UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

LIN YU HAN

RECUPERAÇÃO DA TRAJETÓRIA *ONLINE* DE CARACTERES LATINOS UTILIZANDO *DEEP LEARNING*

DISSERTAÇÃO

PONTA GROSSA

2020

LIN YU HAN

RECUPERAÇÃO DA TRAJETÓRIA *ONLINE* DE CARACTERES LATINOS UTILIZANDO *DEEP LEARNING*

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná.

Orientador: Dr. Erikson Freitas de Morais
Coorientador: Dra. Simone B. K. Aires

PONTA GROSSA
2020

Ficha catalográfica elaborada pelo Departamento de Biblioteca
da Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa
n.60/20

L735 Lin, Yu Han

Recuperação da trajetória online de caracteres latinos utilizando *deep learning*. / Lin Yu Han, 2020.
93 f. : il. ; 30 cm.

Orientador: Prof. Dr. Erikson Freitas de Morais
Coorientadora: Profa. Dra. Simone Bello Kaminski Aires

Dissertação (Mestrado em Ciência da Computação) - Programa de Pós-Graduação em Ciência da Computação. Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2020.

1. Manuscritos. 2. Escrita - Identificação. 3. Conjunto de caracteres (Processamento de dados). 4. Redes neurais (Computação). 5. Sistemas de reescrita (Computação). I. Morais, Erikson Freitas de. II. Aires, Simone Bello Kaminski. III. Universidade Tecnológica Federal do Paraná. IV. Título.

CDD 004

Elson Heraldo Ribeiro Junior. CRB-9/1413. 09/09/2020.

Ministério da Educação
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ CÂMPUS PONTA GROSSA
Diretoria de Pesquisa e Pós-Graduação
Programa de Pós-Graduação em Ciência da Computação

FOLHA DE APROVAÇÃO

Título de Dissertação Nº 24/2020

RECUPERAÇÃO DA TRAJETÓRIA ONLINE DE CARACTERES LATINOS UTILIZANDO DEEP LEARNING

Por

Lin Yu Han

Esta dissertação foi apresentada às **14:30 horas** de **25 de Agosto de 2020**, na sala de Videoconferência Online, como requisito parcial para a obtenção do título de MESTRE EM CIÊNCIA DA COMPUTAÇÃO, Programa de Pós-Graduação em Ciência da Computação. O candidato foi arguido pela Banca Examinadora, composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho APROVADO.

Prof. Dr. Alceu de Souza Britto Junior (PUC)

Prof. Dr. Cinthia Obladen de Almendra Freitas (PUC)

Prof. Dr. Helyane Bronoski Borges (UTFPR)

Prof. Dr. Erikson Freitas de Morais (UTFPR)
Orientador e presidente da banca

Universidade Tecnológica Federal do Paraná
Campus Ponta Grossa

Visto do Coordenador:

Prof. Dr. André Koscianski
Coordenador do PPGCC
UTFPR – Câmpus Ponta Grossa

Aos meus pais e amigos.

AGRADECIMENTOS

Agradecimentos aos meus pais, que sempre me apoiaram em tudo o que fiz. Com sua imensa sabedoria me ensinaram em como ser uma pessoa correta para alcançar meus objetivos sem prejudicar os outros. Me fizeram do que sou hoje e graças a eles consigo enfrentar e superar minhas dificuldades não apenas com conhecimento, mas também com sabedoria. Obrigado aos meus amigos da marcos, que sempre me apoiaram e proporcionaram conversas que alegram os meus dias. Obrigado aos meus amigos e colegas de classe do Sociais PPGCC, que enfrentaram juntos comigo os altos e baixos da pós-graduação. Obrigado a meus orientadores pela guia e paciência para que eu pudesse finalizar esse trabalho. Obrigado a UTFPR, que me graduou e me acolheu na pós-graduação. Obrigado a todos que sempre acreditaram em mim.

“A verdadeira força de nós, seres humanos,
é a capacidade de poder mudar nós mesmos
por nossa própria vontade.”
Saitama - One Punch-Man

RESUMO

LIN, Yu Han. *Recuperação da Trajetória Online de Caracteres Latinos utilizando Deep Learning. 2020.* Dissertação de Mestrado em Ciência da Computação, Universidade Tecnológica Federal Do Paraná. Ponta Grossa, 2020.

Estudos sobre a recuperação de trajetória de manuscritos ganharam espaço na área da pesquisa de reconhecimento de textos manuscritos *offline*. O motivo está no uso de recursos de reconhecimento online, criando técnicas para simular a escrita da palavra manuscrita e inserindo as coordenadas simuladas dos pixels em sistemas que reconhecem palavras de modo *online*. O princípio dessas técnicas é encontrar um traçado ordenado de modo similar àquele feito por uma pessoa durante a escrita; esse processo é conhecido como recuperação da trajetória de manuscritos (*handwriting trajectory recovery* - HTR). Vários trabalhos apresentaram o uso de grafos para realizar a HTR, esqueletizando os caracteres e traçando o caminho correto do grafo, sendo este grafo o esqueleto do caractere. Entretanto, estudos recentes caracterizam o uso das redes neurais artificiais de aprendizagem profunda (*deep learning*) para realizar a HTR. A vantagem de se utilizar as redes de *deep learning* é usufruir da sua capacidade de generalização para atingir taxas de acertos melhores na recuperação dessas trajetórias. Apesar destes trabalhos apresentarem resultados promissores, seus resultados são dificilmente comparados entre si, uma vez que utilizam métricas de avaliação diferentes uma das outras. Baseado nesses fatores, o presente trabalho tem como objetivo apresentar uma proposta para realizar a HTR de caracteres latinos do *dataset* IRONOFF através do uso de redes *deep learning*, além de apresentar um novo modelo de avaliação chamada de SWO (*Segmentation-sliding Window-Ordering*), para avaliar a sequência de coordenadas preditas pelas redes de *deep learning*. Comparada com as métricas existentes na literatura, a avaliação SWO mostrou-se ser mais eficaz tanto quantitativamente, quanto qualitativamente, sendo capaz de verificar se a sequência dos traços e o formato geométrico do caractere foram recuperados. Este trabalho também identifica que a variação na quantidade de pontos de coordenadas por caractere (p/c) afeta no desempenho das redes de *deep learning* para o problema da HTR, evidenciando uma abordagem para melhorar as taxas de avaliação. Os experimentos são conduzidos a partir da transformação da informação *online* em *offline*. Esses dados passam pelo processo de normalização e *data augmentation*. Cinco configurações de redes são implementadas e passam pelo processo de treinamento, validação e testes. Alcança-se como melhores resultados as taxa de $95,31\%$ de acurácia na predição de trajetória de caracteres *single-stroke* e $92,93\%$ de acurácia na predição de trajetória de caracteres *multi-strokes*, apresentando dessa maneira bons resultados para o problema de HTR.

Palavras-chaves: Recuperação de Trajetória. Caracteres Manuscritos. Redes Neurais Artificiais. Aprendizagem Profunda.

ABSTRACT

LIN, Yu Han. *Online Latin Characters Trajectory Recovery using Deep Learning*. Masther's Degree Thesis in Computer Science, FUniversidade Tecnológica Federal Do Paraná. Ponta Grossa, 2020.

Studies on the recovery of trajectories manuscript have gained space in the field of offline handwriting recognition research. The reason is in the use of online recognition resources, creating techniques to simulate the writing of the handwritten word and inserting the simulated coordinates of the pixels in systems that recognize words online. The principle of these techniques is to find an orderly pattern similar to that made by a person during writing; this process is known as handwriting trajectory recovery (HTR). Several works have presented the use of graphs to perform the HTR, skeletonizing the characters and tracing the correct path of the graph, which this graph representing the skeleton of the character. However, recent studies characterize the use of deep learning artificial neural networks to realize the HTR. The great advantage of using deep learning networks is to take their generalization capacity to achieve better accuracy rates in the recovery of handwriting trajectories. Although these works present promising results, their results are difficult to compare with each other, since they use different evaluation metrics. Based on these factors, this work aims to present a proposal to perform the HTR of Latin characters of the IRONOFF dataset through the use of deep learning networks, in addition, present a new evaluation model called SWO (Segmentation-sliding Window-Ordering) to evaluate the sequence of coordinates predicted by the deep learning networks. Compared with the existing metrics in the literature, the SWO evaluation proved to be more effective in both quantitatively and qualitatively aspect, being able to verify whether the sequence of the strokes and the geometric shape of the character were recovered. This work also identifies that the variation in the number of coordinate points per character (p/c) affects the performance of deep learning networks for the problem of HTR, evidencing an approach to improve the evaluation rates. The experiments are conducted from the transformation of information online into offline. This data goes through the normalization and data augmentation process. Five network configurations are implemented and undergo the training, validation and testing process. The best results are achieved with a 95.31% accuracy rate in predicting the trajectory of single-stroke character and 92.93% accuracy in predicting the trajectory of multi-strokes character, thus presenting good results for the HTR problem.

Key-words: Trajectory Recovery. Handwritring Characters. Artificial Neural Networks. Deep Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASCII	*American Standard Code for Information Interchange*
DOR	*Drawing Order Recovery*
DTW	*Dynamic Time Warping*
CNN	*Convolutional Neural Networks*
CT	*Complete Trajectory*
GA	*Genetic Algorithm*
GC	*Graph Cut*
HTR	*Handwriting Trajectory Recovery*
IRONOFF	*The ireste on/off dual handwriting database*
JP	*Junction Point*
LSTM	*Long-Short Term Memory*
PA	Progressão Aritmética
p/c	Pontos por Caractere
RMSE	*Root Mean Square Error*
SP	*Start Point*
SP-JP-CT	Avaliação SP, JP e CT
SVM	*Support Vector Machine*
SWO	*Segmentation - sliding Window - Ordering*
TSF	*Trajectory Segmentation Factor*

SUMÁRIO

1 INTRODUÇÃO

A capacidade de produzir manuscritos é uma habilidade única de cada indivíduo e tem como objetivo transmitir alguma mensagem, pensamento ou ideia através do uso de um determinado idioma (PLAMONDON; SRIHARI, 2000). A escrita é composta por vários ícones, conhecido como caracteres ou letras, cada um com formas específicas, sendo possível combiná-los para atingir um nível linguístico mais alto, como palavras, como é o caso do alfabeto latino (AIRES, 2005). Na Figura 1 temos ilustrado três símbolos que representam os caracteres 'e', 'f' e 'g' do alfabeto latino.

Figura 1 – Exemplos de caracteres manuscritos do alfabeto latino

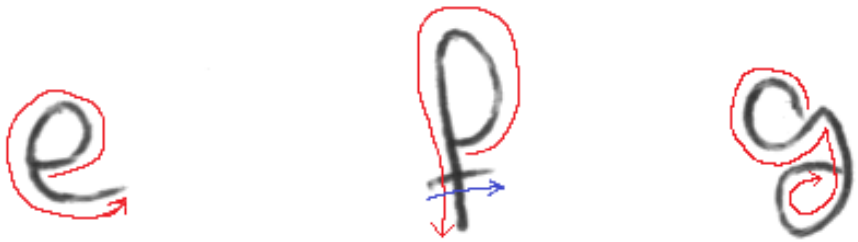
Fonte: (VIARD-GAUDIN *et al.*, 1999)

O reconhecimento de manuscritos consiste em transformar a linguagem representada em sua forma de marcas gráficas em sinais digitais de computador. Um exemplo que se pode citar de ortografias baseadas no alfabeto latino são as palavras de uma carta manuscrita, que poderiam ser escaneadas e reconhecidas em sinais digitais em ASCII de 8 bits ou Unicode em 16 bits (PLAMONDON; SRIHARI, 2000). Para automatizar essa transformação da escrita humana em sinais digitais, são criados sistemas capazes de identificar e reconhecer os manuscritos. Esses sistemas de reconhecimento de manuscritos podem ser classificados em *online* e *offline* (PLAMONDON; SRIHARI, 2000). O reconhecimento *online* é a transcrição de escritas realizada através da disponibilidade de informações dinâmicas e temporais, como a sequência dos pixeis traçados, pressão, velocidade e aceleração do traçado a partir de uma caneta eletrônica ou uma superfície digitalizadora. Já o reconhecimento *offline* é o processo de extração e identificação da escrita em textos de imagens estáticas.

O reconhecimento *offline* tem uma disponibilidade de informações limitada em comparação com o *online*, já que no *online* tem-se informações como velocidade, inclinação, pressão e sequência (NGUYEN; BLUMENSTEIN, 2010) e o *offline* possui apenas a imagem digitalizada por um scanner. Devido a essa diferença, pesquisas mostram que o reconhecimento *online* atingem resultados melhores que o reconhecimento *offline* (NOUBIGH; KHERALLAH, 2017). Um

exemplo disso é o estudo realizado por Zhang, Bengio e Liu (2017), onde pode ser encontrada uma avaliação comparativa dos métodos de reconhecimento *offline* e *online* proposto pelo autor. Isso encorajou muitos pesquisadores a tentar recuperar informações dinâmicas das imagens estáticas. Logo, a recuperação da sequência de traços é considerada uma forma alternativa de adquirir uma característica que possa melhorar a acurácia de sistemas *offline*. Na Figura 2 temos exemplos de traçados de caracteres manuscritos.

Figura 2 – Exemplos de traçado de caracteres manuscritos do alfabeto latino



Fonte: (VIARD-GAUDIN *et al.*, 1999)

Como pode ser observado na Figura 2, os caracteres podem ser traçados a partir de uma ordenação. A ordenação produzida implica que existem pontos que surgiram antes dos outros pontos.

A recuperação da sequência dos traços, ou recuperação da trajetória (*Handwriting Trajectory Recovery* - HTR), é a técnica usada para extrair informações da ordem temporal dos traçados manuscritos contidos em imagens estáticas (NOUBIGH; KHERALLAH, 2017). Com a recuperação dessa informação dinâmica, os sistemas de reconhecimento *online* podem produzir informações que podem auxiliar os sistemas de reconhecimento de manuscritos *offline*. Da mesma forma que uma pessoa possa reproduzir o traçado de um manuscrito realizado por outra pessoa, um sistema de HTR pode aprender a reproduzir também os traçados manuscritos, servindo de auxílio para os sistemas de reconhecimento *offline*.

Durante muito tempo, a trajetória de manuscritos era recuperada através do uso de técnicas heurísticas, como busca gulosa, combinando-o com algoritmos para solução do caminho mais curto de um grafo, como mostrado na pesquisa de Dinh *et al.* (2016). A estratégia é aplicar a esqueletização na imagem estática e partir disso, gerar um grafo para posteriormente utilizar algoritmos de caminhamento em grafos, seguindo os pixeis como se fossem os nós do grafo.

Trabalhos recentes apresentam o uso da *Deep Learning* (Aprendizagem Profunda) para realizar a predição da trajetória dos traçados dos caracteres (ZHAO; YANG; TAO, 2019)(ELBAATI; HAMDI; ALIMI, 2019)(BHUNIA *et al.*, 2018)(ZHANG; BENGIO; LIU, 2017). As abordagens visam utilizar principalmente as redes neurais convolucionais, chamadas de CNN

(*Convolutional Neural Network*), para realizar a extração de características do caractere de imagens *offline* e a partir disso realizar a predição dos pixels ordenados.

Para recuperar o traçado de caracteres chineses, Zhao, Yang e Tao (2019) e Wang *et al.* (2019) utilizam a CNN para extrair as características dos pixels das imagens dos caracteres e, a partir disso, utilizam um algoritmo para interpretar essas características, resultando numa ordenação de sequência de coordenadas. Bhunia *et al.* (2018), utiliza uma combinação da rede CNN e uma rede LSTM (*Long-Short Term Memory*) para recuperar o traçado de caracteres manuscritos telugos. Elbaati, Hamdi e Alimi (2019) também utiliza a combinação da CNN e LSTM para predizer as informações de coordenadas *online* de caracteres latinos de uma imagem; essas coordenadas são interpretadas por uma rede neural classificadora de caracteres *online* para serem avaliadas.

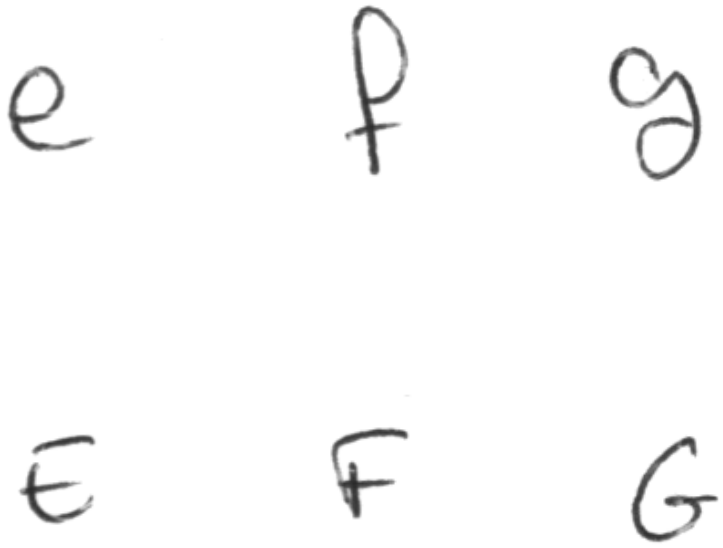
Os trabalhos apresentam semelhanças com relação ao modo de predição: utilização de redes *deep learning*. Entretanto, um assunto que não aparece com relevância na literatura, porém bastante significativo para os sistemas de HTR, é com relação ao modo de avaliação dos resultados.

No processo de reconhecimento de caracteres, os resultados são avaliados através da comparação direta da classificação obtida pelo algoritmo com o conjunto verdade (*grond-truth*). Por exemplo, se uma rede neural classificou a amostra com o rótulo equivalente ao rótulo do conjunto verdade da amostra, aumenta-se a taxa de acerto, caso contrário, aumenta-se a taxa de erro. Esse procedimento não pode ser facilmente adotado para o problema de HTR utilizando redes de *deep learning*. Os autores afirmam que a predição realizada pelas redes de aprendizagem profunda não geram coordenadas exatas (BHUNIA *et al.*, 2018)(ZHAO; YANG; TAO, 2019)(ELBAATI; HAMDI; ALIMI, 2019), ou seja, geralmente, os pixels preditos desviam levemente da posição original em comparação ao pixels do conjunto verdade.

Em razão disso, muitas vezes os autores utilizam métricas próprias (como Bhunia *et al.* (2018) e Zhao, Yang e Tao (2019)), ou utilizam coeficientes estatísticos (como Zhang, Bengio e Liu (2017)) para avaliar seus sistemas. O problema da falta de padronização do método avaliativo é a impossibilidade de se realizar comparações dos resultados entre os trabalhos. Os resultados dos coeficientes estatísticos podem ser comparados, mas necessitam de um limiar (valor de corte) para dizer se as coordenadas preditas estão corretas ou erradas em comparação ao conjunto verdade. Este limiar também não é padronizado. Isso mostra que tanto a predição de coordenadas da trajetória dos caracteres quanto a avaliação dessas coordenadas são problemas que ainda precisam ser exploradas e aprofundadas.

Baseado nisso, este trabalho visa apresentar uma abordagem de HTR utilizando *deep learning* em um léxico limitado ao alfabeto de caracteres minúsculos latinos da base IRON-FOFF, gerando informações dinâmicas que possam auxiliar no processo de reconhecimento de manuscritos *offline*. A escolha do uso de apenas caracteres minúsculos é pelo fado de que os caracteres maiúsculos (apesar de corresponderem aos mesmos rótulos) possuem escritas e formatos geométricos diferentes das minúsculas, como mostrado na Figura 3.

Figura 3 – Exemplos de caracteres minúsculos e maiúsculos do alfabeto latino



Fonte: (VIARD-GAUDIN *et al.*, 1999)

Na Figura 3 os caracteres 'e' e 'E' possuem o mesmo valor simbólico, entretanto, seus formatos geométricos e formas de traçados são bem diferentes. Devido a isso, trabalhou-se neste escopo menor de caracteres.

Com os processos de normalização no pré-processamento de dados, consegue-se construir bases com 20 pontos por caractere (p/c), $30p/c$, $40p/c$, $50p/c$, $60p/c$, $70p/c$ e $100p/c$, para realizar uma análise do impacto que a quantidade de pontos tem sobre a aprendizagem das redes de *deep learning*. Além disso, esta pesquisa também apresenta um modelo de avaliação chamado de SWO (*Segmentation - sliding Window - Ordering*) para avaliar as coordenadas preditas de sistemas de HTR, com o objetivo de conseguir alcançar uma padronização da avaliação dos resultados obtidos por esse tipo de sistema.

1.1 OBJETIVOS

O principal objetivo deste trabalho é desenvolver uma metodologia para a recuperação de trajetória do traçado de caracteres manuscritos latinos *offline* usando *Deep Learning*.

Para alcançar o objetivo principal e validar seu correto funcionamento, é necessário cumprir alguns objetivos específicos:

- Efetuar o pré-processamento da base de dados IRONOFF, normalizando e realçando in-

formações mais significativas do conjunto para contribuir na qualidade do treinamento das redes *deep learning*;

- Aumentar a quantidade de dados da base de dados utilizando a técnica de *Data Augmentation* para obter uma maior generalização das redes *deep learning*;
- Criar de diferentes bases de dados a partir da padronização de cada conjunto em diferentes quantidades de pontos por caracteres ($20p/c$, $30p/c$, $40p/c$, $50p/c$, $60p/c$, $70p/c$ e $100p/c$);
- Desenvolver redes neurais a partir da composição hierárquica de CNN e LSTM para HTR;
- Realizar o treinamento das redes com as bases de dados padronizadas;
- Analisar nos resultados obtidos o impacto da quantidade de pontos por caracteres da entrada, para se conseguir uma melhoria na acurácia da recuperação de trajetória;
- Avaliar os resultados utilizando as métricas de avaliação existentes na literatura;
- Realizar uma análise qualitativa das métricas de avaliação;
- Desenvolver um novo método de avaliação para os resultados dos sistemas de HTR;
- Apresentar um cenário de teste qualitativo como prova de conceito para avaliar e validar o método proposto, analisando os resultados obtidos;
- Realizar um comparativo quantitativo e qualitativo dos resultados obtidos entre o novo método apresentado e as métricas existentes na literatura.

Este documento está organizado conforme segue. No Capítulo 1 são abordados a introdução, objetivos gerais e os objetivos específicos. O Capítulo 2 apresenta alguns trabalhos relacionados da área de HTR. No Capítulo 3 são apresentados conceitos relacionados à reconhecimento de manuscritos, aprendizagem de máquina e redes neurais artificiais de aprendizagem profunda. A metodologia para o desenvolvimento desta pesquisa é apresentada no Capítulo 4. Os experimentos e resultados são apresentados no Capítulo 5. No Capítulo 6 apresenta-se o modelo de avaliação SWO. E por fim, o Capítulo 7 apresenta as conclusões obtidas da pesquisa.

2 TRABALHOS RELACIONADOS

Esta seção apresenta pesquisas relacionadas à utilização de técnicas para a HTR na recuperação da ordem da trajetória de manuscritos, sendo importante ressaltar que tal característica não está presente em uma imagem de manuscrito *offline*.

Durante a pesquisa dos trabalhos verificou-se que existem duas principais estratégias para a recuperação de trajetória utilizadas pelos pesquisadores; a primeira é o uso de grafos para representar a imagem do manuscrito, sendo a trajetória do manuscrito um caminho que percorre esse grafo; a segunda é o uso da inteligência artificial ou aprendizado de máquina, que aproximam a probabilidade do modelo de treinamento para reproduzir os manuscritos.

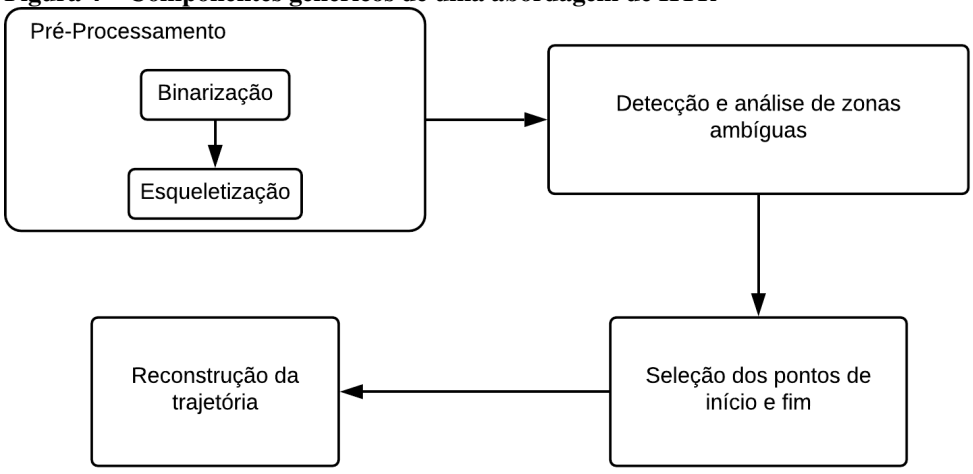
A seguir serão apresentados alguns trabalhos classificados conforme as abordagens citadas e no final do capítulo, será apresentada uma tabela com as taxas de acurácia dos trabalhos elencados e uma breve discussão de algumas abordagens.

2.1 ABORDAGENS UTILIZANDO GRAFOS

Algumas abordagens de HTR têm aderido a hipótese de Rousseau, Camillerapp e Anquetil (2006), onde a principal estratégia consiste em encontrar os pontos de início e fim e através de um algoritmo de recuperação de trajetória encontrar o traçado correto, gerando um conjunto de coordenadas equivalentes a um sinal *online* da imagem para então ser reconhecido em um sistema de reconhecimento *online*.

As pesquisas bibliográficas realizadas por Nguyen e Blumenstein (2010) e Noubigh e Kherallah (2017) mostram as principais técnicas adotadas pelos pesquisadores. A Figura 4 apresenta um fluxograma das técnicas comumente utilizadas nas abordagens de HTR utilizando grafos.

Figura 4 – Componentes genéricos de uma abordagem de HTR



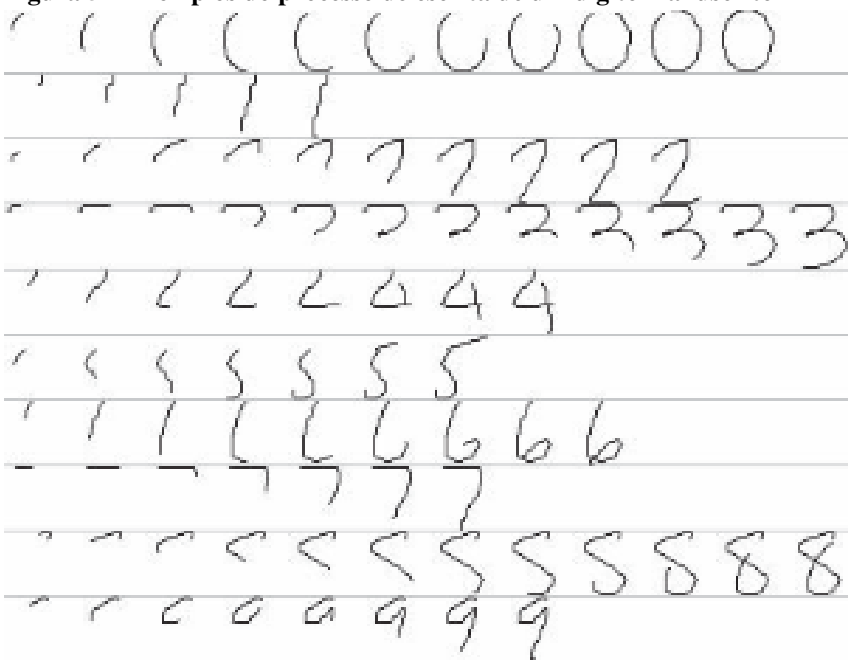
Fonte: Adaptado de Nguyen e Blumenstein (2010)

Como observado na Figura 4, a binarização é uma etapa primordial em qualquer sistema de reconhecimento; após a binarização, é realizada a esqueletização (GONZALEZ; WOODS, 2009) da imagem. Com o resultado da esqueletização, zonas ambíguas são detectadas e analisadas (NGUYEN; BLUMENSTEIN, 2010). A dificuldade de resolver efetivamente as ambiguidades e pontos incertos de regiões de interseção pode afetar diretamente o desempenho de algoritmos de recuperação de trajetória (NOUBIGH; KHERALLAH, 2017). Após a detecção e análise de zonas ambíguas, é realizada a seleção dos pontos de início e fim. Essas duas informações são importantes para a próxima etapa, que consiste na recuperação da trajetória final (NGUYEN; BLUMENSTEIN, 2010).

A pesquisa desenvolvida por Nagoya e Fujioka (2011) propôs um algoritmo para encontrar o melhor caminho Euleriano de um grafo não direcionado, obtido através da esqueletização da imagem da palavra manuscrita. O conceito de *D-line Index* é introduzido e representa a ponte entre duas arestas de grau ímpar (NAGOYA; FUJIOKA, 2011). Este conceito garante diminuir o caminho entre dois vértices de um grafo, sendo possível encontrar o melhor caminho Euleriano como forma de traçar a palavra manuscrita (NAGOYA; FUJIOKA, 2011).

Iwakiri *et al.* (2012) focaram seu trabalho em instâncias de imagens. Uma instância é uma sequência de imagens, que são as fases do processo da escrita (IWAKIRI *et al.*, 2012). A Figura 5 apresenta exemplos de instâncias.

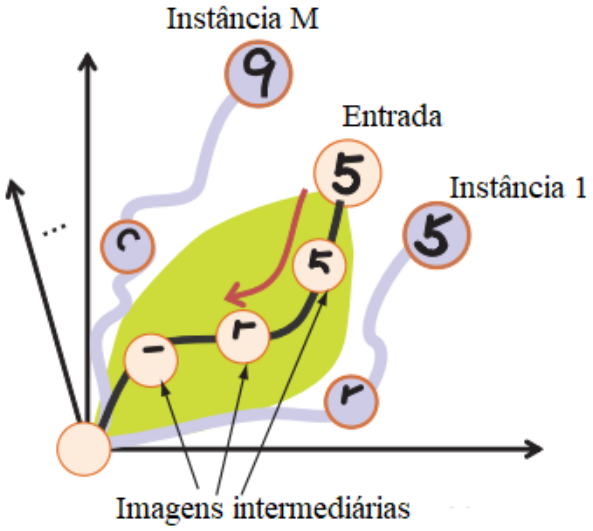
Figura 5 – Exemplos do processo de escrita de um dígito manuscrito



Fonte: Adaptado de Iwakiri *et al.* (2012)

Cada linha da Figura 5 apresenta uma instância do dígito manuscrito, se as mesmas forem ordenadas na sequência correta aproximam-se do modo da escrita humana (IWAKIRI *et al.*, 2012). Na Figura 6 é ilustrada a ideia básica do método proposto por Iwakiri *et al.* (2012), o qual mostra o conceito de instâncias e como cada imagem intermediária é relacionada uma com as outras.

Figura 6 – Ideia básica de recuperação de traços baseado em instâncias



Fonte: Adaptado de Iwakiri *et al.* (2012)

A Figura 6 mostra o problema da HTR abordado por Iwakiri *et al.* (2012) através de um grafo cúbico para o caminho ideal, sendo que cada nó é uma instância e o caminho ideal é a sequência correta das instâncias subsequentes (IWAKIRI *et al.*, 2012). O algoritmo consiste em determinar o melhor caminho de instâncias da imagem de entrada (que consiste a imagem do dígito inteiro) para uma imagem vazia (a origem); após isso, faz-se o caminho inverso do traçado para quando chegar na origem reproduzir o processo da escrita a partir da origem (IWAKIRI *et al.*, 2012). A área verde da Figura 6 mostra o conjunto de imagens intermediárias possíveis para a imagem de entrada. As instâncias são construídas a partir de dados *online* (IWAKIRI *et al.*, 2012).

A pesquisa apresentada por Sharma (2013) apresenta a HTR através do cálculo do código da cadeia. O código da cadeia consiste em uma forma de representação da borda de um objeto na imagem utilizando a posição relativa entre os pixels consecutivos da borda do objeto (PEDRINI; SCHWARTZ, 2008). A imagem passa por estágios de pré-processamento, que inclui a esqueletização. Dois algoritmos são apresentados para recuperar a trajetória do manuscrito usando os códigos em cadeia. Em um trabalho posterior, Sharma (2015) utiliza as características dinâmicas recuperadas de imagens de manuscritos e características estáticas para se realizar o reconhecimento. As características estáticas são a quantidade de pixels brancos em diferentes sub-regiões da imagem e as características dinâmicas são obtidos através do cálculo do código da cadeia do esqueleto e das bordas do objeto na imagem (SHARMA, 2015).

O método proposto por Phan *et al.* (2015) também utiliza a estratégia de duas etapas: esqueletização e recuperação de trajetória. O método de esqueletização utilizado é baseado em contornos poligonais do caractere manuscrito. Este processo é seguido pelo uso do procedimento de triangulação de *Delaunay*, que converte coordenadas cartesianas em uma malha triangular. A partir da malha de triângulos gerada, uma estimativa do esqueleto do caractere é formada a partir da junção dos triângulos. Com isso o autor busca uma esqueletização da imagem que

seja a mais próxima da representação do caractere original. Um grafo é construído a partir da esqueletização e para encontrar o melhor caminho e, após isso, é utilizado um algoritmo básico de traçado (*Basic Trace Algorithm*).

O trabalho de Kinjarapu, Narr e Valurouthu (2016) utiliza também a estratégia de grafos a partir da esqueletização de imagens de caracteres da linguagem Telugu. Sua técnica para recuperar a ordem da escrita envolve oito estruturas de grafos: *S-vertex*, *S-loop*, *Stouching*, *T-vertex*, *T-loop*, *D-line*, *M-loop* e *L-loop* (KINJARAPU; NARR; VALUROUTHU, 2016). Essas técnicas foram identificadas a partir da análise gráfica dos caracteres Telugu, sendo possível ordenar os traços de cima para baixo.

Kha, Kha e Blumenstein (2016) desenvolveram um algoritmo que percorre grafos a partir da esqueletização de 400 caracteres manuscritos da base de dados UNIPEN(GUYON *et al.*, 1994). O algoritmo proposto identifica os pontos de início e fim, verifica os trajetos possíveis para cada nó e analisa os vários tipos de laços encontrados nos caracteres.

O uso da esqueletização é um processo comum para a formação dos grafos na HTR. Um dos problemas enfrentados pela esqueletização, que é um processo utilizado pelos trabalhos anteriormente citados, são as distorções que ocorrem nos pontos de junções, ou seja, locais onde os traços se cruzam (VU; NA; KIM, 2015). O método proposto por Vu, Na e Kim (2015) consiste em transformar esses segmentos distorcidos em um único ponto, através de etapas de pós-processamento que corrigem os traços ilegítimos (aqueles que não fazem parte do manuscrito) causados por uma esqueletização normal. O objetivo desse processo é melhorar o vetor de característica para os sistemas de HTR (VU; NA; KIM, 2015).

2.2 ABORDAGENS UTILIZANDO INTELIGÊNCIA ARTIFICIAL E APRENDIZADO DE MÁQUINA

No trabalho de Elbaati *et al.* (2009) é desenvolvido um algoritmo genético (GA - *Genetic Algorithm*) para otimizar o melhor trajeto dos segmentos obtidos a partir da esqueletização da imagem de caracteres latinos. A população gerada para alimentar o algoritmo genético é uma cadeia de números inteiros entre 1 e o número de segmentos encontrados na imagem (ELBAATI *et al.*, 2009). A codificação do segmento é a ordem que elas aparecem; seleção, cruzamento e mutação são as etapas realizadas pelo algoritmo genético e a função *fitness* é aplicada otimizando o percurso do manuscrito, que é a ordem temporal dos traços (ELBAATI *et al.*, 2009).

Utilizando bases de caracteres chineses, Zhao, Yang e Tao (2019) e Wang *et al.* (2019) também utilizam redes de aprendizagem profunda para recuperar a trajetória de caracteres manuscritos.

O método desenvolvido por Zhao, Yang e Tao (2019) é constituído de três sistemas. O primeiro sistema consiste em extrair a energia estática da escrita (*static writing energy*), que

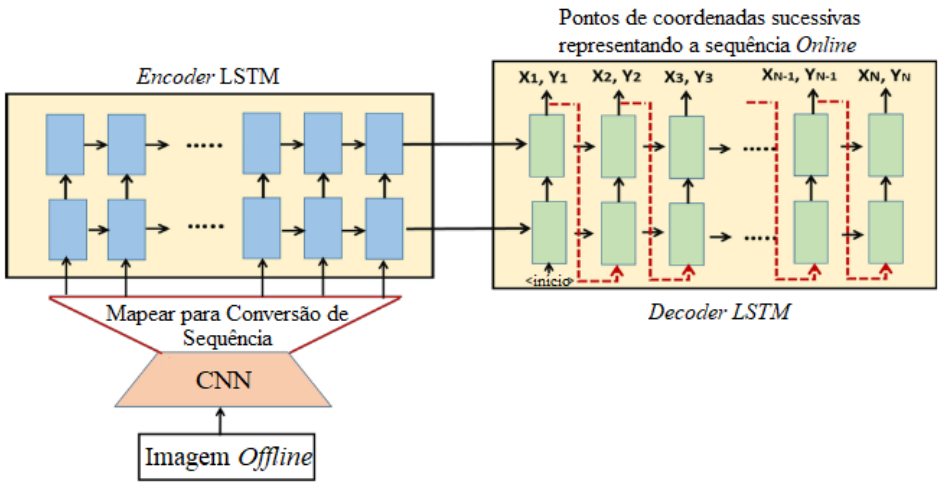
consiste em utilizar uma CNN de duas camadas de convolução e duas camadas totalmente conectadas para extrair o mapeamento de calor (*heat-map*) da imagem; este mapeamento representa quais pontos da imagem são escritas antes (valor alto) e quais pontos são escritos depois (valor baixo). O segundo sistema consiste em identificar a energia dinâmica da escrita (*dynamic writing energy*) através de outra CNN de três camadas convolucionais e três camadas totalmente conectadas; essa energia dinâmica é composta por três estados: e escolha do ramo (*meeting choosing*), a pesquisa do ponto final (*end-point searching*) e a movimentação em linha (*inline moving*). O último sistema é um algoritmo que computa ambas as informações dos sistemas anteriores para produzir a predição ordenada da escrita (*drawing order recovery* - DOR) do caractere da imagem (ZHAO; YANG; TAO, 2019).

Os caracteres chineses são compostos por um ou mais subcaracteres (WANG *et al.*, 2019); a partir disso, Wang *et al.* (2019) criaram um sistema de HTR para caracteres chineses a partir de dois estágios. Dada uma imagem *offline* contendo o caractere chinês, na primeira etapa uma CNN é utilizada para computar a ordem da escrita dos subcaracteres do caractere e uma segunda CNN estima a ordem e direção da escrita dos traços de cada subcaractere. A segunda etapa consiste em utilizar o *graph cut* (GC) (BOYKOV; VEKSLER; ZABIH, 2001) para refinar a estimativa da ordem da escrita. A GC então, produz a sequência de coordenadas que representa a ordem da escrita dos traços do caractere (WANG *et al.*, 2019).

Podemos observar que Zhao, Yang e Tao (2019) e Wang *et al.* (2019) utilizam uma estratégia híbrida, utilizando redes *deep learning* e algoritmos heurísticos, para compor seus sistemas de HTR.

O estudo de sistemas totalmente baseado em *deep learning* para HTR de caracteres indianos é explorado por Bhunia *et al.* (2018). O método consiste em utilizar a arquitetura combinada de CNN e LSTM *encoder-decoder* para gerar as coordenadas dos pixels a serem desenhados. A Figura 7 mostra no geral a arquitetura proposta por Bhunia *et al.* (2018).

Figura 7 – Diagrama da arquitetura proposta por Bhunia *et al.* (2018) para a recuperação de trajetória de manuscritos. LSTM unidirecional é apresentada por questões de simplicidade



Fonte: Adaptado de Bhunia *et al.* (2018)

Como mostrado na Figura 7, Bhunia *et al.* (2018) realizam a extração de características das imagens *offline* é realizada por uma CNN, produzindo um vetor de características de tamanho 256. A arquitetura *encoder-decoder* LSTM utiliza o modelo *seq2seq* proposta por Sutskever, Vinyals e Le (2014) para aprender a probabilidade condicional da entrada para a saída (BHUNIA *et al.*, 2018). A rede codificadora (*encoder*) calcula as características codificadas de uma entrada de tamanho fixo gerada pela CNN; a saída codificada é retirada do último estado oculto da unidade LSTM; a saída codificada é processada na rede decodificadora (*decoder*) para gerar a saída final da sequência dos pontos de coordenadas. Utilizando uma base *online* de caracteres Telugo, Bhunia *et al.* (2018) realiza o pré-processamento na base de dados que consiste em padronizar em 50 pontos o tamanho da sequência de pontos de cada caractere e transformar essa sequência de pontos em imagens *offline*, além de realizar uma etapa de pós-processamento, que consiste em mover as coordenadas preditas para o pixel mais próximo do esqueleto do caractere original gerado pelo *ground-truth*.

Elbaati, Hamdi e Alimi (2019) também construíram seu sistema utilizando CNN e LSTM para a predição de trajetória de caracteres latinos. Entretanto, Elbaati, Hamdi e Alimi (2019) utiliza mais uma rede *deep learning*, chamada de *Beta-LSTM* para realizar a classificação das coordenadas preditas. Em suma, este método realiza a classificação de caracteres baseado na recuperação da ordem temporal do traçado de caracteres em imagens *offline* (ELBAATI; HAMDI; ALIMI, 2019). Esta estratégia monta um sistema original de reconhecimento de caractere que utiliza as informações *online* e *offline*, porém, as informações *online* são produzidas por uma rede e não são avaliadas conforme o conjunto verdade. Elbaati, Hamdi e Alimi (2019) afirma que a ordem temporal predita pelas duas primeiras redes é compatível com o conjunto verdade, porém não apresenta como foi computado essa compatibilidade, além de realizar um processo de reamostragem das coordenadas preditas baseado na curvatura das trajetórias, método que não é detalhado em seu trabalho.

2.3 RESULTADOS DAS ABORDAGENS

A Tabela 1 apresenta os resultados, as respectivas técnicas e base de dados utilizadas nos trabalhos citados anteriormente.

Com base na tabela, pode-se observar que o sistema proposto por Sharma (2015) atingiu uma taxa próxima de 100% de acerto sobre os dígitos manuscritos da base MNIST. Como comentado anteriormente, sua técnica consiste em utilizar informações dinâmicas e estáticas para realizar o reconhecimento dos manuscritos *offline*.

Ainda há uma grande lacuna no reconhecimento de manuscritos baseado em HTR da base IRONOFF, relatado por Elbaati *et al.* (2009), que conseguiram 72% de acurácia. Nota-se também que Elbaati *et al.* (2009) foram os únicos que relatam testes do seu sistema sobre bases com idiomas diferentes, atingindo taxas de acurácia diferentes. Este aspecto mostra que

Tabela 1 – Desempenho e técnicas utilizadas dos trabalhos relacionados à HTR

Autor	*Dataset*	Técnica	Acurácia (%)
Stefano *et al.* (2010)	*Dataset* próprio	Correspondência de pixels + algoritmo para percorrer grafo	82,88
Sharma (2013)	MNIST	Código da Cadeia + algoritmo de percorrer grafo	97,39
Sharma (2015)	MNIST	código da Cadeia + distribuição de pixels + algoritmo para percorrer grafo	99,27
Phan *et al.* (2015)	*Dataset* próprio	Malha triangular + algoritmo para percorrer grafo	94.78
Vu, Na e Kim (2015)	IAM	Melhoria da esqueletização + algoritmo para percorrer grafo	92,12
Kinjarapu, Narr e Valurouthu (2016)	LIPI Toolkit	Algoritmo para percorrer grafos baseado em 8 estruturas	96,35
Kha, Kha e Blumenstein (2016)	Unipen *single stroke* Unipen *multi stroke*	algoritmo para percorrer grafos	96 94
Elbaati *et al.* (2009)	IFN/ENIT *good quality* LMCA IRONOFF IFN/ENIT *random quality*	esqueletização + GA	97 94,44 72,5 72
Zhao, Yang e Tao (2019)	OLHWDB1.1	SEN + DEN + algoritmo DOR	83,1
Wang *et al.* (2019)	CASIA-OLHWDB1.0	CNN + GC	75,87
Bhunia *et al.* (2018)	LIPI Toolkit 7	CNN + *seq2seq* LSTM	96,94
Elbaati, Hamdi e Alimi (2019)	IRONOFF	CNN + LSTM + Beta-LSTM	92,77

Fonte – Autoria própria

manuscritos de línguas diferentes podem apresentar características diferentes.

Assim, é possível observar que a abordagem heurística (utilizando grafos ou não) pode ser eficaz com determinada base de dados, pois analisa a estrutura gráfica da escrita de uma base específica. Esse tipo de método pode não possuir a capacidade de generalizar, logo, pode ser que não seja eficaz se aplicado em outra base de dados do mesmo idioma.

O uso da *deep learning* pode superar a abordagem heurística pois ela trabalha com métodos probabilísticos, adaptando-se à base de treinamento, podendo ser capaz de generalizar com a base de testes. Conforme a revisão da literatura realizada, o uso da *deep learning* na área de recuperação de trajetória de manuscritos é bem recente.

Zhao, Yang e Tao (2019) e Wang *et al.* (2019) utilizam uma estratégia híbrida de redes *deep learning* e algoritmos heurísticos. Em seus trabalhos alegam que a as redes desenvolvidas conseguem predizer as coordenadas, porém não com tanta precisão, por isso foi desenvolvido algoritmos para refinar as coordenadas preditas pelas redes (ZHAO; YANG; TAO, 2019) (WANG *et al.*, 2019).

O trabalho de Bhunia *et al.* (2018) realiza a HTR através de arquiteturas de aprendizagem profunda, limitando o tamanho da sequência *online* de coordenadas preditas para 50 (cinquenta) pontos, logo, o treinamento da rede também utiliza uma base de dados com a uniformização das sequências de coordenadas para essa mesma quantidade. Bhunia *et al.* (2018) também comenta que as imagens *offline* dos manuscritos são redimensionadas para o tamanho 64x64.

No trabalho realizado por Elbaati, Hamdi e Alimi (2019) apresenta uma abordagem diferente que consiste em, através de uma rede *deep learning*, realizar a classificação das coordenadas preditas por uma outra rede *deep learning*, realizando assim, o processo de reconhecimento de caracteres *offline* através do reconhecimento de caracteres *online*.

Apesar dos trabalhos apresentarem resultados promissores para o problema da HTR, a análise dos resultados ainda é necessário ser explorada. As informações recuperadas pelas redes *deep learning* não são avaliadas da mesma forma entre os trabalhos. Cada autor utiliza uma forma de avaliação diferente que muitas vezes é necessária a distorção dos dados recuperados (como o pós-processamento realizado por Bhunia *et al.* (2018)) ou uso de medidas estatísticas (como Zhao, Yang e Tao (2019) e Wang *et al.* (2019)) que não apresentam valores reais de acurácia. Baseado nisso, este trabalho apresenta uma proposta de avaliação de sistemas de *deep learning* para HTR, mostrando sua eficácia em verificar a semelhança geométrica entre as coordenadas preditas e o caractere original e a capacidade de verificar a ordem correta comparando-a com o conjunto verdade. Este trabalho também apresenta um estudo na variação do tamanho de entrada, mostrando que é possível obter melhores resultados realizando etapas de pré-processamento para normalização dos dados.

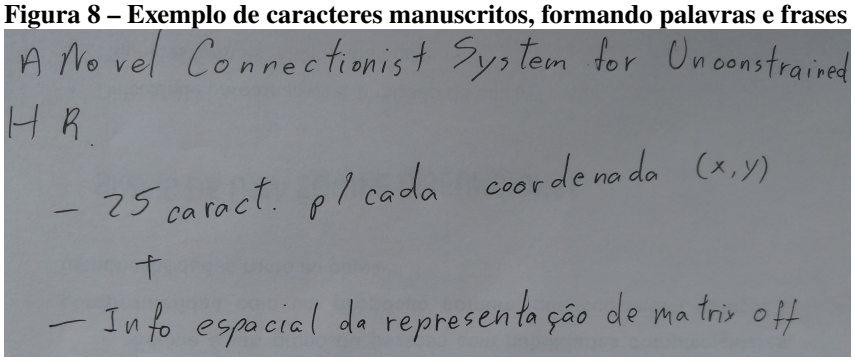
3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda alguns conceitos sobre reconhecimento de manuscritos, recuperação de trajetória de manuscritos e aprendizado de máquina. Também será abordado algumas técnicas de *deep learning* que são utilizados para o reconhecimento de padrões e também para a recuperação de trajetória de manuscritos.

3.1 RECONHECIMENTO DE MANUSCRITOS

Um texto manuscrito é um conjunto de caracteres com formas básicas definidas, sendo possível combiná-las para formar uma representação de uma unidade linguística, como por exemplo, no alfabeto latino, combina-se letras individuais para dar forma às palavras cursivas (AIRES, 2005). A Figura 8 apresenta uma imagem escaneada a partir de um papel que contém exemplos de palavras manuscritas no alfabeto latino.

Figura 8 – Exemplo de caracteres manuscritos, formando palavras e frases



Fonte: Autoria própria

Os métodos para o reconhecimento de caracteres tem como tarefa transformar os pixels identificados na imagem em representações simbólicas digitais, como caracteres em ASCII (PLAMONDON; SRIHARI, 2000). Plamondon e Srihari (2000) classificam os sistemas de reconhecimento de manuscritos em sistemas *online* e sistemas *offline*.

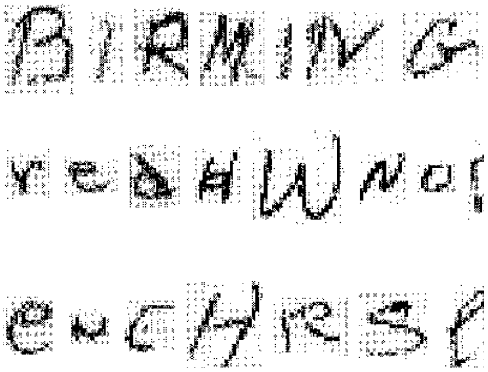
3.1.1 Reconhecimento *Online* e *Offline* de Manuscritos

Um manuscrito é convertido na sua forma digital através do escaneamento do papel contendo a escrita da palavra ou através da escrita com uma caneta eletrônica sobre uma superfície digitalizadora (PLAMONDON; SRIHARI, 2000). Esses dois modos de coleta de dados definem os modos de reconhecimento *offline* e *online*, respectivamente.

No reconhecimento *offline*, os dados das palavras ou dos caracteres são disponibilizados através de uma imagem digitalizada da escrita, sendo necessário realizar uma série de

etapas de pré-processamento da imagem para localizar e registrar o texto apropriado que será reconhecido, passo conhecido como análise de documento. Segundo Pladomon *et al.*, as etapas mais comuns para efetuar esta análise são limiarização, remoção de ruídos, segmentação de linhas e segmentação de palavras (PLAMONDON; SRIHARI, 2000). A Figura 9 mostra alguns exemplos de caracteres manuscritos segmentados a partir de imagens escaneadas.

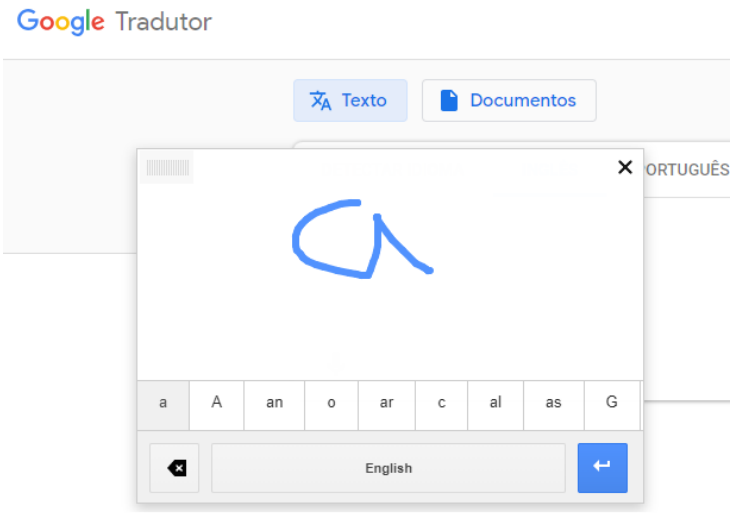
Figura 9 – Exemplo de caracteres manuscritos segmentados



Fonte: Plamondon e Srihari (2000)

O reconhecimento *online*, por outro lado, envolve informações dinâmicas referente ao ato da escrita. Informações como ordem do traçado, velocidade e aceleração são disponibilizadas pelas entradas de dados eletrônicas específicas, como uma superfície digitalizadora ou uma caneta eletrônica. Esses manuscritos são reconhecidos através de métodos estruturais e baseado em regras ou métodos estatísticos (PLAMONDON; SRIHARI, 2000). Um exemplo de reconhecedor *online* de manuscrito é a plataforma de reconhecimento de escrita do Google Tradutor[1], como mostrado na Figura 10. Esta plataforma foi desenvolvida por Keysers *et al.* (2017) e suporta múltiplos idiomas (KEYSERS *et al.*, 2017).

Figura 10 – Método de entrada on-line de manuscritos do Google Tradutor



Fonte: Autoria própria

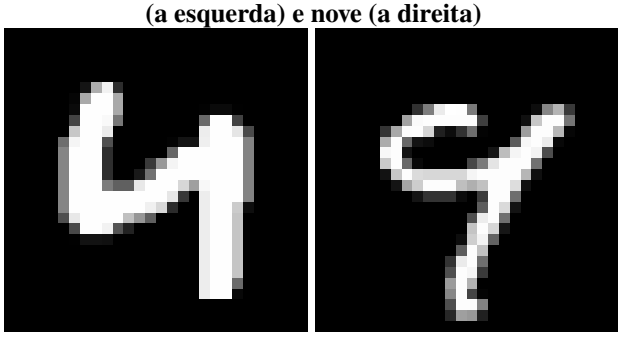
[1] https://translate.google.com.br/

3.2 RECUPERAÇÃO DE TRAJETÓRIA DE MANUSCRITOS - HTR

A recuperação de trajetória de manuscritos (*Handwriting Trajectory Recovery* - HTR) são técnicas cujo o objetivo é encontrar um trajeto orientado similar àquele realizado por um escritor durante a operação de escrita (NOUBIGH; KHERALLAH, 2017). A motivação para esse tipo de pesquisa é a obtenção de características dinâmicas a partir de base de dados de manuscritos que não possuem informações dinâmicas, visando melhorar o desempenho do reconhecimento de manuscritos *offline*. Um exemplo de informação dinâmica é a ordem cronológica dos pontos que compõem o traçado, que é perdida logo após o escritor terminar a operação da escrita.

A partir das informações dinâmicas, pode-se discriminar formas que são geometricamente semelhantes (NOUBIGH; KHERALLAH, 2017). A Figura 11 apresenta duas amostras de dígitos manuscritos: o numeral 4 (quatro) e o 9 (nove).

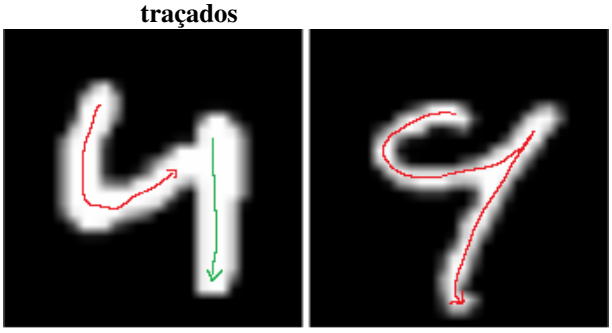
Figura 11 – Imagens dos dígitos manuscritos quatro (a esquerda) e nove (a direita)



Fonte: LeCun *et al.* (1998)

Como pode ser observado na Figura 11, os dígitos quatro e nove possuem um semicírculo e um traço perpendicular. Um algoritmo que utilize as características geométricas para discriminar essas imagens pode classificar esses dois dígitos como sendo iguais. Se analisarmos o traçado da escrita desses dígitos, pode-se ter algo parecido como mostrado na Figura 12.

Figura 12 – Imagens dos dígitos manuscritos quatro (a esquerda) e nove (a direita) com seus traçados



Fonte: Adaptado de LeCun *et al.* (1998)

Um algoritmo classificador que utilize como forma auxiliar as informações dinâmicas, poderia identificar os manuscritos diferentes a partir da quantidade de traçados. Analisando a

Figura 12, para escrever o dígito quatro, utiliza-se dois traços, e para o nove a escrita é realizada com apenas um traçado.

A partir do número de traços do caractere, pode-se realizar uma divisão dos caracteres em dois subgrupos: caracteres de traços singular (*single-stroke)* e caracteres de múltiplos traços (*multi-stroke*) (KHAN; HAIDER, 2010). Esta diferença é caracterizada pelo modo de escrita do autor e é uma informação obtida através de métodos de capturas de entradas eletrônicas de escrita. O dígito quatro, por exemplo, pode ser tanto escrito com apenas um traço (caso o autor realize um traçado contítnuo durante toda a escrita, não interrompendo o contato da caneta sobre a superfície de escrita), ou pode ser escrita com múltiplos traços (quando em algum momento o autor interrompe o contato da caneta sobre a superfície de escrita e retorna o contato com o objetivo de finalizar o caractere).

3.2.1 Método de avaliação de HTR

Para calcular a acurácia dos trajetos recuperados dos sistemas de HTR, os autores utilizam métodos distintos para avaliar seus experimentos, comparando os traçados preditos com o conjunto verdade, chamado de *ground-truth* (NGUYEN; BLUMENSTEIN, 2010), ou seja, apesar de utilizarem o *ground-truth*, existem diversos métodos para avaliar os sistemas de HTR, não havendo uma padronização para as métricas. Dinh *et al.* (2016) utiliza o algoritmo de alinhamento proposto por Needleman e Wunsch (1970) para calcular a taxa de acerto do seu algoritmo através do alinhamento dos traçados reconhecidos com o traçados do *ground-truth* (DINH *et al.*, 2016).

Rousseau, Camillerapp e Anquetil (2006) avalia as etapas do seu sistema separadamente. Seu método de avaliação consiste em verificar primeiramente o acerto dos pontos de início e fim. Após, avalia-se o algoritmo de reconstrução verificando os acertos dos trajetos reconstruídos. Apenas os resultados corretos na primeira etapa passam para a próxima etapa (ROUSSEAU; CAMILLERAPP; ANQUETIL, 2006), ou seja, somente amostras em que foi possível identificar os pontos corretos de início e fim são verificados no algoritmo de reconstrução.

Bhunia *et al.* (2018) realiza um pós-processamento chamado de "tradução"de coordenadas, que consiste em mover as coordenadas preditas para o pixel mais próximo em relação ao esqueleto original (*ground-truth*) do caractere. Após isso, avalia seu sistema de HTR de forma similar em comparação com a avaliação proposta por Rousseau, Camillerapp e Anquetil (2006), utilizando as seguintes métricas para avaliar pontos de início, pontos de junção e trajetória completa:

- Acurácia de Ponto de Início (*Start Point* - SP): predição do ponto de início correto. A

média é calculada pela Equação 3.1;

$$\text{SP} = \frac{\text{Número de imagens com o SP detectado corretamente}}{\text{Número total de imagens de teste}} \tag{3.1}$$

- Acurácia Pontos de Junção (*Junction Points* - JP): na travessia dos pontos de junção (ou seja, traços que se cruzam), caso determinar o caminho correto da entrada e saída do ponto de junção, considera-se a junção positiva. Como uma imagem possui mais de um ponto de junção, a média da acurácia dos pontos de junção é calculada pela Equação 3.2;

$$\text{Acurácia JP} = \frac{\text{Número de JP percorridos corretamente}}{\text{Número total de JP na base de teste}} \tag{3.2}$$

- Acurácia da recuperação da Trajetória Completa (*Complete Trajectory* - CT): quando a trajetória completa é perfeitamente recuperada desde que o ponto de início correto (SP) esteja correto, calculada pela Equação 3.3

$$\text{Acurácia CT} = \frac{\text{Número de imagens onde todos os pontos foram preditos corretamente}}{\text{Número total de SP na base de teste}} \tag{3.3}$$

Os pontos de início e os pontos de junção são elementos importantes para algoritmos que realizam a HTR utilizando grafos, como apresentado por Nguyen e Blumenstein (2010), pois o caminho percorrido por um grafo consiste de um nó inicial e seus nós subsequentes, o qual vai representar o traçado do caractere. É importante ressaltar que a trajetória completa CT representa o acerto total do traçado do caractere, pois este mostra se a HTR foi realizada corretamente.

Outros trabalhos na literatura utilizam medidas estatísticas. Zhao, Yang e Tao (2019) e Phan *et al.* (2015) utilizam DTW (Dynamic Time Warping) (GIORGINO *et al.*, 2009). Phan *et al.* (2015) utiliza o RMSE (*Root Mean Square Error*) (CHAI; DRAXLER, 2014) para a avaliação. Wang *et al.* (2019) utiliza como avaliação o coeficiente de correlação de Pearson (MUKAKA, 2012).

O sistema de avaliação utilizado por Elbaati, Hamdi e Alimi (2019) é baseado na convencional classificação de caracteres. A partir das coordenadas preditas, Elbaati, Hamdi e Alimi (2019) utiliza mais uma rede neural *deep learning* para interpretar as coordenadas e classificalas nos rótulos dos caracteres, ou seja, além de treinar uma rede para a tarefa de predição, deve-se treinar mais uma rede a parte para classificar as coordenadas preditas em caracteres. Logo, Elbaati *et al.* (2009) não avalia diretamente as coordenadas preditas, mas sim, avalia baseado na classificação dos caracteres.

Pode ser observado que cada autor utiliza técnicas diferentes de avaliação para verificar os resultados obtidos de seus experimentos.

3.3 APRENDIZADO DE MÁQUINA

Possuir um comportamento inteligente é possuir a capacidade de aprender (CARVALHO *et al.*, 2011). A capacidade de uma máquina aprender refere-se à sua capacidade de induzir algo a partir de eventos passados, que são informações que uma máquina deve analisar para realizar classificações ou predição de eventos futuros (CARVALHO *et al.*, 2011).

O Aprendizado de Máquina é o processo de indução de uma hipótese que um computador realiza para aprender a resolver um problema a partir da análise de um conjunto de dados que representam instâncias do problema a ser resolvido (CARVALHO *et al.*, 2011). A partir dessa aprendizagem automática, os computadores resolvem os problemas de modo mais autônomo reduzindo a mediação humana sobre o problema. O objetivo principal é aprender um modelo (ou hipótese) através do processamento de um conjunto de treinamento e relacionar os valores de um objeto desse conjunto de treinamento ao valor de seu atributo de saída (CARVALHO *et al.*, 2011).

A necessidade do aprendizado de máquina está na complexidade do problema e na necessidade de adaptação. Segundo Shalev-Shwartz e Ben-David (2014), existem problemas complexos que podem ser auxiliadas por programas que aprendam e se aperfeiçoam baseado nas experiências que lhe são expostas (SHALEV-SHWARTZ; BEN-DAVID, 2014). Tarefas realizadas por humanos como reconhecimento de falas e compreensão de imagens, tarefas além da capacidade humana como previsão do tempo e análise de dados genômicos, tarefas que requerem adaptação através da interação com os dados de entrada como reconhecimento de manuscritos de diferentes usuários são alguns exemplos desses problemas.

Os dois tipos de aprendizagem de máquina mais conhecidos são: a aprendizagem supervisionada e a não supervisionada. Aprender mediante exemplos rotulados de uma base de treinamento e testar em uma base de testes consiste no que se conhece de aprendizagem supervisionada. Já a aprendizagem não supervisionada é quando dado um conjunto de dados de entrada sem rótulos, a máquina organiza os dados em grupos, sendo cada grupo um *cluster* de objetos que são equivalentes entre si (SHALEV-SHWARTZ; BEN-DAVID, 2014). A escolha da abordagem de aprendizagem depende da base de dados envolvida no problema. Caso haja conhecimento sobre os rótulos da base, escolhe-se a aprendizagem supervisionada. Em uma base de dados onde não se conhece as classes e nem quantos tipos de objetos existem na base, então a aprendizagem não supervisionada deve ser utilizada (SHALEV-SHWARTZ; BEN-DAVID, 2014).

Para o reconhecimento de manuscritos, pode-se ter o conhecimento *a priori* sobre os rótulos que representam a classe que a palavra ou caractere pertence. Logo, utiliza-se a aprendizagem supervisionada. A HTR é um processo onde a aprendizagem necessita de supervisão, sendo necessário aprender previamente a ordem dos trajetos a serem recuperados.

3.4 APRENDIZAGEM PROFUNDA

Muitas abordagens de aprendizado de máquina e reconhecimento de padrões foram desenvolvidas para classificar um conjunto de características estatísticas ou estruturais (e.g. histograma de distribuição de pixels e Momentos de Hu, respectivamente), extraído dos dados no seu formato natural (LECUN; BENGIO; HINTON, 2015). A extração desse conjunto de características é um processo conduzido pelo homem, que através de métodos matemáticos e estatísticos criam sistemas que calculam valores para discretizar um determinado objeto do seu conjunto. Tendo esses valores como a entrada, o classificador produz como saída a detecção ou a classificação de padrões.

Com a utilização da aprendizagem profunda (*deep learning*), a extração dessas características se tornou um processo automatizado. A aprendizagem profunda permite que modelos computacionais de multicamadas, ou seja, as redes neurais artificiais, aprendam representações de múltiplos níveis de abstração (LECUN; BENGIO; HINTON, 2015), amplificando aspectos da entrada que são importantes para a discriminação do objeto e suprimindo os aspectos irrelevantes.

Em geral, no âmbito de classificação de imagens, os dados entram na arquitetura de aprendizagem profunda em forma de um vetor com os valores dos pixels. As características aprendidas na primeira camada de representação geralmente representam a presença ou a ausência de bordas e arestas num determinado local da imagem. A segunda camada pode encontrar arranjos particulares das bordas e arestas encontradas na camada anterior. A terceira camada pode localizar partes que são combinações maiores dos arranjos que podem identificar objetos familiares. As camadas subsequentes detectariam os objetos como combinações dessas partes (LECUN; BENGIO; HINTON, 2015). Logo, todas essas camadas são extratores de características que aprendem representações por elas mesmas.

Segundo os trabalhos levantados por LeCun, Bengio e Hinton (2015), os modelos de aprendizagem profunda atingiram o estado da arte em várias áreas da ciência, obtendo grandes avanços para solucionar problemas que resistiram às melhores tentativas da comunidade da inteligência artificial. Dentre esses problemas está o reconhecimento de objetos em imagens (como relatado em Krizhevsky, Sutskever e Hinton (2012), Tompson *et al.* (2014) e Hafemann, Oliveira e Sabourin (2018)) e o reconhecimento de fala (Mikolov *et al.* (2011) e Hinton *et al.* (2012)). A aprendizagem profunda também atingiu bons resultados na predição de dados de relações estrutura-atividade quantitativa como mostrado em Ma *et al.* (2015) e *splicing* alternativo de ácidos ribonucleicos como mostrado em Leung *et al.* (2014), na tradução de idiomas como relatado em Wu *et al.* (2016) e na recuperação da sequência de traçados manuscritos por Bhunia *et al.* (2018), mostrando que modelos de aprendizagem profunda têm produzido resultados promissores em vários campos de pesquisas.

A seguir serão apresentadas as arquiteturas de aprendizagem profunda estudadas no

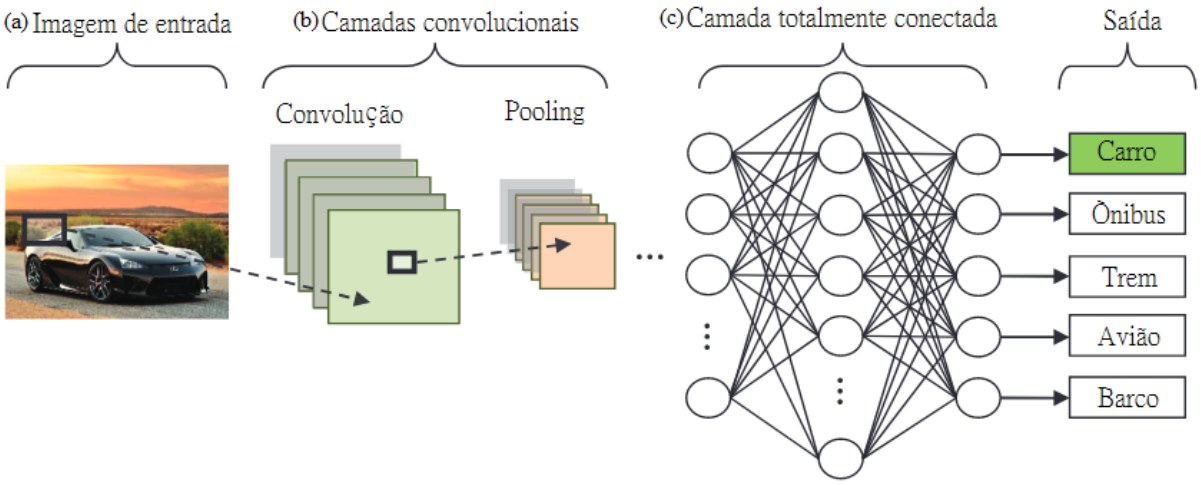
presente trabalho. Entretanto, é importante ressaltar a existência de outras arquiteturas como por exemplo as Redes Neurais Recorrentes, Redes Adversariais Geradoras e as Auto-codificadoras.

3.4.1 Redes Neurais Convolucionais - CNN

Assim como as Redes Neurais Artificiais, as Redes Neurais Convolucionais, ou CNN (*Convolutional Neural Network*) são inspiradas biologicamente no cérebro humano, mais especificamente o córtex visual (RAWAT; WANG, 2017). Sua principal capacidade é aprender os padrões dos dados que lhe são fornecidos. As redes neurais convolucionais são arquiteturas criadas para processar dados multidimensionais (LECUN; BENGIO; HINTON, 2015), como por exemplo, uma imagem bidimensional (2D) composta por pixels de três canais de intensidade de cor (RGB).

A arquitetura da CNN basicamente consiste em várias camadas de convolução e *pooling*, seguido de uma rede neural totalmente conectada (RAWAT; WANG, 2017). A Figura 13 mostra a arquitetura de uma rede CNN para a classificação de imagens.

Figura 13 – Sequência dos módulos da CNN



Fonte: Adaptado de Rawat e Wang (2017)

Como observado na Figura 13, a rede recebe diretamente a imagem crua como entrada (Figura 13-a). A imagem passa por alguns estágios de convolução e *pooling* (Figura 13-b) que tem como função extrair um vetor de características do objeto. Este vetor serve de entrada para uma rede neural totalmente conectada (Figura 13-c) para que seja realizada a sua classificação.

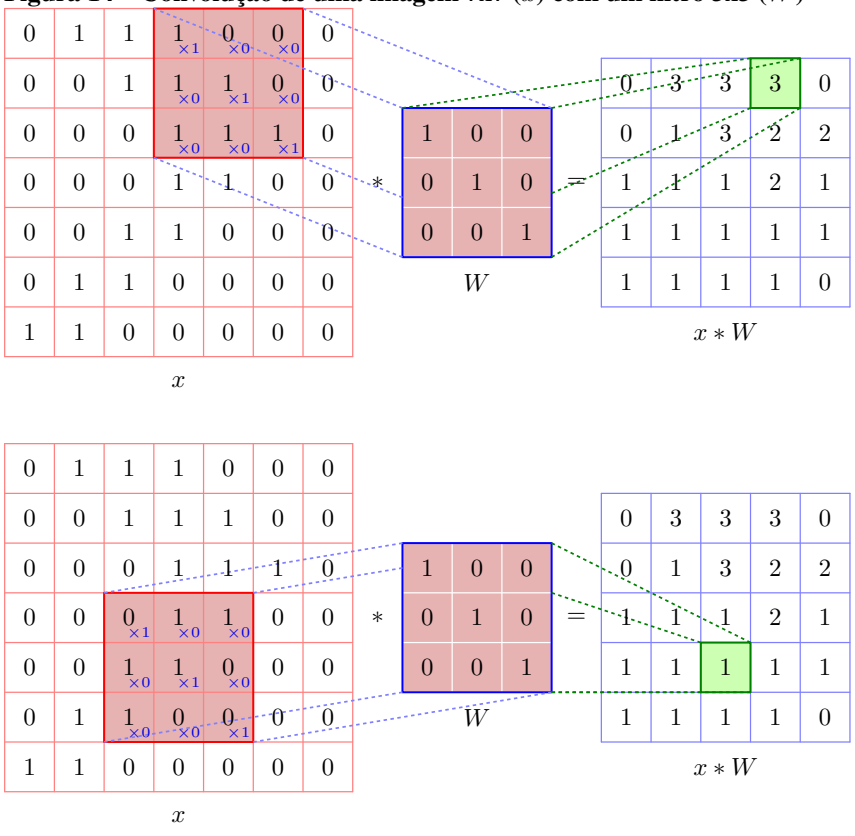
As sessões seguintes detalham brevemente cada uma das etapas da CNN conforme descritos na Figura 13.

3.4.1.1 Camadas de Convolução

As camadas de convolução são usadas para extrair características pois são capazes de aprender representações a partir de uma imagem de entrada (RAWAT; WANG, 2017). Essas camadas são compostas por neurônios que são organizados em mapas de características (*feature map*): cada valor do mapa é conectado no neurônio vizinho da camada anterior através de um conjunto de pesos, também chamados de filtros (*kernel*) (LECUN; BENGIO; HINTON, 2015). O resultado do produto escalar entre os valores dos neurônios e os filtros computam um novo mapa de características. Esse processo é chamado de convolução, que consiste na aplicação dos filtros bidimensionais através de uma "janela deslizante"que percorre (através de saltos - *strides*) o mapa de características, computando o produto escalar dos valores do mapa de característica com os filtros.

A Figura 14 apresenta duas etapas de convolução de uma imagem x, convolucionada por um filtro W de tamanho 3x3, resultando em um mapa de características que é o produto escalar $x * W$.

Figura 14 – Convolução de uma imagem 7x7 (x) com um filtro 3x3 (W)



Fonte: Autoria própria

Na Figura 14 é demonstrado como os neurônios (organizados no mapa de características $x * W$) estão conectados com a camada anterior (x) através do conjunto de pesos (W). O filtro bidimensional W percorre a matriz x com um salto (*stride*) de valor um, calculando o produto escalar $x * W$, que é o novo mapa de características. Para cada filtro aplicado, um neurônio (do

produto $x * W$) está conectado a um subconjunto dos neurônios (uma janela de x) da camada anterior. Logo, para cada camada de convolução da rede, o produto $x * W$ sofre uma nova convolução de um outro filtro, produzindo um novo mapa de características.

No exemplo anterior, a convolução causa uma redução de dimensionalidade (de 7x7 para 5x5). Isso ocorre pois o filtro W percorre dentro dos limites da matriz x. Pode-se manter a mesma dimensionalidade posicionando o elemento central do filtro W (posição [1,1]) no primeiro elemento da matriz (posição [0,0]).

O processo de convolução pode ser computado conforme a Equação 3.4.

$$Y_k = f(W_k * x) \tag{3.4}$$

A função $f(.)$ da Equação 3.4 representa uma função de ativação. Sendo x a imagem de entrada e o filtro W, a operação $W_k * x$ é o produto escalar do modelo de filtro W em cada local da imagem de entrada x, produzindo um novo mapa de características Y_k. A função de ativação é responsável por ativar ou desativar os neurônios de uma determinada camada. Trucco e Verri (1998) define formalmente produto escalar do filtro e da imagem de acordo com a Equação 3.5.

$$I_A(i,j) = I * A = \sum_{h=-\frac{m}{2}}^{\frac{m}{2}} \sum_{k=-\frac{m}{2}}^{\frac{m}{2}} A(h,k) I(i-h, j-k) \tag{3.5}$$

Segundo Trucco e Verri (1998), a operação $*$ representa uma convolução discreta, I uma imagem de tamanho N x M, m um número ímpar menor que N e M, e A o *kernel* do filtro que é uma máscara m x m. O processo de convolução substitui o valor $I(i,j)$ com a soma ponderada dos valores de I na vizinhança (i,j).

A Figura 15 ilustra o resultado do processo de convolução, realizado por um filtro passa-alta em uma imagem em escala de cinza.

Figura 15 – Resultado do filtro de Sobel

(a) (b)

Fonte: Autoria própria

Na Figura 15-a tem-se uma imagem em escala de cinza que possui muitos detalhes como objetos de fundo (detalhes das paredes, vigas, espelho, objetos refletidos no espelho), objetos que compõem a personagem da foto (olhos, nariz, boca, cabelo, ombro, chapéu) e níveis

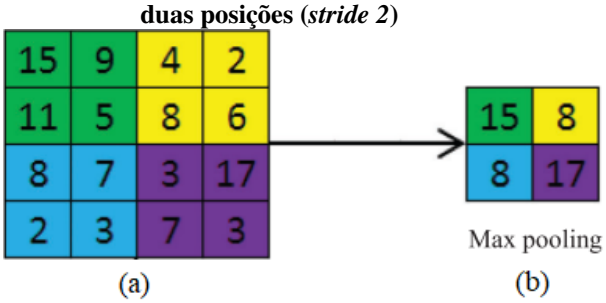
de escala de cinza diferenciados. A Figura 15-b tem-se a convolução realizada por um filtro passa-alta chamada filtro de Sobel, proposto por Sobel e Feldman (1968).

Os filtros passa-alta tem como objetivo destacar as bordas e realçar os contrastes, eliminando os pixels de baixa intensidade (DIAS; SOARES; FONSECA, 2011). O filtro de Sobel é um *kernel* de tamanho 3x3 que pode extrair características referentes às bordas que estão presentes na imagem. A convolução realizada pelos filtros de uma CNN também pode extrair características de bordas, como relatado em Mohamed e Hamida (2014). A diferença entre os filtros de imagens digitais como o Sobel e os filtros de uma CNN é devido ao fato dos valores do filtro de convolução como o de Sobel serem fixos, enquanto que os valores dos filtros de convolução de uma CNN são pesos que se ajustam conforme o treinamento, sendo capaz de generalizar conforme o problema.

3.4.1.2 Camadas de *Pooling*

Após a etapa de convolução, é realizada o processo de *pooling*. A camada de *pooling* tem como objetivo diminuir a dimensionalidade do mapa de características recebido da etapa de convolução, aumentando o desempenho e a capacidade de generalização da rede (RAWAT; WANG, 2017). Uma das técnicas utilizadas é o *max pooling*, que seleciona o maior valor dentro da região do *pooling* (NAGI *et al.*, 2011). A Figura 16 mostra um exemplo de *max pooling* com um filtro de tamanho 2x2.

Figura 16 – *Max pooling* com filtro 2x2 com salto de duas posições (*stride 2*)



Fonte: Adaptado de Rawat e Wang (2017)

Na Figura 16-a, a matriz de tamanho 4x4 sofre uma redução de dimensionalidade a partir de um filtro de tamanho 2x2. A saída do *max pooling* será uma matriz com os maiores valores numéricos de cada região 2x2 (Figura 16-b). As regiões de *pooling* são definidas utilizando um salto (*stride*) de duas posições.

As camadas convolucionais são compostas principalmente pelas etapas de convolução e *pooling*. Pode-se criar várias camadas de convolução, podendo-se diminuir a dimensionalidade do mapa em cada camada da rede. A última camada das camadas convolucionais refere-se ao mapa de características,que será enviado para uma rede neural artificial totalmente conectada.

3.4.1.3 Camada Totalmente Conectada

Essa camada é composta por uma rede neural artificial totalmente conectada. O objetivo dessa rede é interpretar as características extraídas da última camada de convolução e realizar o processo de classificação (RAWAT; WANG, 2017).

Cada neurônio dessa rede está ligado com todos os neurônios da camada seguinte, sendo a última camada contendo o número de neurônios de acordo com o número de classes do problema para realizar a classificação (RAWAT; WANG, 2017).

3.4.1.4 Configurações da CNN

A quantidade de convoluções, o método de *pooling*, o tamanho do mapa de características, o tamanho dos filtros e a quantidade de camadas e de neurônios da rede neural totalmente conectada são parâmetros que mudam de acordo com o problema. Existem casos onde a rede neural totalmente conectada não é utilizada, como relatado em Bhunia *et al.* (2018).

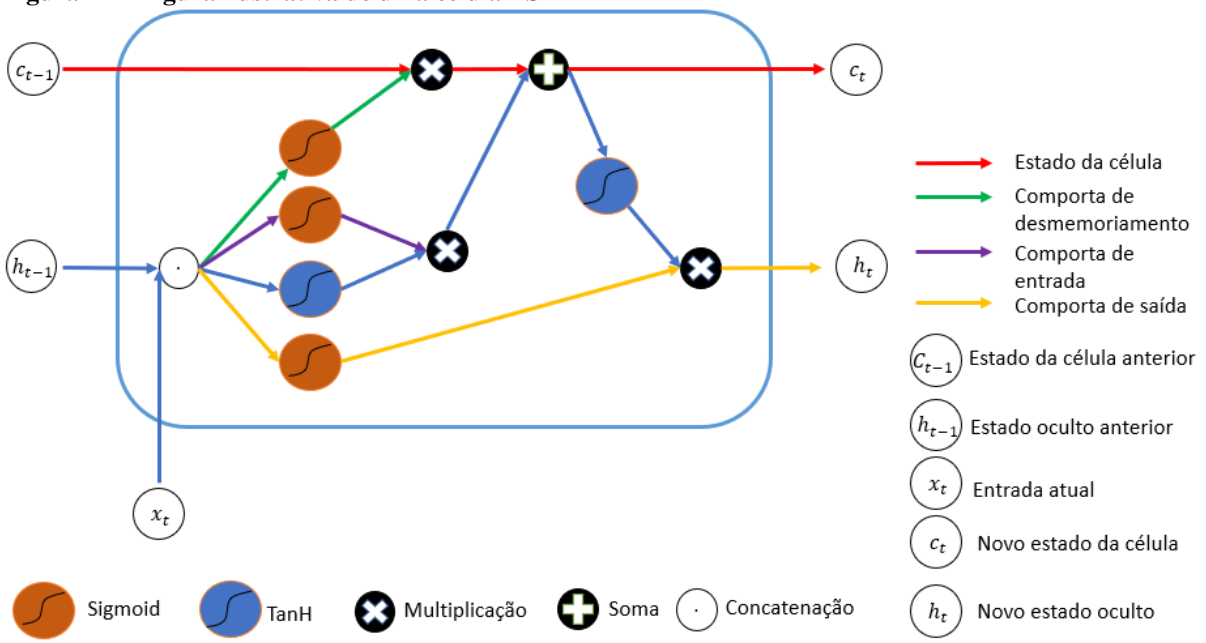
Parkhi *et al.* (2014), em seu sistema de reconhecimento facial, utiliza uma configuração com onze camadas, sendo as oito primeiras de convolução e as últimas três da rede neural totalmente conectada; os filtros possuem tamanhos de 3 a 4096, o tamanho dos saltos dos filtros e do processo de *max pooling* variam entre 1 e 2 (PARKHI *et al.*, 2014). O estudo de Huang *et al.* (2014) para o reconhecimento de emoções na fala, relata o uso de apenas duas camadas convolucionais e utiliza uma SVM (*Support Vector Machine* - máquina de vetores suporte) para realizar a classificação no lugar da camada totalmente conectada.

3.4.2 Redes de Memória de Longo-Curto Prazo - LSTM

As Redes de Memória de Longo-Curto Prazo, ou LSTM (*Long-Short Term Memmory*) são redes neurais recorrentes com células de memória, capaz de aprender dependências de longo prazo criando pontes nos intervalos de tempo, mesmo em casos onde há ruídos ou sequências de entradas incompreensíveis (SCHMIDHUBER; HOCHREITER, 1997). Essa característica faz com que as redes LSTM sejam capazes de predizer novos dados a partir dos dados treinados.

As células de memória são compostas por uma série de operações. Essas operações permitem que a LSTM esqueça ou mantenha informações. A partir dessas informações mantidas, consegue-se ter uma relação da dependência entre os dados para se produzir uma informação nova que se relaciona com os dados fornecidos na entrada. A seguir será apresentado como as células da rede atuam sobre as entradas de dados para memorizar ou descartar as informações. A Figura 17 ilustra graficamente uma célula de memória de uma rede LSTM.

Figura 17 – Figura ilustrativa de uma célula LSTM



Fonte: Autoria própria

Como mencionado anteriormente, a rede LSTM é composta por células. A Figura 17 ilustra uma célula da rede LSTM. Cada célula possui 5 componentes principais: estado da célula (*cell state*), estado escondido (*hidden state*), comporta de entrada (*input gate*), comporta de desmemoriamento (*forget gate*) e a comporta de saída (*output gate*).

As comportas (*gates*) possuem a função de ativação sigmóide (*sigmoid*), que vai projetar os valores recebidos numa escala de 0 a 1. Isso ajuda a atualizar ou esquecer informações, pois qualquer valor multiplicado por zero será anulado (esquecido) e qualquer valor multiplicado por 1 será o mesmo valor, então será mantido (memorizado)(GERS; SCHMIDHUBER; CUMMINS, 1999). Dessa forma a rede consegue aprender quais dados devem ser esquecidos e quais dados devem ser guardados. Logo, as comportas são responsáveis por regular o tráfego de informações nas células.

A comporta de desmemoriamento decide quais informações devem ser descartadas e quais devem ser mantidas (GERS; SCHMIDHUBER; CUMMINS, 1999). Informações do estado oculto anterior e da entrada atual ($h_{t-1} \cdot x_t$) são enviadas para uma função de ativação, resultando em um valor entre zero e um. Quanto mais próximo de zero estiver o valor, maior a probabilidade da informação ser esquecida e quanto mais próximo de 1, maior a probabilidade de ser mantida.

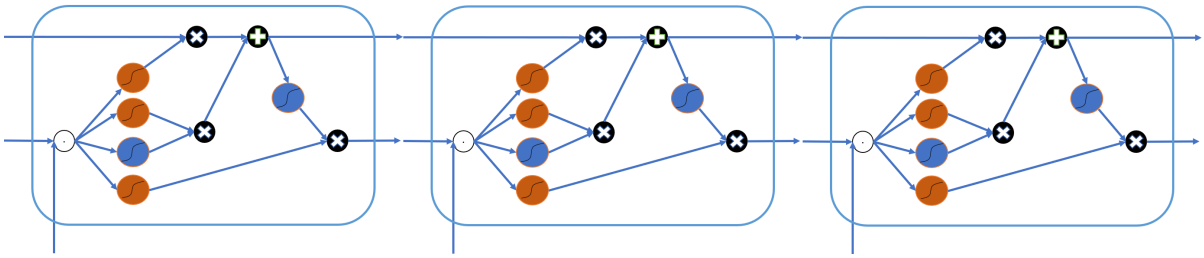
Para atualizar o estado da célula (*cell state*), deve-se calcular o resultado da comporta de entrada (*input gate*). Primeiramente, passa-se o estado oculto da célula anterior e a entrada atual ($h_{t-1} \cdot x_t$) para a função sigmóide, que decide qual valor vai ser atualizado transformando os valores $h_{t-1} \cdot x_t$ num valor entre zero e um, onde zero significa que a informação não é importante e um quer dizer que informação é importante (SCHMIDHUBER; HOCHREITER, 1997). O estado anterior e a entrada atual também são enviados para a função tanh (tangente hiperbólica),

que irá transformar os valores numa escala entre menos um e um. Então multiplica-se a saída da função tanh pela saída da função sigmoide. A função sigmoide irá decidir qual informação da saída da função tanh é importante.

Com as informações da comporta de desmemoriamento e da comporta de entrada, pode-se calcular o estado da célula. Primeiramente, o valor da célula anterior é multiplicado pelo valor da saída da comporta de desmemoriamento, havendo a possibilidade do descarte de valores do estado da célula caso este for multiplicado por um valor próximo a zero. Então soma-se o resultado da comporta de entrada, que irá atualizar os valores da célula de estado, produzindo um novo estado da célula (c_t).

Por último, tem-se a comporta de saída (*output gate*), que decide qual será o próximo estado oculto. O estado oculto contém informações das entradas anteriores, usado para fazer predições (SCHMIDHUBER; HOCHREITER, 1997). Primeiramente o estado oculto da célula anterior e a entrada atual ($h_{t-1} \cdot x_t$) são enviados para a função sigmóide. O novo estado da célula (c_t) é enviado para uma função Tanh. Então, é realizado a multiplicação dos resultados da sigmóide e da tanh. Dessa forma é possível decidir quais informações o estado oculto deve manter (GERS; SCHMIDHUBER; CUMMINS, 1999). O resultado será o novo estado oculto (h_t), que será transmitido para a próxima célula, assim como também o novo estado da célula (c_t). A Figura 18 mostra o exemplo de uma rede LSTM com três células.

Figura 18 – Rede LSTM



Fonte: Autoria própria

A Figura 18 ilustra a conexão realizada entre células de memória, formando uma rede LSTM, onde a saída da célula anterior é a entrada da próxima célula. As multiplicações realizadas pelas comportas de entradas e saídas evitam que as informações irrelevantes continuarem fluindo adiante (SCHMIDHUBER; HOCHREITER, 1997).

As comportas são responsáveis por decidir quando manter, sobrescrever, acessar e prevenir perturbação de informações na memória da célula, possibilitando a rede LSTM extrair informações transmitidas pela ordem temporal das entradas que estão amplamente separadas (GERS; SCHMIDHUBER; CUMMINS, 1999).

A proposta apresentada por este trabalho irá utilizar essa a capacidade de aprendizado de dependências entre os dados para recuperar a sequência de coordenadas de pixels de caracteres manuscritos *offline*.

Durante o traçado de um caractere manuscrito, cada pixel do traçado possui uma coordenada x e y que é uma localização cartesiana do pixel na imagem *offline*. Cada pixel possui

uma conexão com o pixel anterior e pixel posterior. Essa conexão é a ordem em que os pixels são traçados. O conjunto de sequência das coordenadas desses pixels fazem parte da informação *online* do caractere.

A LSTM pode aprender essa ligação que existe entre essas coordenadas. O estado oculto de uma célula LSTM contém informações das entradas anteriores e o estado da célula possui informações da entrada atual. Essas informações são transmitidas na rede LSTM e para cada entrada nova (neste caso, uma coordenada nova) a rede aprende a sua dependência com as entradas anteriores e aprende também a sua dependência com próxima entrada. Com isso, a rede LSTM pode ser capaz de predizer a sequência correta das coordenadas para formar o traçado dos caracteres, recuperando parte da informação *online* que é perdida no processo de escrita *offline*.

4 METODOLOGIA

Neste capítulo são apresentados os elementos necessários para o desenvolvimento deste trabalho. Os métodos usados para o desenvolvimento desta pesquisa serão detalhados nas seções seguintes. Inicialmente, a Seção 4.1 apresenta a base de dados utilizada neste trabalho. Em seguida, a Seção 4.2 as etapas de pré-processamento aplicados na base de dados, detalhando os processos realizados sobre as informações *online* e *offline* da base. A implementação das redes neurais artificiais de aprendizagem profunda é apresentada na Seção 4.3. A Seção 4.4 contém as informações sobre os *frameworks* utilizados e as proporções de treinamento, validação e teste utilizadas nos experimentos.

4.1 BASE DE DADOS IRONOFF

A base de dados utilizada no presente trabalho é a IRONOFF, desenvolvida por Viard-Gaudin *et al.* (1999). Esta base de dados foi escolhida pois possui informações *online* e *offline* de um mesmo caractere manuscrito. As informações *online* são indispensáveis no processo do aprendizado pois é com base nessas informações que a rede pode aprender a ordem dos traçados, utilizando-as como o conjunto verdade (*ground-truth)* durante o processo de aprendizagem, e também como uma forma de verificar a precisão dos resultados.

A base IRONOFF possui 4086 dígitos isolados, 10685 caracteres minúsculos isolados, 10679 caracteres maiúsculos isolados e 31346 palavras isoladas, sendo esse último limitadas a um léxico de 197 palavras diferentes (VIARD-GAUDIN *et al.*, 1999). Os dados *offline* estão disponíveis no formato TIFF na resolução 167x214. As imagens foram escaneadas em uma resolução de 300 dpi com 8 bits por pixel. Os dados *online* estão também numa resolução espacial de 300 dpi, gravados com uma taxa de 100 pontos por segundos, disponíveis em arquivos ASCII no formato UNIPEN. Informações como sexo, idade, nacionalidade e a lateralidade dominante da pessoa também estão disponíveis nos arquivos em ASCII. A Figura 19 apresenta uma amostra de um caractere isolado manuscrito no modo *offline* e sua correspondência de dados no modo *online*.

A Figura 19-a ilustra o arquivo em formato ASCII descrevendo os dados *online* do caractere manuscrito, bem como outras informações como sexo, idade, nacionalidade e a lateralidade dominante (destro/canhoto) da pessoa. A sequência de coordenadas x e y do caractere está entre as etiquetas PEN_DOWN e PEN_UP. A pressão da caneta e também o tempo são disponibilizadas após cada coordenada. A Figura 19-b apresenta a imagem *offline* do caractere.

A caracterização de que o caractere é *single-stroke* ou *multi-strokes* está contido nas informações *online* do caractere. Quando um caractere possui a etiqueta PEN_UP seguida por um PEN_DOWN, significa que o autor da escrita realizou uma interrupção do traçado sobre

Figura 19 – Exemplo de dados *online* e a respectiva imagem da base IRONOFF

```
.DATE       08 07 1998

.COMMENT    Declarations
.X_DIM         3600
.Y_DIM         3600
.X_POINTS_PER_INCH 300
.Y_POINTS_PER_INCH 300
.POINTS_PER_SECOND 100
.COORD      X Y P T

.COMMENT    Data
.WRITER_ID  sk18091962
.COUNTRY    Germany
.HAND       R
.AGE        35
.SEX        M
.SEGMENT CHARACTER ? ? "a"
.PEN_DOWN
88 104 2 0
89 103 12 10
90 101 24 21
90 100 30 31
90 99 39 40
91 97 46 50
92 95 53 62
92 89 57 74
91 86 61 85
90 83 64 97
88 80 67 109
85 78 69 121
80 80 71 132
74 88 70 144
69 101 69 156
64 114 68 167
62 120 68 179
61 127 68 191
62 131 68 203
68 132 67 214
72 130 67 224
81 123 68 234
89 116 68 244
93 113 70 256
94 111 73 272
94 112 75 285
93 119 76 297
92 123 76 308
92 130 76 320
93 136 74 331
96 138 66 340
99 142 15 350
.PEN_UP
```

(a) B1.champs1.unp

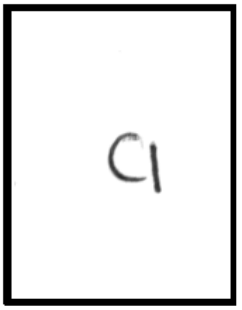
(b) B1.champs1.tif

```
.DATE       11 17 1998

.COMMENT    Declarations
.X_DIM         3600
.Y_DIM         3600
.X_POINTS_PER_INCH 300
.Y_POINTS_PER_INCH 300
.POINTS_PER_SECOND 100
.COORD      X Y P T

.COMMENT    Data
.WRITER_ID  fk16081980
.COUNTRY    France
.HAND       R
.AGE        18
.SEX        M
.SEGMENT CHARACTER ? ? "a"
.PEN_DOWN
97 96 10 0
93 92 94 21
92 91 119 45
84 92 148 57
79 96 154 68
78 98 156 78
77 100 157 88
76 102 159 98
75 109 162 110
75 114 166 121
78 118 173 133
80 120 184 144
83 122 191 155
86 123 191 164
89 123 190 174
94 123 179 186
99 122 70 198
102 120 2 210
.PEN_UP
.PEN_DOWN
106 97 112 352
105 98 155 365
105 102 194 392
105 106 206 402
105 108 211 412
107 113 220 422
107 117 226 432
108 120 228 441
108 122 231 451
108 124 232 462
108 128 232 473
109 129 232 482
109 131 231 490
110 131 207 523
111 129 55 539
.PEN_UP
```

(c) B1.champs17.unp



(d) B221.champs1.tif

Fonte: Autoria própria

a superfície digitalizadora, ocorrendo a perda de contato do objeto utilizado para a escrita com a superfície (PEN_UP) e o contato novamente do objeto com a superfície (PEN_DOWN) para a continuar a escrita. Podemos observar um exemplo disso na Figura 19-c, cujo o caractere 'a', mostrado na Figura 19-d é realizado por dois traços. Isso mostra que uma mesma classe de caractere pode ser escrita tanto por *single-stroke* quanto por *multi-stroke*. Os caracteres de *single-stroke* são um subgrupo dos caracteres de *multi-strokes*, uma vez que os caracteres de *multi-strokes* são caracteres de um ou mais traços.

Neste trabalho, utiliza-se apenas os caracteres minúsculos da base, pois apesar de um caractere minúsculo 'a' e um caractere maiúsculo 'A' serem da mesma classe, em uma recupe-

ração de trajetória, ambos são escritos de maneiras diferentes e possuem uma forma geométrica diferente. Além disso, serão feitos experimentos separando os caracteres de *single-stroke*, realizando um treinamento e recuperação separado do conjunto todo (*multi-stroke*).

Para proceder com os experimentos, é necessário que a base de dados passe por processos de refinamento, para que informações relevantes sejam evidenciadas, padronizadas e normalizadas, melhorando o processo de treinamento das redes *deep learning*. Esse processo é conhecido como pré-processamento, detalhado na seção a seguir.

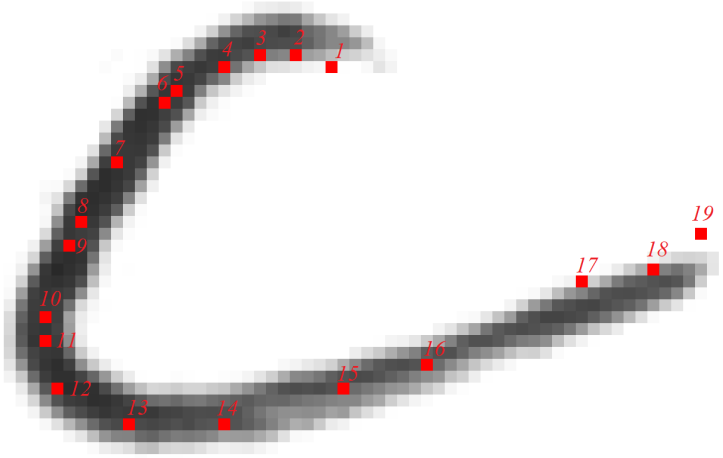
4.2 PRÉ-PROCESSAMENTO

Como mencionado anteriormente, as imagens da base IRONOFF necessitam passar por alguns processos para que informações relevantes sejam padronizadas e normalizadas. As imagens da base estão na dimensão 167x214. O conteúdo principal do caractere não ocupa a imagem toda, podendo haver informações irrelevantes na imagem. Uma forma de contornar isso é realizar o pré-processamento das imagens, assim, apenas as informações relevantes de cada caractere são utilizadas.

A escrita humana nem sempre possui tamanhos iguais. Utilizar uma caixa delimitadora nos caracteres pode resultar em imagens de tamanhos diferentes. Isto impacta diretamente na entrada de uma rede neural, uma vez que as entradas geralmente são de um número de neurônios previamente definido. Uma das técnicas usadas para solucionar este problema é redimensionar as imagens para que fiquem todas no mesmo tamanho, para que possam ser enviadas para a rede, obedecendo o número de neurônios de entrada da rede. Redimensionar as imagens para a entrada de uma rede neural artificial é uma prática usada em diversos trabalhos na literatura, como é o caso de Bhunia *et al.* (2018), Zheng *et al.* (2016) e Huang *et al.* (2014).

A Figura 20 mostra a relação dos dados *online* e *offline* de uma amostra da base.

Figura 20 – Caractere *offline* em escala de cinza com a ordenação do seu trajeto *online*.



Fonte: Autoria própria

Pode ser observado na Figura 20 que as coordenadas *online* do caractere não percorre o centro do traçado original (o esqueleto do caractere) e os pontos não estão equidistantes um dos outros. Logo, as duas informações não estão totalmente equivalentes.

Para que os dados *offline* e *online* estejam em sintonia, utiliza-se as coordenadas x e y das informações *online* para redesenhar os caracteres nas imagens *offline*. Criando imagens *offline* a partir das informações *online* faz com que ambas possam estar em equivalência. Essa estratégia é utilizada por trabalhos como Dinh *et al.* (2016), Bhunia *et al.* (2018), Wang *et al.* (2019) e Zhao, Yang e Tao (2019). A normalização da quantidade de pontos de cada caractere também deve ser realizada, deixando as distâncias entre uma coordenada e a próxima coordenada na sequência sejam iguais. A normalização reduz a estimativa do erro e diminui o tempo do processo de treinamento das redes neurais (SOLA; SEVILLA, 1997).

Neste trabalho, então, utiliza-se apenas as coordenadas x e y das informações *online* para a criação das imagens *offline*. Todos os experimentos são conduzidos utilizando as informações *online* e estas transformadas em *offline* para servir de entrada para as redes de *deep learning*. O pré-processamento é realizado nas coordenadas *online* e consiste das seguintes etapas: redimensionamento, normalização de pontos e *Data Augmentation*. Os dois primeiros processos são para padronizar a base de dados, enquanto que este último é para geração de novas amostras para reduzir o erro por falta de generalização da rede neural artificial. A seguir será detalhada cada uma dessas etapas.

4.2.1 Redimensionamento

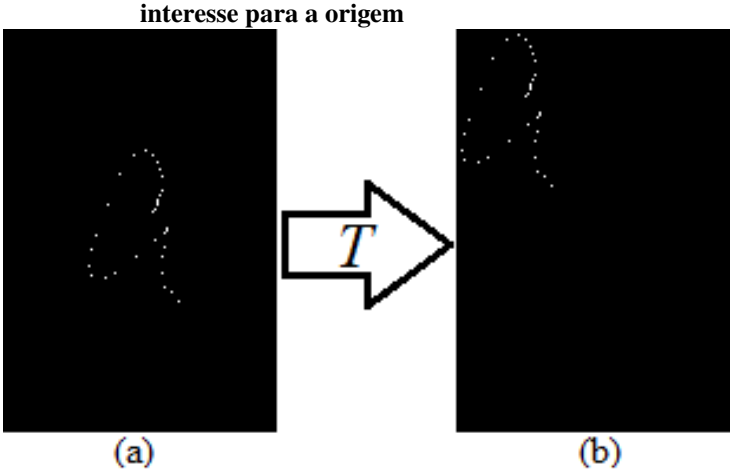
O redimensionamento consiste em deixar as coordenadas dentro de um limite desejado. Como visto na seção 4.1, os caracteres estão centralizados em uma imagem de resolução 167x214, o mesmo acontece com as coordenadas *online*. Para realizar o redimensionamento das coordenadas, primeiramente, é realizado o processo de translação (T) dos pares de coordenadas do caractere até a origem. Sendo S o conjunto de pares ordenados $[x, y]$ que representam a sequência de pixels do caractere, Sx_{min} e Sy_{min} os menores valores de x e y de S e $[x_i, y_i]$ um pixel do caractere. A equação 4.1 apresenta o cálculo para a translação de S para o primeiro quadrante iniciando em $[0, 0]$.

$$ST(i) = S(i) - (Sx_{min}, Sy_{min}) = (Sx_i - Sx_{min}, Sy_i - Sy_{min}) \tag{4.1}$$

A partir do cálculo da equação 4.1, a região de interesse da imagem passa a estar perto da origem, como mostrado na Figura 21.

A Figura 21 mostra as sequências de coordenadas de um caractere desenhadas em uma imagem que passa pela operação geométrica de translação. A translação do objeto de interesse, que neste caso são os pixels da imagem, é realizada em todas as coordenadas, dessa forma, o

Figura 21 – Ilustração do processo de translação da região de interesse para a origem



Fonte: Autoria própria

caractere é transladado para o primeiro quadrante perto da origem.

Neste trabalho, as coordenadas são enquadradas dentro da dimensão 64x64. Sendo w e h as dimensões da imagem, $ratioW$ a razão $w/64$ e $ratioH$ a razão $h/64$, os novos valores de um ponto $[x_i, y_i]$ de uma sequência de coordenadas S é calculada conforme a equação 4.2.

$$S(x_i, y_i) = \left(\frac{x_i}{ratioW}, \frac{y_i}{ratioH}\right) \tag{4.2}$$

Com isso, a base de dados já está em consonância com o tamanho da entrada das redes neurais convolucionais utilizadas neste trabalho. Resta realizar mais duas etapas de pré-processamento, detalhadas nas seções 4.2.2 e 4.2.3.

4.2.2 Normalização

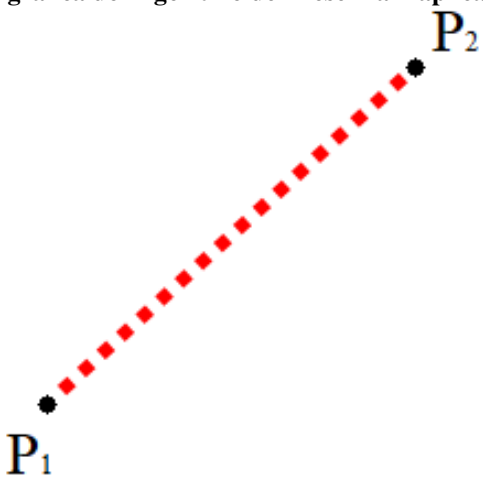
Esta etapa tem como objetivo padronizar a quantidade de pares de coordenadas de todos os caracteres da base. Este processo ajuda no processo de aprendizagem da rede, pois na predição de pontos, a quantidade de parâmetros que a rede deve atualizar será sempre a mesma.

O processo de normalização consiste no uso de dois algoritmos: o Algoritmo de Bresenham (BRESENHAM, 1965) e o cálculo da distribuição uniforme em um intervalo.

Dado dois pontos $P_1(x_1, y_1)$ e $P_2(x_2, y_2)$ em um espaço n-dimensional, o algoritmo de Bresenham é um algoritmo que produz uma série de pontos adicionais para conectar os pontos P_1 e P_2. Dessa maneira, consegue-se produzir uma linha contínua, sem intervalos esparsos entre os pontos. A Figura 22 apresenta graficamente o resultado do algoritmo de Bresenham.

Na Figura 22, os pontos P_1 e P_2 são conectados por uma série de pontos (indicados em vermelho) produzidos pelo algoritmo de Bresenham, o qual conecta as duas extremidades formando uma linha.

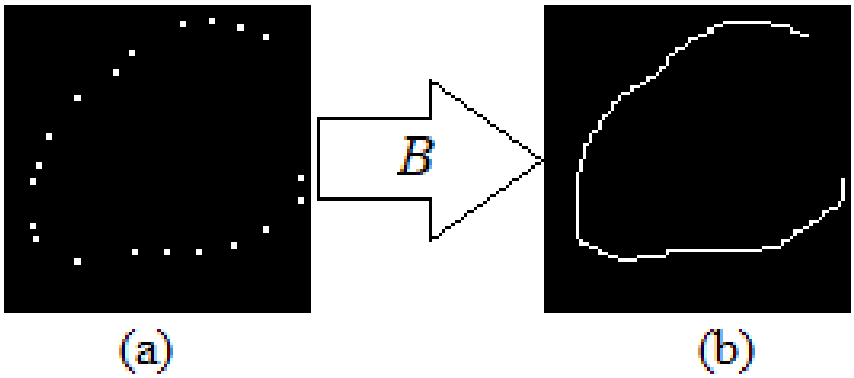
Figura 22 – Representação gráfica do Algoritmo de Bresenham aplicado em dois pontos



Fonte: Autoria própria

Como visto na Figura 20, entre os pontos *online* do caractere, podem ser gerados mais pontos para conectar duas coordenadas, produzindo um esqueleto do caractere. A Figura 23 ilustra o resultado do algoritmo de Bresenham aplicado a uma sequência de pares de coordenadas de um caractere.

Figura 23 – Representação gráfica do Algoritmo de Bresenham aplicado em todas as coordenadas do caractere



Fonte: Autoria própria

Na Figura 23-a temos graficamente as coordenadas *online* S de um caractere. Sendo P_i uma coordenada da sequência S e n a quantidade de coordenadas de S, o algoritmo de Bresenham é aplicado nos pontos P_1 e P_2, P_2 e P_3, e assim sucessivamente até P_{n-1} e P_n, produzindo uma nova sequência S_b que possui a sequência de coordenadas do traçado do esqueleto do caractere, mostrado na Figura 23-b.

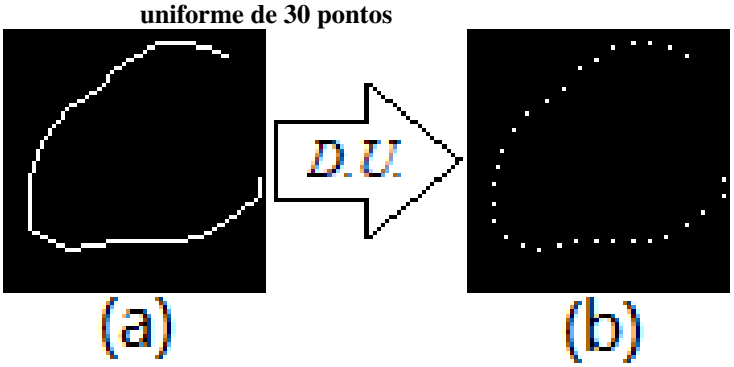
O algoritmo de Bresenham conecta todos os pontos do caractere para que, posteriormente, seja possível calcular uma distribuição uniforme de todo o conjunto.

A distribuição uniforme[1] consiste em, dada uma constante c e um intervalo $[inicio, fim]$, calcular c números distribuídos uniformemente entre $inicio$ e fim. Esses números são usados

[1] Função numpy.linspace de https://docs.scipy.org/doc/numpy/reference/generated/numpy.linspace.html

como índices da sequência S_b, tendo assim uma distribuição uniforme de coordenadas. A Figura 24 ilustra o resultado deste processo.

Figura 24 – Representação gráfica do resultado da distribuição uniforme de 30 pontos



Fonte: Autoria própria

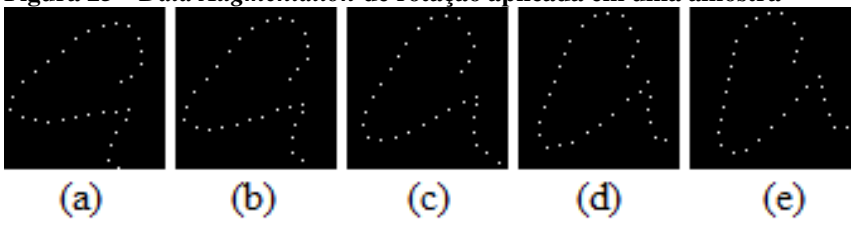
Como pode ser observado na Figura 24, a distribuição uniforme das coordenadas produz um resultado normalizado das coordenadas do caractere. Na Figura 24-b, trinta coordenadas são calculadas para que as distâncias entre dois pontos P_i e P_{i+1} do caractere sejam equidistantes. Este processo é realizado em todas as amostras da base de dados.

4.2.3 *Data Augmentation*

A técnica de ampliação de dados (em inglês, *data augmentation*) é utilizada como um método para aumentar a quantidade de amostras de uma base de dados, ampliando a capacidade de generalização da rede de aprendizagem. Dentre várias técnicas existentes na literatura, este trabalho utilizará apenas a técnica de rotação.

Como o problema se limita a trajetória de pontos de coordenadas, outras técnicas podem alterar demasiadamente o conteúdo principal da imagem. Caso seja aplicada a transformação de escala, as extremidades dos caracteres podem ultrapassar o limite do tamanho da imagem, pois já foi aplicado a técnica de *boudingbox* para retirar a área de interesse. O mesmo acontece com a translação: se movimentarmos os caracteres para alguns pixels em qualquer direção, o caractere perderá parte de sua estrutura e a rede estará aprendendo um caractere que pode não existir no alfabeto latino. A Figura 25 ilustra exemplos do resultado das rotação aplicadas em uma das amostras da base.

Figura 25 – *Data Augmentation* de rotação aplicada em uma amostra



Fonte: Autoria própria

Na Figura 25 tem-se uma amostra que passou por quatro categorias de rotação. Da Figura 25-a até a Figura 25-e, temos a representação gráfica das rotações de $30°$, $15°$, $0°$, $-15°$ e $-30°$ das coordenadas de uma amostra, sendo a Figura 25-c, de $0°$, a imagem que representa as coordenadas no seu estado original, sem rotação. A rotação aplicada inclina o a amostra no sentido horário e anti-horário. Para cada amostra, são gerados quatro novos casos, o que aumenta a base de dados em cinco vezes comparado à quantidade original da base.

Como são caracteres latinos, rotacionar ou até mesmo espelhar um caractere pode fazer com que este se torne um outro caractere. Por exemplo, espelhar o caractere 'b' escrito em letra de forma, transforma-se em 'd'; o mesmo acontece com 'p', que vira 'q'; em outros casos, ao rotacionar a imagem contendo o caractere 'c' em $90°$ acabam virando 'u'. Outros casos podem não afetar, por exemplo espelhar os caracteres 'e' e 'i' de manuscritos cursivos. Neste sentido, optou-se por não utilizar o espelhamento e não utilizar mais rotações pois como a rede aprende a trajetória, espelhar ou rotacionar demasiadamente as amostras implica no aprendizado de outras formas de caracteres, isso pode afetar no desempenho e o tempo de aprendizado da rede, além do fato de que este trabalho tem-se como objetivo realizar somente o aprendizado de traçado de caracteres do alfabeto latino.

4.2.4 Considerações da Seção

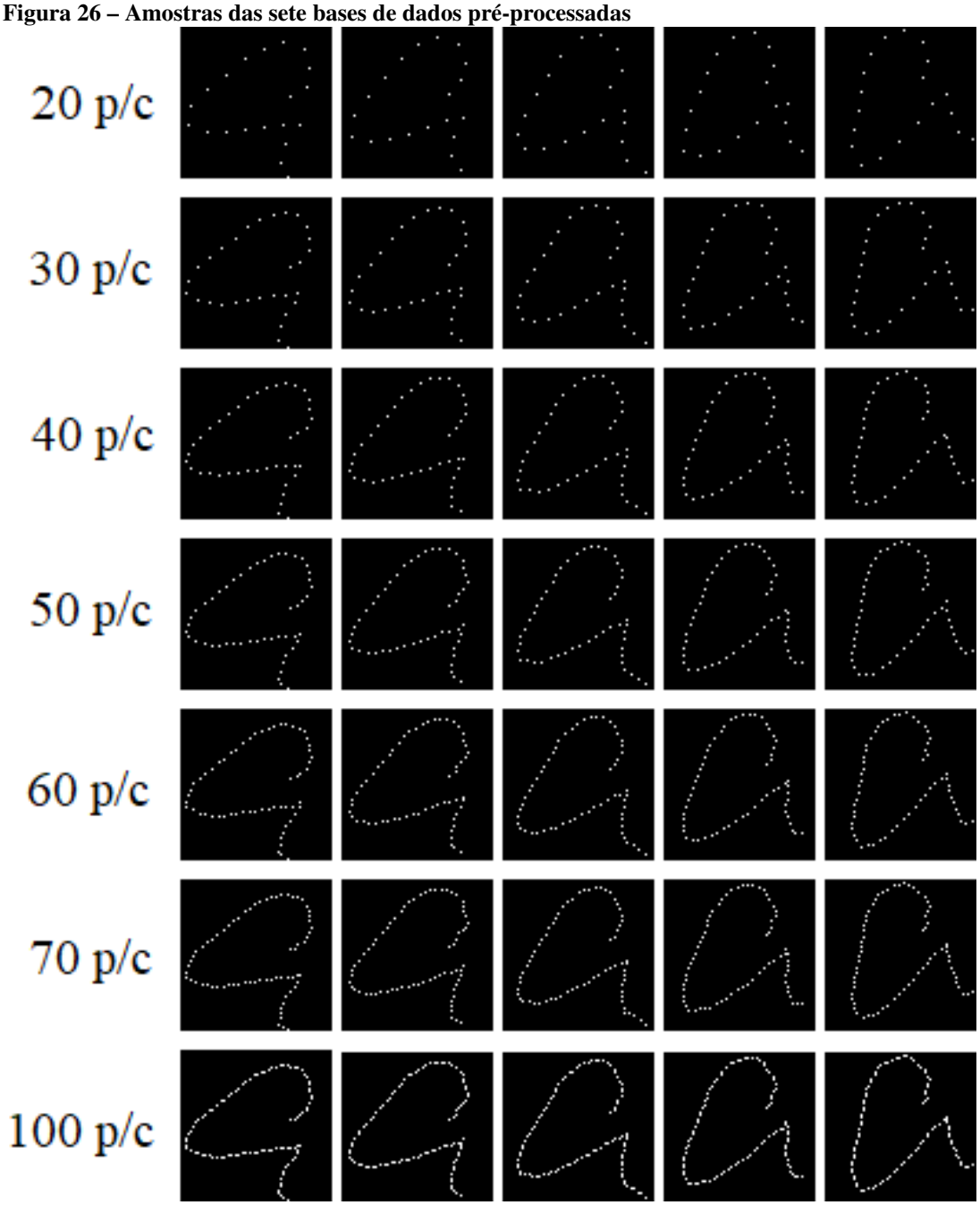
Nesta seção descreveu-se o pré-processamento realizado na base de dados IRONOFF. Como já comentado no subitem 4.2.2, deseja-se predizer uma quantidade fixa de pontos iguais para todos os caracteres, por esta razão utilizou-se o algoritmo de Bresenham e feito uma distribuição uniforme.

A normalização da distribuição uniforme pode trazer uma quantidade fixa de pontos para todas as amostras da base. Logo, optou-se por utilizar essa propriedade para gerar novas bases de dados. Como parte do experimento, um conjunto de números foi escolhido para produzir sete base de dados diferentes: 20 pontos por caractere (p/c), $30p/c$, $40p/c$, $50p/c$, $60p/c$, $70p/c$ e $100p/c$, com o objetivo de analisar o comportamento das redes de aprendizagem profunda sobre diferentes quantidades de coordenadas a serem preditas. A Figura 26 mostra exemplos de amostras das sete bases.

Na Figura 26, temos exemplos das sete bases que são utilizadas neste trabalho. Cada uma delas é usada separadamente nas redes de aprendizagem profunda para o treinamento e testes. Desta forma, pode-se analisar qual é a melhor quantidade de pontos que deve ser fixada, uma vez que diminuir ou aumentar a quantidade de pontos não altera o formato geométrico do caractere, porém, interfere diretamente nos pesos que serão atualizados dentro das redes neurais.

Com a técnica de *data augmentation*, conseguiu-se gerar bases com cinco vezes mais amostras que a original, totalizando 53425 amostras em cada base. A base completa constitui-se de caracteres consideradas *multi-strokes* (ou seja, de um ou mais traços). A recuperação de

Figura 26 – Amostras das sete bases de dados pré-processadas



Fonte: Autoria própria

trajetória também será realizada sobre caracteres de *single-stroke* (ou seja, de apenas um traço), o qual totaliza 40975 amostras.

4.3 ARQUITETURA DE REDES NEURAIS ARTIFICIAIS DE *DEEP LEARNING*

As arquiteturas de aprendizagem profunda presentes na literatura apontam que é possível realizar a recuperação de informações *online* a partir de uma imagem *offline*. Elbaati, Hamdi e Alimi (2019) eBhunia *et al.* (2018) utilizam uma CNN para a extração de características de ima-

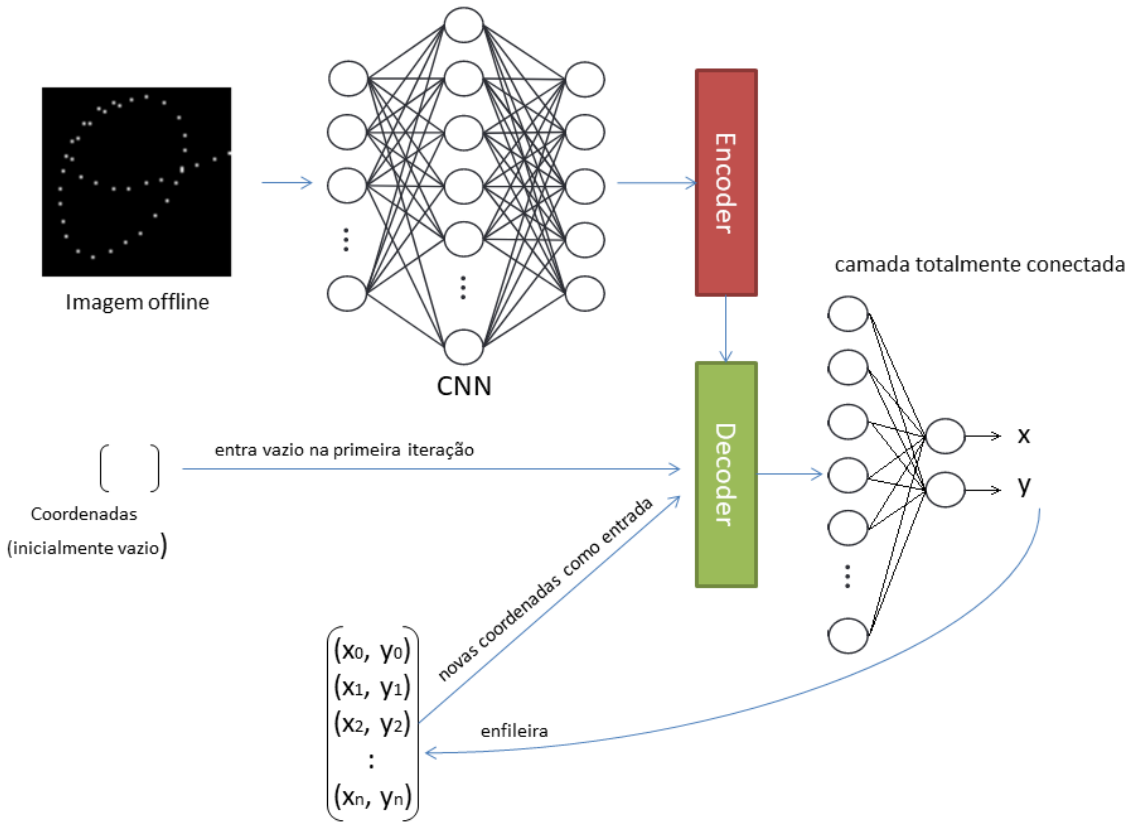
gens. As características extraídas são interpretadas por uma rede LSTM, que produz a sequência de coordenadas da trajetória do caractere da imagem.

A partir dessa estrutura hierárquica reportada pelos trabalhos mencionados acima, este trabalho pretende explorar essa estrutura com novos parâmetros, realizando o treinamento de novos pesos em um novo contexto, para o alfabeto latino.

4.3.1 CNN e LSTM

Após o preprocessamento realizado na base IRONOFF, as imagens de predição são inseridas na CNN, que tem como objetivo de extrair as características de alto nível, e neste caso, as características extraídas representam a sequência de coordenadas dos pixels. As camadas convolucionais produzem um conjunto de características da imagem de entrada. Esse conjunto (vetor) de características servirá de entrada para a rede LSTM, que irá interpretar e traduzir para a sequência de coordenadas de pixels. A Figura 27 apresenta um grafo com o processo de predição.

Figura 27 – Redes de CNN e LSTM *encoder-decoder* para predição de coordenadas



Fonte: Autoria própria

Na Figura 27, a CNN recebe a imagem *offline* construída a partir de informações *online*

e produz um vetor de características, que será a entrada da rede LSTM *encoder-decoder* no modelo *seq2seq* proposto por Sutskever, Vinyals e Le (2014), parametrizado por uma rede LSTM bidirecional proposta por Graves, Jaitly e Mohamed (2013). A LSTM unidirecional pode não obter os resultados pois captura informações sequenciais apenas da esquerda para a direita, logo, a bidirecional é adotada para aprender dependências nas duas direções (BHUNIA *et al.*, 2018).

Cada camada da LSTM possui 256 células. A primeira rede LSTM é uma rede codificadora. O estado oculto da última célula LSTM é a saída codificada dessa rede, o qual também é a entrada da rede LSTM decodificadora. A segunda rede LSTM é a rede decodificadora.

A ideia básica dessa arquitetura é gerar uma saída de tamanho diferente do tamanho da entrada, pois dada uma entrada (que no nosso caso é de tamanho 256), a rede codificadora irá codificar o vetor de características de alta dimensão em um estado oculto. A rede decodificadora recebe este estado oculto na primeira célula da cadeia para decodifica-lo e usá-lo para produzir novos estados decodificados que servirão de entrada para as próximas células da própria cadeia da rede decodificadora, treinando e aprendendo as sequência de dependências de dados para a predição de coordenadas.

Para gerar as coordenadas, para cada intervalo de tempo t da saída da rede LSTM decodificadora, aplica-se uma camada totalmente conectada, com 256 neurônios na primeira camada (recebendo as saídas das células da rede LSTM decodificadora) e dois neurônios de saída. Esta saída representa os pontos x e y a cada intervalo de tempo t. O modelo gera continuadamente como saída os pontos de coordenadas preditos, produzindo a sequência inteira de traços do caractere.

Após gerada a sequência de coordenadas, é realizado o cálculo do custo para atualizar os valores dos pesos da rede. No problema em questão, este cálculo permite dizer o quanto a rede está acertando nas predições das coordenadas. Quanto menor o custo, menos deve-se ajustar os pesos da rede. Utiliza-se a função da distância L1 apresentada na Equação 4.3, onde realiza-se o cálculo para cada coordenada predita $\vec{z_t}$ com sua respectiva coordenada *ground-truth* $\vec{Z}_t$.

$$L = \frac{1}{n'} \sum_{t=1}^{n'} ||\vec{z_t} - \vec{Z}_t|| \tag{4.3}$$

4.3.2 Configurações das Redes

Neste trabalho foram implementadas cinco configurações de redes distintas para treinar nas sete diferentes bases de dados criadas, totalizando 35 redes neurais artificiais de aprendizagem profunda. Inicialmente foi reproduzida a rede proposta por Bhunia *et al.* (2018) (neste trabalho chamaremos de *netBh*) e, a partir disso, outras quatro redes foram desenvolvidas. A arquitetura da primeira rede, nomeada de *net-v0*, possui 6 camadas convolucionais e 3 camadas de LSTM em cada *encoder-decoder*; a segunda rede, nomeada de *net-v1* possui 8 camadas

convolucionais e 3 camadas de LSTM em cada *encoder-decoder*; a terceira rede *net-v2* possui 12 camadas convolucionais e 3 camadas LSTM em cada *encoder-decoder*; a quarta e última rede *net-v3* possui 16 camadas convolucionais e 3 camadas LSTM em cada *encoder-decoder*. A Tabela 2 apresenta a quantidade de neurônios presente em cada camada de cada rede.

Tabela 2 – Quantidade de neurônios de cada camada das redes implementadas.

Camada	*netBh*	*net-v0*	*net-v1*	*net-v2*	*net-v3*
conv1	4096	4096	4096	4096	4096
conv2	64	64	64	64	64
conv3	128	128	128	128	128
conv4	256	256	256	256	256
conv5	256	256	256	256	256
conv6	256	256	256	256	256
conv7	-	-	256	256	256
conv8	-	-	256	256	256
conv9	-	-	-	256	256
conv10	-	-	-	256	256
conv11	-	-	-	256	256
conv12	-	-	-	256	256
conv13	-	-	-	-	256
conv14	-	-	-	-	256
conv15	-	-	-	-	256
conv16	-	-	-	-	256
lstm1-*Encoder*	2x256	3x256	3x256	3x256	3x256
lstm2-*Decoder*	2x256	3x256	3x256	3x256	3x256
fullconn1	256	256	256	256	256
fullconn2	2	2	2	2	2

Fonte – Autoria própria

Na *netBh*, o *max-pooling* é realizado em todas as camadas convolucionais menos na conv6. A normalização de lotes (*batch normalization*) é realizada nas conv3, conv4 e conv6. A rede *encoder-decoder* possui no total quatro camadas, duas para *encoder* (lstm1-*Encoder*) e duas para a *decoder* (lstm2-*Decoder*). A camada totalmente conectada *fullconn1* recebe os valores da saída dos neurônios da lstm2-*Decoder* que irá computar as saídas para a última camada de dois neurônios, o qual produz dois valores: x e y, sendo elas as coordenadas de um pixel.

A segunda rede é chamada de *net-v0*. Esta rede possui um pequeno diferencial em relação a *netBh*: ao invés de ser duas camadas em cada *encoder-decoder*, a *net-v0* possui três camadas de LSTM em cada *encoder-decoder*. A convolucionais seguem a mesma configuração da *netBh*.

A terceira rede é chamada de *net-v1*. Esta rede possui 8 camadas convolucionais e 3 camadas de LSTM em cada *encoder-decoder*, igual na rede *net-v0*. O *max-pooling* é realizado em todas as camadas menos na conv3, conv6 e conv8. A normalização de lotes é aplicada nas camadas conv3, conv6 e conv8.

A quarta rede, *net-v2*, possui 12 camadas convolucionais. O *max-pooling* é aplicado

em todas as camadas convolucionais, menos na conv3, conv6, conv8 e conv12. Normalização de lotes na conv3, conv6, conv8 e conv12. Possui três camadas de LSTM em cada *encoder-decoder*.

A quinta e última rede, nomeada de *net-v3*, possui 16 camadas convolucionais. Nas convolucionais da *net-v3*, o *max-pooling* é aplicado em todas as camadas, menos na conv3, conv6, conv8 e conv16. A normalização é realizada na conv3, conv6, conv8 e 16. Também possui três camadas de LSTM em cada *encoder-decoder*.

O intuito de trabalhar com mais camadas convolucionais é fazer com que a rede seja capaz de melhorar a extração de características de informações de entradas mais robustas (como as bases de $70p/c$ e $100p/c$). Como foram criados sete base de dados, sendo cada uma delas contendo mais informações em cada amostra (20 pontos por caractere, 30 pontos por caractere, etc.), intuitivamente, as redes configuradas tiveram a quantidade de camadas da CNN aumentadas (6, 8, 12 e 16 camadas de convolução) para analisar o comportamento dessas configurações nas diferentes bases.

4.4 FRAMEWORKS E TREINAMENTO

Para a implementação deste trabalho, foi utilizado a linguagem *Python*[2] e o framework *open source* de aprendizagem de máquina *Tensorflow*[3]. O framework *OpenCV*[4] também é utilizado para realizar as operações nas imagens.

Os experimentos foram realizados em uma máquina com a placa GPU da Nvidia GeForce GTX Titan de 6GB, um processador i7-3770K de 3.50GHz e 8GB de memória RAM. Todas as cinco redes são treinadas com 200 épocas com a validação cruzada. A divisão das bases consiste em 70% de treinamento, 15% validação e 15% de testes.

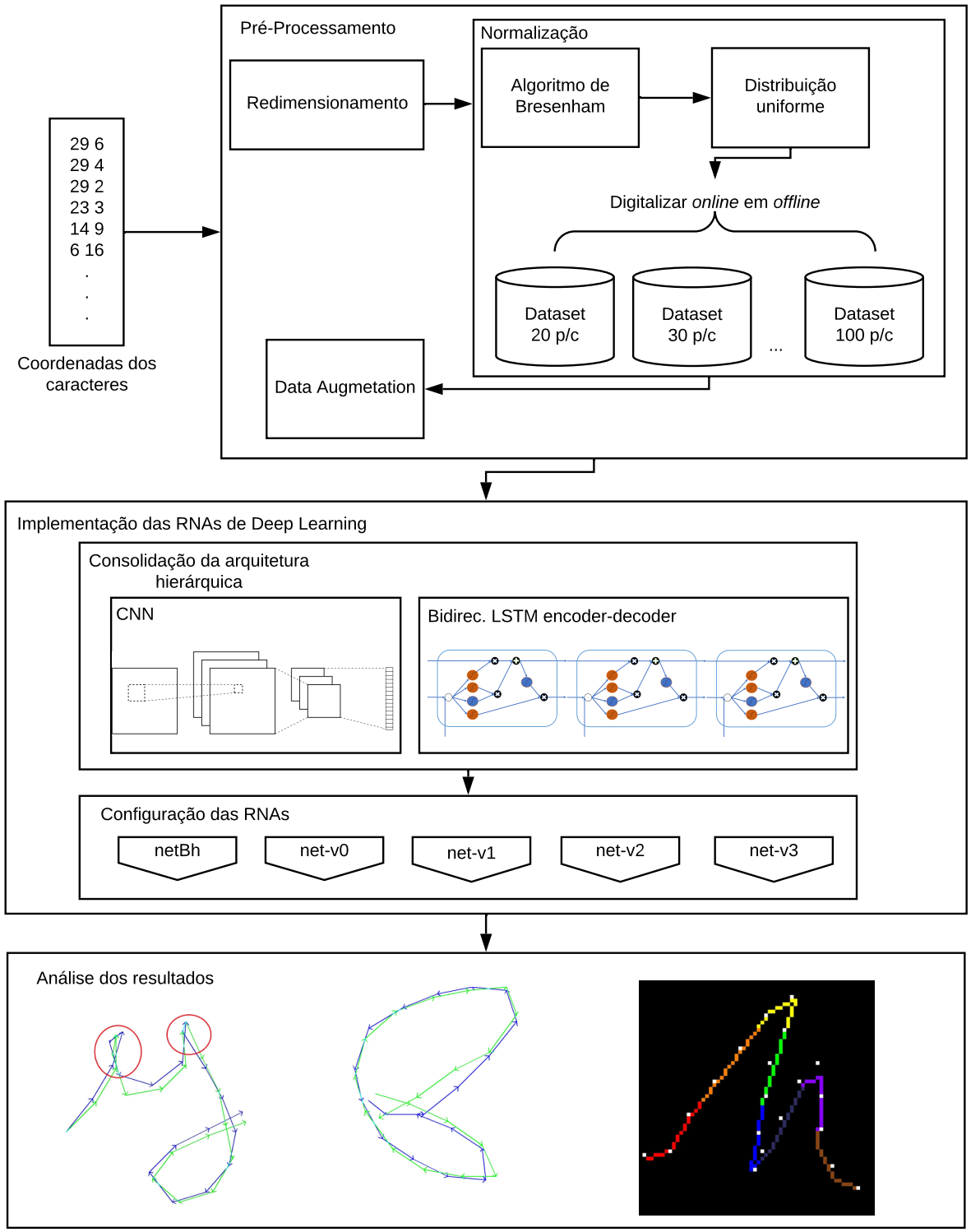
No total, são treinadas 70 redes (5 configurações de redes por 7 bases de *single-stroke* e 7 bases de *multi-strokes*). O processo de treinamento durou cerca de oito horas para a *netBh* com a base de $20p/c$ *single-stroke* e trinta e duas horas para *net-v3* com a base de $100p/c$ *multi-strokes*.

4.5 CONSIDERAÇÕES FINAIS DO CAPÍTULO

Este capítulo descreveu os detalhes de cada etapa para produzir os resultados da pesquisa. Na Figura 28 tem-se a ilustração dos processos realizados nesta pesquisa, includindo a etapa de análise dos resultados.

[2] https://www.python.org/
[3] https://www.tensorflow.org/
[4] https://opencv.org/

Figura 28 – Visão geral do método proposto para a recuperação de caracteres manuscritos



Fonte: Autoria própria

Em suma, é realizado primeiramente o pré-processamento, que consiste em redimensionamento, normalização e *data augmentation* das informações *online*. Essas informações são pixeladas em matrizes digitais, tornando-se informações *offline*. Com o uso algoritmo de Bresenham para a normalização, foi possível criar sete bases diferentes, sendo cada uma delas contendo uma quantidade fixa de pontos (específica para cada base) para cada amostra. Em seguida,

implementa-se as redes neurais artificiais de aprendizagem profunda, o qual consiste numa relação hierárquica de redes convolucionais para a extração de características e redes LSTM bidirecionais, que aprendem a dependência de dados temporais. Cinco redes foram criadas para treinar as 14 bases de dados (7 de *single-stroke* e 7 de *multi-strokes*), totalizando em 70 redes treinadas. As redes são diferentes em quantidade de camadas, sendo possível verificar qual obteve melhor resultado em cada uma das bases de dados criadas. Por fim, realiza-se a análise dos resultados, que serão discutidos com detalhes no Capítulo 5 e Capítulo 6.

5 RESULTADOS

Neste capítulo discute-se os resultados quantitativos e qualitativos inicias obtidos a partir dos experimentos realizados para a predição de trajetória de coordenadas de caracteres com um traço. Primeiramente, as métricas de avaliação utilizadas nesta fase são descritas na Seção 5.1. Em seguida, os resultados quantitativos das bases de caracteres *single-stroke* e *multi-stroke* são apresentados na Seção 5.2. Na Seção 5.3 são apresentadas as taxas de acerto por classe de caractere. A análise qualitativa da avaliação aplicada em amostras são apresentadas na Seção 5.4, mostrando o comportamento da avaliação em trajetória de caracteres *latinos*.

5.1 MÉTODO AVALIATIVO SP-JP-CT

Para o cálculo quantitativo dos resultados iniciais, são utilizadas as métricas de avaliação formalizadas por Bhunia *et al.* (2018) e Rousseau, Camillerapp e Anquetil (2006), descritas na Seção 3.2.1. São elas: SP (*Starting Point* - taxa de acerto de pontos de início), JP (*Junction Point* - taxa de acerto de pontos de junção) e CT (*Complete Trajectory* - taxa de acerto da trajetória total).

Nos resultados apresentados neste capítulo, a taxa de trajetória completa (CT) é resultado do número total de amostras preditas corretamente dividido pelo número total de amostras da base de teste. Esta informação é importante uma vez que a avaliação encontrada na literatura não divide a quantidade de trajetórias completas pela quantidade total de amostras, mas sim, pelo número de amostras que apresentam o ponto de início correto (SP).

Decidiu-se realizar o cálculo dessa taxa de maneira diferente em comparação da encontrada na literatura pois reflete melhor o que realmente a rede está predizendo. Quando se utiliza a quantidade de amostras que apresentam SP corretos como dividendo do cálculo, aumenta-se a taxa de acerto de CT, o qual resulta em uma taxa que não é em relação ao número total de amostras da base de teste. Por exemplo, em um cenário onde uma base possui 100 amostras de teste e a rede acertar 80 pontos de início e 75 trajetórias corretas. Na métrica original, a taxa CT seria calculada por $CT = (75/80)$, o que resultaria em uma taxa CT de $93,75\%$. Neste trabalho, a taxa CT será calculada por $CT = (75/100)$, dividindo o acerto pela quantidade total de amostras da base de testes, resultando em uma taxa CT de 75%, mostrando o real acerto da rede sobre a base inteira.

Optou-se por utilizar a avaliação SP-JP-CT pois é um método detalhado na literatura por Bhunia *et al.* (2018) e Rousseau, Camillerapp e Anquetil (2006) o qual se utiliza o conjunto verdade para produzir um valor que reflete de maneira mais fiel a acurácia das coordenadas preditas. Alguns trabalhos da literatura utilizam medidas de comparação de séries temporais ou estatísticos, como DTW (Dynamic Time Warping) descrito em Giorgino *et al.* (2009) e é

utilizado por Zhao, Yang e Tao (2019) e Phan *et al.* (2015), ou o RMSE (*Root Mean Square Error*), descrito em Chai e Draxler (2014) e utilizado por Phan *et al.* (2015) e o coeficiente de correlação de Pearson, descrito em Mukaka (2012) e utilizado por Wang *et al.* (2019). Entretanto, esse tipo de cálculo não estabelece uma linha de corte para saber se o resultado da predição está correto ou errado. Portanto, os autores que utilizam esse tipo de métrica mostram suas taxas mediante a diferentes valores de corte estabelecidos por eles. Com a avaliação *SP-JP-CT* não é necessário levantar um coeficiente de corte, pois são medidas que comparam diretamente cada coordenada predita com o *ground-truth*.

5.2 RESULTADOS QUANTITATIVOS

Nesta seção serão apresentados os resultados obtidos das redes nas sete bases produzidas ($20p/c$, $30p/c$, $40p/c$, $50p/c$, $60p/c$, $70p/c$ e $100p/c$) de caracteres *single-stroke* e *multi-strokes*. Para cada rede, é apresentado a tabela contendo as taxas de acertos de pontos de início (SP), pontos de junção (JP), trajetória completa (CT) e a média desses valores em cada base de dados comentada na seção 4.2.4. Ao final desta seção, será apresentado o gráfico com o conjunto das taxas de acertos CT de todas as redes.

5.2.1 Resultados SP-JP-CT da *netBh*

Os primeiros resultados a serem apresentados são os resultados da rede *netBh*, que consiste na rede com as configurações proposta por Bhunia *et al.* (2018). A Tabela 3 apresenta as taxas de acertos SP, JP e CT.

Tabela 3 – Taxas de acerto da rede *netBh* nas sete bases

Base	**SP**		**JP**		**CT**		**Média**	
	single	***multi***	***single***	***multi***	***single***	***multi***	***single***	***multi***
$20p/c$	$93,60\%$	$93,13\%$	$85,38\%$	$84,65\%$	***84,56%***	***82,21%***	$87,85\%$	$86,67\%$
$30p/c$	$92,91\%$	$91,71\%$	$85,09\%$	$83,42\%$	$79,22\%$	$76,25\%$	$85,74\%$	$83,80\%$
$40p/c$	$90,92\%$	$91,02\%$	$82,04\%$	$81,39\%$	$71,86\%$	$67,85\%$	$81,61\%$	$80,09\%$
$50p/c$	$91,53\%$	$90,12\%$	$81,28\%$	$79,05\%$	$67,76\%$	$61,62\%$	$80,19\%$	$76,93\%$
$60p/c$	$90,94\%$	$89,43\%$	$79,40\%$	$76,14\%$	$56,12\%$	$48,71\%$	$75,49\%$	$71,43\%$
$70p/c$	$90,38\%$	$88,97\%$	$76,78\%$	$73,61\%$	$49,32\%$	$43,16\%$	$72,16\%$	$68,58\%$
$100p/c$	$89,13\%$	$87,87\%$	$68,82\%$	$64,01\%$	$31,99\%$	$20,81\%$	$63,31\%$	$57,56\%$

Fonte – Autoria própria

Pode-se observar na Tabela 4 que as taxas de acertos são maiores nos pontos de início e pontos de junção, porém, a trajetória completa não consegue valores acima de 85%. Todos os valores vão decaindo conforme o número de pontos por caractere da base de dados vai aumentando. Logo, pode-se concluir que quanto maior for a quantidade de informações (quantidade

de pontos do caractere) que a rede deverá aprender, mais difícil será para ela conseguir realizar a predição.

As resultados de caracteres *multi-strokes* não apresentaram melhores resultados em comparação aos caracteres de *single-strokes* para o mesmo p/c. Por serem o conjunto total de caracteres de um ou mais traços, a complexidade aumenta. A rede passa a tentar entender que existem caracteres cujo possuem interrupções de traçados que podem iniciar perto ou longe do local interrompido. Nos resultados da Tabela 3, apenas a SP de *multi-strokes* na base $40p/c$ superou por 1 ponto percentual a $40p/c$ *single-stroke*.

A rede *netBh* foi projetada para caracteres Telugo com caracteres de 50 pontos de coordenada. Conforme os resultados apresentados na Tabela 3, entende-se que é possível melhorar os resultados de Bhunia *et al.* (2018) diminuindo a quantidade de pontos por caractere.

Os resultados apresentados nas próximas seções são resultados de configurações de redes criadas para o desenvolvimento desta pesquisa, visando melhorar as taxas de acertos de HTR das bases de caracteres latinos.

5.2.2 Resultados SP-JP-CT da *net-v0*

Os resultados obtidos pela rede *net-v0* são apresentados na Tabela 4.

Tabela 4 – Taxas de acerto da rede *net-v0* nas sete bases

Base	SP		JP		CT		Média	
	single	***multi***	***single***	***multi***	***single***	***multi***	***single***	***multi***
$20p/c$	$93,13\%$	$92,39\%$	$85,14\%$	$84,04\%$	***84,65%***	***82,37%***	$87,64\%$	$86,27\%$
$30p/c$	$92,72\%$	$92,07\%$	$84,53\%$	$83,90\%$	$79,17\%$	$77,56\%$	$85,47\%$	$84,51\%$
$40p/c$	$91,51\%$	$90,73\%$	$83,16\%$	$81,54\%$	$74,33\%$	$69,21\%$	$83,00\%$	$80,49\%$
$50p/c$	$91,25\%$	$90,06\%$	$80,98\%$	$78,66\%$	$68,02\%$	$63,63\%$	$80,09\%$	$77,45\%$
$60p/c$	$91,61\%$	$89,23\%$	$79,70\%$	$76,51\%$	$60,84\%$	$58,46\%$	$77,39\%$	$74,73\%$
$70p/c$	$90,73\%$	$88,39\%$	$76,44\%$	$72,17\%$	$46,02\%$	$38,45\%$	$71,06\%$	$66,34\%$
$100p/c$	$89,46\%$	$87,56\%$	$68,60\%$	$64,20\%$	$29,87\%$	$20,11\%$	$62,64\%$	$57,29\%$

Fonte – Autoria própria

Comparando os dados da Tabela 4 com a Tabela 3, a rede *net-v0* conseguiu um aumento da taxa de acerto em algumas bases. Com a base $20p/c$ *multi-strokes* conseguiu-se um aumento de $0,16$ pontos percentuais em comparação com a $20p/c$ *multi-strokes* da *netBh*. Para a base de $40p/c$ também pode ser observada um aumento de acerto para ambos *single-stroke* e *multi-strokes*, chegando a uma diferença de quase $2,5$ pontos percentuais de aumento.

A diferença entre essas duas redes está na quantidade de camadas na rede LSTM em cada *encoder-decoder*. Logo, nesta etapa pode-se afirmar que, com mais neurônios nas camadas *encoder-decoder*, a rede conseguiu interpretar melhor as características extraídas pela CNN e pode realizar uma predição mais precisa. O mesmo ocorre com os resultados *multi-strokes*, que conseguem também maiores taxas em CT nas bases $20p/c$, $40p/c$, $50p/c$ e $60p/c$.

5.2.3 Resultados SP-JP-CT da *net-v1*

Na rede *net-v1* aumentou-se o número de camadas da CNN com o objetivo de melhorar a extração de características das amostras de entrada. A Tabela 5 apresenta os resultados obtidos da rede *net-v1*.

Tabela 5 – Taxas de acerto da rede *net-v1* nas sete bases

Base	SP		JP		CT		Média	
	single	***multi***	***single***	***multi***	***single***	***multi***	***single***	***multi***
$20p/c$	$93,31\%$	$93,15\%$	$85,37\%$	$84,94\%$	***85,17%***	***83,39%***	$87,95\%$	$87,16\%$
$30p/c$	$92,67\%$	$91,86\%$	$85,17\%$	$83,55\%$	$81,18\%$	$77,22\%$	$86,34\%$	$84,21\%$
$40p/c$	$90,58\%$	$90,47\%$	$81,80\%$	$81,00\%$	$72,27\%$	$68,08\%$	$81,55\%$	$79,85\%$
$50p/c$	$91,16\%$	$90,31\%$	$81,10\%$	$79,06\%$	$66,42\%$	$62,52\%$	$71,32\%$	$77,30\%$
$60p/c$	$90,67\%$	$88,58\%$	$78,72\%$	$75,99\%$	$54,84\%$	$51,29\%$	$74,74\%$	$71,95\%$
$70p/c$	$90,35\%$	$88,10\%$	$76,93\%$	$72,76\%$	$55,04\%$	$44,71\%$	$74,11\%$	$68,53\%$
$100p/c$	$89,30\%$	$87,14\%$	$69,40\%$	$64,07\%$	$39,55\%$	$23,86\%$	$66,09\%$	$58,36\%$

Fonte – Autoria própria

Nos dados apresentados na Tabela 5 pode-se observar um aumento nas taxas de acurácia CT em algumas bases, como em $20p/c$, $30p/c$, $70p/c$ e $100p/c$ em comparação com a rede *net-v0*. Através disso, conseguiu-se verificar que a rede com mais camadas de CNN consegue apresentar melhores resultados para bases com amostras de mais pontos por caracteres. Pode-se afirmar que as camadas de convolução a mais da CNN podem ter conseguido extrair melhor as características adicionais contidas nas imagens, entretanto, os resultados da *net-v1* ainda não superam os resultados para as bases de $40p/c$, $50p/c$ e $60p/c$.

5.2.4 Resultados SP-JP-CT da *net-v2*

Na Tabela 6 são apresentados os resultados da rede *net-v2*.

Tabela 6 – Taxas de acerto da rede *net-v2* nas sete bases

Base	SP		JP		CT		Média	
	single	***multi***	***single***	***multi***	***single***	***multi***	***single***	***multi***
$20p/c$	$93,05\%$	$92,63\%$	$85,14\%$	$84,23\%$	***85,01%***	***82,38%***	$87,74\%$	$86,41\%$
$30p/c$	$92,28\%$	$90,05\%$	$84,37\%$	$82,70\%$	$79,23\%$	$76,13\%$	$85,29\%$	$82,96\%$
$40p/c$	$91,57\%$	$87,48\%$	$82,79\%$	$80,48\%$	$73,05\%$	$68,68\%$	$82,47\%$	$78,88\%$
$50p/c$	$91,60\%$	$84,90\%$	$81,08\%$	$78,73\%$	$64,11\%$	$62,03\%$	$78,93\%$	$75,22\%$
$60p/c$	$91,56\%$	$82,33\%$	$79,03\%$	$76,85\%$	$56,19\%$	$55,18\%$	$75,59\%$	$71,45\%$
$70p/c$	$91,31\%$	$79,75\%$	$76,91\%$	$74,98\%$	$48,37\%$	$48,33\%$	$72,20\%$	$67,69\%$
$100p/c$	$88,63\%$	$69,44\%$	$66,84\%$	$59,07\%$	$25,70\%$	$20,94\%$	$60,39\%$	$49,82\%$

Fonte – Autoria própria

Em comparação com a *net-v1*, os resultados da *net-v2* da média e da taxa de acertos CT são melhores nas bases de $40p/c$, $60p/c$ e $70p/c$.

5.2.5 Resultados SP-JP-CT da *net-v3*

Na Tabela 7 são apresentados os resultados obtidos das predições da rede *net-v3*.

Tabela 7 – Taxas de acerto da rede *net-v3* nas sete bases

Base	SP		JP		CT		Média	
	single	***multi***	***single***	***multi***	***single***	***multi***	***single***	***multi***
$20p/c$	$92,54\%$	$92,27\%$	$83,81\%$	$83,49\%$	***81,89%***	***80,84%***	$86,08\%$	$85,53\%$
$30p/c$	$91,90\%$	$90,60\%$	$83,26\%$	$81,03\%$	$76,00\%$	$71,08\%$	$83,72\%$	$80,91\%$
$40p/c$	$90,99\%$	$83,93\%$	$81,12\%$	$78,57\%$	$67,81\%$	$61,32\%$	$79,97\%$	$74,61\%$
$50p/c$	$88,90\%$	$82,26\%$	$77,72\%$	$76,11\%$	$57,92\%$	$51,57\%$	$74,84\%$	$69,98\%$
$60p/c$	$90,32\%$	$73,91\%$	$76,80\%$	$73,65\%$	$51,57\%$	$41,81\%$	$72,90\%$	$63,13\%$
$70p/c$	$89,69\%$	$72,24\%$	$74,19\%$	$71,19\%$	$44,68\%$	$32,06\%$	$69,52\%$	$58,50\%$
$100p/c$	$88,62\%$	$70,58\%$	$65,32\%$	$58,89\%$	$25,28\%$	$19,46\%$	$59,74\%$	$49,64\%$

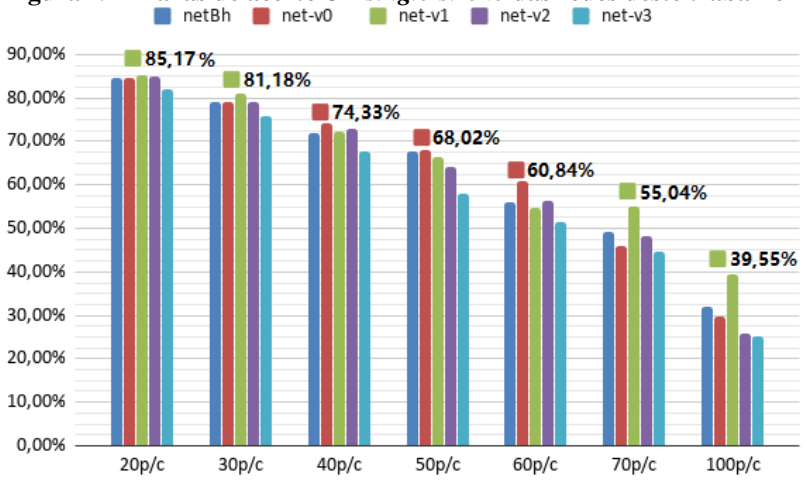
Fonte – Autoria própria

A rede *net-v3* não apresentou resultados melhores em comparação à outras redes. Por ser a rede que contém o maior número de camadas convolucionais em sua arquitetura, acredita-se que o tempo de treinamento deve ser ampliado para conseguir um melhor ajustes dos pesos dos neurônios e conseguir maior capacidade de generalização. Esse fator também pode ser levado em consideração para a rede *net-v2*.

5.2.6 Resultados CT *single-stroke*

O gráfico com os resultados das taxas de acerto das bases de *single-stroke* de todas as redes treinadas deste trabalho é mostrado na Figura 29.

Figura 29 – Taxas de acerto *CT single-stroke* das redes deste trabalho



Fonte: Autoria própria

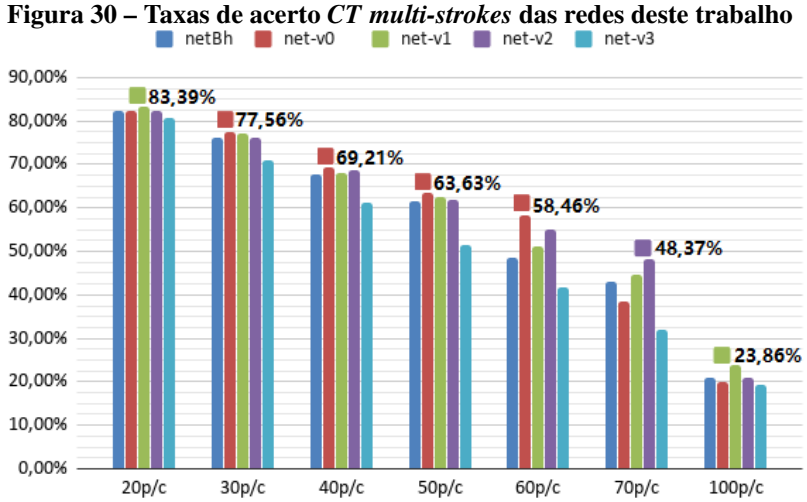
Pelo gráfico ilustrado na Figura 29 pode-se observar um comparativo entre as redes em cada base. A *net-v1* atingiu uma maior diferença nas taxas CT para as bases de $70p/c$ e $100p/c$

em comparação com as outras redes. Isso mostra que, baseado nas variáveis de treinamento, validação e testes, ela consegue generalizar melhor sobre dados mais robustos. Além disso, alcançou melhores resultados também para a base de $20p/c$ e $30p/c$.

5.2.7 Resultados CT *multi-strokes*

A Figura 30 apresenta um gráfico com os resultados das taxas de acerto das bases de *multi-strokes* de todas as redes treinadas deste trabalho.

Figura 30 – Taxas de acerto *CT multi-strokes* das redes deste trabalho



Fonte: Autoria própria

Para as bases de *multi-stroke*, a *net-v2* alcançou o melhor resultado na base $70p/c$. A *net-v1* continua sendo melhor na $20p/c$. Para as bases de $30p/c$, $40p/c$, $50p/c$ e $60p/c$ a rede *net-v0* atingiu melhores resultados.

Vale ressaltar que todas as redes passaram pelas mesmas configurações de treinamento, validação e testes. Isso quer dizer que algumas redes podem necessitar de mais épocas de treinamento comparadas às outras ou necessitar de mais amostras para o conjunto de validação, para conseguir uma maior capacidade de generalização.

Na seção seguinte, apresenta-se a CT por caracteres *single-stroke* e *multi-strokes* das redes na base $20p/c$, podendo ser observado o comportamento de cada rede sobre cada classe de caractere.

5.3 RESULTADOS POR CARACTERES COM A AVALIAÇÃO CT

Nesta seção apresenta-se os resultados de CT por caractere das redes treinadas na base $20p/c$ de *single-stroke* e *multi-stroke*. Escolheu-se apresentar apenas os resultados da base de $20p/c$ pois foi a base onde as redes obtiveram maiores taxas de acerto. Vale ressaltar que, neste

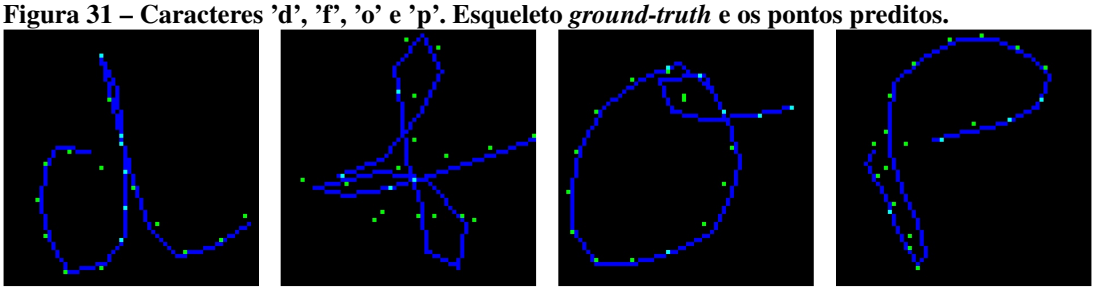
trabalho, os experimentos são conduzidos com os caracteres minúsculos da base IRONOFF e, para uma mesma classe de caractere, pode-se ter a escrita em *single-stroke* e *multi-stroke*, como descrito na Seção 4.1. As taxas de acertos são apresentados na Tabela 8. O cálculo do desvio padrão também é apresentado para mostrar o grau de dispersão de cada base *single-stroke* e *multi-strokes* em cada rede.

Pode-se observar na Tabela 8 que para a classe de caracteres *single-stroke* 'i' conseguiu-se acertos de 100%. Esse constitui-se de um conjunto pequeno de caracteres (composto apenas por cinco caracteres). Isso acontece pois são poucos autores que escrevem o caractere 'i' com apenas um traço. Geralmente, o caractere 'i' é realizado por dois traços, que é o caso da base *multi-strokes* da IRONOFF, que compõem-se de 345 caracteres.

Verificando os melhores resultados por caractere *single-stroke*, a *net-v2* apresenta 9 melhores resultados, *net-v1* e *netBh* com 8, empate no caractere 'i' em todas as redes e a *net-v3* obteve o melhor resultado no caractere 'h'. Para *multi-strokes*, 12 melhores resultados foram obtidos pela rede *net-v1*, 6 foram dos resultados da *net-v2*, 4 da *net-v0* e as redes *netBh* e *net-v3* ficaram com 2 melhores resultados cada.

Segundo os dados da Tabela 8, os caracteres de classes 'd', 'f', 'o' e 'p' são caracteres que obtiveram menores taxas de acerto. Esses são caracteres que possuem laços ou que possuem traços que se sobrepõem ou que estão muito próximos um dos outros, fazendo com que as redes passem por um desafio maior para aprender e predizer essas classes de caracteres. A Figura 31 mostra essas situações.

Figura 31 – Caracteres 'd', 'f', 'o' e 'p'. Esqueleto *ground-truth* e os pontos preditos.



(a) (b) (c) (d)

Fonte: Autoria própria

Podemos observar pela Figura 31-b que o caractere da classe 'f' é bastante complexo em laços e traços que estão muito próximos um dos outros, dificultando a predição. Os outros caracteres são caracteres que possuem suas predições parecidas com o esqueleto original, porém são consideradas predições incorretas. Essas avaliações levou a necessidade de realizar uma análise mais detalhada do sistema de avaliação SP-JP-CT, utilizado até o momento.

Tabela 8 – Taxas de acerto por caractere da base $20p/c$ utilizando a métrica CT

Caractere	*netBh*		*net-v0*		*net-v1*		*net-v2*		*net-v3*		*Best Single*	*Best Multi*
	single	*multi*	*single*	*multi*	*single*	*multi*	*single*	*multi*	*single*	*multi*		
a	84,15%	72,84%	75,85%	72,84%	69,06%	73,73%	80,38%	70,45%	76,60%	67,76%	84,15%	73,73%
b	78,71%	77,91%	71,94%	78,51%	80,65%	76,72%	83,55%	74,93%	75,81%	74,33%	83,55%	78,51%
c	99,40%	96,86%	99,40%	98,29%	99,70%	98,86%	99,40%	98,00%	99,10%	96,86%	99,70%	98,86%
d	62,63%	59,37%	52,63%	57,46%	55,79%	60,95%	67,37%	62,86%	62,63%	62,86%	67,37%	62,86%
e	82,06%	83,19%	77,35%	80,00%	85,00%	84,64%	82,94%	80,87%	80,29%	82,61%	85,00%	84,64%
f	65,20%	62,57%	58,40%	61,14%	60,80%	64,00%	66,40%	62,00%	57,60%	54,29%	66,40%	64,00%
g	74,33%	67,35%	65,00%	71,47%	65,00%	72,94%	73,33%	74,41%	67,00%	72,35%	74,33%	74,41%
h	96,51%	96,97%	90,79%	98,18%	93,65%	97,88%	96,19%	96,97%	96,51%	97,58%	96,51%	98,18%
i	100,00%	82,32%	100,00%	86,38%	100,00%	84,93%	100,00%	83,48%	100,00%	81,74%	100,00%	86,38%
j	85,71%	69,86%	77,14%	70,72%	91,43%	68,99%	88,57%	65,80%	77,14%	61,45%	91,43%	70,72%
k	91,79%	88,70%	80,00%	88,99%	81,54%	92,75%	88,72%	88,12%	87,18%	88,99%	91,79%	92,75%
l	86,57%	85,35%	78,57%	87,89%	84,29%	88,73%	86,29%	87,32%	85,71%	84,51%	86,57%	88,73%
m	90,14%	88,17%	82,32%	91,27%	80,00%	93,24%	91,59%	92,68%	90,14%	90,42%	91,59%	93,24%
n	96,18%	96,57%	93,53%	97,71%	97,06%	98,57%	97,65%	98,57%	97,06%	97,43%	97,65%	98,57%
o	50,00%	51,90%	51,67%	46,47%	56,94%	54,08%	49,44%	51,09%	41,11%	51,09%	56,94%	54,08%
p	52,77%	52,90%	42,13%	51,61%	48,09%	51,29%	48,94%	50,32%	41,70%	43,87%	52,77%	52,90%
q	84,79%	87,07%	80,56%	85,85%	83,66%	83,66%	84,79%	81,95%	81,41%	83,66%	84,79%	87,07%
r	93,85%	92,57%	91,69%	92,86%	94,46%	93,14%	94,77%	93,43%	94,15%	91,71%	94,77%	93,43%
s	95,87%	93,73%	94,92%	91,94%	98,10%	91,64%	97,14%	93,73%	93,65%	89,85%	98,10%	93,73%
t	93,04%	87,67%	81,74%	89,59%	89,57%	90,41%	91,30%	88,77%	85,22%	86,85%	93,04%	90,41%
u	96,39%	94,85%	93,11%	93,64%	94,10%	94,55%	95,41%	95,15%	95,08%	95,45%	96,39%	95,45%
v	96,72%	94,71%	94,63%	95,29%	97,01%	95,00%	98,51%	96,18%	95,52%	93,82%	98,51%	96,18%
w	93,21%	92,00%	82,14%	92,31%	87,14%	92,31%	95,00%	92,62%	92,86%	91,38%	95,00%	92,62%
x	75,29%	81,52%	60,00%	81,21%	80,00%	80,91%	80,00%	80,30%	78,82%	81,82%	80,00%	81,82%
y	85,42%	89,86%	84,41%	89,86%	90,85%	91,88%	89,49%	91,59%	87,12%	88,12%	90,85%	91,88%
z	87,00%	88,00%	81,67%	87,38%	87,33%	89,54%	85,00%	88,31%	84,00%	76,00%	87,33%	89,54%
Média	84,56%	82,21%	84,65%	82,37%	85,17%	83,39%	85,01%	82,38%	81,89%	80,84%	86,33%	84,03%
Desvio Padrão	13,81	13,69	15,46	14,58	14,76	13,83	13,99	14,28	16,18	15,14	13	13,59
Mediana	86,79%	87,37%	81,12%	87,64%	86,07%	89,14%	88,64%	87,72%	85,47%	84,08%	91,14%	89,14%

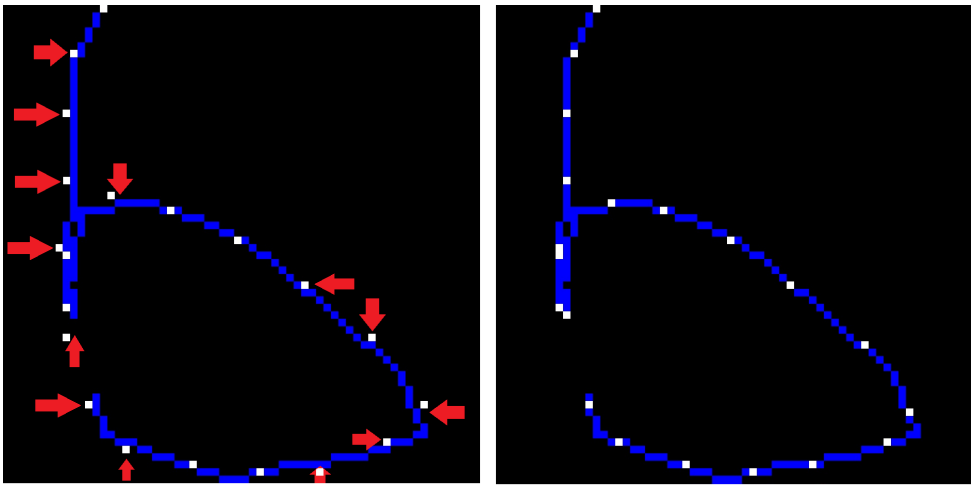
Fonte – Autoria própria

Além dos resultados quantitativos de *single-stroke* e *multi-stroke*, foi realizada também uma análise qualitativa, que consiste em verificar visualmente as imagens formadas pela predição e as imagens dos caracteres originais, acompanhando a trajetória dos pixels de ambos. Nessa análise, cerca de 15% das amostras foram avaliadas como falso negativas ou falsos positivas, ou seja, existem amostras que deveriam ser consideradas verdadeiras e incluídas nas taxas de trajetória completa, porém são consideradas falsas e também amostras que deveriam ser consideradas falsas, porém foram consideradas verdadeiras. A Seção 5.4 a seguir apresentará com detalhes algumas dessas amostras.

5.4 ANÁLISE QUALITATIVA

Para avaliar qualitativamente as amostras, é necessário relembrar que o método existente na literatura para avaliar as coordenadas oriundas de sistemas de HTR utilizando *deep learning* realiza um processo de pós-processamento antes de aplicar as métricas de avaliação, comentada na Seção 3.2.1. Esse processo consiste em traduzir os pontos preditos para o ponto mais próximo do esqueleto da imagem offline (neste trabalho, este processo é chamado de Tradução de Pixels). A Figura 32 ilustra esse processo de tradução.

Figura 32 – Tradução dos pixels preditos no esqueleto do conjunto verdade do caractere



(a) (b)

Fonte: Autoria própria

A tradução dos pixels nada mais é do que uma operação geométrica dos mesmos. O motivo de se aplicar essa operação pode ser observado na Figura 32-a. Nem todos os pontos preditos (em branco) estão dentro do esqueleto do conjunto verdade (em azul). A predição muitas vezes não consegue acertar exatamente o local do esqueleto, então é necessário realizar uma operação geométrica sobre essas coordenadas. Na Figura 32-a mostra os locais mais próximos em que as coordenadas preditas estão do conjunto verdade, indicando (por setas vermelhas) para onde devem ser movidas. A Figura 32-b apresenta o resultado final deste processo de tradução,

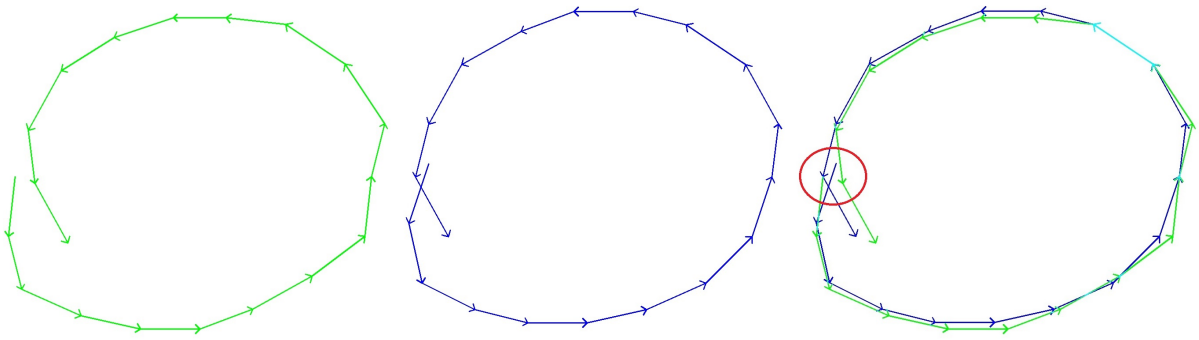
onde todos os pixels preditos passam a pertencer parte do trajeto original do esqueleto.

Na maioria dos casos, este processo funciona para se conseguir avaliar as coordenadas preditas. Entretanto, foi verificado que essa métrica acusa resultados incorretos quando na verdade são corretos (falsos negativos) e também acusa resultados corretos quando na verdade não são (falsos positivos). Alguns desses casos serão apresentados nas seções a seguir, mostrando o motivo pelo qual o modelo de avaliação existente pode não ser o suficiente para computar os resultados de um sistema de HTR.

5.4.1 Falsos Negativos Ocasionados pela Tradução de Pixels

A tradução dos pixels preditos para o mais próximo do esqueleto, comentado anteriormente na Seção 5.4, pode traduzir o ponto predito para o segmento errado do esqueleto. Essa tradução errada é causada pela sobreposição (laços) de traços durante o processo de escrita. Na Figura 33 temos um exemplo de sobreposição de traços.

Figura 33 – Trajetórias do caractere 'o' original (em verde), da predição (em azul) e a sobreposição dos dois



(a) (b) (c)

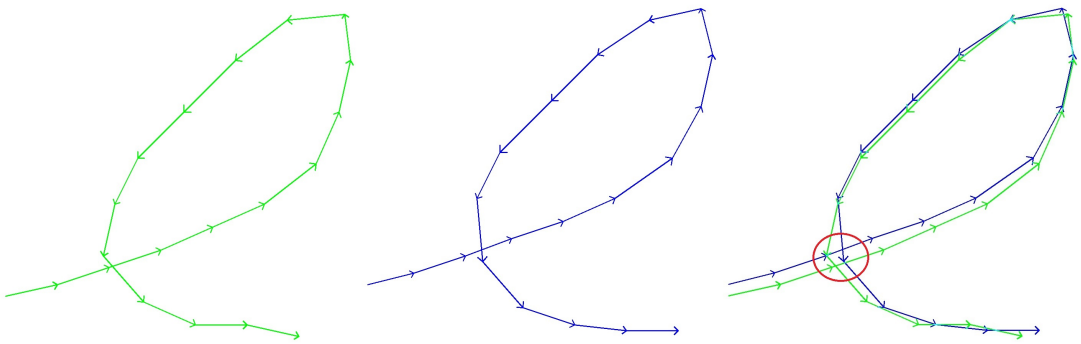
Fonte: Autoria própria

Na Figura 33, a sequência e direção dos traços são indicados pelas setas. Na cor verde temos o conjunto verdade do caractere (Figura 33-a) e em azul temos o a predição realizada pela rede (Figura 33-b. Pode-se observar que o ponto de início que a rede prediz é diferente da original, porém, todo o resto da trajetória está correta. Quando realiza-se a tradução, os pontos da área destacada em vermelho (na Figura 33-c) entram em conflito: o ponto inicial é traduzido para o final da trajetória, pois está mais próximo do traço final do conjunto verdade dele.

Esse tipo de laço ocorre pela peculiaridade da escrita do autor; um 'o' perfeitamente escrito teria seu ponto de início e final iguais. Porém essa não é uma situação que acontece com frequência na realidade.

Em um outro caractere que apresenta laço em sua estrutura de escrita e não depende da peculiaridade de escrita do autor é mostrado na na Figura 34 com o caractere 'e' escrita de maneira cursiva.

Figura 34 – Trajetórias do caractere 'e' original (em verde), da predição (em azul) e a sobreposição dos dois

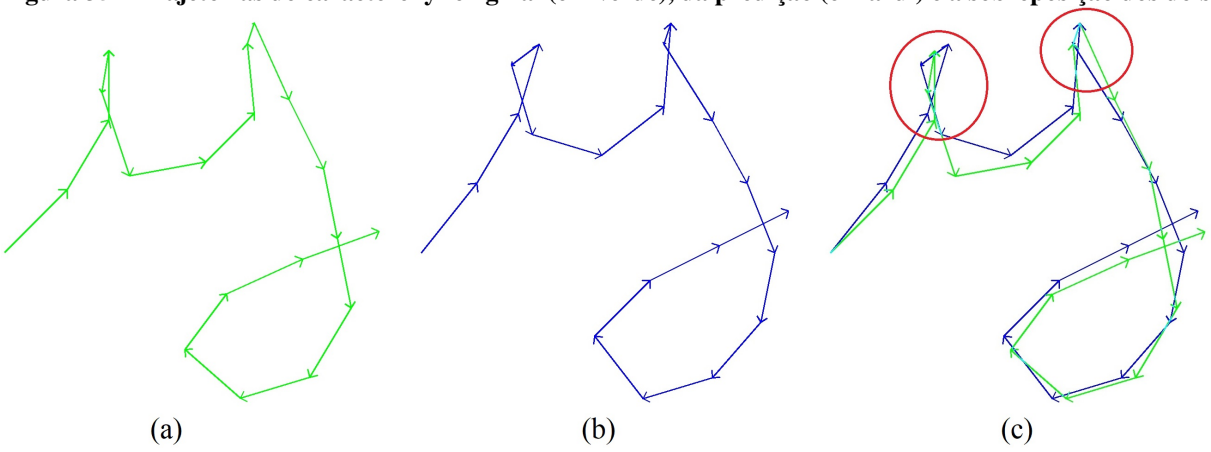


Fonte: Autoria própria

Na Figura 34 a rede prediz corretamente o ponto de início. Entretanto, a tradução na região de sobreposição, mostrado na Figura 34-c, faz com que um dos pontos iniciais do traçado seja movido para o final do traçado original.

A mesma situação ocorre também em casos contendo várias regiões de sobreposição, como mostra na Figura 35.

Figura 35 – Trajetórias do caractere 'y' original (em verde), da predição (em azul) e a sobreposição dos dois



Fonte: Autoria própria

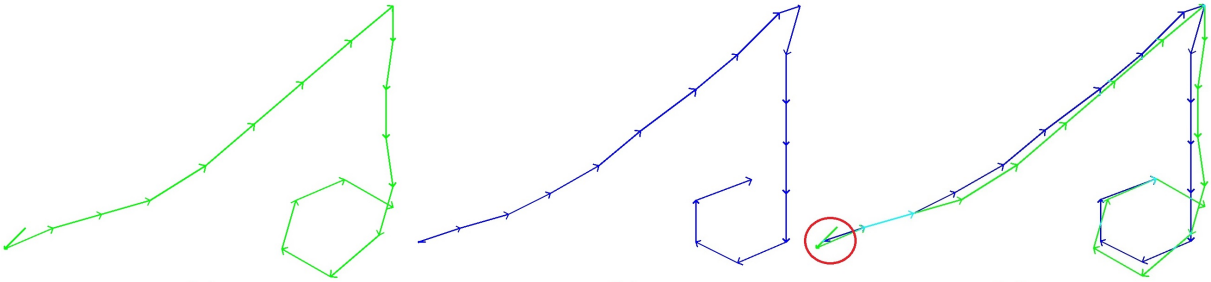
O caractere 'y' possui três regiões de sobreposição, dois deles são pequenos laços, indicados na Figura 33-c em vermelho. Essas regiões também são mal traduzidas e, consequentemente, mal interpretadas, pois a tradução do pixel predito move-o para uma ordenação diferente da que ela realmente é, fazendo com que a predição seja considerada incorreta.

5.4.2 Falsos Negativos Ocasionados pela Peculiaridade da Escrita

A questão da peculiaridade da escrita consiste no estilo de escrita característico do autor. No processo de tradução para o mais próximo do esqueleto, a tradução não alcança alguns traços que foram realizados durante a escrita, pois o processo da escrita possui um processo pe-

culiar, característica do próprio do autor da escrita. Na Figura 36 temos um exemplo do caractere 's'.

Figura 36 – Trajetórias do caractere 's' original (em verde), da predição (em azul) e a sobreposição dos dois



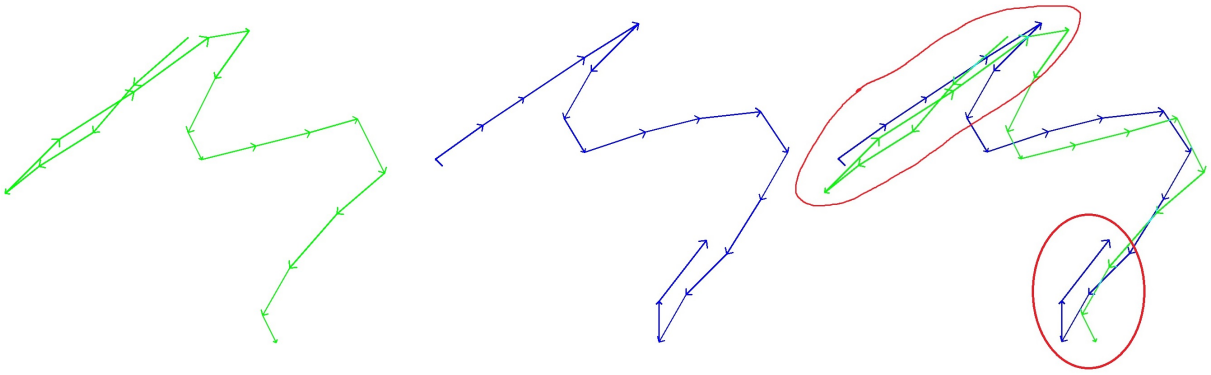
(a) (b) (c)

Fonte: Autoria própria

A escrita é diferente de pessoa para pessoa. Na Figura 36, o autor da escrita do caractere 's' inicia a escrita com um pequeno "gancho". Essa característica, na tipografia, é chamada de serifa[1]. A predição realiza a serifa, porém, não com muita precisão. A tradução dos pontos preditos para o mais próximo do esqueleto produz uma sequência errada, pois o ponto inicial da sequência predita é traduzida para uma coordenada após as coordenadas preditas posteriormente ao ponto inicial.

O caractere 'm' da Figura 37 também mostra uma peculiaridade da escrita do autor.

Figura 37 – Trajetórias do caractere 'm' original (em verde), da predição (em azul) e a sobreposição dos dois



(a) (b) (c)

Fonte: Autoria própria

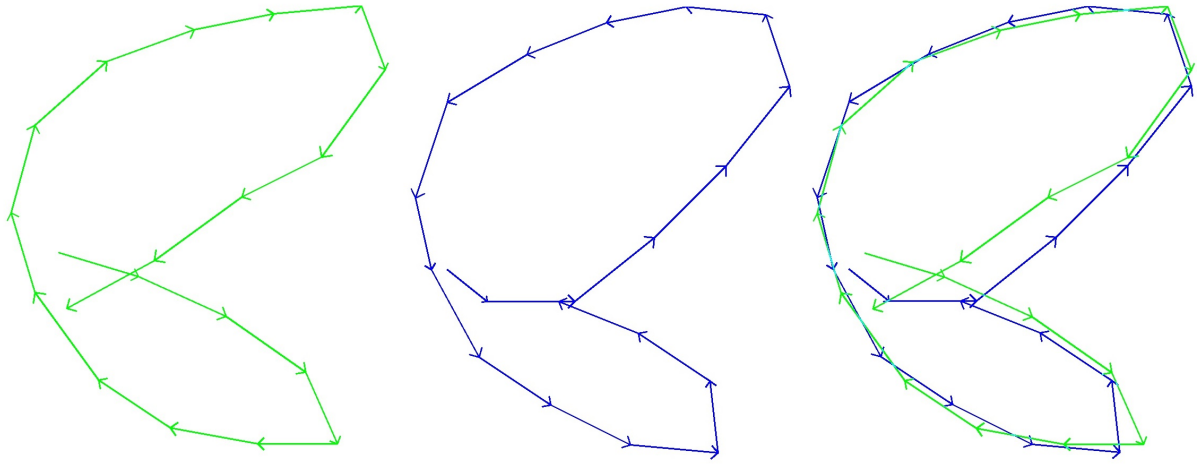
A predição realizada pela rede não consegue interpretar a sobreposição do autor no início do traçado, como ilustrado na Figura 37-c. A rede entende como um único traçado na diagonal para cima. No final, como ainda faltam pontos para serem preditos, ela realiza um gancho no fim do traçado. A tradução faz com que a predição desse segmento inicial fique incorreta. Este é um caso que também pode ser considerado um caso de *outlier* (ponto fora da curva), pois a rede prediz corretamente o formato do caractere e a ordem da escrita dos traços, porém a tradução do ponto predito para o mais próximo fez com que a predição seja considerada incorreta.

[1] https://pt.wikipedia.org/wiki/Serifa

5.4.3 Falsos Negativos Ocasionados pela Ordem Contrária de Escrita

No processo de aprendizagem, a rede recebeu apenas traçados numa determinada ordem do caractere. Quando foi lhe apresentada uma imagem desse caractere, a rede conseguiu reproduzir a ordem que foi aprendida, reproduzindo totalmente o formato geométrico do caractere, porém, no conjunto verdade, o caractere foi escrito com uma ordem diferente. Na Figura 38 temos um exemplo dessa situação.

Figura 38 – Trajetórias do caractere 'p' original (em verde), da predição (em azul) e a sobreposição dos dois



(a) (b) (c)

Fonte: Autoria própria

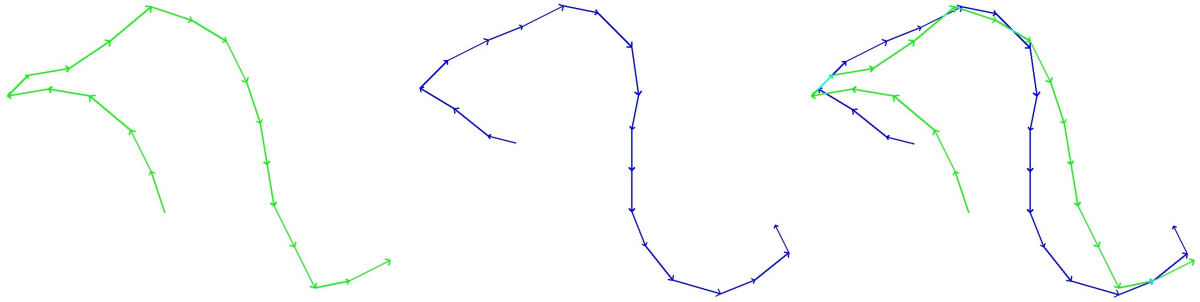
Na Figura 35-a tem-se o traçado original do autor e na Figura 38-b o traçado predito pela rede. Em ambos os casos mostram a escrita do caractere 'p', possuindo o formatos geométricos similares. Entretanto, o autor escreve o caractere com uma ordenação de traçado peculiar e esse tipo de ordenação não está contido na base de treinamento, o que fez com que a rede produzisse o traçado na direção contrária do autor. Quando os dois traçados são sobrepostos, pode-se observar que os dois traçados ocupam praticamente a mesma região, mostrando que ambas possuem o mesmo formato geométrico. A avaliação utilizando SP-JP-CT não consegue avaliar corretamente estes casos pois a ordem dos traçados está totalmente diferente comparada ao conjunto verdade.

5.4.4 Falsos Positivos Ocasionados Pela Falta de Verificação da Forma Geométrica

No processo de predição, a rede gerou apenas uma parte do caractere que no pós-processamento, a tradução moveu as coordenadas preditas para dentro do esqueleto do conjunto verdade, porém não completa o trajeto inteiro do caractere. Estas são situações de falsos positivos, pois o método de avaliação verifica se todos os pontos estão corretamente ordenados dentro da área do esqueleto, que de fato está, pois realiza-se a tradução para o mais próximo do esqueleto. Porém, não verifica se o formato geométrico do caractere foi de fato realizado.

Na Figura 39 temos um exemplo de situação onde o formato geométrico do caractere predito não é verificado.

Figura 39 – Trajetórias do caractere 'n' original (em verde), da predição (em azul) e a sobreposição dos dois



(a) (b) (c)

Fonte: Autoria própria

Quando uma predição não consegue de fato representar o formato do caractere, é considerado, então, que a predição falhou e produziu um caractere diferente. No exemplo da Figura 39 temos o caso de uma predição que não consegue construir o formato do caractere 'n'. Porém, a avaliação o considera como correta, pois ao ser traduzida para o esqueleto do caractere original, todos os pontos preditos estão na ordem correta do esqueleto. Este é um dos casos de falsos positivos encontrados.

5.5 CONSIDERAÇÕES DO CAPÍTULO

A partir das amostras analisadas, foi constatada uma necessidade de propor uma nova avaliação para obter com mais precisão a taxa de acerto de sistemas de HTR utilizando redes *deep learning*. Baseado nas amostras expostas anteriormente, o método SP-JP-CT apresentou resultados falsos negativos e positivos (totalizando 15% das amostras), prejudicando assim, o resultado quantitativo final produzido pela rede.

A partir dos falsos positivos e negativos analisados, pode-se concluir que um método de avaliação de coordenadas de HTR precisa levar em consideração dois fatores: a ordenação correta dos pixels e o formato geométrico similar ao caractere original. O método SP-JP-CT não atende os dois fatores por necessitar um pós-processamento de tradução de coordenadas, o qual traduz a coordenada e altera sua ordenação, e por não verificar a geometria formada pelas coordenadas preditas.

Baseado nisso, uma nova avaliação é proposta para atender esses dois requisitos. Esta nova avaliação é apresentada no Capítulo 6, o qual descreve uma abordagem mais precisa para avaliar as coordenadas preditas, buscando aumentar a confiabilidade do resultado da taxa de acerto pois possibilita a correção dos resultados falsos negativos e falsos positivos do método SP-JP-CT.

6 MODELO DE AVALIAÇÃO SWO

Neste capítulo será apresentado o modelo de avaliação para sistemas de HTR desenvolvido neste trabalho, chamado de SWO (*Segmentation - sliding Window - Ordering*). Inicialmente apresenta-se o motivo pelo qual se deve desenvolver uma avaliação. Em seguida, na Seção 6.2 a avaliação SWO será detalhada. Os resultados quantitativos da avaliação SWO serão apresentados na Seção 6.3. A Seção 6.4 irá apresentar as taxas de acurácia por classe de caractere. Na Seção 6.5 serão apresentados as análises qualitativas da avaliação SWO mostrando os detalhes e justificando sua eficácia. Por fim, na Seção 6.6 realiza-se uma comparação dos resultados da avaliação SWO com a avaliação CT.

6.1 JUSTIFICATIVA

A necessidade de criar um novo método de avaliação veio com base na análise dos resultados da avaliação SP-JP-CT, comentadas na Seção 5.4. Na Seção 3.2.1 levantou-se as avaliações que os autores utilizam para calcular o resultado das predições. A partir dos trabalhos analisados como de Zhao, Yang e Tao (2019), Wang *et al.* (2019), Bhunia *et al.* (2018), o estudo de Nguyen e Blumenstein (2010) com trabalhos anteriores, assim como também de outros trabalhos comentados no Capítulo 2, pode se concluir que os autores utilizam métodos avaliativos que melhor se encaixam em seus próprios experimentos, ou seja, não existe uma padronização para se realizar a avaliação.

Em razão desses fatores, é proposto neste trabalho um modelo de avaliação cujo o objetivo é avaliar as coordenadas preditas de sistemas de HTR, verificando a similaridade do formato geométrico produzido pelas coordenadas e a ordenação das das mesmas com o *ground-truth*.

6.2 AVALIAÇÃO SWO

Nesta seção apresenta-se o método de avaliação proposto, SWO, que melhora os resultados dos testes pois elimina os falsos negativos e falsos positivos resultantes da avaliação SP-JP-CT. Além disso, elimina a necessidade de realizar qualquer tipo de pós-processamento sobre as coordenadas preditas, possibilitando resultados de taxa de acerto mais precisos, permitindo também que seja aplicado a qualquer modelo de HTR.

Inicialmente, define-se um valor n para dividir o esqueleto de cada caractere original da base em n segmentos (*Segmentation*). A escolha do valor n produz segmentos uniformes para cada caractere, ao ponto de que cada segmento seja significativo para o caractere. Ser significativo quer dizer que a falta deste segmento pode impactar no formato geométrico original

do caractere. Segmentar excessivamente o traçado de um caractere faz com que a retirada de um segmento não seja tão significativo no formato geométrico do mesmo. Segmentar de maneira insuficiente pode afetar de mais no formato ao ponto de não ser reconhecido de nenhuma maneira.

A definição do valor n pode afetar diretamente no resultado da avaliação. Caso n seja um valor muito grande, a predição pode não encontrar algum segmento. Isso acarretará em perda de acurácia, porém não representa de fato que a rede não conseguiu com sucesso a predição do formato geométrico do caractere. Em contrapartida, um n muito baixo faz com que a avaliação resulte sempre em resultados positivos.

Nos experimentos realizados, verificou-se que quanto maior a quantidade de pontos a serem preditos por caractere (p/c - pontos por caractere), maior deve ser também a quantidade de segmentos (n) que se deve dividir durante a avaliação. Para estabelecer essa relação, houve a necessidade de criar um padrão para definir o valor de n de modo que, toda vez que se aumenta a p/c, o valor de n também deve aumentar. Isso constitui-se de valores diretamente proporcionais.

Valores para n foram testados de maneira incremental, até que se conseguiu definir uma forma de computar a quantidade de segmentos a ser dividida de forma padronizada de acordo com a quantidade de pontos que base possui em cada caractere. Nomeia-se a constante n de TSF - *Trajectory Segmentation Factor* (Fator de Segmentação de Trajetória), pelo fato da mesma resultar em um valor para segmentar a trajetória do caractere. O cálculo para obter a TSF é apresentada pela seguinte equação:

$$TSF = \left\lfloor 2 * \sqrt{p/c} \right\rfloor \tag{6.1}$$

onde TSF é a quantidade de segmentos pelo qual a trajetória deve ser dividida e p/c é a quantidade de pontos por caractere na predição. Então, o TSF é definida pelo piso da raiz quadrada do valor de p/c multiplicado pela constante 2. Quando a função matemática de piso (em inglês *floor*, de Iverson (1962)) é aplicada, o resultado é convertido no maior número inteiro menor ou igual a ele. Realiza-se essa operação da função piso pois nesta etapa, deseja-se otimizar a quantidade de segmentos para conseguir maximizar a taxa de acerto. A Tabela 9 apresenta o comportamento da TSF em comparação as progressões aritméticas PA_1 e PA_2 de primeiro termo $a_1 = 8$ e razão $r = 1$ e $r = 2$, respectivamente, para diferentes quantidades de p/c.

Pode-se observar pelos resultados dos cálculos apresentados na Tabela 9 que a fórmula do TSF consegue permanecer mais estável quando se aumenta a quantidade de p/c. É importante ressaltar essa característica pois a diferença entre um caractere que contém, por exemplo, 21 pontos e um caractere que contém 22 pontos é bastante pequena; suas diferenças começam a ser notadas quando se aumenta uma quantidade mais considerável de pontos.

A fórmula do TSF consegue se manter constante até que se tenha uma diferença considerável de p/c. Dessa forma, as quantidades de segmentos divididos passam a ser maiores quando a quantidade de pontos tiver um aumento maior; situação que não ocorre em PA.

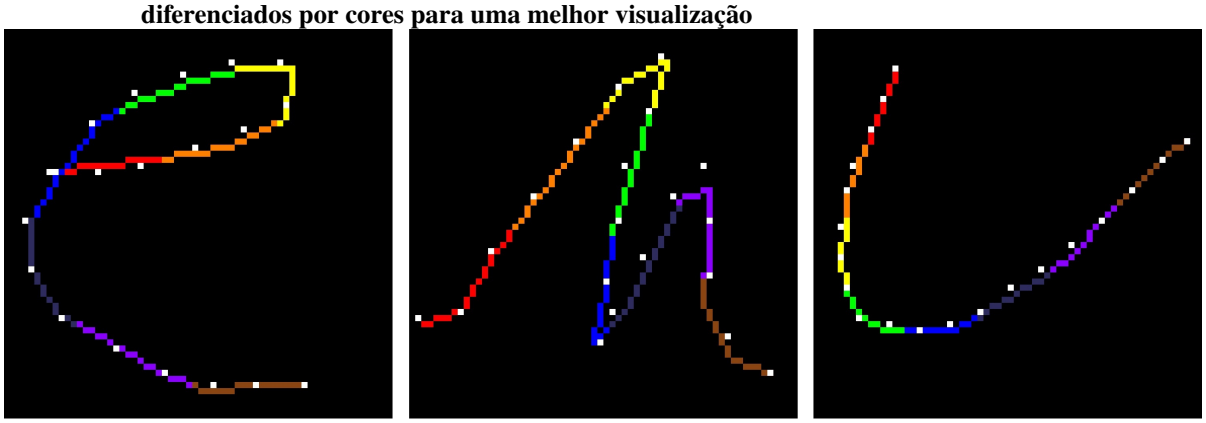
Tabela 9 – Comportamento da TSF e PA em valores de p/c

	TSF	PA_1	PA_2
p/c	$\lfloor 2 * \sqrt{p/c} \rfloor$	$a_1 = 8, r = 1$	$a_1 = 8, r = 2$
20	8	8	8
21	9	9	10
22	9	10	12
24	9	12	16
25	10	13	18
28	10	16	24
30	10	18	28
40	12	28	48
50	14	38	68
60	15	48	88
70	16	58	108
100	20	88	168

Fonte – Autoria própria

Na Figura 40 apresenta-se exemplos de esqueletos de caracteres de $20p/c$, resultando $TSF = 8$, onde cada segmento está representado com uma cor diferente; cada caractere possui vinte pontos brancos que são a predição da rede.

Figura 40 – Esqueleto dos caracteres 'u', 'm' e 'e', respectivamente, divididos em sete segmentos, diferenciados por cores para uma melhor visualização

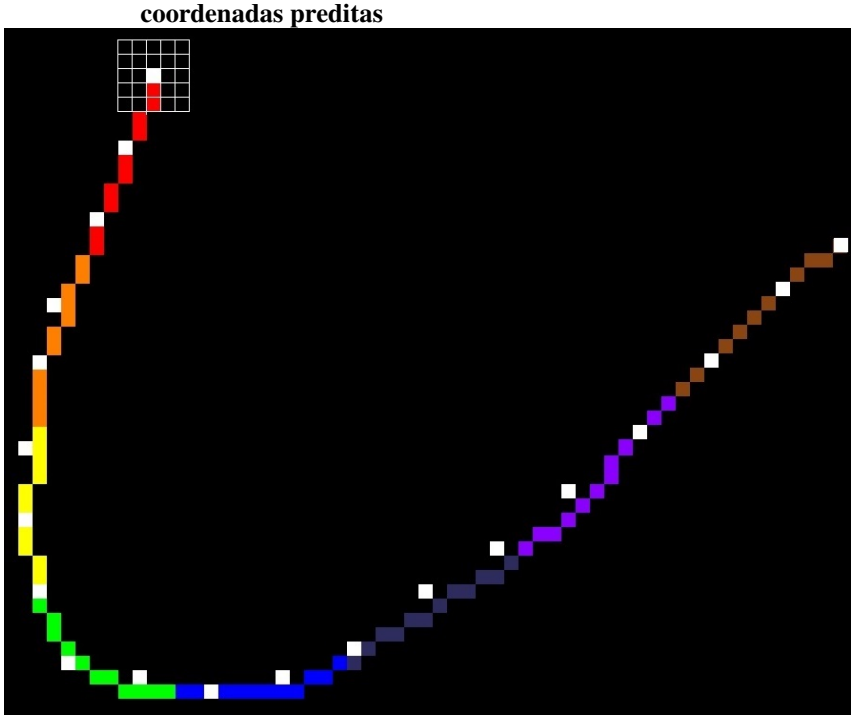


Fonte: Autoria própria

A Figura 40 apresenta exemplos que foram divididos em oito segmentos. Cada um desses segmentos estão representados com uma cor para uma melhor visualização da divisão. Assim como a ordem das coordenadas, os segmentos também possuem uma ordenação, que seria em primeiro o vermelho, em seguida o laranja, amarelo, verde, azul, anil, violeta e por fim marrom.

Após a divisão de segmentos, cria-se uma janela deslizante (*sliding Window*) de tamanho 5x5, o qual será centrado na trajetória do esqueleto do caractere original e irá percorrer o esqueleto verificando se há pontos preditos pela rede neural em cada segmento separado que estão dentro da área coberta pela janela, atribuindo o ponto encontrado no segmento que está sendo percorrido. Na Figura 41 temos uma ilustração da matriz deslizante sobre um caractere.

Figura 41 – Representação da matriz deslizante iniciando no ponto do início do esqueleto. Os pontos brancos representam as coordenadas preditas



Fonte: Autoria própria

Como pode ser observado na Figura 41, os pontos preditos (que estão na cor branca) podem estar no trajeto do esqueleto do caractere ou muito próximo ao esqueleto. A janela deslizante inicia seu percurso no início da trajetória do esqueleto verificando se os pontos preditos estão dentro do alcance da matriz de tamanho 5x5.

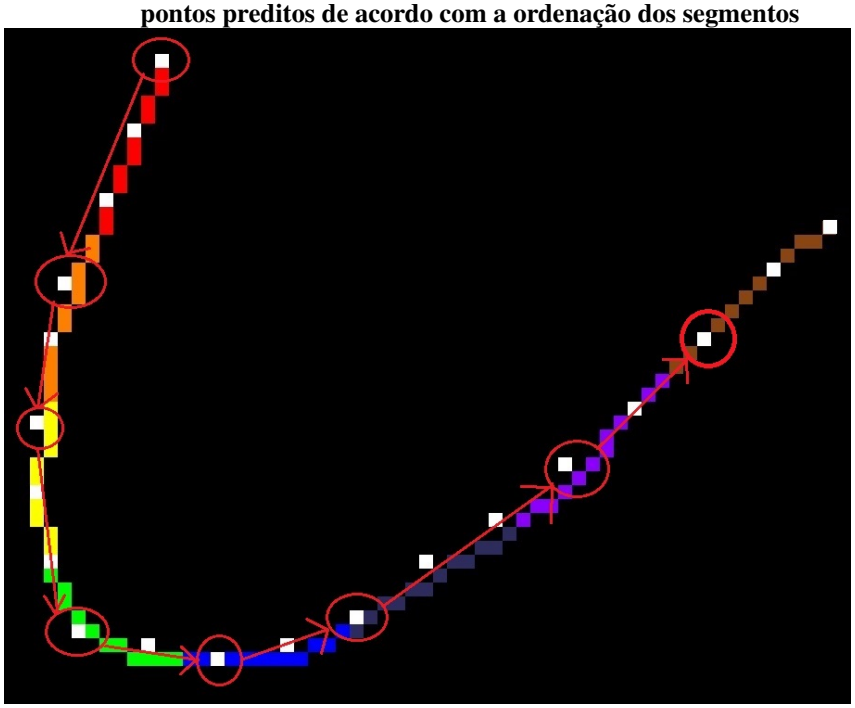
Este tamanho de janela foi definido pois as predições realizadas pela rede podem desviar da trajetória original do esqueleto. Uma janela muito pequena (por exemplo 3x3) pode não alcançar a área de predição. Em contrapartida, uma janela muito grande pode alcançar regiões preditas que não correspondem ao segmento em avaliação. Nos experimentos realizados neste trabalho, onde o tamanho das imagens é de 64x64, conseguiu-se através de uma janela de 5x5 avaliar de maneira mais precisa o alcance dos pontos preditos a partir do esqueleto *ground-truth*. Imagens com caracteres maiores podem necessitar de janelas de tamanhos maiores, o mesmo pode ocorrer com imagens de caracteres menores, que podem necessitar janelas de tamanho menores.

Quando a janela deslizante encontra pontos preditos, o segmento em que ela percorre é considerado predito. Quando algum segmento não for predito, ou seja, não foram encontrados pontos preditos na região da janela deslizante, quer dizer que não foi possível predizer o caractere. Existirão pontos em que não vão ser abordados pela janela deslizante. Esses pontos são descartados, uma vez que, nesta etapa, o objetivo é verificar se a predição conseguiu abordar todos os segmentos do esqueleto original. Quando um ponto predito pertencer aos dois segmentos, este ponto vai pertencer ao segmento que ainda não tiver pontos atribuídos.

Após verificar se os segmentos foram ou não preditos, é necessário verificar se a ordem

dos pontos preditos estão corretos (*Ordering*). Para isso, é verificado se existe uma ordenação entre um ponto de cada segmento. Ainda utilizando o caractere 'u' como exemplo, a Figura 42 ilustra os traços ordenados e as predições que cada segmento possui.

Figura 42 – Representação da ordenação dos segmentos. Os pontos brancos representam as coordenadas preditas. As setas mostram um exemplo da ordenação correta encontrada pelos pontos preditos de acordo com a ordenação dos segmentos



Fonte: Autoria própria

Na Figura 42, para cada segmento encontra-se um ponto que corresponda a ordenação daquele segmento. Se for possível encontrar pelo menos uma combinação de um ponto de cada segmento que corresponda com a ordenação dos traços, então considera-se que foi possível predizer a ordem do traçado desse caractere. É possível encontrar mais de uma combinação de ordenação correta em cada caso.

Na seção 5.4.3 foi levantado que há casos de falsos positivos pela ordem contrária da escrita. Para contornar esses casos, caso as verificações anteriores não tiverem um resultado positivo (de caractere predito), uma última verificação é adicionada invertendo a ordem de traçado do esqueleto e verificando se é possível encontrar pelo menos uma combinação de um ponto de cada segmento que corresponda a ordenação invertida dos traços. Se sim, então esta predição é considerada correta.

Na próxima seção serão apresentados os resultados quantitativos da predição das redes utilizando a avaliação SWO. Ao final da seção apresenta-se um gráfico com o resultado de todas as redes.

6.3 RESULTADOS QUANTITATIVOS COM A AVALIAÇÃO SWO

Nesta seção apresenta-se os resultados quantitativos das predições das redes utilizando a avaliação SWO.

As taxas de acurácia das bases de *sinle-stroke* são apresentadas na Tabela 10.

Tabela 10 – Avaliação SWO do resultado das redes nas bases *single-stroke*

Base	netBh	net-v0	net-v1	net-v2	net-V3
$20p/c$	$91,36\%$	$90,75\%$	***95,31%***	$90,79\%$	$85,17\%$
$30p/c$	$91,67\%$	$94,34\%$	$95,14\%$	$92,39\%$	$76,97\%$
$40p/c$	$87,28\%$	$92,51\%$	$91,06\%$	$89,86\%$	$76,42\%$
$50p/c$	$86,04\%$	$88,56\%$	$87,97\%$	$86,46\%$	$65,75\%$
$60p/c$	$83,69\%$	$86,59\%$	$85,95\%$	$81,01\%$	$59,24\%$
$70p/c$	$79,52\%$	$83,02\%$	$86,93\%$	$78,27\%$	$50,36\%$
$100p/c$	$39,73\%$	$44,82\%$	$45,13\%$	$45,45\%$	$23,29\%$

Fonte – Autoria própria

Os dados da Tabela 11 apresentam os resultados das redes nas bases *multi-strokes*.

Tabela 11 – Avaliação SWO do resultado das redes nas bases *multi-strokes*

Base	netBh	net-v0	net-v1	net-v2	net-V3
$20p/c$	$88,66\%$	$92,21\%$	***92,93%***	$89,59\%$	$81,91\%$
$30p/c$	$88,19\%$	$92,13\%$	$92,69\%$	$86,40\%$	$73,37\%$
$40p/c$	$79,59\%$	$87,69\%$	$87,31\%$	$83,13\%$	$72,89\%$
$50p/c$	$77,64\%$	$84,31\%$	$83,42\%$	$80,89\%$	$59,33\%$
$60p/c$	$72,05\%$	$81,44\%$	$78,76\%$	$75,26\%$	$50,49\%$
$70p/c$	$57,99\%$	$70,48\%$	$72,28\%$	$65,85\%$	$47,30\%$
$100p/c$	$32,19\%$	$32,45\%$	$33,02\%$	$33,15\%$	$22,98\%$

Fonte – Autoria própria

A rede *net-v1* apresentou a maior taxa de acerto para a base *single-stroke* e *multi-strokes* de $20p/c$. Comparando com outras bases, a *net-v1* também se mantém superior para as bases $30p/c$ e $70p/c$. A rede *net-v0* supera os resultado nas acurácias das bases de $40p/c$, $50p/c$ e $60p/c$. Para a base com $100p/c$, a rede *net-v2* atingiu maiores acertos.

Com esses resultados podemos observar que dada as mesmas condições de treinamento, validação e teste para as redes, a *net-v1* conseguiu generalizar mais rapidamente as bases com menor número de pontos por caractere ($20p/c$, $30p/c$), enquanto que a *net-v0* consegui para quantidade de pontos por caractere maiores ($40/pc$, $50p/c$, $60p/c$). Entretanto, a *net-v1* apresentou melhores resultados para a base de $70p/c$, que é uma quantidade superior de pontos.

Na seção a seguir, apresenta-se os resultados por classe de caracteres das bases de *single-stroke* e *multi-stroke* de $20p/c$, o qual foi a base que se conseguiu atingir maior acurácia na predição das redes.

6.4 RESULTADOS POR CARACTERES COM A AVALIAÇÃO SWO

Nesta seção apresenta-se os resultados por caractere das redes treinadas na base $20p/c$ de *single-stroke* e *multi-stroke* utilizando a avaliação SWO. Os resultados da base de $20p/c$ foram escolhidos pois foi a base onde as redes obtiveram maiores taxas de acerto. Vale ressaltar mais uma vez que os experimentos deste trabalho são conduzidos com os caracteres minúsculos da base IRONOFF e, para uma mesma classe de caractere, pode-se ter a escrita em *single-stroke* e *multi-stroke*, como descrito na Seção 4.1. Os resultados são apresentados na Tabela 12 juntamente com o cálculo do desvio padrão para mostrar o grau de dispersão de cada conjunto *single-stroke* e *multi-stroke* em cada rede.

A partir dos dados da Tabela 12, pode-se notar um aumento de 40% na taxa de acurácia em cada classe dos caracteres 'd', 'f', 'o' e 'p', em comparação com a avaliação CT por caracteres comentadas na Seção 5.3. Na *net-v1*, conseguiu-se 100% de acerto no caractere 'c' de *single-stroke*. Em outras classes de caracteres também conseguiu-se observar um aumento na taxa de acurácia.

No caractere da classe 'x', a avaliação SWO conseguiu identificar os falsos positivos, como por exemplo, na rede *netBh*, onde a taxa de acerto neste caractere caiu em comparação com a avaliação CT mostrada na Tabela 8. A rede *net-v1* consegue predizer melhor os caracteres da classe 'x', obtendo resultados melhores. O grau de dispersão dos conjuntos também é menor comparada aos dados da Tabela 8, mostrando que os resultados das predições são mais uniformes e homogêneas. Levantando os melhores resultados por caractere, a *net-v1* obteve 23 dos 26 melhores resultados por caractere *single-stroke*. Para *multi-strokes*, a *net-v1* obteve 18 dos 26 melhores.

Para validar esse aumento de acurácia, na próxima seção realiza-se uma análise qualitativa da avaliação SWO, mostrando a eficácia da avaliação para corrigir os falsos negativos e falsos positivos da avaliação SP-JP-CT.

Tabela 12 – Taxa de acurácia SWO por caractere das redes na base $20p/c$

Caractere	*netBh*		*net-v0*		*net-v1*		*net-v2*		*net-v3*		*Best Single*	*Best Multi*
	single	*multi*	*single*	*multi*	*single*	*multi*	*single*	*multi*	*single*	*multi*		
a	94,72%	91,64%	92,08%	92,24%	93,96%	92,84%	92,45%	92,54%	88,30%	84,48%	94,72%	92,84%
b	95,81%	98,51%	94,52%	99,10%	99,35%	98,51%	95,48%	98,21%	94,52%	90,45%	99,35%	99,10%
c	99,10%	98,00%	98,81%	99,14%	100,00%	99,71%	98,81%	99,43%	98,21%	96,00%	100,00%	99,71%
d	90,53%	87,30%	86,84%	93,02%	95,26%	91,43%	91,58%	89,21%	86,84%	79,37%	95,26%	93,02%
e	98,82%	98,55%	97,65%	97,97%	99,12%	99,42%	98,82%	98,55%	97,06%	95,36%	99,12%	99,42%
f	84,40%	82,57%	80,40%	86,29%	88,00%	90,29%	82,40%	81,71%	65,60%	70,86%	88,00%	90,29%
g	90,67%	82,94%	83,00%	87,35%	96,00%	90,88%	86,67%	85,00%	76,67%	74,71%	96,00%	90,88%
h	96,83%	96,67%	93,97%	96,97%	98,41%	97,58%	95,87%	96,36%	91,43%	92,42%	98,41%	97,58%
i	100,00%	84,06%	80,00%	88,70%	100,00%	91,01%	80,00%	82,61%	100,00%	72,75%	100,00%	91,01%
j	85,71%	73,62%	91,43%	81,74%	97,14%	83,48%	85,71%	75,07%	60,00%	56,52%	97,14%	83,48%
k	87,18%	82,32%	89,23%	89,57%	91,28%	88,12%	85,64%	86,67%	72,82%	76,81%	91,28%	89,57%
l	88,00%	88,17%	91,43%	93,80%	95,14%	95,77%	90,00%	88,17%	88,00%	82,82%	95,14%	95,77%
m	83,48%	83,38%	90,14%	91,27%	90,72%	89,58%	85,22%	88,73%	84,35%	86,48%	90,72%	91,27%
n	95,88%	95,71%	97,35%	97,43%	97,35%	98,00%	95,00%	96,86%	91,76%	93,14%	97,35%	98,00%
o	90,56%	93,21%	87,50%	93,75%	91,39%	91,85%	93,33%	90,76%	88,33%	85,05%	93,33%	93,75%
p	85,96%	90,00%	88,51%	90,65%	94,89%	92,26%	89,36%	89,35%	79,57%	80,97%	94,89%	92,26%
q	92,11%	91,95%	89,30%	93,90%	96,62%	92,44%	87,04%	90,00%	81,69%	76,83%	96,62%	93,90%
r	89,85%	88,29%	91,08%	90,29%	95,38%	91,71%	89,54%	88,00%	83,69%	87,43%	95,38%	91,71%
s	97,14%	92,54%	93,97%	97,61%	98,10%	94,63%	95,56%	92,54%	84,76%	86,27%	98,10%	97,61%
t	93,04%	86,58%	92,17%	86,85%	97,39%	91,23%	87,83%	83,56%	80,00%	69,32%	97,39%	91,23%
u	97,38%	97,27%	98,36%	97,58%	98,69%	98,79%	98,03%	97,88%	96,07%	94,24%	98,69%	98,79%
v	94,03%	94,12%	96,72%	97,94%	97,01%	96,76%	96,12%	95,88%	91,34%	93,82%	97,01%	97,94%
w	87,50%	84,62%	89,64%	89,54%	91,79%	91,08%	88,93%	88,62%	84,64%	87,08%	91,79%	91,08%
x	70,59%	73,33%	60,00%	85,15%	85,88%	85,76%	69,41%	84,85%	60,00%	70,30%	85,88%	85,76%
y	88,81%	85,22%	85,42%	90,43%	93,56%	91,88%	81,69%	88,41%	74,58%	71,88%	93,56%	91,88%
z	81,33%	84,00%	81,67%	89,23%	92,67%	91,38%	82,00%	80,92%	71,67%	76,00%	92,67%	91,38%
Média	91,36%	88,66%	90,75%	92,21%	95,31%	92,93%	90,79%	89,59%	85,17%	81,91%	95,45%	93,44%
Desvio Padrão	6,63	7,02	8,02	4,77	3,68	4,15	6,89	6,22	11,06	9,97	3,61	4,19
Mediana	90,61%	88,23%	90,61%	91,75%	95,69%	91,87%	89,45%	88,97%	84,70%	83,65%	95,69%	92,55%

Fonte – Autoria própria

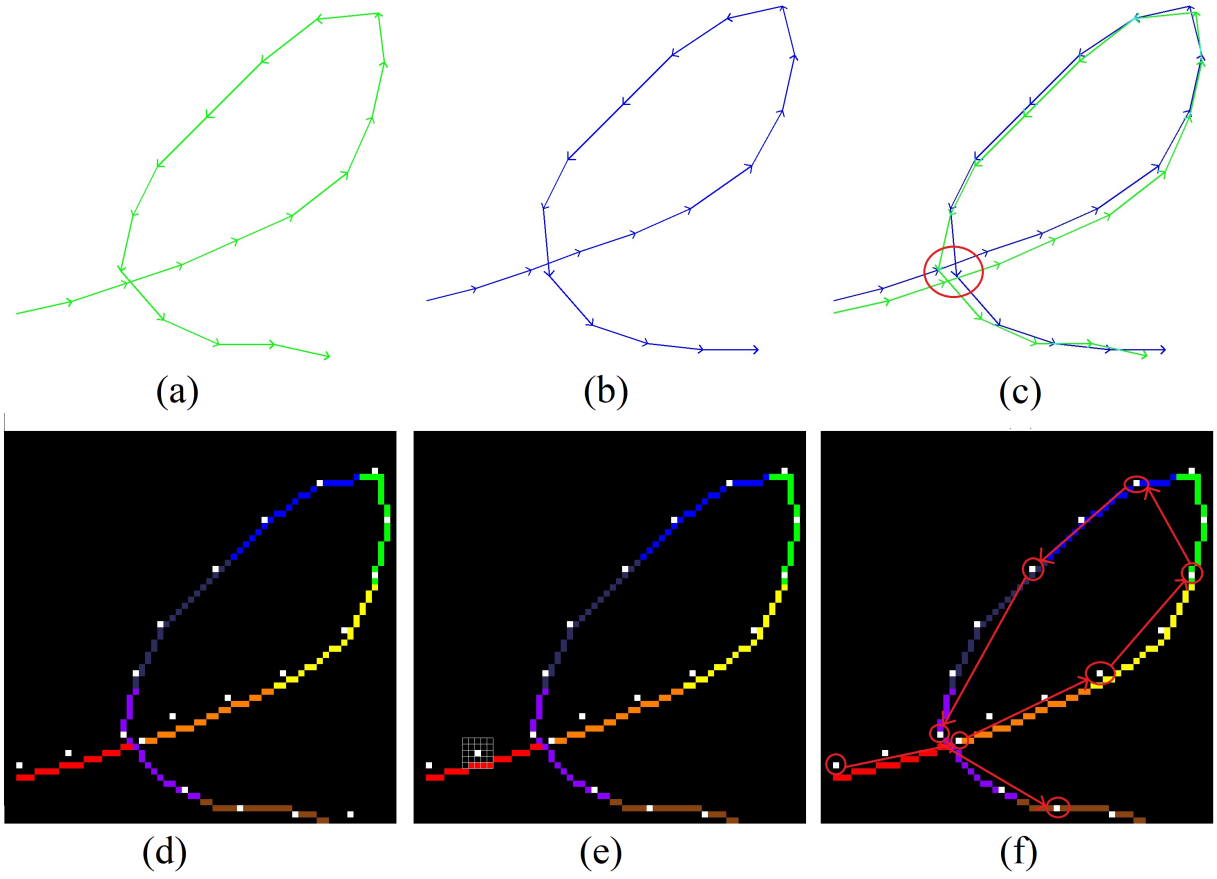
6.5 ANÁLISE QUALITATIVA COM A AVALIAÇÃO SWO

Nesta seção será mostrado os casos de falso negativos e falsos positivos comentados na seção 5.4 utilizando a avaliação SWO explicada na Seção 6.2. Desta maneira pode-se comprovar a eficácia do novo método para avaliar sistemas de HTR utilizando redes neurais artificiais.

Para ilustrar as características da avaliação SWO, utiliza-se algumas amostras da base $20p/c$ com a predição da *net-v1*. Para uma base de $20p/c$, o TSF será igual a 8. Cada segmento será representado por uma cor. As cores são: vermelho, laranja, amarela, verde, azul, anil, violeta, e marrom.

A primeira categoria identificada de casos de falsos negativos são oriundos de um processo de pós-processamento, que consiste na tradução do ponto predito para o mais próximo em relação ao esqueleto original do caractere (comentada na Seção 5.4). Como a avaliação SWO não necessita de um pós-processamento, logo, os problemas causados por este processo são anulados. A janela deslizante resolve o problema de pontos que estão distantes ou próximos do esqueleto original do caractere. Vejamos um exemplo mostrado na Figura 43.

Figura 43 – Trajetória do caractere 'e' e ilustração da nova avaliação.



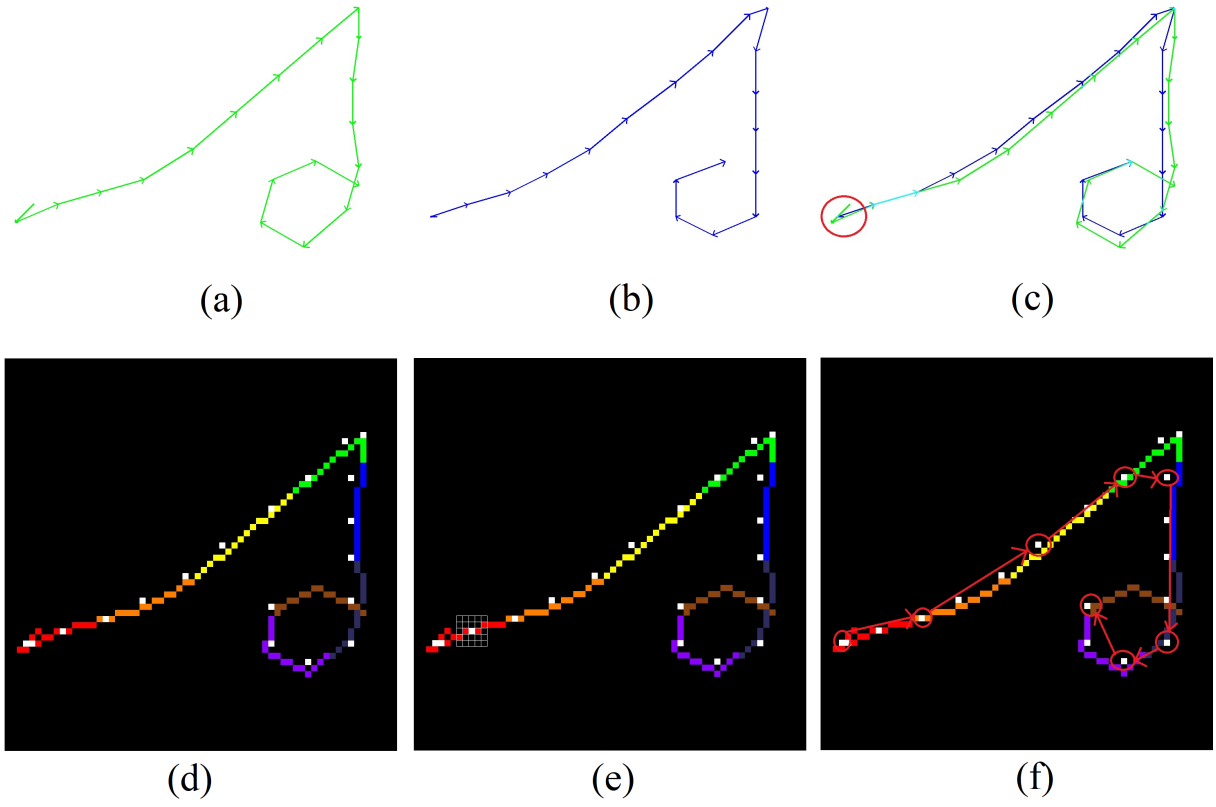
Fonte: Autoria própria

Como pode ser observado na Figura 43-a, Figura 43-b e Figura 43-c, o caractere 'e' é avaliado incorretamente pela SP-JP-CT devido a tradução incorreta nas zonas de sobreposição

de segmentos, pois o terceiro ponto é traduzido incorretamente para praticamente o final do traçado. Na avaliação SWO (Figura 43-d, Figura 43-e, Figura 43-f), apesar do terceiro ponto entrar no traço violeta, que é o sétimo traço do caractere, ainda se consegue encontrar uma ordenação correta dos pontos preditos por segmento que corresponde com a ordenação dos traços. Na Figura 43-f, podemos observar que o terceiro ponto predito está mais próximo do traço em violeta. Porém, ele não está predito na ordem errada, apenas está longe do traçado que ele deveria estar, que seria o laranja. Ainda na Figura 43-f, no traço laranja também temos um ponto que está no traço errado. Durante a procura da combinação correta, este ponto é ignorado pois não está na ordenação correta do traçado.

A próxima categoria de falsos negativos são com relação a peculiaridades da escrita. Estes casos também são corrigidos pela avaliação SWO, uma vez que a divisão dos traços abrange essas peculiaridades e os pontos preditos cobre os traços que estão com as peculiaridades, conforme o exemplo ilustrado na Figura 44.

Figura 44 – Trajetória do caractere 's' e ilustração da nova avaliação.

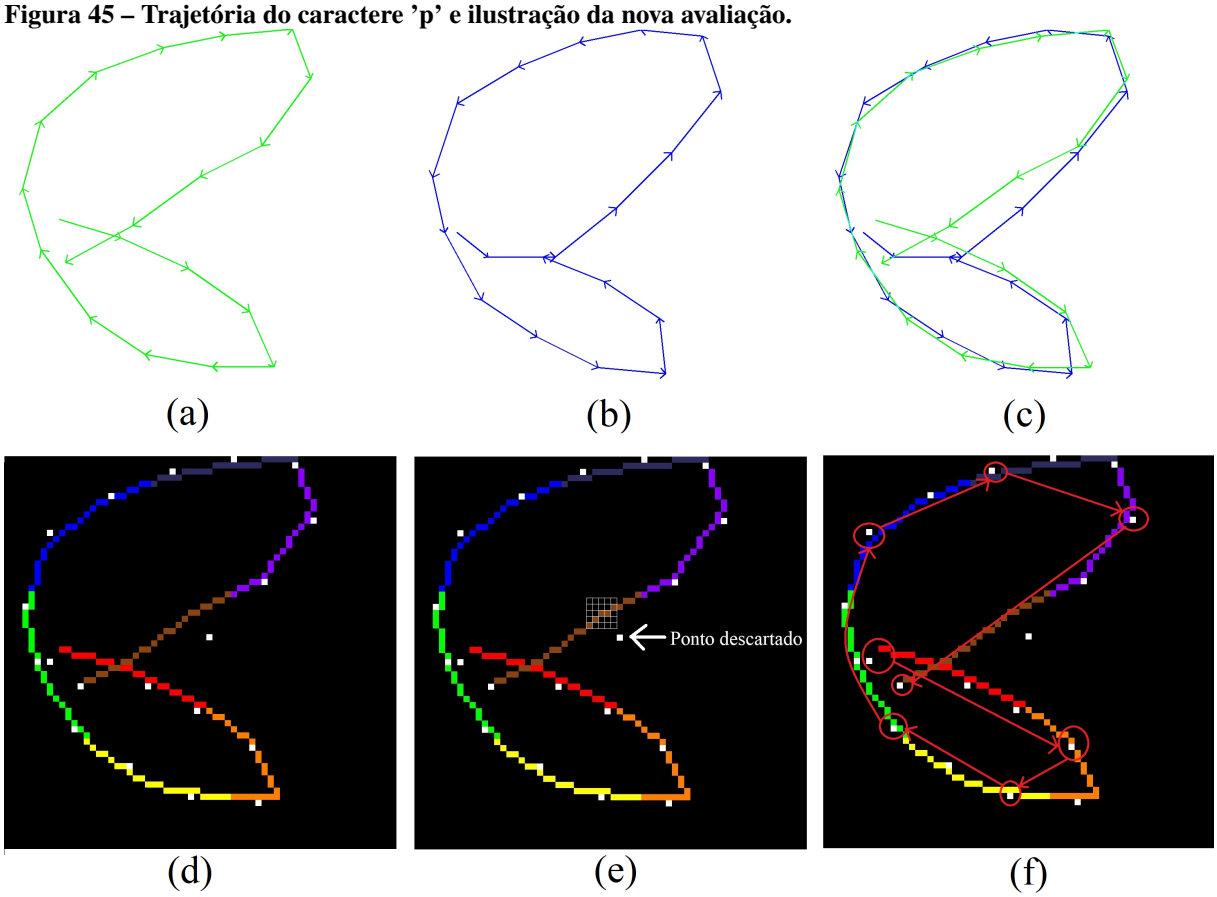


Fonte: Autoria própria

A má interpretação causada no início do traçado é contornada pela avaliação SWO, como mostrado na Figura 44-d, Figura 44-e e Figura 44-f. Isso ocorre pois durante a procura de uma ordenação de segmentos, é requerido apenas um ponto por segmento. No exemplo da Figura 44-f, os três pontos do segmento vermelho vêm antes dos pontos preditos que estão no segmento laranja. Esses casos são contornados pela avaliação SWO.

A outra categoria de falsos negativos são ocasionadas pela ordem contrária de traçado. Como já foi comentado anteriormente, esta situação ocorre quando a rede aprende uma determinada ordem de predição de um caractere e é lhe apresentada o mesmo caractere, porém o autor de escrita realiza o traçado na ordem contrária. Na Figura 45 temos um exemplo deste caso.

Figura 45 – Trajetória do caractere 'p' e ilustração da nova avaliação.



Fonte: Autoria própria

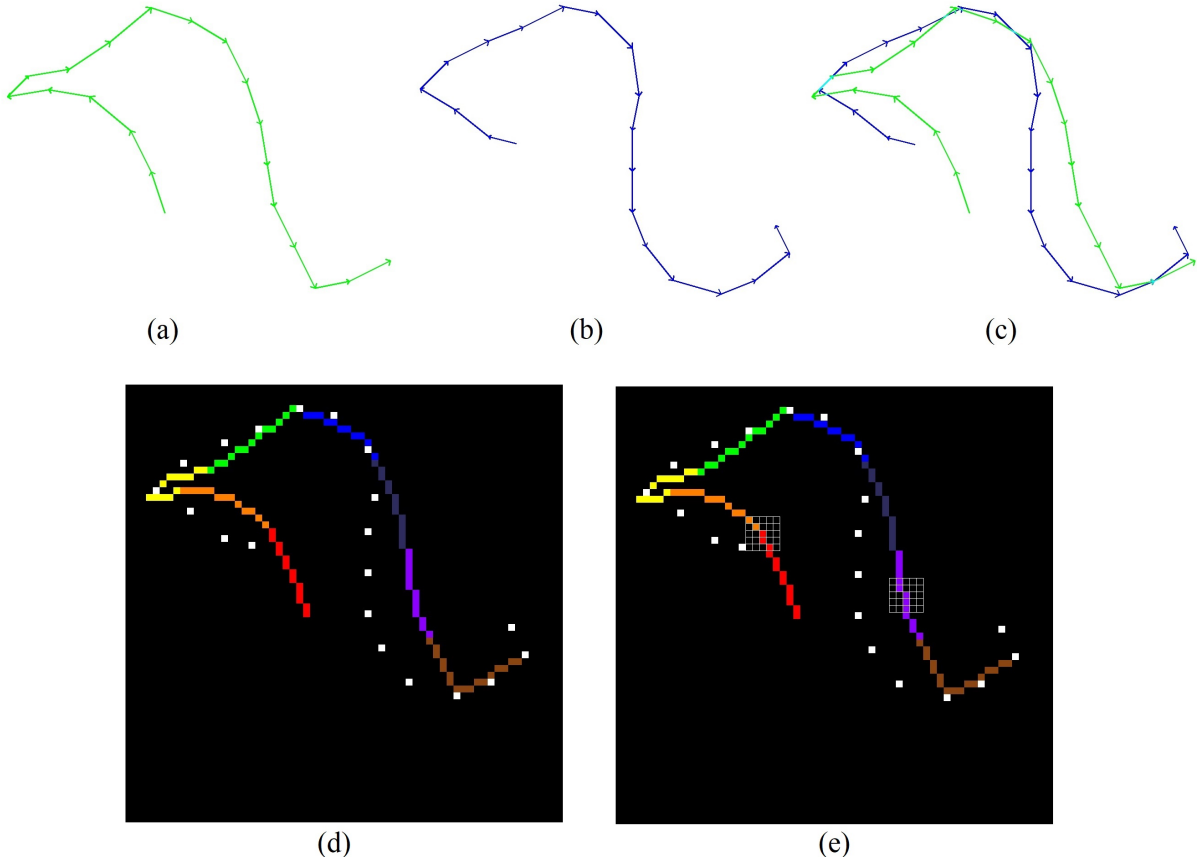
No exemplo da Figura 45-e, temos um dos pontos que foi descartado na avaliação SWO (no segmento marrom). Isto não impede que o traço seja predito, pois existe outro ponto que alcança o esqueleto original no segmento marrom. O ponto então é descartado e não será considerado para as próximas etapas.

A ordenação inicia no traço vermelho, porém a predição realiza o traçado no traço marrom. A avaliação SWO conta com a etapa de realizar a verificação dos pontos preditos na ordem contrária desde que estes também estejam na ordenação correta quando a ordem for invertida, por isso consegue abranger esses caracteres como do exemplo da Figura 45-f.

A última categoria é de falsos positivos. Falsos positivos são quando a predição está errada, porém a avaliação declara a predição como correta. Os falsos positivos são causados principalmente pelo pós-processamento da tradução de pontos, pois muitas vezes a predição não abrange o traçado inteiro do esqueleto do caractere, mas pela avaliação *SP-JP-CT*, que verifica apenas se os pontos preditos estão na ordem correta dentro do esqueleto original, resulta que a amostra foi predita com sucesso.

A avaliação SWO contorna esses casos e atribui corretamente como verdadeiros negativos pois utiliza a janela deslizante, o qual limita a área que a predição deve ocorrer, e também utiliza a divisão do traçado do caractere em segmentos. Na Figura 46 temos um exemplo de falso negativo e a avaliação proposta que classifica como verdadeiro negativo.

Figura 46 – Trajetória do caractere 'n' e ilustração da nova avaliação.



Fonte: Autoria própria

Na Figura 46-e é ilustrado dois segmentos o qual a rede não consegue predizer, que são descritos pelas cores vermelhos e violeta. Na avaliação de *SP-JP-CT* realiza-se a tradução dos pontos preditos para o ponto mais próximo do esqueleto. Neste caso, todos os pontos vão conseguir ficar em sequência correta (até mesmo o último ponto predito, que irá ficar na mesma posição do penúltimo ponto). Se avaliar a sequência após a tradução, vai ser considerada correta, pois o ponto de início predito está em primeiro comparado aos outros e todos os pontos de junção também estão corretos.

Entretanto, pode-se observar que a forma geométrica do caractere predito não está semelhante com a forma geométrica do caractere original. Isso ocorreu pois a rede não interpretou corretamente o segmento inicial, produzindo uma leve curva na região perto do segmento violeta e ainda realizou um gancho no final do traçado. No final, o formato geométrico produzido se assemelha com o caractere 's'. A avaliação SWO consegue avaliar corretamente esses casos, pois a janela deslizante é bastante precisa em identificar se há segmentos faltantes (segmento vermelho) e se há segmentos não correspondentes (segmento violeta), como mostrado na Figura-46-e.

Na próxima seção, serão apresentados um comparativo entre a métrica CT e a avaliação SWO, mostrando a diferença de acurácia de ambas as avaliações.

6.6 COMPARAÇÃO QUANTITATIVA ENTRE CT E SWO

Nesta seção serão apresentados os resultados quantitativos da avaliação SWO em comparação com a métrica de avaliação CT. É utilizado apenas a CT como comparação pois é a métrica que também avalia a trajetória completa das coordenadas. Aliás, uma amostra com o CT correto possui também o SP e seus JP corretos, então seus resultados também são computados dentro de CT.

A Tabela 13 apresenta os resultados da avaliação SWO com a rede *netBh* em comparação com os resultados da avaliação CT.

Tabela 13 – Taxas de acurácia da rede *netBh* na avaliação CT e SWO

Base	**single-stroke**		**multi-stroke**	
	CT	SWO	CT	SWO
$20p/c$	$84,58\%$	$91,36\%$	$82,21\%$	$88,66\%$
$30p/c$	$79,20\%$	$91,67\%$	$76,25\%$	$88,19\%$
$40p/c$	$71,86\%$	$87,82\%$	$67,85\%$	$79,59\%$
$50p/c$	$67,76\%$	$86,04\%$	$61,62\%$	$77,64\%$
$60p/c$	$56,12\%$	$83,69\%$	$48,71\%$	$72,05\%$
$70p/c$	$49,32\%$	$79,52\%$	$43,16\%$	$57,99\%$
$100p/c$	$32,01\%$	$39,73\%$	$20,81\%$	$32,19\%$

Fonte – Autoria própria

De acordo com os dados da Tabela 13, a nova avaliação, o qual possibilita a correção dos falsos negativos e falsos positivos ocasionados pela avaliação CT, mostrou que as predições da rede *netBh* possuem resultados melhores comparado aos resultados comentados no Capítulo 5. Foi possível mostrar que a rede possui resultados promissores sobre a base *single-stroke* de $70p/c$, atingindo cerca de 30 pontos percentuais a mais comparada a avaliação CT.

Na avaliação SWO, os resultados com as bases *single-stroke* com $20p/c$ e $70p/c$ diferem em cerca de 12 pontos percentuais, enquanto que na avaliação CT a diferença é de quase 36 pontos percentuais. Essa diferença mostra que a avaliação CT não consegue computar com eficiência as coordenadas preditas oriundas de imagens com mais pontos por caractere. A avaliação proposta, por utilizar uma janela deslizante e por dividir o caractere em segmentos para buscar por pontos ordenados de cada segmento, consegue mostrar que as coordenadas preditas correspondem a uma ordenação correta de traçado no formato geométrico correto do caractere original. A diferença de acurácia também é notada para as bases de *multi-strokes*.

A Tabela 14 apresenta os resultados comparativos das outras redes implementadas neste trabalho em todas as bases de *single* e *multi-stroke*.

Tabela 14 – Taxas de acerto de trajetórias completas das redes desenvolvidas na avaliação *CT* e na avaliação proposta por este trabalho.

Base	*net-v0*				*net-v1*				*net-v2*				*net-v3*			
	single-stroke		***multi-strokes***		***single-stroke***		***multi-strokes***		***single-stroke***		***multi-strokes***		***single-stroke***		***multi-strokes***	
	CT	**SWO**	**CT**	**SWO**	**CT**	**SWO**	**CT**	**SWO**	**CT**	**SWO**	**CT**	**SWO**	**CT**	**SWO**	**CT**	**SWO**
20p/c	84,65%	90,75%	82,37%	92,21%	85,17%	95,31%	83,39%	92,93%	85,01%	90,79%	82,38%	89,59%	81,89%	85,17%	80,84%	81,91%
30p/c	79,17%	94,34%	77,56%	92,13%	81,18%	95,14%	77,22%	92,69%	79,23%	92,39%	76,13%	86,40%	76,00%	76,97%	71,08%	73,37%
40p/c	74,33%	92,51%	69,21%	87,69%	72,27%	91,06%	68,08%	87,31%	73,05%	89,86%	68,68%	83,13%	67,81%	76,42%	61,32%	72,89%
50p/c	68,02%	88,56%	63,63%	84,31%	66,42%	87,97%	62,52%	83,42%	64,11%	86,46%	62,03%	80,89%	57,92%	65,75%	51,57%	59,33%
60p/c	60,84%	86,59%	58,46%	81,44%	54,84%	85,95%	51,29%	78,76%	56,19%	81,01%	55,18%	75,26%	51,57%	59,24%	41,81%	50,49%
70p/c	46,02%	83,02%	38,45%	70,48%	55,04%	86,93%	44,71%	72,28%	48,37%	78,27%	48,33%	65,85%	44,68%	50,36%	32,06%	47,30%
100p/c	29,87%	44,82%	20,11%	32,45%	39,55%	45,13%	23,86%	33,02%	25,70%	45,45%	20,94%	33,15%	25,28%	23,29%	19,46%	22,98%

Fonte – Autoria própria

Como observado na Tabela 14, existe um aumento considerável no quesito de acurácia de trajetória completa quando se compara a avaliação CT e a avaliação SWO. As acurácias melhoram tanto para *single-stroke* quanto para *multi-strokes* das bases de $60p/c$ e $70p/c$ para as redes *net-v0*, *net-v1* e *net-v2*, tendo uma média de 80% de acerto para SWO, enquanto que a CT possui uma média de 55%. A diferença de acurácia entre uma base e outra é bastante grande quando se compara as duas métricas de avaliação. A avaliação CT entre as bases *single-stroke* de $20p/c$ e a de $70p/c$ chega a ser de 40%. Essa diferença não acontece na avaliação proposta neste trabalho.

A base $100p/c$ apresentou comportamentos parecidos para as duas avaliações pois em ambas resultaram em taxas baixas de acurácia. Isso pode ter ocorrido pelo fato de que quanto maior a quantidade de informações que a rede precisa aprender, mais tempo (épocas) ela precisá levar para aprender e será preciso mais dados para a base de treinamento para aumentar a capacidade de generalização. As redes provavelmente não tiveram um treinamento suficiente para trabalhar com a complexidade dos dados ao ponto de apresentar uma capacidade de generalização maior para as amostras mais robustas.

A partir dos dados da Tabela 14 pode ser observado que a predição da trajetória de caracteres *multi-strokes* é uma tarefa de complexidade maior. Por ser de complexidade maior, as taxas de acurácia das redes diminuíram em relação as taxas de *single-stroke*. Apenas na rede *net-v0* em $20p/c$ foi possível ter resultados melhores de *multi-strokes*.

A partir dos resultados, pode-se deduzir que as redes de aprendizagem profunda necessitam de mais tempo e mais amostras de treinamento para se conseguir atingir um nível maior de generalização. Entretanto, é possível diminuir a complexidade desses caracteres e conseguir resultados satisfatórios diminuindo a quantidade de pontos por caractere. Através de uma configuração de rede específica, a rede *net-v1* atingiu as maiores taxas de acurácias de $95,31\%$ para caracteres *single-stroke* e $92,93\%$ para caracteres de *multi-strokes* com $20p/c$.

6.7 CONSIDERAÇÕES DO CAPÍTULO

A partir dos falsos positivos e falsos negativos analisados da avaliação SP-JP-CT comentados na Seção 5.4, verificou-se as falhas desse método avaliativo e foi possível desenvolver uma avaliação mais eficaz que avalia a corretude da trajetória completa. Essa nova avaliação é chamada de SWO (*Segmentation - sliding Window - Ordering*), a qual faz uma verificação tanto na forma geométrica da predição quanto na ordenação dos pontos preditos.

A análise qualitativa da avaliação SWO mostra a eficácia do método avaliativo utilizado. A segmentação por TSF (*Trajectory Segmentation Factor*) faz com que os pontos preditos sejam amarrados a determinado segmento através da janela deslizante. Essa janela realiza a validação da similaridade geométrica dos pontos preditos com o esqueleto *ground-truth*.

A partir dos resultados analisados, a correção dos falsos negativos da avaliação SP-JP-CT mostrou uma melhoria dos resultados, sendo possível alcançar 95% de taxa de acurácia, mostrando um maior potencial das redes de *deep learning* desenvolvidas neste trabalho.

7 CONCLUSÕES

Este trabalho apresentou uma metodologia para realizar a HTR utilizando redes de *deep learning*. Apresentou-se também um estudo sobre o treinamento das redes em diferentes quantidades de pontos por caracteres, além de propor uma métrica nova para avaliação da predição de pontos.

A base de caracteres minúsculos latinos IRONOFF foi utilizada para os experimentos. As informações *online* dos caracteres passam pelo processo de redimensionamento, normalização e *data augmentation*. Com a normalização, foi possível criar sete bases diferentes: 20 pontos por caractere (p/c), $30p/c$, $40p/c$, $50p/c$, $60p/c$, $70p/c$ e $100p/c$. Além disso, separou-se dos caracteres de *multi-strokes* (um ou mais traços) os caracteres *single-stroke* (um traço), totalizando 14 bases de dados para os experimentos. As informações *online* são pixeladas em matrizes digitais, transformando-as em imagens *offline* que servirão de entrada para as redes de *deep learning*.

Cinco configurações diferentes de redes foram implementadas. Todas compõem-se de uma estrutura hierárquica de CNN e LSTM bidirecional, sendo a primeira para extração de características da imagem e a segunda para aprender a dependência de dados temporais das coordenadas. As redes diferem de quantidade de camadas e são treinadas em todas as bases de dados mencionadas, totalizando 70 redes de *deep learning* treinadas para HTR.

Para a análise dos resultados, as coordenadas preditas pelas redes são inicialmente avaliadas utilizando as métricas encontradas na literatura. São elas: avaliação dos pontos de início (SP), avaliação dos pontos de junção (JP) e avaliação da trajetória completa (CT). Esta avaliação mostrou resultados promissores, atingindo como melhor resultado a taxa de 85% de acurácia para a base *single-stroke* de $20p/c$. Entretanto, as análises qualitativas da avaliação mostraram que cerca de 15% das amostras são avaliadas incorretamente. Predições que não tinham similaridade geométrica com o *ground-truth* eram incorretamente avaliadas como certas, causando uma inconsistência no resultado calculado final.

A partir disso, foi proposto um novo método avaliativo para poder avaliar com mais eficiência as coordenadas preditas. Este método, chamado de SWO (*Segmentation - sliding Window - Ordering*), conta com um sistema para verificação da similaridade geométrica através de uma janela deslizante e um fator de segmentação de trajetória TSF (*Trajectory Segmentation Factory*) para auxiliar na avaliação da ordenação das coordenadas preditas. Através das análises qualitativas e quantitativas do método SWO, foi possível mostrar sua eficácia para avaliar as coordenadas preditas das redes *deep learning* para HTR.

Com a avaliação SWO, conseguiu-se atingir uma taxa de $95,31\%$ de acurácia para a base *single-stroke* de $20p/c$ e $92,93\%$ de acurácia para a base *multi-strokes* de $20p/c$. Dessa forma, conseguiu-se mostrar que é possível obter melhores resultados com menos informações de entrada. Os resultados comparativos entre a avaliação CT e SWO mostram que a avaliação

proposta é promissora para avaliar as coordenadas preditas de sistemas de HTR, além de mostrar que as redes desenvolvidas neste trabalho são capazes de realizar a HTR para uma base de dados de caracteres latinos.

7.1 TRABALHOS FUTUROS

Este trabalho define um ponto de partida para uma série de trabalhos futuros que ainda podem ser realizados a fim de explorar mais as *deep learning* para a HTR. Alguns dos trabalhos são listados a seguir:

- Aumentar o tempo de treinamento para as redes configuradas com maior número de camadas convolucionais, para verificar se há uma melhoria na taxa de acurácia;
- A partir das redes treinadas, ampliar o tamanho de entrada para verificar se é possível recuperar a trajetória de conjuntos de caracteres, como sílabas e palavras;
- Aplicar os métodos de *Transfer Learning*, utilizando redes pré-treinadas com imagems, como a VGG16, proposta por Simonyan e Zisserman (2014), e a ImageNet, proposta por Russakovsky *et al.* (2015);
- Realizar a *Transfer Learning* entre as bases criadas com diferentes pontos por caractere para realizar a predição de caracteres que possuem quantidades de pontos de coordenadas variadas;
- Ainda com *Transfer Learning*, utilizar as redes treinadas neste trabalho para treinar com caracteres de outras bases de dados, como a UNIPEN, proposta por Guyon *et al.* (1994);
- Aplicar e analisar quantitativamente e qualitativamente o sistema de avaliação SWO em sistemas HTR de *deep learning* aplicado a outras bases de caracteres latinas, como a UNIPEN, e também bases de outros idiomas, como a base de caracteres chineses OLHWDB proposto por Liu *et al.* (2013);
- Explorar a capacidade de recuperar partes faltantes dos caracteres utilizando o zoneamento proposto por Freitas *et al.* (2007), verificando se é possível predizer a trajetória desta parte faltante;
- Desenvolver uma rede *deep learning* para classificar caracteres baseando-se nas informações *offline* e *online*, sendo esta última, geradas por um sistema de HTR utilizando *deep learning* como as redes desenvolvidas nesta pesquisa;
- Construir um sistema de seleção que consiga decidir a melhor rede para recuperar a trajetória da imagem de entrada, uma vez que, ao realizar uma comparação entre as redes

desenvolvidas neste trabalho, algumas redes apresentam melhores taxas de acertos em determinados caracteres, podendo ter a possibilidade de melhorar a taxa de acurácia.

REFERÊNCIAS

AIRES, S. B. K. **Reconhecimento de caracteres manuscritos baseados em regiões perceptivas**. 2005. 97f. Dissertação de Mestrado em Informática Aplicada — Pontificia Universidade Católica Do Paraná, Curitiba, 2005.

BHUNIA, A. K. *et al.* Handwriting trajectory recovery using end-to-end deep encoder-decoder network. In: INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION. **Proceedings...** [S.l.], 2018. p. 3639–3644.

BOYKOV, Yuri; VEKSLER, Olga; ZABIH, Ramin. Fast approximate energy minimization via graph cuts. **IEEE Transactions on pattern analysis and machine intelligence**, IEEE, v. 23, n. 11, p. 1222–1239, 2001.

BRESENHAM, Jack E. Algorithm for computer control of a digital plotter. **IBM Systems journal**, IBM, v. 4, n. 1, p. 25–30, 1965.

CARVALHO, A. *et al.* **Inteligência Artificial–uma abordagem de aprendizado de máquina**. [S.l.]: Rio de Janeiro: LTC, 2011.

CHAI, Tianfeng; DRAXLER, Roland R. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. **Geoscientific model development**, Copernicus GmbH, v. 7, n. 3, p. 1247–1250, 2014.

DIAS, F. G.; SOARES, H. C.; FONSECA, L. M. G. Estudo da aplicação de filtros de detecção de bordas na identificação da frente termal da corrente do brasil. In: SIMPÓSIO BRASILEIRO DE SENSORIAMENTO REMOTO. **Anais...** [S.l.], 2011. p. 7086–7092.

DINH, M. *et al.* Recovery of drawing order from multi-stroke english handwritten images based on graph models and ambiguous zone analysis. Expert Systems with Applications, Elsevier, 2016. Disponível em: <https://doi.org/10.1016/j.eswa.2016.08.017>. Acesso em: 23 de janeiro de 2019.

ELBAATI, Abdelkarim; HAMDI, Yahia; ALIMI, Adel M. Handwriting recognition based on temporal order restored by the end-to-end system. In: IEEE. **2019 International Conference on Document Analysis and Recognition (ICDAR)**. [S.l.], 2019. p. 1231–1236.

ELBAATI, A. *et al.* Temporal order recovery of the scanned handwriting. In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION. **Proceedings...** [S.l.], 2009. p. 1116–1120.

FREITAS, Cinthia OA *et al.* Handwritten character recognition using nonsymmetrical perceptual zoning. **International Journal of Pattern Recognition and Artificial Intelligence**, World Scientific, v. 21, n. 01, p. 135–155, 2007.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: Continual prediction with lstm. **Neural Computation**, IET, v. 10, n. 12, p. 2451—-2471, 1999. Disponível em: <https://doi.org/10.1162/089976600300015015>. Acesso em: 20 de março de 2019.

GIORGINO, Toni *et al.* Computing and visualizing dynamic time warping alignments in r: the dtw package. **Journal of statistical Software**, v. 31, n. 7, p. 1–24, 2009.

GONZALEZ, R. C.; WOODS, R. C. **Processamento digital de imagens .** [S.l.]: Pearson Educación, 2009.

GRAVES, A.; JAITLY, N.; MOHAMED, A. Hybrid speech recognition with deep bidirectional lstm. In: WORKSHOP ON AUTOMATIC SPEECH RECOGNITION AND UNDERSTANDING. **Proceedings...** [S.l.], 2013. p. 273–278.

GUYON, Isabelle *et al.* Unipen project of on-line data exchange and recognizer benchmarks. In: IEEE. **Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3-Conference C: Signal Processing (Cat. No. 94CH3440-5)**. [S.l.], 1994. v. 2, p. 29–33.

HAFEMANN, L. G.; OLIVEIRA, L. S.; SABOURIN, R. Fixed-sized representation learning from offline handwritten signatures of different sizes. **International Journal on Document Analysis and Recognition**, Springer, v. 21, n. 3, p. 219–232, 2018. Disponível em: <https://doi.org/10.1007/s10032-018-0301-6>. Acesso em: 15 de novembro de 2018.

HINTON, G. *et al.* Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. **Signal processing magazine**, IEEE, v. 29, n. 6, p. 82–97, 2012. Disponível em: <https://doi.org/10.1109/MSP.2012.2205597>. Acesso em: 3 de abril de 2019.

HUANG, Z. *et al.* Speech emotion recognition using cnn. In: ACM INTERNATIONAL CONFERENCE ON MULTIMEDIA. **Proceedings...** [S.l.], 2014. p. 801–804.

IVERSON, Kenneth E. A programming language. In: **Proceedings of the May 1-3, 1962, spring joint computer conference**. [S.l.: s.n.], 1962. p. 345–351.

IWAKIRI, Y. *et al.* On the possibility of instance-based stroke recovery. In: INTERNATIONAL CONFERENCE ON FRONTIERS IN HANDWRITING RECOGNITION. **Proceedings...** [S.l.], 2012. p. 29–34.

KEYSERS, D. *et al.* Multi-language online handwriting recognition. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 39, n. 6, p. 1180–1194, 2017.

KHA, V. A.; KHA, H. H.; BLUMENSTEIN, M. Extraction of dynamic trajectory on multi-stroke static handwriting images using loop analysis and skeletal graph model. **REV Journal on Electronics and Communications**, IEEE, v. 6, n. 1-2, 2016. Disponível em: <http://dx.doi.org/10.21553/rev-jec.131>. Acesso em: 23 de janeiro de 2019.

KHAN, Kamran Ullah; HAIDER, Ihtesham. Online recognition of multi-stroke handwritten urdu characters. In: IEEE. **2010 International Conference on Image Analysis and Signal Processing**. [S.l.], 2010. p. 284–290.

KINJARAPU, A. K.; NARR, S.; VALUROUTHU, K. P. Writing order recovery from telugu character images. **International Journal of Computer Systems**, v. 3, n. 4, p. 347–355, April 2016. Disponível em: <http://www.ijcsonline.com/IJCS/Vol03_Issue04/Writing_Order_Recovery_from_Telugu_Character_Images.php>. Acesso em: 3 de fevereiro de 2019.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. **Proceedings...** [S.l.], 2012. p. 1097–1105.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group, v. 521, n. 7553, p. 436–444, 2015. Disponível em: <https://doi.org/10.1038/nature14539>. Acesso em: 9 de setembro de 2019.

LECUN, Y. *et al.* Gradient-based learning applied to document recognition. **https://doi.org/10.1109/5.726791/**, v. 86, p. 2278–2324, 1998. Disponível em: <https://doi.org/10.1109/5.726791>. Acesso em: 22 de abril de 2018.

LEUNG, M. K. *et al.* Deep learning of the tissue-regulated splicing code. **Bioinformatics**, Oxford University Press, v. 30, n. 12, p. i121–i129, 2014. Disponível em: <https://doi.org/10.1093/bioinformatics/btu277>. Acesso em: 15 de outubro de 2019.

LIU, Cheng-Lin *et al.* Online and offline handwritten chinese character recognition: benchmarking on new databases. **Pattern Recognition**, Elsevier, v. 46, n. 1, p. 155–162, 2013.

MA, J. *et al.* Deep neural nets as a method for quantitative structure–activity relationships. **Journal of chemical information and modeling**, ACS Publications, v. 55, n. 2, p. 263–274, 2015. Disponível em: <https://pubs.acs.org/doi/10.1021/ci500747n>. Acesso em: 16 de outubro de 2019.

MIKOLOV, T. *et al.* Strategies for training large scale neural network language models. In: WORKSHOP ON AUTOMATIC SPEECH RECOGNITION AND UNDERSTANDING. **Proceedings...** [S.l.], 2011. p. 196–201.

MOHAMED, A. E.; HAMIDA, A. M. S. Convolutional neural network for edge detection in sar grayscale images. **Journal of VLSI and Signal Processing**, v. 4, p. 75–83, 2014. Disponível em: <https://doi.org/10.9790/4200-04217583>. Acesso em: 22 de março de 2019.

MUKAKA, Mavuto M. A guide to appropriate use of correlation coefficient in medical research. **Malawi medical journal**, Medical Association of Malawi, v. 24, n. 3, p. 69–71, 2012.

NAGI, J. *et al.* Max-pooling convolutional neural networks for vision-based hand gesture recognition. In: INTERNATIONAL CONFERENCE ON SIGNAL AND IMAGE PROCESSING APPLICATIONS. **Proceedings...** [S.l.], 2011. p. 342–347.

NAGOYA, T.; FUJIOKA, H. A graph theoretic algorithm for recovering drawing order of multi-stroke handwritten images. In: INTERNATIONAL CONFERENCE ON INTELLIGENT NETWORKING AND COLLABORATIVE SYSTEMS. **Proceedings...** [S.l.], 2011. p. 569–574.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. **Journal of molecular biology**, Elsevier, v. 48, n. 3, p. 443–453, 1970. Disponível em: <https://doi.org/10.1016/0022-2836(70)90057-4>. Acesso em: 20 de janeiro de 2019.

NGUYEN, V.; BLUMENSTEIN, M. Techniques for static handwriting trajectory recovery: a survey. In: INTERNATIONAL WORKSHOP ON DOCUMENT ANALYSIS SYSTEMS. **Proceedings...** [S.l.], 2010. p. 463–470.

NOUBIGH, Z.; KHERALLAH, M. A survey on handwriting recognition based on the trajectory recovery technique. In: INTERNATIONAL WORKSHOP ON ARABIC SCRIPT ANALYSIS AND RECOGNITION. **Proceedings...** [S.l.], 2017. p. 69–73.

PARKHI, Omkar M *et al.* Deep face recognition. In: BRITISH MACHINE VISION CONFERENCE. **Proceedings**. [S.l.], 2014. p. 41.1–41.12.

PEDRINI, H.; SCHWARTZ, W.R. **Análise de imagens digitais: princípios, algoritmos e aplicações**. THOMSON PIONEIRA, 2008. ISBN 9788522105953. Disponível em: <https://books.google.com.br/books?id=13KAPgAACAAJ>.

PHAN, D. *et al.* Triangulation based skeletonization and trajectory recovery for handwritten character patterns. **Ksii Transactions on Internet & Information Systems**, v. 9, n. 1, 2015. Disponível em: <http://www.itiis.org/digital-library/manuscript/940>. Acesso em: 2 de maio de 2018.

PLAMONDON, R.; SRIHARI, S. N. Online and off-line handwriting recognition: a comprehensive survey. **IEEE Transactions on pattern analysis and machine intelligence**, IEEE, v. 22, n. 1, p. 63–84, 2000.

RAWAT, Waseem; WANG, Zenghui. Deep convolutional neural networks for image classification: A comprehensive review. **Neural Computation**, v. 29, n. 9, p. 2352–2449, 2017. PMID: 28599112. Disponível em: <https://doi.org/10.1162/neco_a_00990>.

ROUSSEAU, L.; CAMILLERAPP, J.; ANQUETIL, E. What knowledge about handwritten letters can be used to recover their drawing order? In: INTERNATIONAL WORKSHOP ON FRONTIERS IN HANDWRITING RECOGNITION. **Proceedings**. [S.l.], 2006.

RUSSAKOVSKY, Olga *et al.* Imagenet large scale visual recognition challenge. **International journal of computer vision**, Springer, v. 115, n. 3, p. 211–252, 2015.

SCHMIDHUBER, J.; HOCHREITER, S. Long short-term memory. **Neural Computation**, v. 9, n. 8, p. 1735–1780, 1997. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>. Acesso em: 14 de dezembro de 2018.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. **Understanding machine learning: From theory to algorithms**. [S.l.]: Cambridge university press, 2014.

SHARMA, A. Recovery of drawing order in handwritten digit images. In: INTERNATIONAL CONFERENCE ON IMAGE INFORMATION PROCESSING. **Proceedings...** [S.l.], 2013. p. 437–441.

____. A combined static and dynamic feature extraction technique to recognize handwritten digits. **Vietnam Journal of Computer Science**, Springer, v. 2, n. 3, p. 133–142, 2015.

SIMONYAN, Karen; ZISSERMAN, Andrew. Very deep convolutional networks for large-scale image recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SOBEL, Irwin; FELDMAN, Gary. A 3x3 isotropic gradient operator for image processing. **a talk at the Stanford Artificial Project in**, p. 271–272, 1968.

SOLA, J; SEVILLA, Joaquin. Importance of input data normalization for the application of neural networks to complex industrial problems. **IEEE Transactions on nuclear science**, IEEE, v. 44, n. 3, p. 1464–1468, 1997.

STEFANO, C. D. *et al.* Reading cursive handwriting. In: INTERNATIONAL CONFERENCE ON FRONTIERS IN HANDWRITING RECOGNITION. **Proceedings**. [S.l.], 2010. p. 95–100.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. **Proceedings...** [S.l.], 2014. p. 3104–3112.

TOMPSON, J. J. *et al.* Joint training of a convolutional network and a graphical model for human pose estimation. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. **Proceedings...** [S.l.], 2014. p. 1799–1807.

TRUCCO, Emanuele; VERRI, Alessandro. **Introductory techniques for 3-D computer vision**. [S.l.]: Prentice Hall Englewood Cliffs, 1998. v. 201.

VIARD-GAUDIN, C. *et al.* The ireste on/off (ironoff) dual handwriting database. In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION. **Proceedings...** [S.l.], 1999. p. 455–458.

VU, H. N.; NA, I. S.; KIM, S. H. Correction of text character skeleton for effective trajectory recovery. **International Journal of Contents**, v. 11, n. 3, 2015.

WANG, Yujung *et al.* Two-stage fully convolutional networks for stroke recovery of handwritten chinese character. In: SPRINGER. **Asian Conference on Pattern Recognition**. [S.l.], 2019. p. 321–334.

WU, Yonghui *et al.* Google's neural machine translation system: Bridging the gap between human and machine translation. **CoRR**, abs/1609.08144, 2016.

ZHANG, X. Y.; BENGIO, Y.; LIU, C. L. Online and offline handwritten chinese character recognition: A comprehensive study and new benchmark. **Pattern Recognition**, Elsevier, v. 61, p. 348–360, 2017.

ZHAO, Bocheng; YANG, Minghao; TAO, Jianhua. Drawing order recovery for handwriting chinese characters. In: IEEE. **ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.], 2019. p. 3227–3231.

ZHENG, Liang *et al.* Good practice in cnn feature transfer. **CoRR**, 2016.