - Estrutura de pastas do dataset 39
Figura 24 - Exemplos de casos de marcações problemáticas 41
Figura 25 - Gráfico de comparação da função loss 44
Figura 26 - Gráfico de comparação da função de acurácia 44
Figura 27 - Gráfico de comparação da função de perda com *overfit* 46
Figura 28 - Gráfico de comparação da função de acurácia com *overfit* 46
Figura 29- Tela de Home 48
Figura 30 - Tela de Correção 48
Figura 31 - Tela de Resultado 48
Figura 32 - Tela de Histórico 48

LISTA DE ABREVIATURAS E SIGLAS

ROI	*Region of Interest* (região de interesse)
PDI	Processamento Digital de Imagem
VC	Visão Computacional
ANN	*Artificial Neural Network*
CNN	*Convolution Neural Network*
MLP	*Multilayer Perceptron*
FOV	*Focus on view*
CBPF	Centro Brasileiro de Pesquisas Físicas
JSON	*JavaScript Object Notation*
API	Interface de Programação de Aplicações
DETRAN	Departamento Estadual de Trânsito
IFPB	Instituto Federal de Educação, Ciência e Tecnologia da Paraíba
RGB	*Red, Green, Blue*
CCD	Dispositivo de Carga Acoplada

LISTA DE TABELAS

Tabela 1 - Distribuição do conjunto de dados completo 31
Tabela 2 - Descrição das camadas da rede neural 34
Tabela 3 - Descrição do dataset sem aumento de dados 36
Tabela 4 - Descrição do dataset com aumento de dados 37

SUMÁRIO

1 **INTRODUÇÃO** **12**
1.1 CONSIDERAÇÕES GERAIS 12
1.2 PROBLEMÁTICA 12
1.3 JUSTIFICATIVA 14
1.4 OBJETIVOS 14

2 **FUNDAMENTAÇÃO TEÓRICA** **14**
2.1 CORREÇÃO DO TESTE DE SETAS 15
2.2 PROCESSAMENTO DIGITAL DE IMAGENS 15
2.2.1 AQUISIÇÃO E DIGITALIZAÇÃO 16
2.2.2 REPRESENTAÇÃO DE UMA IMAGEM DIGITAL 16
2.2.3 PRÉ-PROCESSAMENTO 17
2.2.4 ESCALA DE CINZA 18
2.2.5 FILTRAGEM ESPACIAL 18
2.2.6 MEDIAN BLUR 19
2.2.7 SEGMENTAÇÃO 19
2.2.8 CLASSIFICAÇÃO 21
2.3 VISÃO COMPUTACIONAL 21
2.4 REDES NEURAIS ARTIFICIAIS 21
2.4.1 MULTILAYER PERCEPTRON (MLP) 23
2.4.2 DEEP LEARNING 25
2.4.3 REDES NEURAIS CONVOLUCIONAIS 25
2.4.4 TENSORFLOW 26
2.4.5 API REST 26
2.5 ESTADO DA ARTE 27

3 **METODOLOGIA** **28**
3.1 TECNOLOGIAS 28
3.1.1 FRAMEWORKS E BIBLIOTECAS 28
3.1.2 BASE DE DADOS 28
3.1.3 TREINAMENTO 29
3.2 CONTEXTUALIZAÇÃO E ESTRATÉGIAS DESENVOLVIDAS 30
3.2.1 MODELO DE DETECÇÃO DO GABARITO 30
3.2.2 MODELO DE IDENTIFICAÇÃO DA REGIÃO DAS RESPOSTAS 30
3.2.3 MODELO DE TREINAMENTO DA REDE NEURAL 31
3.3 ETAPAS DO DESENVOLVIMENTO 31
3.4 LIMITAÇÕES E AMEAÇAS 41

4 **RESULTADOS** **40**

5 CONSIDERAÇÕES FINAIS **45**

6 REFERÊNCIAS BIBLIOGRÁFICAS **46**

1. INTRODUÇÃO

Neste capítulo iremos descrever uma visão geral deste projeto de conclusão de curso. Inicialmente abordaremos um embasamento teórico sobre o teste na subseção 1.1, em seguida apresentaremos a problemática na subseção 1.2, seguida pela justificativa do projeto na subseção 1.3 e por fim os objetivos do projeto na subseção 1.3.

1.1.CONSIDERAÇÕES GERAIS

O teste de psicotécnico é um método de avaliação de personalidade que permite avaliar o comportamento de um indivíduo, assim como suas reações em determinadas situações. Este teste é exigido pelo DETRAN em diversas situações, como concursos públicos, emissão de porte de armas, classificação de habilitação e outros processos onde é obrigatório comprovar saúde mental. Este teste, conhecido popularmente como exame psicotécnico, é amplamente difundido por ser exigido no processo de aquisição da Carteira Nacional de Habilitação.

A aplicação deste teste pode ocorrer em clínicas médicas ou no próprio DETRAN, todavia deve ser aplicado por profissionais da saúde especializados na área da psicologia. Além da aplicação, a correção também deve ser feita por um profissional com as mesmas competências.

1.2.PROBLEMÁTICA

No contexto explicado na subseção anterior, surge então a problemática da quantidade de trabalho manual desempenhado pelos respectivos profissionais, visto que não há nenhuma ferramenta especializada na correção desses testes. Sendo assim, desenvolver uma ferramenta capaz de fornecer uma correção quantitativa dos exames se faz de grande proveito.

A aplicação do exame psicotécnico acontece de forma manual, onde cada avaliado recebe uma folha de papel impressa com os exercícios propostos e os responde conforme as orientações do profissional. Da mesma forma, a correção dos testes é feita de forma manual pelos respectivos profissionais de saúde.

A coleta dos dados quantitativos do exame, seu armazenamento e a avaliação são feitas por profissionais que não dispõem de uma ferramenta especializada para auxiliá-los. Eles são responsáveis pela aplicação, correção quantitativa dos testes e avaliação qualitativa dos resultados.

O teste de setas (figura 1) é um dos exercícios que compõem o teste do psicotécnico. Neste exercício, o avaliado deve identificar as setas-alvo, localizadas na parte superior da folha, dentro do retângulo e marcar com um risco as setas da matriz de respostas que sejam iguais a alguma das setas-alvo, em um curto intervalo de tempo.

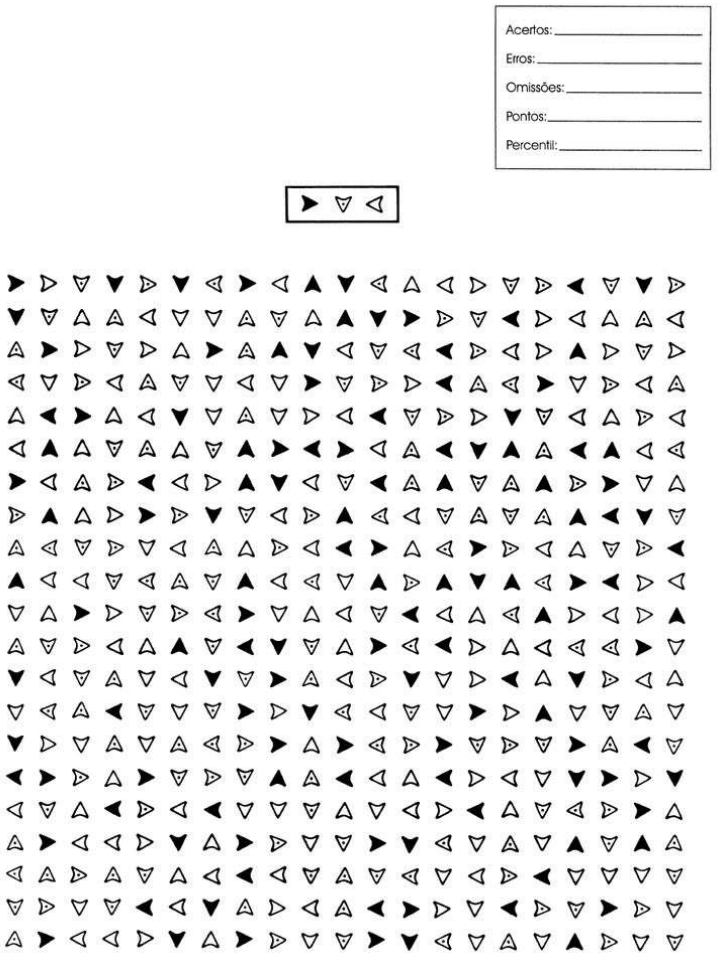


Figura 1 - Teste de setas (Fonte: Google Imagens)

Como é possível observar na imagem acima, pela quantidade de setas e o curto período de tempo, o exercício exige a aplicação de uma alta carga de

concentração por parte do avaliado para sua resolução completa e correta. A correção deste exercício é feita por um profissional da saúde com o auxílio de um gabarito oficial, e este avalia fazendo a contagem de erros, acertos e omissões em cada linha e ao final propõe uma análise qualitativa das respostas. Dessa forma, o profissional encarregado leva muito tempo para realizar as correções e fornecer o resultado, além da possibilidade de falhas na correção devido ao estresse mental.

1.3.JUSTIFICATIVA

Tendo como base a problemática apresentada, a proposta deste projeto é fornecer um sistema que, obtendo uma imagem ou um conjunto de imagens de exames, faça uma análise quantitativa das respostas marcadas no teste, fornecendo um relatório de erros, acertos e omissões para cada imagem. Para que isso seja possível, serão utilizadas técnicas de processamento digital de imagens, visão computacional e aprendizagem de máquina.

O sistema utilizará diferentes técnicas com o objetivo de realizar o tratamento na imagem, buscar pelas regiões de interesse, encontrar padrões e por fim fornecer o relatório quantitativo.

Esta problemática faz parte do contexto atual de diversos profissionais de saúde que atuam, especialmente, em trabalhos para o DETRAN. Dessa forma, a grande motivação para o desenvolvimento deste projeto se dá pela relevância para esses profissionais, visto que não há uma ferramenta eficaz, eficiente e disponível que os auxilie. Este projeto propõe uma solução, de fácil acesso e utilizando-se de ferramentas tecnológicas para obtermos a eficiência e eficácia no resultado proposto pela solução.

1.4.OBJETIVOS

Objetivo geral

- Utilizar de técnicas de processamento digital de imagem, visão computacional e aprendizado de máquina para desenvolver uma ferramenta que seja versátil, confiável e eficaz em fazer a correção quantitativa do teste de atenção concentrada.

Objetivos específicos

- Desenvolver uma ferramenta mobile e web para melhor versatilidade.
- Desenvolver um modelo de rede neural com acurácia superior a 90%.
- Executar uma correção quantitativa do teste de setas.
- Automatizar o processo de correção do teste de setas.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo será apresentado o embasamento teórico necessário para compreensão das etapas subsequentes no desenvolvimento efetivo do projeto. Desta forma, na subseção 2.1 será apresentado como a correção do teste é efetuada, na subseção 2.2 serão apresentadas as principais técnicas de PDI utilizadas

2.1. CORREÇÃO DO TESTE DE SETAS

No teste de setas, uma seta é representada por um símbolo que pode assumir diversas variáveis, conforme representado nas figuras a seguir:

Figura 2: Exemplos de Setas (Fonte: própria)

No teste há uma matriz contendo uma variedade de símbolos e o objetivo final é assinalar quais símbolos presentes na matriz, demonstrada na figura 1, que são exatamente iguais as da região do gabarito, demonstrado na figura 3 abaixo:

Figura 3 : Imagem do gabarito (Fonte: própria)

Na correção manual do teste de setas, o profissional de saúde faz o levantamento de três métricas principais, são elas:

- Erros: marcações em setas que não estão presentes no gabarito.
- Acertos: marcação de seta que está presente no gabarito.
- Omissões: não marcação de seta que está presente no gabarito.

Através destes dados quantitativos o respectivo profissional faz a avaliação qualitativa, indicando a aprovação ou reprovação do candidato.

2.2.PROCESSAMENTO DIGITAL DE IMAGENS

Entende-se por processamento digital de imagens, o conjunto de técnicas voltadas para manipulação, análise e classificação de dados multidimensionais, sendo assim, a entrada e a saída do processamento é composta por imagens e/ou informações relativas às mesmas[1].

O processamento digital de imagens subdivide-se em diferentes etapas, a seguir serão demonstradas e explicitadas de forma sucinta as etapas mais utilizadas. Segue o diagrama das principais etapas representado na figura 4:

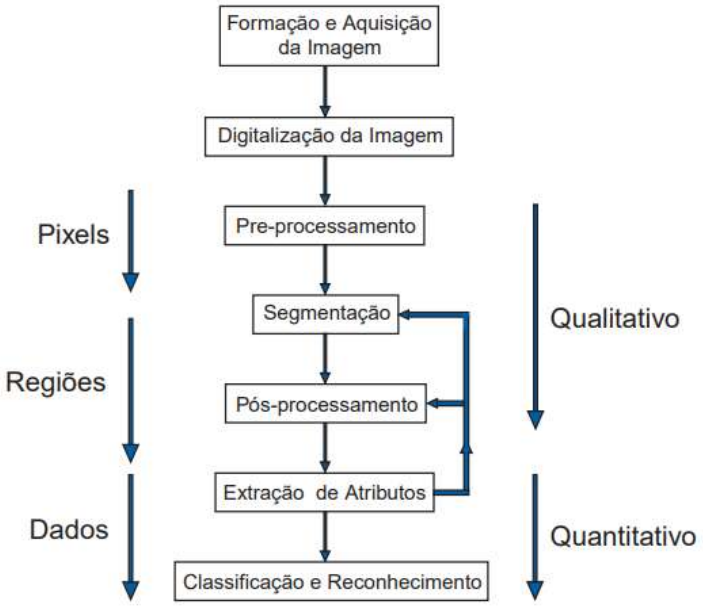


Figura 4 - Etapas de um sistema de PDI (Fonte: CBPF)

2.2.1. AQUISIÇÃO E DIGITALIZAÇÃO

O processo de aquisição de imagens digitais faz o uso de sensores capazes de perceber a luz do ambiente e convertê-la em uma imagem. Em sua maioria, as imagens digitais são originadas a partir de um sensor matricial CCD, presente em câmeras digitais[1].

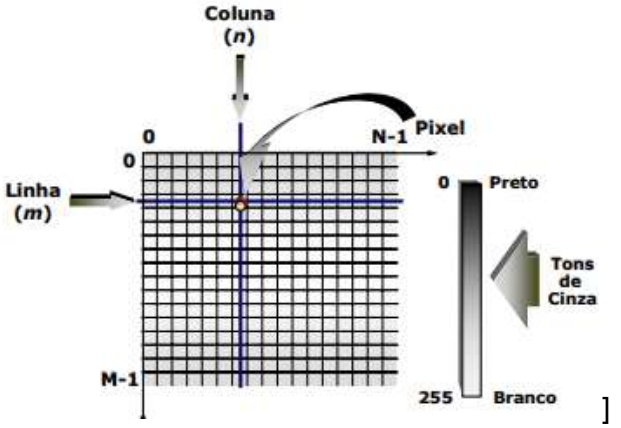
A primeira função realizada pelo sistema de aquisição de imagens, é coletar a energia de entrada e projetá-la em um plano de imagem. Se a iluminação for luz a

entrada frontal do sistema de aquisição de imagens é uma lente óptica que projeta a cena vista sobre o plano focal da lente. O arranjo de sensores, que coincide com o plano focal, produz saídas proporcionais à integral da luz recebida em cada sensor. Circuitos digitais e analógicos realizam uma varredura nessas saídas e as convertem em um sinal analógico, que é então digitalizado por um outro componente do sistema de aquisição de imagens[1].

2.2.2. REPRESENTAÇÃO DE UMA IMAGEM DIGITAL

Uma imagem digital monocromática é representada por uma função bidimensional contínua expressa na forma $f(x, y)$, no qual x e y são coordenadas espaciais e o valor de f em qualquer ponto (x, y) é proporcional a intensidade luminosa no ponto considerado[1].

Cada ponto na matriz bidimensional que representa a imagem digital é denominado elemento da imagem ou pixel. Na figura 5, está representada a notação matricial usual para a localização de um pixel no arranjo bidimensional.



]

Figura 5 - Representação de uma imagem bidimensional (Fonte: Research Gate)

Em uma imagem digital colorida no sistema RGB(*Red, Green, Blue*) um pixel pode ser representado como uma composição de três imagens monocromáticas, na qual cada imagem representa um canal de cor com a respectiva intensidade luminosa[3].

Figura 6 - Imagem digital sob perspectiva de cada canal de cor (Fonte: própria).

2.2.3. PRÉ-PROCESSAMENTO DE IMAGEM

As técnicas de pré-processamento têm a função de melhorar a qualidade da imagem a fim de aumentar a probabilidade de sucesso nos processos subsequentes e, para isso, são aplicadas técnicas que operam no domínio espacial e técnicas que operam no domínio da frequência[2].

Nesta etapa, são implementadas técnicas para redução de ruídos, aumento de contraste, calibração radiométrica, correção de distorções geométricas e técnicas de realce[2]. Podemos citar dentre as principais técnicas: Filtro Gaussiano, realce por equalização de histograma, realce logaritmo, realce linear, Chessboard Detection, etc.

2.2.4. ESCALA DE CINZA

Conforme exemplificado no tópico 2.1.2, uma imagem digital é representada através de uma matriz e, no caso de uma imagem colorida é representada por uma matriz tridimensional, em que as dimensões são: altura, largura e a profundidade representada pelos canais de cores. Uma técnica utilizada para otimizar o processamento da imagem, é reduzir os canais de cores da imagem para somente um canal normalmente denominado monocromático ou escala de cinza[3].

Em uma imagem em escala de cinza, cada pixel é representado pela quantidade de luz refletida nele, isto proporciona maior nitidez para delimitar as

regiões presentes na imagem. Sendo assim, esta etapa precede todas as outras etapas de processamento e segmentação[3].

2.2.5. FILTRAGEM ESPACIAL

A filtragem espacial se fundamenta em uma operação de convolução de uma máscara (*mask* ou *kernel*) e da imagem digital considerada. A máscara é um arranjo matricial de dimensões inferiores às da imagem a ser filtrada e, em geral, quadrado, cujos valores são definidos como fatores de ponderação (pesos) a serem aplicados sobre os pixels da imagem. A operação é executada progressivamente sobre os pixels da imagem, coluna a coluna, linha a linha[3], como ilustrado na figura 7 abaixo e na equação 2.1:

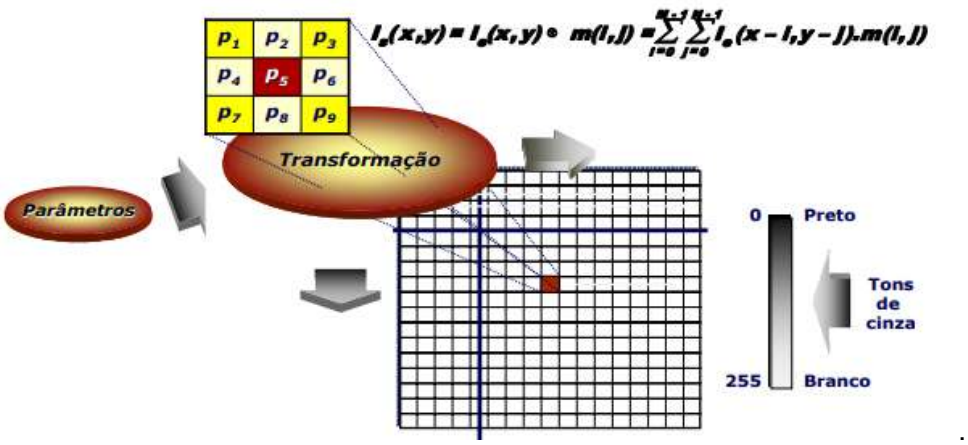


Figura 7 - Representação gráfica do processo de filtragem espacial. Fonte (Queiroz, 2003)[3].

$$I_s(x,y) = I_e(x,y) \otimes m(i,j) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I_e(x-i, y-j).m(i,j) \qquad (2.1)$$

2.2.6. MEDIAN BLUR

O filtro *median blur* ou filtro da mediana, é uma técnica de processamento de imagem que utiliza um *kernel* para operar sobre cada pixel da imagem, aplicando a mediana dos pixels vizinhos ao pixel analisado[3[. Este filtro reduz os ruídos da

imagem, de forma que quanto maior o *kernel*, menor será a nitidez da imagem e maior a quantidade de ruído.

Na figura 6 temos um conjunto de imagens representando a aplicação do filtro. A imagem com o título “original” representa a imagem com ruídos, enquanto as outras imagens demonstram o comportamento do resultado da aplicação do filtro de acordo com o tamanho do *kernel* aplicado, variando de 5 a 15. Dessa forma, é possível perceber a influência do filtro, de forma que quanto maior o *kernel*, ilustrado pela imagem à direita com *kernel* 15, maior será a quantidade de ruídos removidos.

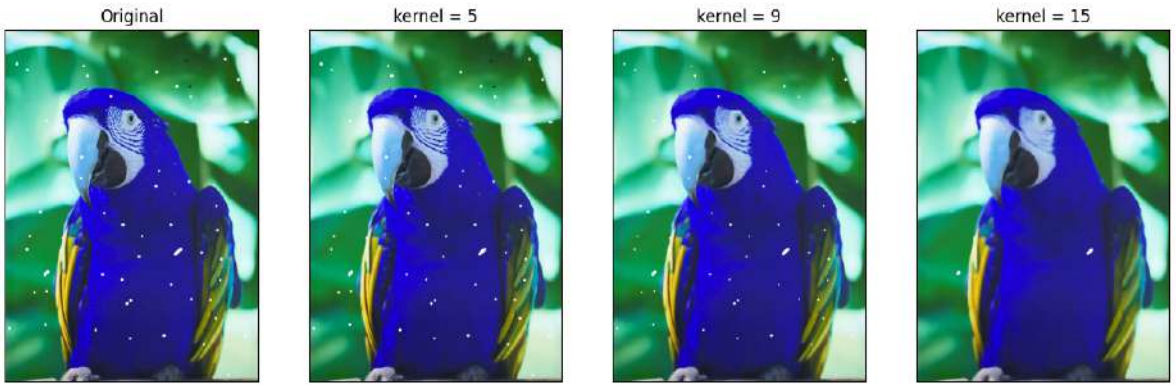


Figura 8 - Aplicação do Median Blur com diferentes kernels/máscaras. (Fonte: própria)

2.2.7. SEGMENTAÇÃO

Nesta etapa, a imagem é decomposta em regiões. Para as regiões as quais estamos buscando, denominamos região de interesse(*ROI*), enquanto as demais regiões podem ser consideradas como fundo da imagem ou *background*[3]. A segmentação é um processo empírico e adaptativo para cada sistema e para cada tipo de imagem requisitada. É importante avaliar e adequar os métodos e os parâmetros de segmentação para cada tipo de imagem, pois, de forma geral, a qualidade da imagem afeta diretamente esta etapa[3].

No tocante à segmentação de imagens monocromáticas, os algoritmos fundamentam-se, em essência, na descontinuidade e na similaridade dos níveis de cinza. A fundamentação na descontinuidade consiste no particionamento da imagem em zonas caracterizadas por mudanças bruscas dos níveis de cinza[3]. O interesse recai usualmente na detecção de pontos isolados, de linhas e de bordas da imagem.

Por outro lado, a fundamentação na similaridade consiste na limiarização e no crescimento de regiões[3].

A técnica de limiarização consiste em agrupar níveis de cinza (em geral, considera-se uma faixa de 0 a 255, em que 0 representa a cor preta e 255 a cor branca) de uma imagem de acordo com a definição de um limiar. No caso mais simples, um único limiar é definido. Nessa técnica, todos os pixels do histograma da imagem são rotulados como sendo do objeto ou do *background*[12]. A técnica de limiarização é aplicada no exemplo da Figura 1.

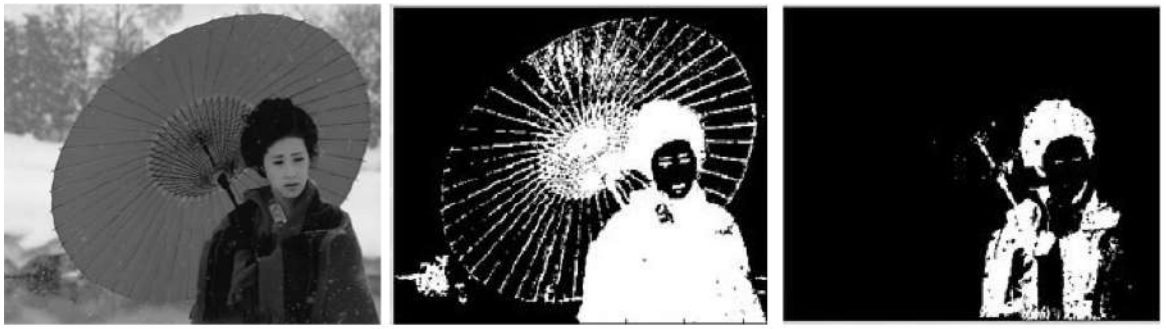


Figura 9 – Da esquerda para a direita, a imagem original, seguida pela mesma imagem aplicando-se um limiar 30 e 10, respectivamente. (Fonte: PetNews UFCG)

2.2.8. CLASSIFICAÇÃO DE OBJETOS

Após a etapa de segmentação, as regiões de interesse estarão em evidência, tornando possível a aplicação de técnicas de reconhecimento de padrões e extração de características da imagem. Para isso, diferentes técnicas podem ser aplicadas e se tratando de inteligência artificial, podem ser aplicadas técnicas de aprendizado de máquina supervisionado e não supervisionado.

As técnicas de aprendizado de máquina subdividem-se em: aprendizagem supervisionada e não supervisionada. Para a técnica com aprendizado supervisionado, a classificação é realizada baseada em características extraídas de um conjunto de dados de entrada, ao qual chamamos de base de conhecimento. Para a técnica com aprendizado não supervisionado, a classificação é realizada

baseado em características similares entre objetos, sem parâmetros de entrada previamente definidos, formando grupos ou Clusters [3].

2.3. VISÃO COMPUTACIONAL

A visão computacional tem por objetivo extrair características e informações importantes de imagens aquisitadas por dispositivos de captura (MILANO; HONORATO). Desta forma, a visão computacional utiliza-se de técnicas de processamento de imagens para realizar tarefas, e dentre elas estão:

- Reconhecimento de padrões;
- Reconstrução de cenários em 3D;
- Análise de movimentos;
- Identificação de objetos ou pessoas;
- Detecção da ocorrência de eventos.

2.4. REDES NEURAIS ARTIFICIAIS

As redes neurais artificiais têm por objetivo emular o comportamento biológico de um neurônio[5]. A primeira rede neural foi desenvolvida em 1958 por Frank Rosenblatt e ficou conhecida como Perceptron. Este modelo de rede possui duas camadas, uma de entrada e uma de saída, no qual sua saída é dada pelo somatório do produto das entradas pelos seus respectivos pesos[5], conforme é possível observar na figura abaixo e na equação 2.2 expressa abaixo:

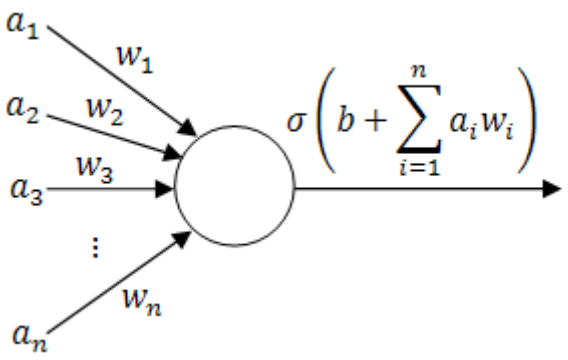


Figura 10 - Representação de neurônio MCP da rede Perceptron

$$\sigma(b + \sum_{i=1}^{n} a_i w_i) \qquad (2.2)$$

Os pesos (ω) são multiplicados com as entradas (a), a soma dos dois é aplicada uma função σ, que geralmente podem ser de três tipos:

- Função sigmóide: $\sigma(x) = \frac{1}{1+exp(-x)}$
- Tangente hiperbólica: $\sigma(x) = tanh(x)$
- Linear simples: $\sigma(x) = x$

Caso o resultado de X seja positivo, o resultado é 1, caso contrário é 0. Sendo assim, os pesos são ajustados de acordo com a seguinte equação:

$$w_i(t+1) = w_i(t) + \alpha(d_j - y_j(t))\, x_{j,i} \qquad (2.3)$$

Dessa forma $w_i(t+1)$ É o novo peso ajustado levando em conta o peso anterior e o erro. Que por sua vez, é calculado pela diferença entre o valor esperado d_j e o valor encontrado y_j multiplicado pela entrada associada àquele peso e ao valor da taxa de aprendizado. Este tipo de rede é muito utilizado para classificar dados em conjuntos.

A estrutura base de uma Rede Neural Artificial(RNA) possui dois componentes principais: conexões e neurônios. As conexões são responsáveis por transmitir as informações entre os neurônios e dependendo do modelo de rede implementado, entre as diferentes camadas. As conexões possuem um peso sináptico que define a modularização da informação propagada, se positivo, a conexão é excitatória, se negativo, a conexão é inibitória, caso seja zero, não há conexão[5].

Uma RNA pode ser representada por um grafo orientado em que as conexões entre os nós possuem apenas um sentido, conforme representado na figura:

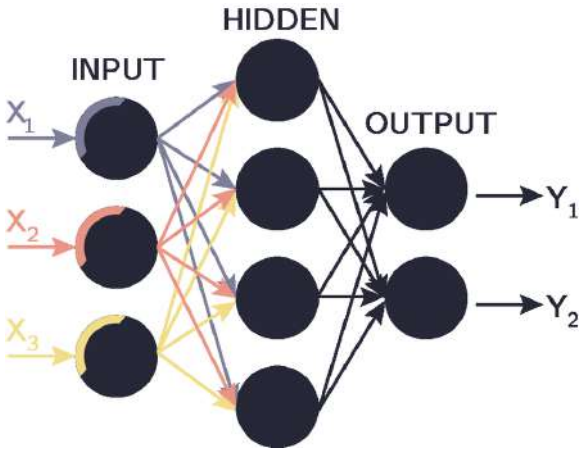


Figura 11 - Representação em formato de grafo orientado de uma RNA.
(Fonte: PNG Wing)

Cada neurônio é responsável por uma parte do processamento da rede. Ele pode ser definido como tendo três elementos base:

- Conjunto de sinais de entrada: são as saídas dos neurônios de camadas anteriores
- Pesos sinápticos: são os pesos das conexões entre os neurônios adjacentes.
- Função de soma
- Função de ativação

2.4.1. REDE *MULTILAYER PERCEPTRON* (MLP)

Uma rede *Multilayer Perceptron* tem como principal diferença a quantidade de camadas, possuindo além das camadas de entrada e saída, um conjunto de camadas ocultas denominadas *Hidden Layer*. Enquanto nas redes do tipo Perceptron o neurônio deve ter uma função de ativação que impõe um limiar, como ReLU ou sigmoid, na *Multilayer Perceptron* podem ser utilizadas qualquer função de ativação arbitrária. Segue na figura 12 o diagrama do modelo da arquitetura de uma rede *multilayer perceptron*:

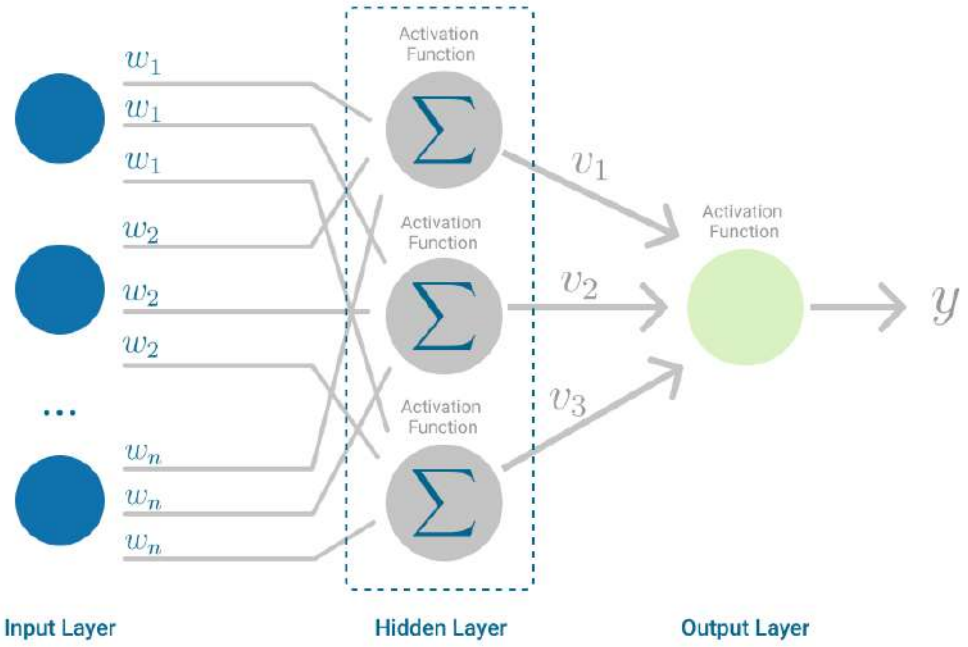


Figura 12 - *Multilayer Perceptron* com multicamadas (Fonte: Towards Data Science)

Este tipo de rede se enquadra na categoria de algoritmos *feedforward*, pois as entradas são combinadas com os pesos iniciais em uma soma ponderada e submetidas a função de ativação, semelhante ao perceptron. Porém a diferença é que cada combinação linear é propagada para a próxima camada. Dessa forma, cada camada está alimentando a próxima com o resultado de seus cálculos[9].

O grande diferencial deste algoritmo é a técnica de backpropagation que permite que os pesos da rede sejam ajustados interativamente com o objetivo de minimizar o custo da função. Para que o backpropagation funcione corretamente, a função de combinação das entradas deve ser diferenciável, ou seja, precisam ter uma derivada limitada[9].

Apesar da sua eficiência na resolução de problemas de classificação, as MLP não possuem a mesma eficiência quando o conjunto de entradas são dados bidimensionais, como imagens. Sendo assim, para solucionar estes tipos de problemas, surgiram as Redes Neurais Convolucionais ou CNNs [10].

2.4.2. DEEP LEARNING

Algoritmos de deep learning usam Redes Neurais Artificiais como sua estrutura principal[9]. O que os diferencia de outros algoritmos é que eles não exigem a entrada de dados indicando as caracteristicas que devem ser observadas durante a fase de aprendizado, podendo assim aprender as características dos dados de forma autónoma[9].

Os algoritmos de *deep learning* absorvem o conjunto de dados e aprendem seus padrões, aprendendo como representar os dados baseado em padrões. Em seguida, eles combinam diferentes representações do conjunto de dados, cada uma identificando um padrão ou característica específica, em uma representação mais abstrata e de alto nível do conjunto de dados. Essa abordagem prática, sem muita intervenção humana no design e extração de recursos, permite que os algoritmos se adaptem muito mais rapidamente aos dados disponíveis[9].

2.4.3. REDES NEURAIS CONVOLUCIONAIS

As redes neurais convolucionais ou simplesmente CNNs, se popularizaram como a principal técnica para classificação de imagens em sistemas de visão computacional. Em suma, a CNN é um modelo de *deep learning* que extrai features automaticamente de um conjunto de imagens fornecidas[8].

Na estrutura de uma CNN, podemos contar com três camadas principais:

- Camada Convolucional: esta camada tem por função extrair features de alto nível dos dados de entrada e os encaminha para a próxima camada através de um mapa de features.
- Camada de Pooling: camada utilizada para reduzir as dimensões dos dados aplicando agrupamento. Esta camada recebe a saída da camada convolucional e prepara um mapa de características condensadas.
- Camada Totalmente Conectada: esta camada é responsável por classificar. As probabilidades são calculadas para cada rótulo de classe por uma função de ativação, normalmente chamada de softmax.

2.4.4. TENSORFLOW

O TensorFlow é uma plataforma completa de código aberto para machine learning. Ele tem um ecossistema abrangente e flexível de ferramentas, bibliotecas e recursos da comunidade que permite aos pesquisadores levar adiante técnicas de *machine learning* de última geração e aos desenvolvedores criar e implantar aplicativos com tecnologia de *Machine Learning*[13].

Através do TensorFlow é possível a criação de forma fácil de modelos, usando-se de APIs intuitivas de alto nível, como o Keras com execução rápida, para criar e treinar modelos de machine learning com facilidade. Elas permitem iteração imediata de modelos e depuração simplificada[13].

2.4.5. API REST

API é um acrônimo para Application Programming Interface, ou Interface de programação de aplicações, este tipo de interface permite que, através do desenvolvimento de rotinas e padrões, seja estabelecida uma comunicação de diferentes dispositivos com as mesmas regras de negócio[6].

A arquitetura da API REST garante mais segurança das regras de negócio e melhor interoperabilidade entre diferentes aplicações, dado que toda comunicação e troca de dados ocorre utilizando requisições HTTP.

De forma geral, uma API REST recebe requisições HTTP que podem ser dos tipos POST, PUT, CREATE, UPDATE, DELETE e em seguida executa as regras de negócio e fornece um retorno ao cliente através de um formato previamente definido, que na maioria das aplicações modernas é o JSON.

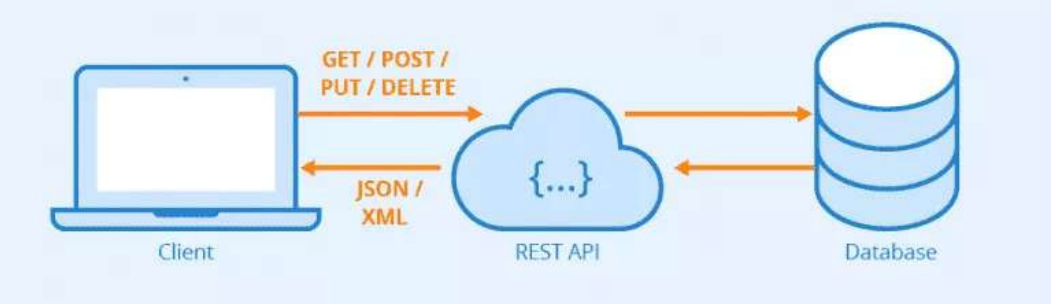


Figura 13 - Representação da arquitetura de uma API. (Fonte: Seobility)

2.5. ESTADO DA ARTE

Atualmente existem no mercado algumas soluções de software que fazem a criação e correção de testes de lógica, todavia não são de fácil acesso, não estão disponíveis nas plataformas mobile e não fazem a correção quantitativa do teste de setas. A seguir serão descritas as principais plataformas e suas funcionalidades.

VOL - Vetor On-line: esta plataforma é voltada para psicólogos do setor de Recursos Humanos, sendo necessário realizar uma assinatura e apresentar no ato de cadastro o certificado de registro no Conselho Regional de Psicologia(CRP). Esta plataforma permite criar e corrigir diversos testes de raciocínio lógico e cognitivo, entretanto o teste de setas não faz parte desse conjunto[14].

SKIP - Sistema de Correção Informatizada do Palográfico: esta plataforma é voltada para correção do teste de palográfico. Através dela é possível gerar relatórios quantitativos e qualitativos sobre a correção deste tipo de teste, entretanto não permite a correção do teste de setas[13].

Foi possível perceber dessa forma, que não há um software com proposta semelhante a nossa, dado que nosso objetivo é desenvolver a ferramenta de forma acessível, cobrando uma assinatura que independe da quantidade de testes que serão corrigidos. Neste projeto, foi tomado como inspiração os softwares citados acima e os softwares que fazem a correção de gabaritos para testes de múltipla escolha, como o Exame Nacional do Ensino Médio.

3. METODOLOGIA

Neste capítulo iremos descrever como o projeto foi desenvolvido, abordando as tecnologias utilizadas e as etapas realizadas no decorrer do desenvolvimento. Além disso, serão descritas as estratégias de processamento de imagem utilizadas, assim como as estratégias para o desenvolvimento da rede neural.

3.1.TECNOLOGIAS

3.1.1. FRAMEWORKS E BIBLIOTECAS

No âmbito do desenvolvimento de sistemas utilizando visão computacional, processamento de imagem e aprendizado de máquina já existem muitas ferramentas e tecnologias disponíveis no mercado, as quais provêem uma gama de ferramentas e métodos que abstraem as operações mais complexas, possibilitando assim que maior parte do tempo seja destinado propriamente ao desenvolvimento da aplicação. Para o desenvolvimento deste projeto foi escolhido como ferramenta principal a linguagem Python, pois proporciona a utilização do framework FastAPI que nos proverá a criação de uma API Rest além de possibilitar fácil integração com frameworks para o desenvolvimento de redes neurais artificiais. Como ferramenta para lidar com o processamento de imagem e visão computacional foi selecionado como principal biblioteca o framework OpenCV, devido a quantidade de conteúdo disponível e por ser de código aberto. Para o desenvolvimento front-end da aplicação, foi escolhido o framework Flutter, pois nos provê a flexibilidade através da sua característica multi-plataforma.

3.1.2. BASE DE DADOS

Foram criados dois datasets com os seguintes propósitos:

- Dataset de testes que serão corrigidos pelo sistema: neste caso, foram respondidos manualmente diversos exemplares do teste, das seguintes maneiras: testes com todas as setas marcadas, variando a maneira que a seta é marcada, testes com algumas setas marcadas, e testes sem nenhuma seta marcada.

- Dataset utilizado para o treinamento e validação da rede neural: neste caso, foi extraída cada seta como uma imagem individual de testes em que todas as setas estavam marcadas, a fim de obter um conjunto de imagens de setas que devem ser consideradas como marcadas. Foi repetido esse processo para um teste em que nenhuma seta foi marcada, a fim de obter um conjunto de imagens de setas que devem ser consideradas como não marcadas.

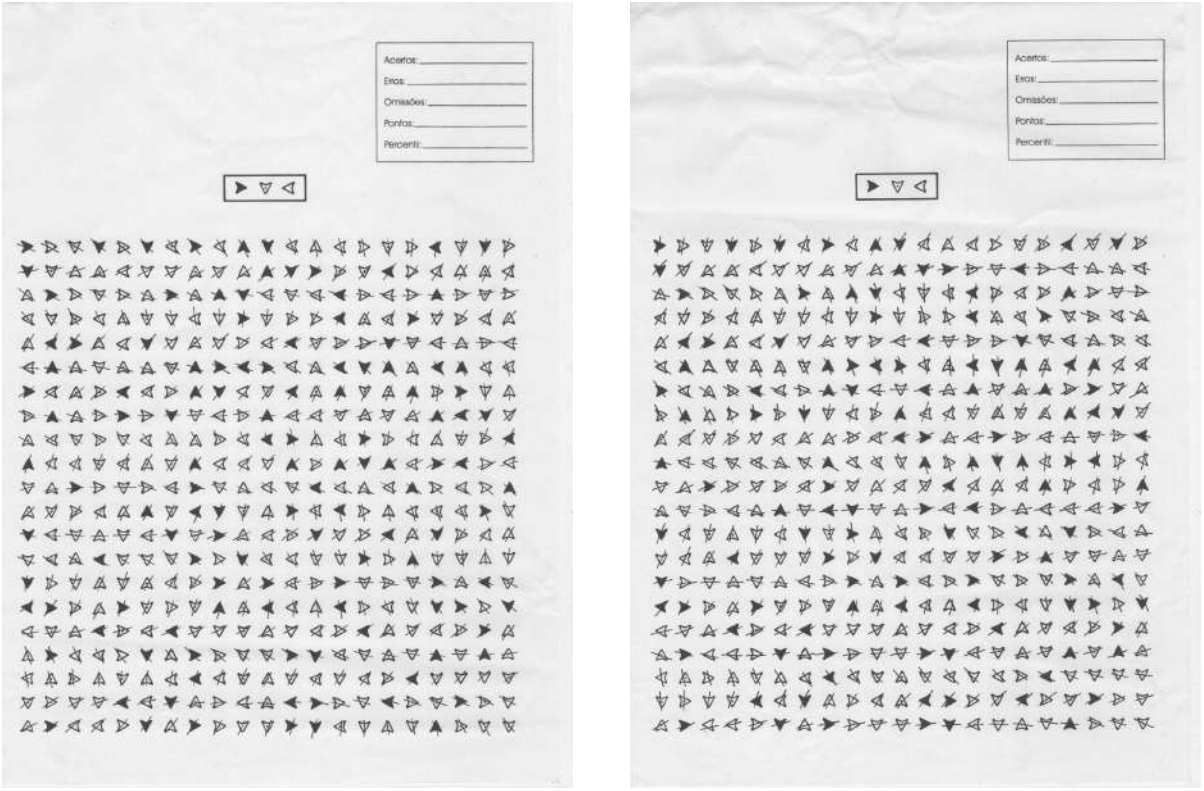


Figura 14 - Exemplos de testes respondidos. (Fonte: própria)

3.1.3. TREINAMENTO

Os modelos de aprendizado de máquina baseado em redes neurais artificiais necessitam de uma etapa de treinamento que pode demandar alto custo computacional, em especial de GPUs, devido a sua capacidade de executar muitos cálculos paralelamente, comparado às CPUs. Pensando nisso, os códigos responsáveis pelo treinamento e validação da CNN foram executados na plataforma Google Colab, que disponibiliza de maneira gratuita servidores com elevada

capacidade de processamento, comparado a dispositivos de uso doméstico. Além disso, o Colab já possui muitas bibliotecas instaladas de forma nativa.

3.2. CONTEXTUALIZAÇÃO E ESTRATÉGIAS DESENVOLVIDAS

Visando alcançar os objetivos estabelecidos para a solução, na qual teremos apenas uma imagem ou uma sequência de imagens que serão analisadas individualmente, foi criado um conjunto de modelos de estratégias para chegar aos objetivos de desenvolvimento. Dessa forma, serão listados os principais modelos desenvolvidos para os principais eventos do sistema, sendo eles: identificação da região do gabarito, identificação da região das respostas e modelo de treinamento da rede neural.

3.2.1. MODELO DE DETECÇÃO DO GABARITO

A região denominada como gabarito está localizada na parte superior da imagem, mais especificamente no primeiro terço da imagem, conforme ilustrado na figura 13. Sendo assim, visando reduzir o tempo gasto e o custo computacional na busca dessa região, a busca dessa região foi reduzida ao primeiro terço da imagem, e foi utilizado o método de detecção de bordas *canny edge* para efetuar a busca pela região retangular.

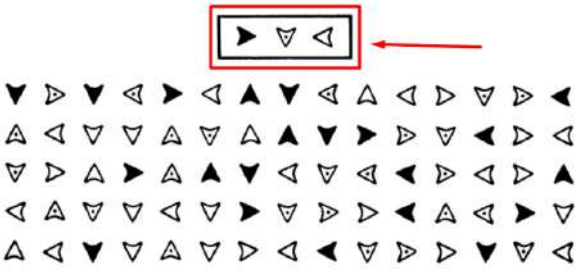


Figura 15 - Região do gabarito. (Fonte: própria)

3.2.2. MODELO DE IDENTIFICAÇÃO DA REGIÃO DAS RESPOSTAS

Para a detecção da região onde são marcadas as respostas, descartamos a região de interesse denominada gabarito, e dessa forma nosso FOV ficou somente na região de marcação das respostas. Logo em seguida, foi implementado o método

de segmentação desenvolvido durante o projeto que será explicado adiante. Como resultado deste modelo foi possível definir os limites superiores e inferiores de cada linha de setas na matriz de respostas, e os limites laterais de cada seta em cada linha. Segue uma representação da saída desta etapa na figura 16 listada abaixo:

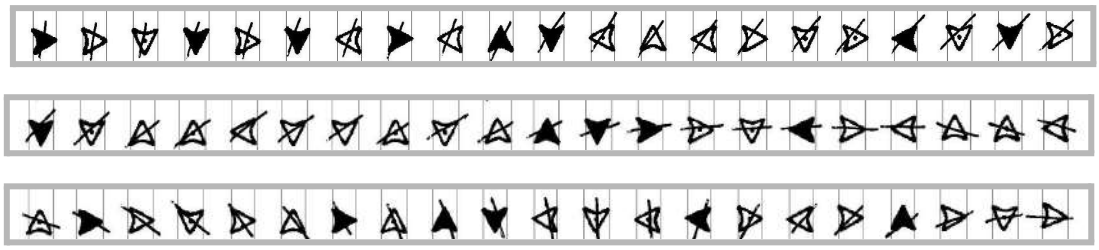


Figura 16 - Resultado da etapa de segmentação das respostas. (Fonte: própria)

3.2.3. MODELO DE TREINAMENTO DA REDE NEURAL

A rede neural convolucional desenvolvida teve como principal motivação classificar e agrupar o conjunto de setas em dois grupos: marcadas e não marcadas. Dessa forma, a responsabilidade do modelo é fornecer uma saída binária.

A correção será proporcionada através da comparação entre duas imagens, um gabarito oficial fornecido como entrada e um teste a ser corrigido. Após a identificação de quais setas estão marcadas no gabarito oficial através da rede, suas coordenadas são salvas em uma matriz e logo em seguida a imagem do teste a ser corrigido é analisada e comparada com as respostas presentes na matriz com as respostas oficiais.

3.3. ETAPAS DO DESENVOLVIMENTO

Nesta subseção, serão apresentadas as etapas realizadas para o desenvolvimento do projeto, iniciando pela configuração do ambiente, passando pelas etapas de processamento digital, aprendizado de máquina, desenvolvimento da interface, desenvolvimento da API Rest, finalizando com o relatório.

Etapa 1 - Preparação do ambiente

Para o desenvolvimento e testes na máquina local, foi necessário instalar as principais ferramentas listadas abaixo:

- IDE Pycharm Community
- Python 3.9
- OpenCV 4.5
- FastAPI 0.75.2
- Flutter 3.0.1
- TensorFlow 2.9.0

Etapa 2 - Criação de testes respondidos

Nesta etapa, foram impressos em folha A4 algumas cópias do teste de setas que foram respondidos da seguinte maneira: 3 testes tiveram todas as setas da matriz de respostas marcadas, 3 testes tiveram algumas setas da matriz de respostas marcadas e 1 teste não teve nenhuma seta da matriz de respostas marcada. Buscou-se sempre variar a maneira que as setas eram marcadas. Após isso, esses testes foram escaneados em colorido por uma impressora EPSON L4160, com 300 ppi de resolução, gerando imagens de 2500 x 3500 pixels, aproximadamente. Essas imagens serão utilizadas para auxiliar o desenvolvimento do sistema e validar o funcionamento do mesmo em vários aspectos.

Etapa 3 - Pré Processamento

Esta etapa envolve passos como a aplicação de filtros para remoção de ruídos, correção de distorções ocasionados pelos sensores de captura, binarização da imagem e redução dos canais de cores da matriz da imagem, convertendo-a de RGB (Red, Green, Blue) para escala de cinza, obtendo assim apenas o canal monocromático.

A etapa de pré-processamento foi de fundamental importância para garantir a eficácia da etapa de segmentação. O primeiro processo desta etapa foi converter a imagem de modo RGB para o modo de escala de cinza, reduzindo-a de três canais de cores para um, tendo em vista que as cores são dispensáveis no contexto da aplicação, e a redução de canais reduz o gasto de recursos computacionais. Através da aplicação do filtro *median blur*, foi possível suavizar marcas indesejáveis que atrapalharam o processo de detecção dos limites de cada linha e cada seta durante a etapa de segmentação, que será explicada adiante. O último processo desta etapa foi a binarização da imagem, através dela a imagem ficou com apenas dois valores de pixel possíveis, que foram 0 e 255, com o 0 correspondendo ao pixel preto e o 255 correspondendo ao pixel branco, o que facilitou a determinação de limites durante o processo de segmentação. A figura 17 listada abaixo, demonstra um exemplo de imagem antes e após aplicação dos filtros.

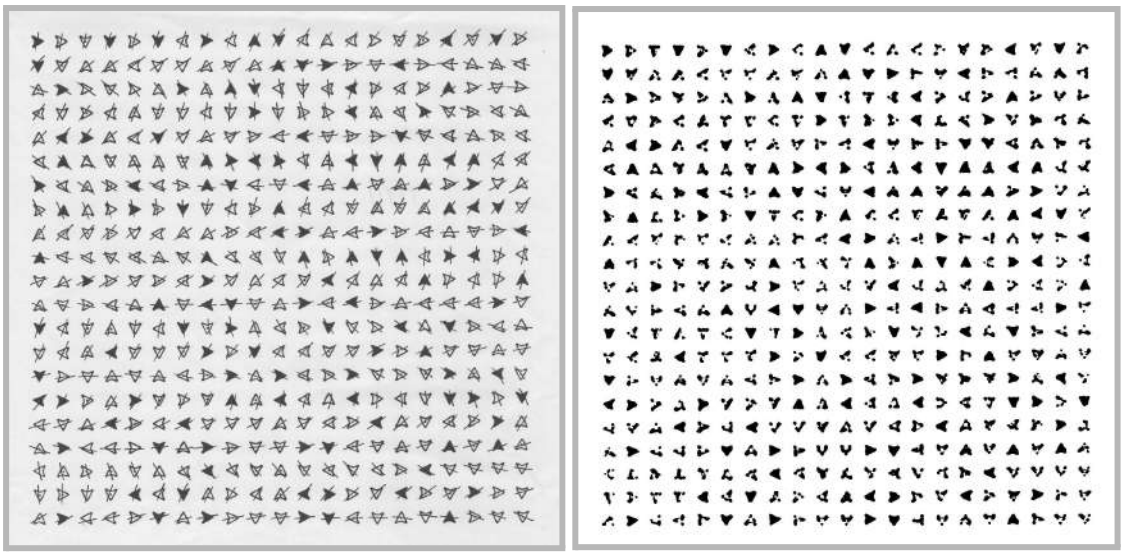


Figura 17 - Matriz de respostas sem modificações e com a aplicação de filtros, respectivamente. (Fonte: própria)

Etapa 4 - Segmentação das Setas

Nesta etapa, foi desenvolvido o algoritmo capaz de separar cada seta na matriz de respostas. Primeiro obtêm-se as coordenadas no eixo das ordenadas que representam os limites superiores e inferiores de cada linha da matriz de respostas.

Para obter esses limites, o algoritmo percorre linha a linha da imagem, de cima para baixo, à procura da primeira linha que contenha um pixel preto, iniciando pela primeira linha abaixo das setas-alvo. Encontrando-a, sua coordenada é salva para ser utilizada como limite superior da linha de setas. Após isso, continua-se percorrendo linha a linha, de cima para baixo, mas agora procurando uma linha que não contenha nenhum pixel preto. Encontrando-a, sua coordenada é salva para ser utilizada como limite inferior da linha de setas.

Para as linhas de setas subsequentes à primeira, a estratégia é a mesma, apenas utiliza-se o limite inferior da linha de setas anterior como coordenada inicial de procura do limite superior da próxima linha de setas. Segue um exemplo de matriz de respostas com os limites superiores e inferiores de cada linha de setas marcados na própria matriz de respostas.

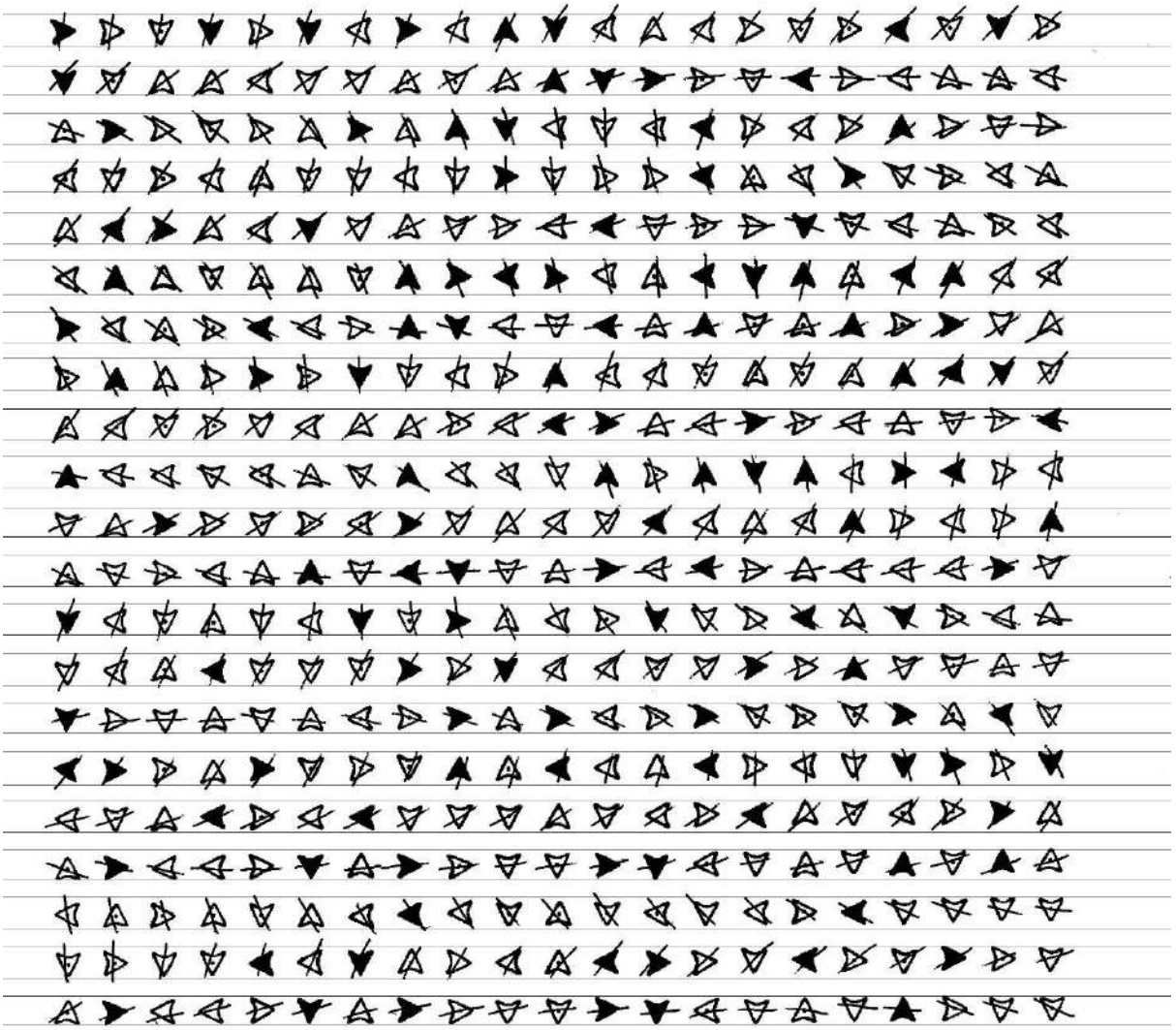


Figura 18 - Matriz de respostas com limites das linhas de setas demarcadas. (Fonte: própria)

Observando a imagem, é possível concluir que as linhas de setas não podem ser recortadas nos limites obtidos, pois assim haveria perda de informação importante para a posterior classificação das setas. Para contornar esse problema, para cada linha de setas, recorta-se a matriz de respostas na média entre o limite superior da linha de setas atual e o limite inferior da linha de setas anterior, e na média entre o limite inferior da linha de setas atual e o limite superior da próxima linha de setas, produzindo um recorte semelhante ao seguinte em todas as linhas de setas:

Figura 19 - Linha recortada corretamente. (Fonte: própria)

Com a linha de setas devidamente recortada, o algoritmo percorre coluna a coluna da linha de setas recortada, da esquerda para a direita, procurando pela primeira coluna que contenha um pixel preto, iniciando na ponta esquerda. Encontrando-a, sua coordenada é salva para ser utilizada como limite esquerdo da seta. Após isso, continua-se percorrendo coluna a coluna, da esquerda para a direita, mas agora procurando uma coluna que não contenha nenhum pixel preto. Encontrando-a, sua coordenada é salva para ser utilizada como limite direito da seta. Para as setas subsequentes à primeira, a estratégia é a mesma, apenas utiliza-se o limite direito da seta anterior como coordenada inicial de procura do limite esquerdo da próxima seta.

Figura 20 - ROI da seta identificado e delimitado. (Fonte: própria)

Da mesma forma que as linhas de setas, as setas não podem ser recortadas nos limites obtidos, pois assim haveria perda de informação importante para a

posterior classificação das setas. Para contornar esse problema, para cada seta, recorta-se a linha da seta na média entre o limite esquerdo da seta atual e o limite direito da seta anterior, e na média entre o limite direito da seta atual e o limite esquerdo da próxima seta, produzindo um recorte semelhante ao seguinte em todas as setas:

Figura 21 - Seta cortada corretamente. (Fonte: própria)

Etapa 5 - Modelagem da Rede Neural

O modelo de rede neural mais utilizado e indicado para classificar e reconhecer características em imagens são as CNNs ou Redes Neurais Convolucionais, conforme explicitado no capítulo 2. Nesta etapa, foi desenvolvido um modelo de CNN que pode ser representado pela arquitetura demonstrada na imagem abaixo:

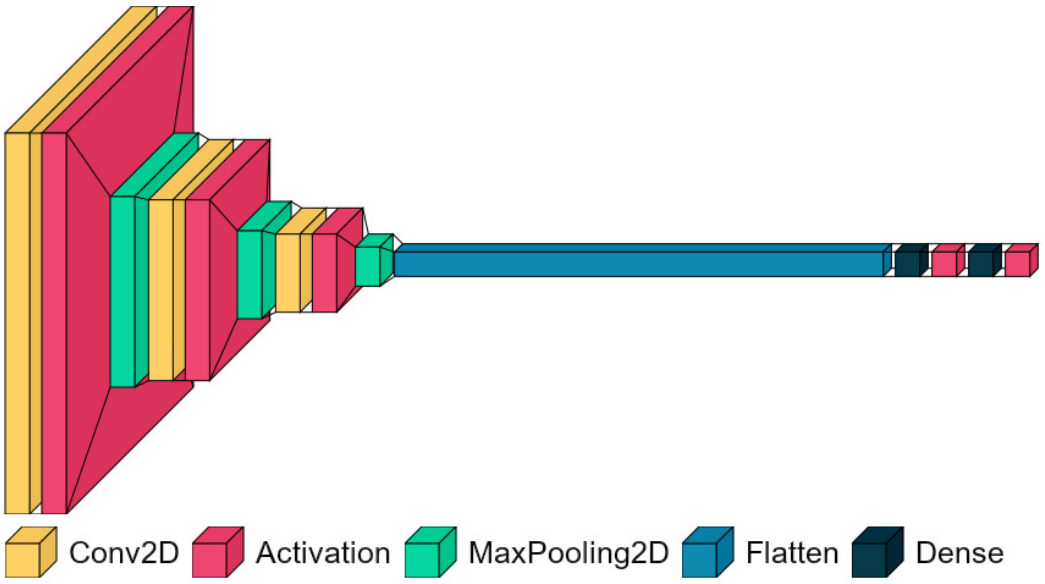


Figura 22 - Arquitetura da rede neural desenvolvida. (Fonte: própria)

A Tabela 2 representa a descrição do modelo de rede treinado com os respectivos parâmetros de saída de cada camada e as dimensões da imagem em cada camada.

Layer	Output Shape	Parameters
Conv2d (Conv2D)	(None, 78, 78, 32)	320
Activation (Activation)	(None, 78, 78, 32)	0
MaxPooling2D	(None, 39, 39, 32)	0
conv2d_1 (Conv2D)	(None, 37, 37, 32)	9248
activation_1 (Activation)	(None, 37, 37, 32)	0
max_pooling2d_1	(None, 18, 18, 32)	0
conv2d_2 (Conv2D)	(None, 16, 16, 64)	18496
activation_2 (Activation)	(None, 16, 16, 64)	0
max_pooling2d_2	(None, 8, 8, 64)	
flatten (Flatten)	(None, 4096)	0
dense (Dense)	(None, 64)	262208
activation_3 (Activation)	(None, 64)	0
dense_1 (Dense)	(None, 2)	130
activation_4 (Activation)	(None, 2)	0

Tabela 2 - Descrição das camadas da rede neural

Etapa 6 - Criação do dataset e treinamento

Nesta etapa, foi escolhida uma das imagens da etapa 2 que tem todas as setas marcadas e, com o auxílio do processo de segmentação desenvolvido na etapa 4,

todas as setas desse teste foram salvas localmente como imagens individuais. Das 441 setas obtidas, 265 setas ou aproximadamente 60% do total de setas obtidas foram separadas para serem utilizadas como treino da rede neural, 88 setas ou aproximadamente 20% do total de setas obtidas foram separadas para serem utilizadas como teste no treino da rede neural, e 88 setas ou aproximadamente 20% do total de setas obtidas foram separadas para serem utilizadas como validação do treinamento da rede neural.

Este processo de extração das setas e separação em três conjuntos foi repetido para a imagem em que nenhuma seta foi marcada, obtida na etapa 2. Dessa maneira, foi construído o dataset para o treino supervisionado da rede neural. Segue a visualização em árvore da estrutura de pastas resultante:

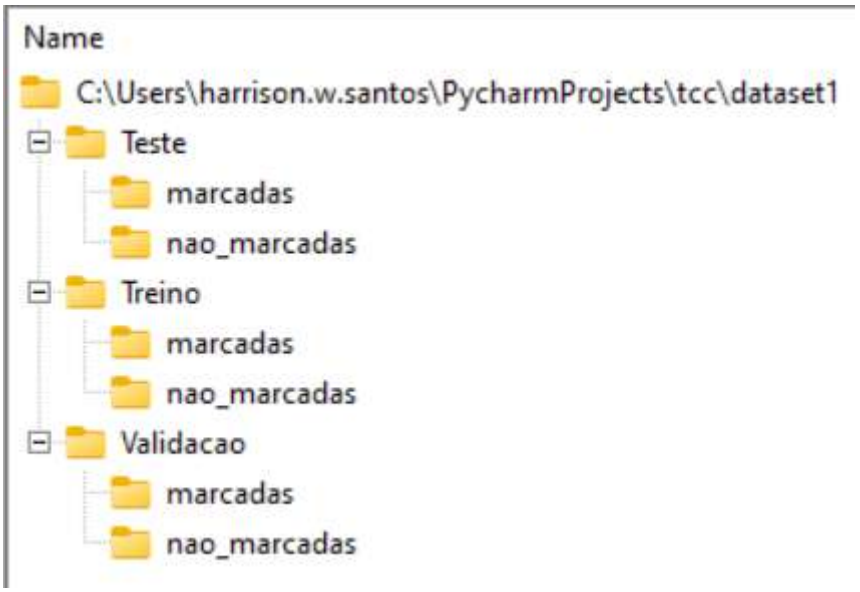


Figura 23 - Estrutura de pastas do dataset. (Fonte: própria)

Após isso, foi feita a compilação do modelo e o treinamento da rede neural com o dataset construído, utilizando o otimizador *rmsprop*, função de loss *binary_crossentropy* e 10 epochs de treinamento.

Etapa 7 - Submissão das setas ao modelo

Nesta etapa, cada seta, extraída da imagem original através da segmentação desenvolvida na etapa 4, é submetida a rede neural treinada na etapa anterior, e a

rede, por sua vez, proporciona uma saída binária que representa sua classificação, sendo uma saída de valor 0 uma imagem classificada como não marcada e uma saída de valor 1 uma imagem classificada como marcada. Ao submeter todas as setas da imagem a rede e obter a classificação de cada, foi possível obter uma matriz que contém a classificação da rede neural de todas as setas.

Etapa 8 - Criação do relatório

Nesta etapa, com auxílio dos avanços obtidos na etapa anterior, foi obtida uma matriz-alvo, que é uma matriz de zeros e uns que representa as classificações de setas de um teste respondido perfeitamente, sendo o 0 uma seta que não deve estar marcada e o 1 uma seta que deve estar marcada. Agora, durante o processo de segmentação das setas da imagem, a classificação de cada seta, obtida da rede neural, é comparada a classificação da seta de mesma posição da matriz-alvo.

Se a classificação obtida for igual a esperada, considera-se um acerto. Se o esperado for uma seta não marcada e a classificação obtida for marcada, considera-se um erro. Se o esperado for uma seta marcada e a classificação obtida for não marcada, considera-se omissão.

Considerando as métricas quantitativas utilizadas pelos profissionais de saúde, o sistema fornece como saída os dados quantitativos dos erros, acertos e omissões, conforme explicado no capítulo dois. Outra possibilidade fornecida através do relatório é o cálculo das métricas propostas para cada linha de setas, possibilitando que o profissional faça uma análise individual das métricas quantitativas de cada linha de setas.

Etapa 9 - Desenvolvimento da API

Nesta etapa, foi desenvolvida uma API Rest no framework FastAPI, que será responsável por receber a imagem que será enviada pela aplicação do usuário através de um método POST, invocar o algoritmo de correção passando a imagem recebida como entrada a este, e devolver o relatório quantitativo da correção obtido na saída do algoritmo na *response* da chamada.

Etapa 10 - Desenvolvimento da Interface

Nesta etapa, foi desenvolvida uma aplicação do tipo *front-end,* que tem como objetivo disponibilizar uma interface amigável para que o usuário final seja capaz de submeter imagens de testes para serem processadas e logo em seguida obterem o resultado quantitativo da correção.

Esta aplicação foi desenvolvida com o framework Flutter, e através dele foi possível gerar a aplicação para sistemas operacionais Windows, Android, IOS e ambiente web, através do navegador.

3.4. LIMITAÇÕES E AMEAÇAS

Uma estratégia que foi inicialmente adotada para atuar no processo de correção era classificar o tipo de cada seta da matriz de respostas, através de uma rede neural, e comparar com o tipo das setas presentes na região do gabarito, para concluir se determinada seta deve estar marcada ou não. No entanto, esta estratégia se mostrou ineficiente, pois foram detectados dois tipos de setas que a rede não conseguiu distinguir eficientemente um ao outro: setas marcadas que tinham ponto no centro e setas marcadas que não tinham ponto no centro, pois a reta desenhada, muitas vezes, passava no centro do ponto, impossibilitando a detecção dessa característica. As setas abaixo foram exemplos de setas que a rede obteve confusão ao realizar a classificação:

Figura 24 - Exemplos de casos de marcações problemáticas. (Fonte: própria)

Sendo assim, foi necessário modificar a estratégia, modificando a rede para que se esta se tornasse um classificador binário, em que deve fornecer como resultado a condição da seta, se está marcada ou não. Desta forma, se fez necessário colocar como entrada uma imagem de um gabarito oficial passado como

referência, e a partir da identificação das respostas deste, foi possível corrigir outros testes comparando com outros testes de entrada.

No tocante ao desenvolvimento do front-end, a maior dificuldade encontrada foi no envio dos arquivos de imagem para a API Rest, visto que há uma diferença no formato de upload nos ambientes Android e Windows. Para contornar este problema, foi utilizado um método de upload e busca de arquivos diferentes para cada ambiente.

4. RESULTADOS

Neste capítulo serão apresentados os resultados obtidos com o desenvolvimento deste projeto considerando os objetivos propostos, com ênfase nos resultados obtidos na etapa de treinamento da rede neural, modelada para classificação das imagens propostas .

Treinamento da Rede Neural

Os resultados do treinamento da rede neural utilizada para classificação das imagens será demonstrado a seguir, possibilitando analisar o comportamento mediante modificações na base de dados utilizados para treinamento.

Nos gráficos apresentados abaixo, foi considerado o dataset sem augmentação de dados, correspondente a seguinte distribuição:

	Sem augmentação	
	Marcada	Não marcada
Treino	445	445
Teste	88	88
Validação	88	88

Tabela 3 - Descrição do dataset sem aumento de dados

Através dos gráficos abaixo, é possível perceber que a função de perda durante o treinamento(loss) e a função de perda durante a validação(val_loss) permaneçam bastante próximas e isto é uma evidência de que o modelo não está apenas memorizando os dados de treinamento, mas aprendendo as propriedades gerais das duas classes. Neste teste, foram utilizados dez epochs para o treinamento.

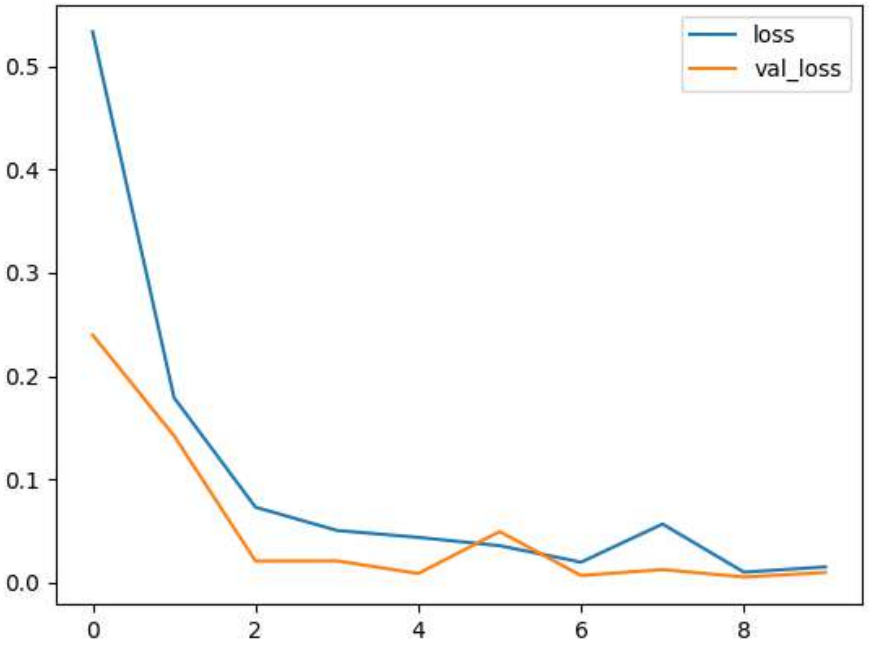


Figura 25 - Gráfico de comparação da função de perda durante o treinamento e validação.

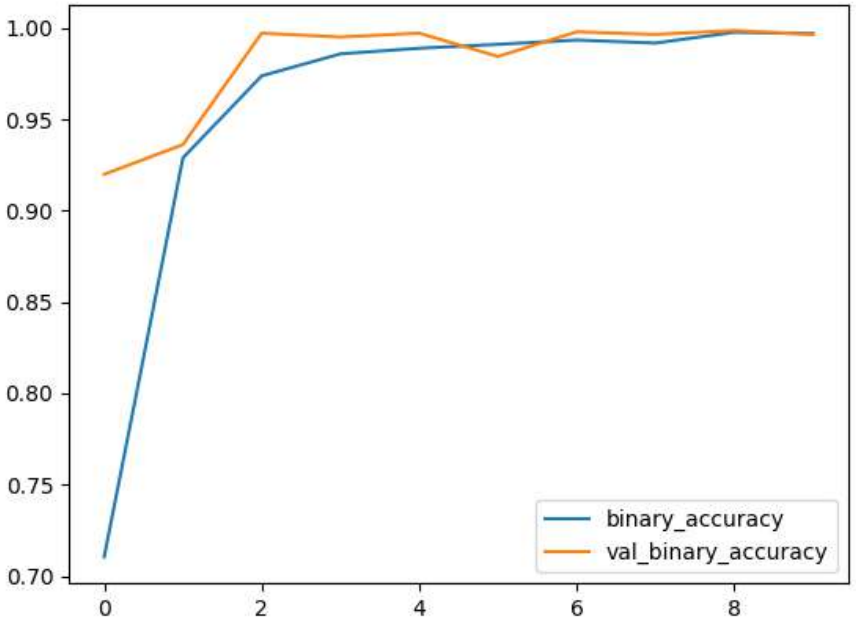


Figura 26 - Gráfico de comparação da função de acurácia durante o treinamento e validação.

Para desenvolver um novo dataset mais generalizado mas utilizando as imagens do dataset anterior, cada imagem de seta do dataset anterior foi descentralizada em diferentes direções e rotacionada em diferentes ângulos, gerando uma grande quantidade de imagens ao final dessa estratégia de augmentation. O dataset resultante teve a seguinte distribuição:

	Com augmentation	
	Marcada	Não marcado
Treino	38.102	38.102
Teste	12.702	12.702
Validação	12.700	12.700

Tabela 4 - Descrição do dataset com augmentation

Ao treinar a rede neural com esse dataset, foram obtidos os gráficos abaixo, onde é possível perceber que a função de perda durante o treinamento(loss) e a função de perda durante validação(val loss) estão descorrelacionados desde o instante inicial, isto é uma evidência de que o modelo pode estar memorizando os dados de treinamento e não aprendendo as propriedades gerais das duas classes. Neste teste, a rede não está usando seu potencial de generalidade, mas somente atribuindo um rótulo baseado em alguma imagem que foi usada na base de dados de treinamento, de forma a ocasionar o que chamamos de overfit.

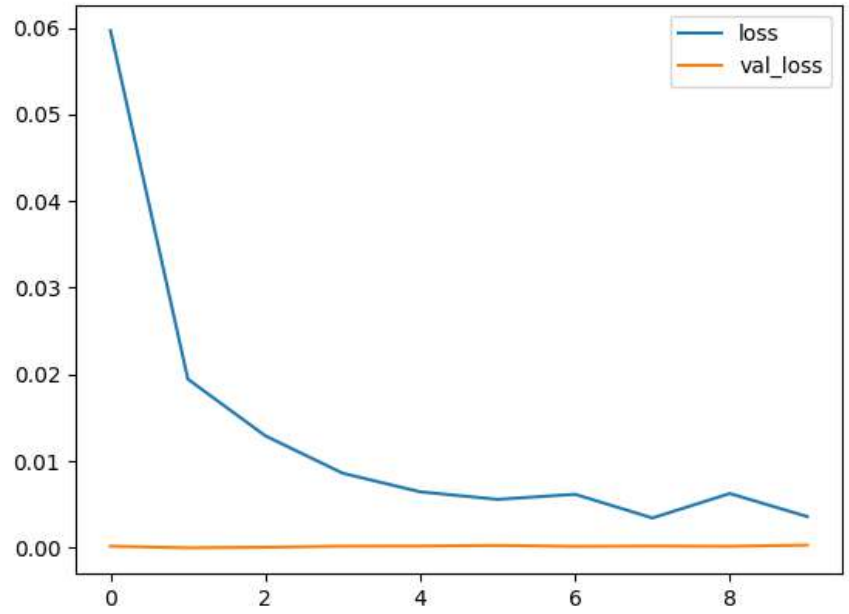


Figura 27 - Gráfico de comparação da função de perda com overfit.

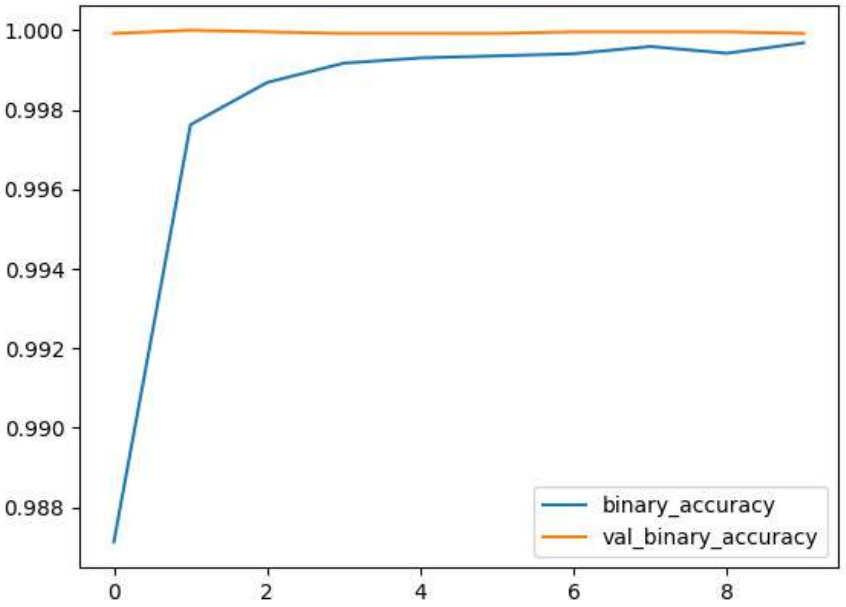


Figura 28 - Gráfico de comparação da função de acurácia com overfit.

Ferramenta Mobile e Web

Como resultado do desenvolvimento da interface do usuário, obtivemos uma interface simplificada que permite seu funcionamento em ambiente Android e Web. A interface possui as telas Histórico, Home, Correção e Resultado. A funcionalidade de cada tela está descrita abaixo assim como estarão visíveis nas figuras 29, 30, 31 e 32.

- Home (figura 29): esta página/tela é o centro da plataforma, através dela é possível acessar histórico, nova correção e perfil.
- Nova Correção(figura 30): nesta página/tela é responsável receber os dados e imagens dos testes que serão corrigidos.
- Resultado(figura 31): esta página/tela exibirá o resultado da correção efetuada.
- Histórico(figura 32): nesta página/tela são armazenadas todas as correções já efetuadas.

Automatização e Correção Quantitativa do Teste de Setas

A correção foi realizada utilizando o modelo de rede neural proposto e todo o processo pode ser realizado através da interface desenvolvida garantindo a automatização e correção quantitativa dos testes. A acurácia da rede neural para a classificação foi superior a 90%, garantindo confiabilidade e assertividade.

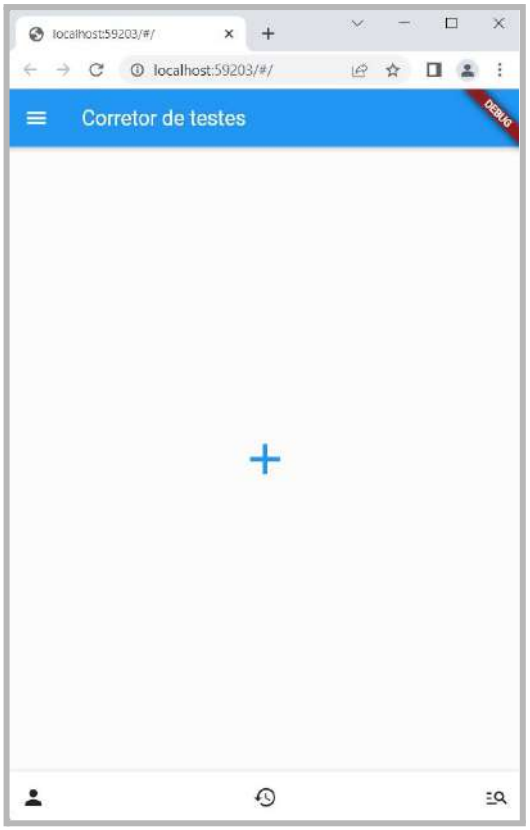


Figura 29 - Tela Home

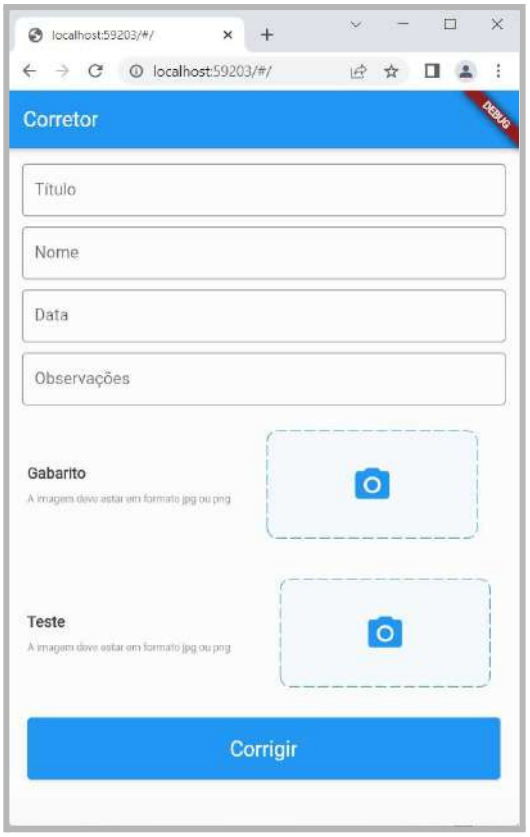


Figura 30- Tela de correção

Figura 31 - Tela de resultado

Figura 32 - Tela de histórico

5. CONSIDERAÇÕES FINAIS

A visão computacional possui um conjunto de ferramentas amplo e sofisticado que permitem o desenvolvimento de tecnologias de alto desempenho e eficácia. Aliado a inteligência artificial, mais especificamente ao aprendizado de máquina, torna possível automatizar e melhorar muitos procedimentos que antes eram feitos de forma manual, como é o caso da correção do teste de setas.

Neste projeto, a utilização das técnicas de visão computacional e aprendizado de máquina mostraram resultados efetivos na correção do teste de setas, obtendo métricas de acurácia e *loss* bastante satisfatórias durante o treinamento e a validação do modelo de rede neural desenvolvido.

O modelo de rede neural treinado para a resolução deste problema foi do tipo CNN e as camadas propostas podem sofrer futuras alterações, assim como as funções de otimização. Foi possível observar a alteração do comportamento da rede de acordo com o conjunto de dados ao aplicar técnica de augmentation.

No desenvolvimento da interface da aplicação algumas dificuldades foram encontradas em relação a experiência do usuário, então foi desenvolvido somente o essencial para comunicação entre a interface do usuário e o back-end e/ou processamento.

A perspectiva futura deste projeto é tornar esta aplicação mais amigável ao usuário, através da aplicação de técnicas de User Experience e User interface(UI/UX). Logo em seguida, realizar mais testes com os profissionais de saúde da área e fazer as alterações conforme necessário. Além disso, outra perspectiva é buscar por formas de otimização do modelo de rede desenvolvido visando redução do tempo de correção.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[1] GONZALEZ, Rafael C.; WOODS, Richard E. **Processamento de imagens digitais**. Editora Blucher, 2000.

[2] ANDRADE, I. E., Albuquerque, M. P. e Albuquerque, M. P., **Processamento Digital de Imagens**, 2003. Disponível em: <https://www.cbpf.br/cat/pdsi/pdf/cap3webfinal.pdf>. Acesso em: 15 Julho 2022

[3] DE QUEIROZ, José Eustáquio Rangel; GOMES, Herman Martins. **Introdução ao processamento digital de imagens**. Rita, v. 13, n. 2, p. 11-42, 2006. Acesso em: 14 Julho 2022.

[4] ROBERT, Kanasz, **Single Layer Perceptron as Linear Classifier**, 2010. Disponível em: </www.codeproject.com/Articles/125346/Single-Layer-Perceptron-as-Linear-Classifier>. Acesso em: 15 Julho 2022.

[5] CARDON, André; MÜLLER, Daniel Nehme. **Introdução às Redes Neurais Artificiais**. Instituto de Informática / UFRGS: [s.n.], 1994. Disponível em: <http://docplayer.com.br/36890934-Introducao-as-redes-neurais-articiais.html>. Acesso em: 15 Julho 2022.

[6] NOLETO, Cairo. **API REST: o que é e como montar uma API sem complicação?**. 2022. Disponível em: <https://blog.betrybe.com/desenvolvimento-web/api-rest-tudo-sobre/#1>. Acesso em: 15 Julho 2022.

[7] NAEEM, Tehreem. **Definição de API REST: Noções básicas de APIs REST**. 2020. Disponível em: <https://www.astera.com/pt/tipo/blog/definição-de-api/>. Acesso em 15 Julho 2022

[8] RAJWAL, Swati. **Classification of Handwritten Digits Using CNN**. 2021. Disponível em: <https://www.analyticsvidhya.com/blog/2021/07/classification-of-handwritten-digits-using-cnn/> Acesso em: 17 Julho 2022

[9] BENTO, Caroline. **Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis**. 2021. Disponível em: <https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141> Acesso em: 17 Julho 2022

[10] DOS SANTOS, Valéria Nunes. **Reconhecimento de um Objeto usando Redes Neurais Convulsionais**. Pdf, UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ DEPARTAMENTO ACADÊMICO DE INFORMÁTICA, 2018.

[11] **Uma plataforma completa de código aberto para machine learning**, TensorFlow, 2022.
Disponível em: <https://www.tensorflow.org>
Acesso em: 30 Julho 2022.

[12] MELO, Natã. **Abordagens do processo de Segmentação: Limiarização, Orientada a Regiões e Baseada em Bordas.** 2020.
Disponível em:
<http://www.dsc.ufcg.edu.br/~pet/jornal/setembro2011/materias/recapitulando.html>
Acesso em: 30 Julho 2022

[13] **Palográfico - SKIP - Sistema de Correção Informatizada do Palográfico.** Mkt Vetoreditora, 2020. Disponível em:
<https://www.psicovita.com.br/loja/palografico-skip-sistema-de-correcao-informatizada-do-palografico>
Acesso em: 08 Setembro 2022

[14] **Plataforma de Correção Online.** Psicovita, 2020. Disponível em:
<http://www.mktvetoreditora.com.br/plataforma-correcao-online.php>
Acesso em: 08 Setembro 2022

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Campina Grande
R. Tranqüilino Coelho Lemos, 671, Dinamérica, CEP 58432-300, Campina Grande (PB)
CNPJ: 10.783.898/0003-37 - Telefone: (83) 2102.6200

Documento Digitalizado Ostensivo (Público)

Versão Final do TCC

Assunto: Versão Final do TCC
Assinado por: Claudio Alberto
Tipo do Documento: Projeto
Situação: Finalizado
Nível de Acesso: Ostensivo (Público)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Claudio Alberto de Sousa Bezerra, ALUNO (201711250018) DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO - CAMPINA GRANDE**, em 26/09/2022 18:44:27.

Este documento foi armazenado no SUAP em 26/09/2022. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 635180
Código de Autenticação: e7327c8860