DOI: 10.36558/rsc.v14i1.8705

Uso de Reconhecimento Óptico de Caracteres para Extração de Textos em Imagens de Redações

Using Optical Character Recognition to Extract Text from Essays Images

Filipe A. Sampaio
Departamento de Computação
Universidade Federal do Piauí
felipealvessampaio@hotmail.com

Raimundo S. Moura
Departamento de Computação
Universidade Federal do Piauí
rsm@ufpi.edu.br

Kelson R. T. Aires
Departamento de Computação
Universidade Federal do Piauí
kelson@ufpi.edu.br

RESUMO

Avaliação Automática de Redação é uma tarefa da área de Processamento de Linguagem Natural, cujo objetivo é avaliar e pontuar textos em prosa escrita. Uma das principais dificuldades desta tarefa é a deficiência de datasets de redações anotadas com o valor obtido em cada competência. Assim, este trabalho propõe uma solução eficaz para capturar as redações escritas por alunos, através de técnicas de visão computacional e reconhecimento óptico de caracteres. Esse trabalho segmenta palavras da imagem do texto da redação e processa cada palavra, reconhecendo então o texto de cada imagem. Ao final, ordena todas as palavras na sequência correta da leitura, obtendo desempenho moderado.

Palavras-chave

Visão Computacional. Reconhecimento Óptico de Caracteres. Rede Neural Recorrente Convolucional.

ABSTRACT

Automatic Essay Scoring is a task in the area of Natural Language Processing, whose objective is to evaluate and score written prose texts. One of the main difficulties of this task is the lack of datasets of essays annotated with the value obtained in each competence. Thus, this work proposes an effective solution to capture essays written by students, through computer vision and optical character recognition techniques. This paper segments words from the image of the essay text and processes each word, then recognizing the text of each image. At the end, it orders all the words in the correct reading sequence, obtaining moderate performance.

Keywords

Computer Vision. Optical Character Recognition. Convolutional Recurrent Neural Network.

CCS Concepts

•**Computing methodologies** → **Neural networks;**

1. INTRODUÇÃO

Atualmente, no Brasil, o Exame Nacional do Ensino Médio (ENEM) possui a maior prova de redação do país em termos de participantes [9]. Na edição de 2023, segundo [7], houve mais de 3,3 milhões de inscrições, com 2,3 milhões realizando a prova. Apenas 60 candidatos obtiveram nota 1.000 na redação do exame, com 1,63% dos participantes zerando a nota de redação. No ano de 2022, 18 participantes conseguiram a nota máxima na redação, com 1,91% dos participantes zerando a mesma. Já ano de 2021, 22 candidatos obtiveram nota máxima, com 4,57% zerando a prova de redação.

Observa-se, segundo [8], que entre as edições de 2014 e 2020, o desempenho das notas de redação mantiveram uma certa estabilidade na média geral, com oscilações entre 506,5 e 531,9. Os temas abordados nessas edições foram variados, abrangendo desde temas sociais, políticos, econômicos e culturais.

Porém, é notado um desafio constante entre as provas, onde em cada edição os elaboradores buscaram abordar temas complexos e de grande relevância social, exigindo mais dos candidatos em apresentar argumentos sólidos e bem fundamentados e propor soluções viáveis para os problemas discutidos.

Já nas recentes edições de 2020 a 2023, segundo [7], a análise das notas de redações do Enem revela uma tendência de crescimento, especialmente em 2023. Houve um aumento significativo da média desse ano em questão, com 641,6, em comparação com 2020, que foi 531,9. Segundo [8], os possíveis fatores que contribuíram para esse crescimento estão relacionados ao retorno presencial das aulas com foco na recuperação da aprendizagem, uma maior familiaridade com o formato da prova junto com as expectativas da banca e

uma preparação mais direcionada.

Segundo [2], os resultados que um participante terá na redação está relacionado ao quanto ele pratica antes de realizar a prova. Quanto menor for o volume de produções textuais dos candidatos durante o período de estudos para o exame, menor será seu resultado. Além disso, a falta de leitura por parte dos mesmos também influencia no seu repertório textual.

A evolução do desempenho dos candidatos para elaboração de boas redações está ligada diretamente também ao *feedback* que professores e profissionais da área de Letras língua portuguesa fornecem ao corrigir seus textos, informando o que deve ser melhorado.

Nota-se também um aumento na sobrecarga do trabalho desses profissionais, que acompanham os alunos durante todo o período de preparação para o exame. Para auxiliar nesse problema, há serviços onde os candidatos digitam suas redações e submetem em formulários para avaliação por um profissional.

Um exemplo é o UOL Brasil Escola[1], que todo mês disponibiliza um tema de redação diferente para os alunos. A redação é corrigida manualmente e o aluno pode acompanhar sua evolução com suas notas recebidas.

Há também protótipos de ferramentas para correção automática de redações. Essas ferramentas são projetadas para auxiliar o profissional na correção, diminuindo sua carga de trabalho e otimizando o acompanhamento mais eficiente para com o aluno. Um exemplo desse tipo de ferramenta é o AAREM (Avaliador Automático de Redações para o Ensino Médio) [12], que apresenta estratégias para avaliação automática de redações escritas em português por meio de uma abordagem baseada na definição de *features* e modelos específicos para cada competência da matriz de referência do ENEM.

Observa-se também que geralmente na proposta de correção manual, o candidato submete uma imagem da redação que foi escrita a mão. Assim, o tempo da correção e retorno do *feedback* está ligado diretamente na qualidade da imagem submetida e da escrita do aluno.

Já a proposta de correção por ferramentas automáticas depende que o candidato digite seu texto em um formulário, prejudicando o mesmo na prática da escrita em folha. A submissão de texto para as ferramentas automáticas de correção é um processo muito valioso para os sistemas de NLP (*Natural Language Processing* - Processamento de Linguagem Natural), pois segundo [11] há poucos *datasets* disponíveis para treinar modelos de correção.

Para incentivar o desenvolvimento de modelos eficientes, há algumas competições que visam fomentar o estudo na área. Um exemplo é a competição PROPOR'24 [15], que busca desenvolver sistemas computacionais capazes de avaliar automaticamente redações para auxiliar os professores em sala de aula, aprimorando estratégias de *feedback*, permitindolhes focar em áreas específicas da redação que requerem aprimoramento de seus alunos.

Logo, é de grande importância que o candidato tenha a prática da escrita como uma constante em seus estudos diários. Assim também é necessário um *feedback* das correções textuais a partir do uso de ferramentas de correção, que pode otimizar o trabalho do profissional.

A proposta deste trabalho é possibilitar essa ligação, oferecendo uma metodologia para extrair textos de imagens e posterior submissão aos sistemas de correção. Tal metodologia utiliza de técnicas de visão computacional e reconhecimento óptico de caracteres através de redes neurais.

Este trabalho é uma versão estendida desenvolvida e apresentada no trabalho [16]. A principal contribuição desse trabalho é a análise e construção de uma metodologia simplificada capaz de realizar o reconhecimento de escrita manuscrita em imagens de palavras isoladas.

Atualmente, o principal problema dessa metodologia está ligada à qualidade da imagem e da escrita do candidato. Isso porque a metodologia leva em consideração que a imagem de entrada é uma imagem de uma redação tirada através de uma foto de celular ou digitalizada por impressora.

No primeiro caso, a qualidade da imagem é bastante afetada devido ao aparelho utilizado para a fotografia, e juntamente a isso, algumas características presentes na imagem afeta no processo de segmentação, como inclinação da fotografia, existencia de paulas horizontais e linhas de borda na folha da redação.

É importante ressaltar também o problema da própria escrita manual realizada por uma pessoa, que é variável, o que afeta no desempenho do modelo proposto.

O restante deste trabalho está organizado nas seguintes seções: na Seção 2 são citados os principais trabalhos selecionados sobre o assunto abordado; na Seção 3 é apresentada a proposta metodológica utilizada; a Seção 4 apresenta os experimentos realizados; na Seção 5 são discutidos os resultados obtidos nos experimentos; por fim, a Seção 6 apresenta a conclusão e os trabalhos futuros.

2. TRABALHOS RELACIONADOS

Existe na literatura alguns trabalhos que procuram estudar OCR (*Optical Character Recognition* - Reconhecimento de Caractere Optico) e ICR (*Intelligent Character Recognition* - Reconhecimento Inteligente de Caracteres) ligado a língua portuguesa, onde definem métodos novos ou sugerem melhorias em modelos atuais.

Observa-se também que há poucos trabalhos que focam no estudo de metodologias de extração de caracteres manuscritos no formato *offline*, principalmente ligado à língua portuguesa.

Na literatura, o reconhecimento de caligrafia automática, ou também conhecida como *handwriting text recognition*, é uma técnica onde utiliza-se um computador para receber e interpretar entradas de texto contidas em papel, documento, tela sensível ao toque ou fotografias.

Há dois tipos de reconhecimento de caligrafia. O *online handwriting recognition* ou reconhecimento online, envolve a conversação automática do texto conforme ele é escrito em um digitalizador especial. No dispositivo onde está sendo realizado a escrita, que pode ser tanto físico como digital, há um sensor ou mecanismo que capta os movimentos da ponta da caneta, bem como a alternância dos movimentos. Esse tipo de dados é conhecido como tinta digital e pode ser considerado dinâmico.

Já o *offline handwriting recognition* ou reconhecimento de escrita manual offline, envolve a conversão automática de texto de uma imagem em código de letras que pode ser usado em computadores e aplicativos de processamento de texto. O reconhecimento *offline* é comparativamente difícil, pois pessoas diferentes têm estilos de escrita diferentes.

Segundo [3], o reconhecimento de palavras da escrita cursiva abstrata é o problema de transformar uma palavra da forma icônica da escrita cursiva em sua forma simbólica. O

[1]https://brasilescola.uol.com.br/

autor mesmo descreve vários processos de um sistema de reconhecimento para palavras isoladas de escrita cursiva *offline*.

O autor descreve sistema de escrita *offline* como a falta de informações sobre a ordem ou padronização da escrita em uma folha de papel, durante a escrita de uma pessoa, que não necessariamente possui um padrão na disposição dos caracteres ou das palavras, tornando a leitura e reconhecimento por um software complexa.

Sua abordagem consiste em transformar uma imagem de palavra através de uma hierarquia de níveis de representação: pontos, contornos, características, letras e palavras. Uma representação de recurso exclusiva é gerada de baixo para cima a partir da imagem usando dependências estatísticas entre letras e recursos.

As classificações para palavras parcialmente formadas são calculadas usando um algoritmo de pilha. Várias novas técnicas para processamento de nível baixo e intermediário para escrita cursiva são descritas, incluindo heurística para localização de linhas de referência, segmentação de letras baseada na detecção de mínimos locais ao longo do contorno inferior e áreas com perfis verticais baixos, codificação simultânea de contornos e suas relações topológicas, extraindo recursos como *loop* intermediário, traço da zona superior e encontrando eventos orientados à forma.

Em [6] os autores se concentraram especialmente no reconhecimento *offline* de palavras manuscritas em inglês, detectando primeiro caracteres individuais.

As principais abordagens para reconhecimento de palavras manuscritas *offline* podem ser divididas em duas classes: holística e baseada em segmentação.

A abordagem holística é usada no reconhecimento de vocabulário de tamanho limitado, onde são consideradas características globais extraídas de toda a imagem da palavra. À medida que o tamanho do vocabulário aumenta, a complexidade dos algoritmos baseados em holísticas também aumenta e, consequentemente, a taxa de reconhecimento diminui rapidamente.

As estratégias baseadas na segmentação, por outro lado, empregam abordagens *bottom-up*, partindo do traço ou do nível do caractere e indo em direção à produção de uma palavra significativa.

Após a segmentação, o problema fica reduzido ao reconhecimento de simples caracteres ou traços isolados e, portanto, o sistema pode ser empregado para vocabulário ilimitado. Os autores adotaram o reconhecimento de palavras manuscritas baseado em segmentação, onde redes neurais são usadas para identificar caracteres individuais.

Foi explorado essas técnicas para projetar um sistema ideal para reconhecimento de palavras manuscritas em inglês no formato *offline*, baseado no reconhecimento de caracteres. Técnica de pós-processamento que usa léxico é empregada para melhorar a precisão geral do reconhecimento.

Em [20] os autores propuseram um método de detecção de texto de cena que consiste em dois estágios: uma rede totalmente convolucional e um estágio de fusão NMS (*Non-Maximum Suppression* - Supressão não máxima).

Segundo os autores, as abordagens anteriores para detecção de texto de cena já alcançaram desempenhos promissores em vários *benchmarks*, porém, os resultados ficam aquém ao lidar com cenários desafiadores, como cenas de ambientes abertos, mesmo quando equipados com modelos de redes neurais profundas, porque o desempenho geral é determinado pela interação de vários estágios e componentes no processo de detecção.

É proposto uma metodologia mais simples, que produz detecção de texto rápida e precisa em cenas naturais. O modelo desenvolvido prevê diretamente a posição das palavras ou linhas de texto, em orientações arbitrárias, eliminando etapas intermediárias desnecessárias (por exemplo, agregação de candidatos e particionamento de palavras), com uma única rede neural.

A rede totalmente conectada produz regiões de texto diretamente, excluindo etapas intermediárias redundantes e demoradas. A abordagem dos autores é mais simples que outras da literatura, que produz detecção de texto rápida e precisa em cenas naturais.

Experimentos em conjuntos de dados padrão, incluindo ICDAR 2015, COCO-Text e MSRA-TD500, demonstram que o algoritmo proposto supera significativamente os métodos de última geração em termos de precisão e eficiência. No conjunto de dados ICDAR 2015, o algoritmo proposto atinge um F-score de 0,7820 a 13,2 fps com resolução de 720p.

O trabalho de [19], propõe um sistema HTR (*Handwritten Text Recognition* - Reconhecimento de texto manuscrito) baseado em RNA (*Artificial Neural Network* - Rede Neural Artificial). Os métodos de pré-processamento aprimoram as imagens de entrada e, portanto, simplificam o problema do classificador.

Esses métodos incluem normalização de contraste e aumento de dados para balancear o conjunto de dados. Opcionalmente, o texto manuscrito é colocado na vertical por um algoritmo de inclinação.

O classificador possui camadas de CNN (*Convolutional Neural Network* - Rede Neural Convolucional) para extrair recursos da imagem de entrada e camadas de RNN (*Recurrent Neural Network* - Rede Neural Recorrente) para propagar informações através da imagem.

O processo consiste em uma operação sequencial, onde a imagem do texto entra em uma CNN, que são treinadas para extrair recursos relevantes da imagem. Logo após, os dados da CNN são passados para uma RNN, com uma sequência de 256 *features* por intervalo de tempo, onde o RNN propaga informações relevantes por meio dessa sequência.

O autor utilizou então uma LSTM (*Long Short Term Memory* - Memória de Curto Prazo Longa) devido sua capacidade de propagar informações por distâncias maiores e oferecer características de treinamento mais robustas.

A ideia de usar RNN é devido a geração de uma matriz que contém uma distribuição de probabilidade sobre os caracteres em cada posição da imagem. A decodificação dessa matriz produz o texto final e é feita pela operação CTC (*Connectionist Temporal Classification* - Classificação Temporal Conexionista).

Com a matriz gerada pela RNN sendo passada então para a CTC, é realizado um pós-processamento de texto no final, para corrigir erros ortográtifos no texto decodificado. É feito então a decodificação utilizando uma LM (*Language Model* - Modelo de Linguagem) implementada pelo autor, denominada *word beam search*, aumentando a acurácia final do processo.

O autor utilizou cinco conjuntos de dados públicos para avaliar o modelo proposto. A precisão é comparada a alguns trabalhos da literatura, apresentando resultados competitivos ao estado da arte.

[14] propõe uma metodologia a partir da análise de RNN,

que é utilizada para descobrir a disposição dos caracteres, introduzindo uma rede neural recorrente para reconhecer textos escritos à mão.

Há, segundo o autor, vários OCRs efetivamente acessíveis para múltiplos idiomas, porém a maioria são efetivos apenas para texto formal, mas para textos cursivos são incomuns, mostrando baixa precisão quando testados em textos do tipo *offline*.

Destaca também que os principais trabalhos focam em analisar textos escritos em inglês, devido a quantidade de textos científicos escritos nessa língua ser maior que qualquer outra lingua.

O autor propõe uma arquitetura simples contendo camadas de convoluções ligadas a camadas de recorrência, onde ao final é atribuído no processo uma camada CTC.

3. METODOLOGIA

Para realização do trabalho, foi planejada a implementação de algumas etapas, apresentadas no diagrama da Figura 1.

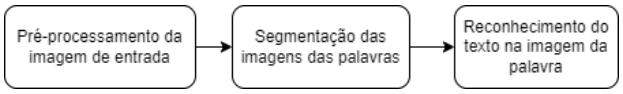


Figure 1: Diagrama apresentando o processo de reconhecimento de palavras.

No pré-processamento, é realizado o *upscaling* (aumento de resolução) da imagem de entrada, para melhorar a extração de características das palavras pelo reconhecimento do texto na imagem da palavra.

Na Figura 2 é apresentado o resultado da utilização do algoritmo FSRCNN[2], realizando aumento de 4x na resolução da imagem de entrada.

Figure 2: Aplicação do *upscaling* de 4x.

Logo em seguida é aplicado o algoritmo *deskew* (endireitamento), que aplica dilatações e detecção de borda para então calcular o ângulo de inclinação da imagem. É possível visualizar o resultado na Figura 3 e 4.

Na segmentação das palavras foi usado então um modelo proposto por [20] e [1], que classifica cada *pixel* como palavra (parte interna ou envolvente) ou *pixel* de fundo.

Para cada *pixel* da classe de palavra interna, é prevista uma AABB (*Axis Aligned Bounding Box* - Caixa Delimitadora de Eixo Alinhado) em torno da palavra.

Ao final é feito um agrupamento aos AABB previstos. O modelo é treinado no conjunto de dados *IAM Dataset* [13], onde é mostrado sua execução na Figura 5.

[2] *Fast Super Resolution Convolutional Neural Network* ou Rede Neural Convolucional de Super Resolução Rápida, é um modelo de aprendizado de máquina que pode ser usado para aumentar a resolução de imagens, implementado na biblioteca OpenCV.

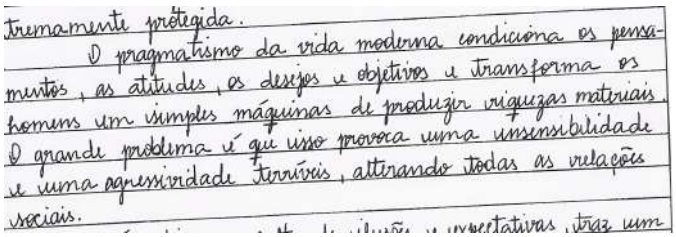


Figure 3: Imagem de entrada antes do *deskew*.

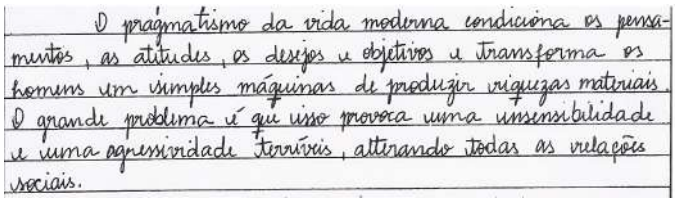


Figure 4: Aplicação de *deskew* na imagem de entrada.

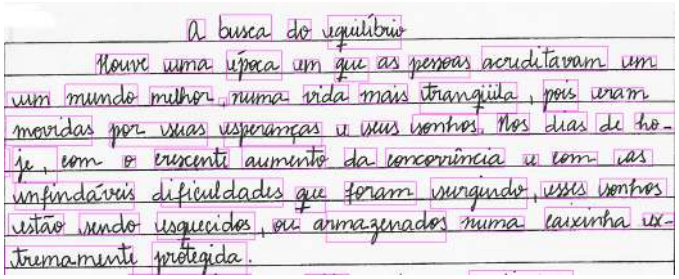


Figure 5: Modelo de detecção de palavras.

A segmentação das palavras utilizando o modelo de deteção de AABB é realizado no eixo vertical, devido o seu treinamento para detecção. Ou seja, o modelo realiza a detecção lendo a imagem de cima para baixo, na ordem em que os pixels são lidos.

O que significa que a segmentação das palavras não necessariamente será na ordem de leitura do texto, que é feito da esquerda para direita, de cima para baixo.

Para ajustar a segmentação na ordem correta da leitura, foi implementado um mecanismo que agrupa o centroide de cada AABB, de tal maneira que é possível plotar uma reta que conecta todos os centroides horizontais. Observe na Figura 6 que tal estratégia adiciona linhas horizontais cruzando o centróide de todos os AABBs detectados.

É feito a leitura do primeiro centroide do primeiro AABB, e logo na sequência, todos os centróides alinhados a direita deste serão conectados, formando então uma linha que irá conectar todas as palavras da linha horizontal.

Com isso, foi possível reordenar a segmentação para que cada palavra segmentada tivesse a sequência correta da leitura.

Para a remoção das linhas horizontais foram implementadas então técnicas de processamento de imagem, utilizando detecção de bordas e técnicas de fechamento e abertura, ajustando com dilatações e erosões para então detectar contornos e, ao final, remover as linhas horizontais. Na Figura 7 é possível visualizar o resultado.

Já no reconhecimento do texto nas imagens das palavras, foram realizados vários testes na arquitetura, iniciando a

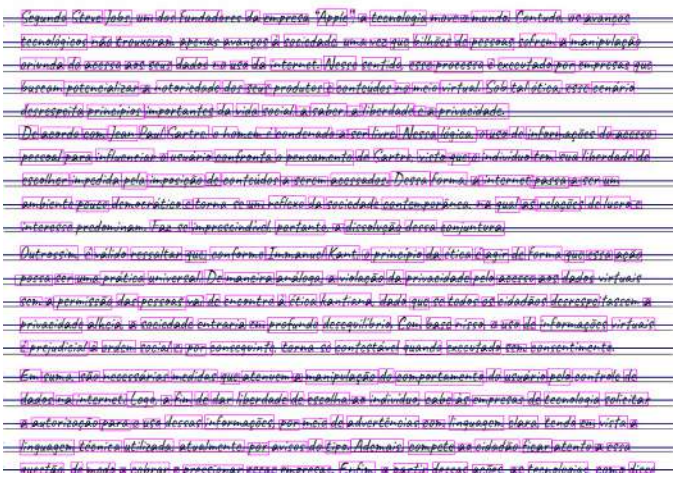


Figure 6: Criação de linhas horizontais para detectar ordem de leitura.

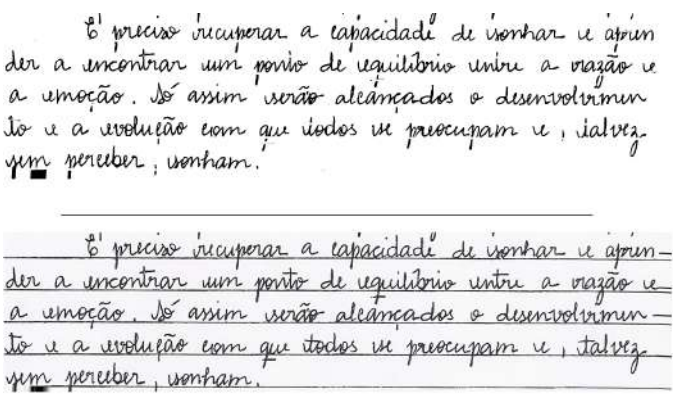


Figure 7: Remoção de linhas horizontais.

partir do modelo proposto por [19], que previa o reconhecimento de linhas de palavras.

O modelo proposto neste trabalho analisa imagens de palavras isoladas, para tentar otimizar os resultados e diminuir a complexidade do modelo, com uma entrada de 128x32 *pixels*. Na Figura 9 pode-se observar a arquitetura proposta.

A arquitetura em questão é denominada pela literatura de CRNN (*Convolutional Recurrent Neural Network* - Rede Neural Recorrente Convolucional), que consiste na concatenação de uma rede convolucional a uma rede recorrente.

Na Tabela 1 é apresentado a arquitetura desenvolvida, bem como suas camadas. Tal arquitetura foi projetada para reconhecer uma única palavra por imagem inserida. Ao todo são três camadas de convolução, seguidas por *batch normalization*, *max pooling* e *dropout*. Todos os hiperparametros foram definidos de forma empírica.

Ao final, a rede passa pela camada de transcrição, onde há o CTC, que gera pontuações de caracteres para cada elemento da sequência, que é representado por uma matriz.

Tal camada CTC será responsável por propagar a atualização da rede e também será usada para a inferência no momento da decodificação do texto final.

O CTC é um tipo de algoritmo usado para treinar redes neurais para detecção de fala, escrita e outros problemas

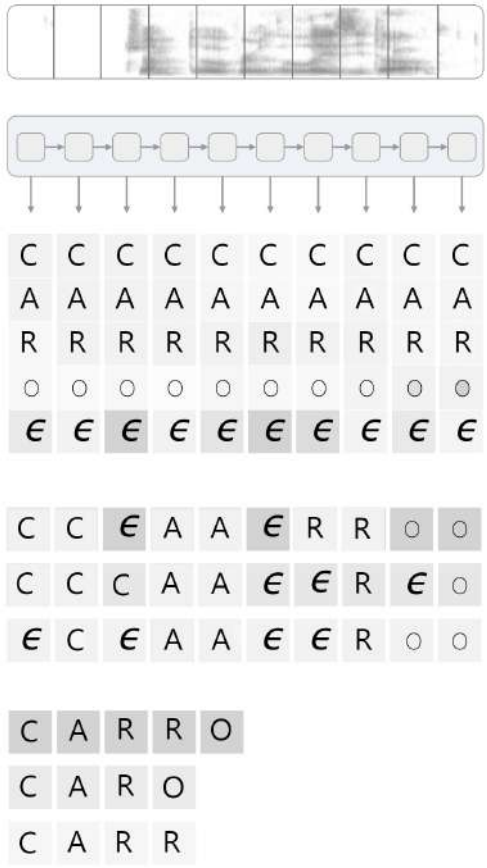


Figure 8: Funcionamento do CTC.

de sequência. É usado para contornar o não conhecimento do alinhamento entre a entrada e saída. Isso porque, para o problema de reconhecimento de palavras manuscritas, é impossível realizar anotações nas imagens das palavras para definir onde começa e onde termina cada letra. Observe na Figura 8 o processo de cálculo da CTC.

Após o cálculo das matriz gerada pela RNN, a rede fornecerá para o CTC uma distribuição sobre as saídas, ou seja, todas as possíbilidades de combinação entre os caracteres supostamente decodificados pelo modelo.

Dada as seguintes expressões, onde X são como palavras, para a sequência de saída correspondente, e Y como transcrições. O objetivo de usar o CTC é mapear X para Y.

$$X = [x1, x2, ..., xT] \tag{1}$$

$$Y = [y1, y2, ..., yT] \tag{2}$$

O algoritmo CTC pode atribuir uma probabilidade para qualquer Y, dado um X. A chave para calcular esta probabilidade é como o CTC calcula sobre os alinhamentos entre entradas e saídas.

O algoritmo CTC não tem alinhamento, ou seja, não requer alinhamento entre a entrada e a saída. No entanto, para obter a probabilidade de uma saída dada uma entrada, o CTC funciona somando a probabilidade de todos os alinhamentos possíveis entre os dois.

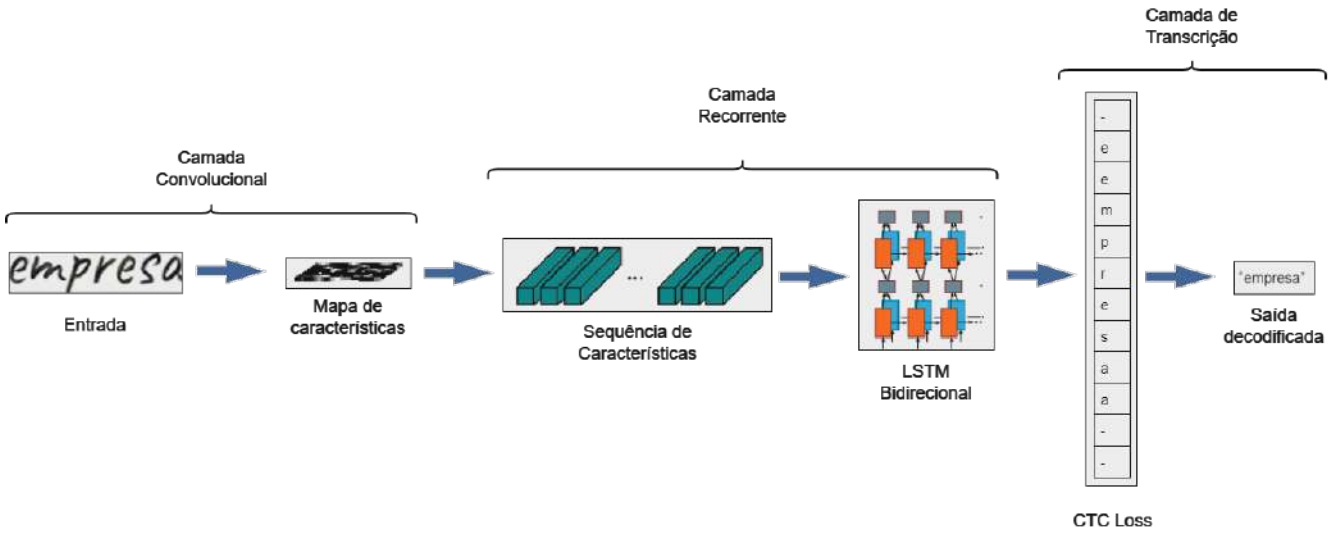


Figure 9: Arquitetura proposta para a tarefa de reconhecimento de caracteres.

Table 1: Arquitetura desenvolvida para detecção de palavras.

Layer	Description	Size of output
input_data	Word image in gray tone	(128, 32, 1)
conv2d + conv2d_1 + batch_normalization + max_pooling2d + dropout	filter: 32, kernel: 3x3, activation: relu, maxpool: 2x2, dropout: 0.25	(64, 16, 32)
conv2d_2 + conv2d_3 + batch_normalization_1 + max_pooling2d_1 + dropout_1	filter 64:, kernel: 3x3, activation: relu, maxpool: 2x2, dropout: 0.25	(32, 8, 64)
conv2d_4 + conv2d_5 + batch_normalization_2	filter: 128, kernel: 3x3, activation: relu	(32, 8, 128)
reshape	Remove dimension	(32, 1024)
bidirectional + dense + batch_normalization_3	units: 128, dropout: 0.35, dense units: 256	(32, 256)
bidirectional_1	units: 128, dropout: 0.35	394240
input_label	Word with the most characters	(none, 21)
Dense_output	Network output: word with up to 32 characters, with 92 character possibilities	(32, 92)

Assim, como observado na Figura 8, a entrada tem comprimento 10 e Y é a palavra carro. A maneira de alinhar X a Y é atribuir um caractere de saída a cada etapa de entrada e recolher as repetições.

O CTC introduz um novo token no conjunto de saídas permitidas. Esse novo token é chamado de token em branco, representado pela letra E na Figura 8. Esse token não corresponde a nada e é simplesmente removido da saída.

Os alinhamentos permitidos pelo CTC têm o mesmo comprimento da entrada. Permitimos qualquer alinhamento que mapeie para Y depois de mesclar repetições e remover os E.

Se Y tem dois caracteres iguais seguidos, então um alinhamento válido deve ter um E entre eles. Com esta regra, é possível diferenciar entre alinhamentos que tendem para "carro" e aqueles que tendem para "caro".

Usar o CTC é interessante pois não seria necessário anotar a posição exata dos caracteres nas imagens de entrada. O CTC guia o treinamento usando a matriz da saída da RNN e o texto *ground truth* (palavra real que corresponde ao texto real na imagem), tenta todos os alinhamentos possíveis do *ground truth* na imagem e obtém a soma de todas as pontuações.

Dessa forma, a pontuação de um *ground truth* é alta se a soma das pontuações de alinhamento tiver um valor alto.

4. DATASETS

Durante o desenvolvimento da pesquisa, foi analisado alguns *datasets* para treinamento do modelo proposto. Observando o trabalho de [19], foi analisado cinco *datasets* possíveis.

O autor utilizou tais conjuntos de dados para avaliar seu sistema HTR proposto. Para ter uma ideia das características dos textos manuscritos dos *datasets*, é apresentado na Figura 10 algumas amostras.

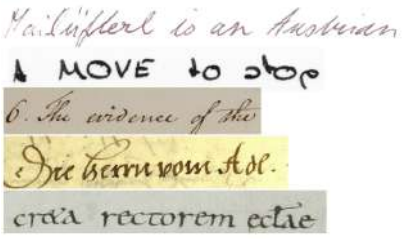


Figure 10: De cima para baixo: CVL, IAM, Bentham, Ratsprotokolle e SaintGall.

Na Tabela 2 é feito uma comparação dos *datasets* usados no trabalho de [19]. O número de caracteres está entre 48 (SaintGall) e 93 (Bentham).

Palavras únicas são calculadas dividindo um texto em palavras e depois contando apenas as únicas. O *dataset* IAM tem o maior número de palavras únicas.

Enquanto o CVL *dataset* tem o número mais baixo. Palavras fora do vocabulário (OOV) são palavras contidas no conjunto de teste, mas não no conjunto de treinamento ou validação.

O desempenho de um modelo de linguagem (LM) em nível

Table 2: Estatísticas dos datasets analisados.

Dataset	#chars	#unique words			OOV	#lines		
		train	valid.	test		train	valid.	test
CVL	54	383	275	314	1.91%	11438	652	1350
IAM	79	11242	2842	3707	37.87%	10244	1144	1965
Bentham	93	8274	2649	1911	17.63%	9198	1415	860
Ratsprotokolle	90	6752	1471	1597	32.06%	8366	1014	1138
SaintGall	48	4795	895	1293	55.45%	1052	143	215

de palavra é influenciado pela porcentagem de palavras desconhecidas. Portanto, o número de palavras OOV dá uma dica se o uso de tal LM faz sentido.

Segue abaixo algumas características dos conjuntos de dados:

- CVL [10]: este conjunto de dados é dedicado principalmente para tarefas de recuperação e localização de palavras. Contém caligrafia de 7 textos escritos por 311 escritores. Os textos estão em Inglês e Alemão. Há apenas anotações para palavras, mas não para sinais de pontuação. Os dados são anotados em nível de palavra.
- IAM [13]: contém texto em Inglês. Este *dataset* foi ampliado, onde segundo o site responsável pelo *dataset* [3] lista 1.539 formulários e 657 escritores. Os dados são anotados em nível de palavra e em nível de linha.
- Bentham [17]: contém 433 páginas. Principalmente escrito em Inglês, algumas partes são gregas e francesas. Os dados são anotados em nível de linha e parcialmente em nível de palavra.
- Ratsprotokolle [18]: textos de reuniões do conselho em Bozen de 1470 a 1805. Um número desconhecido de escritores com alta variabilidade no estilo de escrita criou este conjunto de dados. Está escrito em Alemão moderno. Os dados são anotados em nível de linha.
- SaintGall [5]: este conjunto de dados contém 60 páginas com 24 linhas por página. Foi escrito em latim por uma única pessoa no final do século IX. Descreve a vida de Saint Gall. Os dados são anotados em nível de linha.

Observa-se que todos os conjuntos de dados utilizados por [19] foram escritos em inglês e em alguns casos, alemão ou grego. Nota-se também que a maioria dos modelos de reconhecimento de texto manuscrito, de fato, focam em reconhecer textos excritos em inglês.

Porém, um modelo que consegue reconhecer minimamente uma língua, não necessariamente conseguirá ser utilizado em uma outra língua. Segundo [4], há várias características que diferenciam as escritas das pessoas. Isso envolve tanto a cultura onde a escrita está inserida como o próprio estilo do escritor.

Por exemplo, na Figura 11 é apresentado algumas variações de escrita, feita por pessoas diferentes. Observe que o estilo muda, podendo haver uma escrita contínua em uma única linha ou uma quebra, mas representando a mesma palavra.

Figure 11: Amostras de escritas diferentes feitas por escritores diferentes. Palavras retirados do IAM *dataset*.

Outras escritas, como alemão, russo, vietnamita, grego e georgiano, segundo [4] apresentam características próprias.

O chinês tem um conjunto muito maior de caracteres complexos e com escritores que tem o costume de sobrepor caracteres uns nos outros.

O coreano, sendo uma língua alfabética, agrupa letras em sílabas levando a um grande alfabeto de sílabas. A escrita em hindi geralmente contém uma linha de conexão chamada *shirorekha*, onde os caracteres podem formar estruturas maiores, que influenciam a forma escrita das palavras.

Até mesmo os emojis são símbolos *Unicode* não textuais e que também podem ser usados na escrita, dependendo do texto.

Assim, analisando tais conjuntos de dados apresentados por [19], foi observado que o IAM *dataset* seria o mais adequado para o trabalho de desenvolvimento de um modelo de reconhecimento de texto escrito em português, devido suas características de escritas mais próxima.

Para melhorar o desempenho do modelo proposto e para avaliar a estrutura de escrita do IAM *dataset*, foi proposto um conjunto de dados próprio, semelhante ao IAM, nomeado de *HCAO Dataset*[4].

O HCAO *dataset* foi gerado de forma artificial, copiando o estilo de fonte e escrita contidas no IAM *dataset*. Para a

[3] https://fki.tic.heia-fr.ch/databases/iam-handwriting-database

[4] *Handwritten Calligraphy with Accents and Overlays* - Caligrafia Manuscrita com Acentos e Sobreposições

construção do conjuto de dados, foi selecionado 108 fontes[5] cursivas únicas. Junto a isso, foi escolhido as 1432 palavras do português brasileiro mais utilizadas. Observe na Figura 12 uma amostra comparativa entre uma palavra do IAM *dataset* com o HCAO *dataset*.

Para cada palavra, foi criado variações utilizando as fontes, resultando em 151500 imagens, com as 108 fontes cursivas diferentes. O *IAM Dataset* possui 115320 imagens de palavras isoladas, com 657 fontes diferentes.

Figure 12: Amostra entre palavras do IAM *dataset* (acima) e HCAO *dataset* (abaixo).

5. EXPERIMENTOS

Foi realizado uma validação com 15 amostras de redações escritas por candidatos, analisando a qualidade da escrita e qualidade da imagem, classificando os resultados em 5 categorias: ótimo, bom, médio, ruim e péssimo. Foram analisadas 3 imagens de redações por categoria.

Para redações categorizadas como ótimas, o reconhecimento dos caracteres da palavra junto com a decodificação foram acertivas com a maioria das palavras da redação, principalmente na etapa de detecção de palavras, onde a criação das AABB foram feitas corretamente sobre cada palavra.

Para a categoria bom, o reconhecimento dos caracteres das palavras foram acertivas, havendo mais falhas na decodificação e na etapa de detecção das palavras, devido a presença de artefatos na imagem.

Para a categoria médio, tanto o reconhecimento quanto a decodificação apresentaram falhas em algumas palavras, afetadas em alguns casos pela detecção das palavras. Para a categoria ruim, a detecção das palavras foram satisfatórias, mas o reconhecimento dos caracteres foram ruins, afetando a decodificação. E para a categoria péssimo, a criação das AABB foram ruins, influenciando fortemente tanto o reconhecimento dos caracteres quanto a decodificação.

Nas Figuras 13, 14, 15, 16 e 17 é apresentado os resultados obtidos com o reconhecimento final do modelo proposto para cada uma das redações analisadas. Nelas, é possível visualizar o modelo realizando o reconhecimento sequencial das palavras contidas na imagem. Foi utilizado a arquitetura proposta, treinada com a junção dos *datasets IAM Dataset* e *HCAO Dataset*.

Cada imagem mostra o reconhecimento isolado de cada palavra e seu respectivo reconhecimento pelo modelo treinado. Inicialmente, foi feito alguns experimentos, isolando os *datasets* utilizados no treinamento, que foram *IAM Dataset* e *HCAO Dataset*.

Primeiro, utilizando apenas o *IAM Dataset*, verificou-se que o mesmo apresentou bons resultados quando aferindo suas taxas de acerto com palavras de teste do próprio *dataset*.

Ao treinar o modelo utilizando o *HCAO Dataset*, o mesmo apresentou resultados semelhantes, quando aferido com palavras do próprio *dataset*.

Porém, ao aferir o modelo treinado apenas com o *IAM Dataset* nas palavras de teste do *HCAO Dataset*, o modelo apresentou problemas consideráveis. O mesmo acontece quando é testado o modelo treinado apenas com o *HCAO Dataset* e aferindo com palavras do *IAM Dataset*.

Verificou-se então um problema de generalização do modelo, devido ao tamanho dos *datasets* em comparação com a complexidade da arquitetura proposta, que, embora simples, possui muitos parâmetros treináveis.

Outro problema verificado foi as características da escrita do português brasileiro, que consiste de letras sobrepostas e ligaduras entre os caracteres, onde tais caracteristicas não são marcantes no *IAM Dataset*, além de acentos gráficos e pontuações.

Assim, foi utilizado o *HCAO Dataset* mesclado com o *IAM Dataset*, para aumentar a generalização do modelo, pois o *HCAO Dataset* foi construído com caracteristicas que remete as escritas manuscritas, ou seja, letras sobrepostas, ligaduras entre os caracteres e fontes propriamente cursiva.

Também foram avaliados dois decodificadores: *Beam Search* e *Lexicon Search*. Esses decodificadores foram usados após a etapa de transcrição do modelo, e tem como objetivo predizer uma palavra real de um léxico[6].

O *Beam Search* é um algoritmo de busca usado para encontrar a melhor sequência de palavras em uma frase. Funciona através da expansão de um "feixe" de candidatos a frase, onde cada candidato é uma sequência parcial de palavras. O feixe é ordenado por uma pontuação que avalia a probabilidade de cada candidato ser a frase completa, mas no caso deste trabalho, ele completa palavras isoladas.

Já o *Lexicon Search* é um algoritmo de busca usado para encontrar todas as ocorrências de uma palavra ou frase contidas em um léxico. O algoritmo percorre todo o léxico e compara cada palavra com a palavra ou frase de busca. No caso deste trabalho, a palavra de busca é a palavra predita pelo modelo.

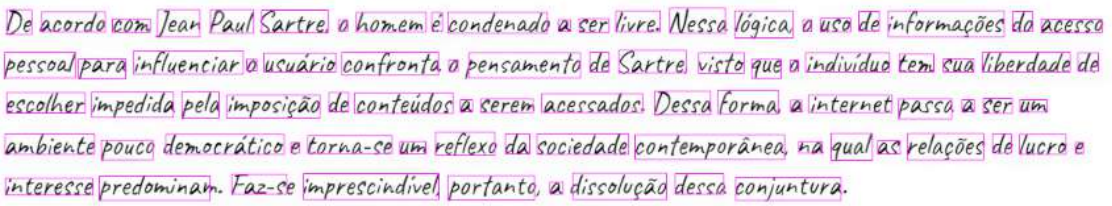
Nos experimentos, o *Lexicon Search* apresentou melhorias significativas em comparação ao primeiro, que utiliza apenas a decodificação resultante da própria arquitetura.

Porém, para que o *Lexicon Search* apresente melhores resultados, o mesmo depende de um grande léxico. Esse aumento acarreta em perda de desempenho na decodificação, pois quanto maior for a árvore do léxico, mais custoso se torna o cálculo de aproximação dos caracteres com as inúmeras palavras disponíveis, fazendo a decodificação ser demorada. No experimento final com a junção dos *datasets*, foi utilizado apenas o *Lexicon Search*, que apresentou melhores resultados.

Assim, no experimento, foi construído um dicionário com apenas 1.000 palavras mais usadas do português brasileiro, para utilizar no decodificador. Na prática, os testes com textos mais diversificados no léxico, o decodificador não teve

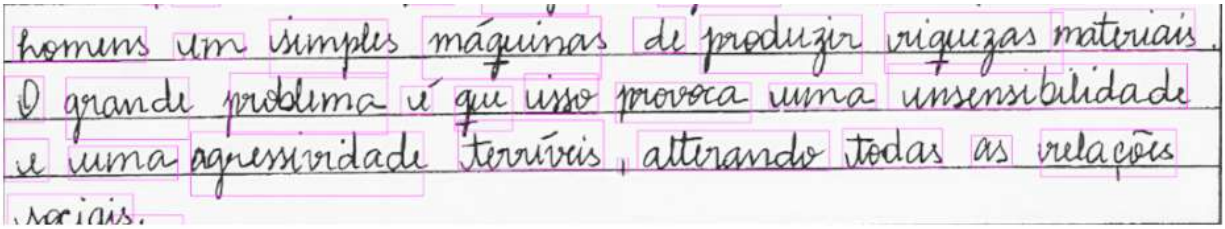
[5]https://www.dafont.com/pt/

[6]Léxico é o conjunto de palavras existente em um determinado idioma. Funciona como um dicionário, que mapeia palavras em seus significados



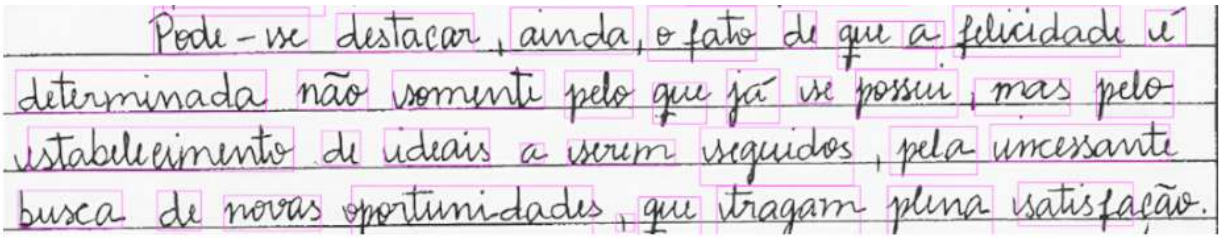
de acordo com jean paul sartre , homem e condenado a ser livre essa logica , uso de informacoes do acesso pessoal para influenciar " usuario confronta , pensamento de sartre visto que , individuo tem sua liberdade de escolher impedida pela imposicao de conteudos a serem acessados dessa forma a internet passa a ser um ambiente pouca democratico . torna-se um reflexo da sociedade contemporanea no qual as relacoes de lucro a interesse predominam faz-se imprescindivel portanto a dissolucao dessa conjuntura

Figure 13: Experimento realizado com redações reais, com resultado ótimo.



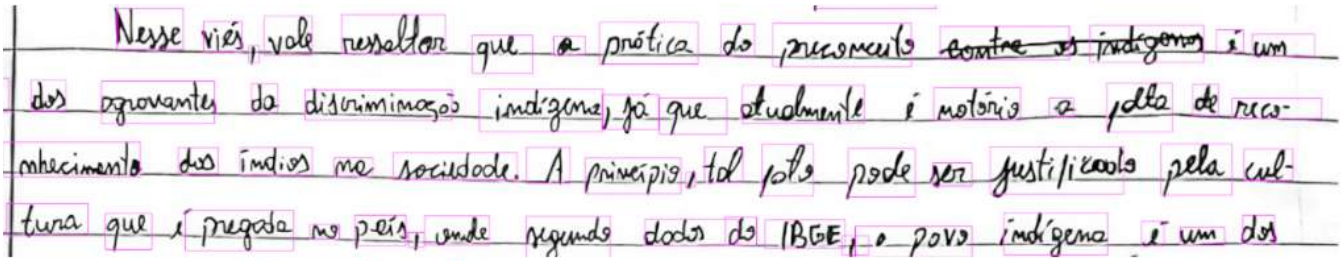
homens um simples máquina de produzir riquezas materiais
2 grande se que isso preve mas insensibilidade
se uma agressividade terriveis A interesse todas as relacoes
sociais 0 ilusoes

Figure 14: Experimento realizado com redações reais, com resultado bom.



Pode-se destacar ainda fato de que as felicidade e
determinada nao somente pela que ja uso possui as ele
estabelecimento de ideais a um seguidos ele incessante
busca de as oportunidades ! que tragam plena series

Figure 15: Experimento realizado com redações reais, com resultado médio.



esse vies , vale ressaltar que a pratica do tem , um
. dos agravantes do indigena ira que atualmente notorio a de os se
do indios no sociedade 1 . tal jobs pode ser justificado pela cabe
sua que pregada nos pais , onde segundo dados do 1888 o por indigena a um dos

Figure 16: Experimento realizado com redações reais, com resultado ruim.

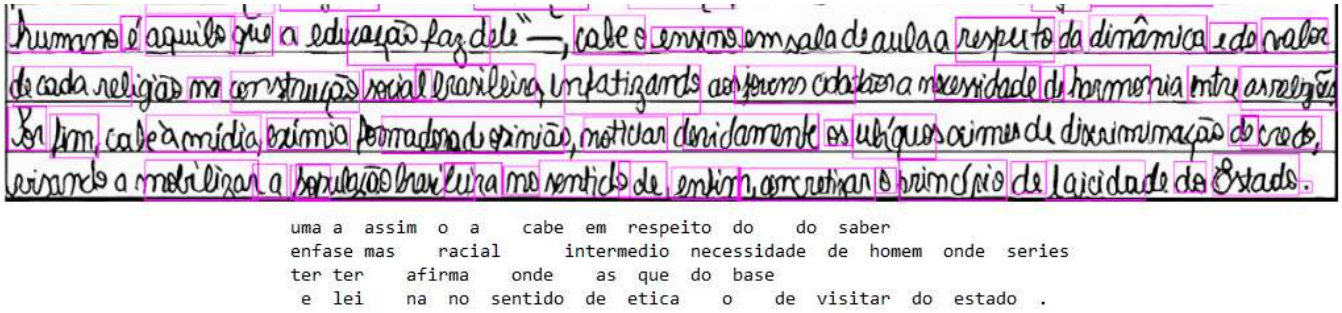


Figure 17: Experimento realizado com redações reais, com resultado péssimo.

bons resultados devido ao tamanho reduzido de 1.000 palavras.

Porém, utilizando um léxico de mais de 100.000 palavras, os resultados de decodificação melhoram, pois há mais palavras para realizar comparações de proximidade. Com um léxico acima de 10.000 palavras, a decodificação passa a ser significativamente demorada, o que impossibilita seu uso em ambiente de produção.

É observado que, mesmo para os experimentos classificados como ótimo, há várias palavras decodificadas de forma errada. Isso se dá por alguns motivos analisados, como a necessidade de aumentar o léxico de palavras do dicíonário, que é utilizado na etapa da decodificação, assim como uma melhor investigação no volume de amostras utilizadas no treino.

É possível visualizar na Figura 18, que há algumas falhas na decodificação, onde o modelo prevê uma palavra muito diferente da real. Também é possível visualizar erros na etapa de detecção de palavras, onde na Figura 18 observa-se na palavra "satisfação", o AABB não detectou toda a região da palavra.

Isso faz com que a palavra, no momento do reconhecimento dos caracteres, recorte a imagem inteira da palavra, fazendo o modelo não interpretar todos os caracteres da palavra, e, por consequência, decodificando uma palavra de forma errada.

Figure 18: Exemplo de decodificação errada.

Outra falha de difícil resolução é a densidade de letras em uma única palavra. É notado na Figura 19 uma amostra, onde a palavra "dinâmica" é decodificada como "do". É comum esse erro quando a imagem da palavra possui muitos caracteres sobrepostos, onde o modelo acaba prevendo apenas uma parte da palavra.

Porém, em alguns casos, a arquitetura tem boa eficiência. Isso ocorre principalmente quando a escrita é bem esparsa, com caracteres bem definidos e minimas ligaduras entre as letras, como pode-se ver na Figura 20.

Outro ponto observado é a diversidade das amostras nos

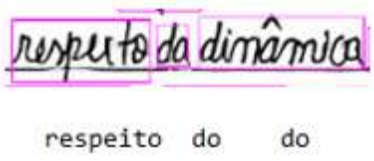


Figure 19: Exemplo de decodificação errada quando há caracteres colados.

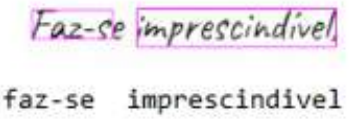


Figure 20: Exemplo de decodificação certa.

datasets utilizados. Entre as amostras utilizadas para treinamento e amostras reais escritas por candidatos, ainda há diferenças consideráveis, principalmente quando se observa as sobreposições de caracteres durante a escrita em uma redação, como no ENEM.

Outro ponto analisado é sobre os erros dos próprios candidatos ao escreverem na redação. Em alguns exemplos, o candidato escreve palavras erradas ou rasuradas. Porém, o modelo acaba decodificando a palavra em uma correta, levando então a uma piora nos resultados. Na Figura 21 é apresentado essa situação.

Figure 21: Rasura de palavra feita pelo candidato e o modelo prevê uma palavra.

Table 3: Comparativo das taxas CER e WER.

Autor	Língua	CER(%)	WER(%)
Scheidl, Harald (2018)	Inglês	4.86	10.15
Flor (2020)	Inglês	8.58	27.90
Puigcerver (2017)	Inglês	9.39	29.34
Bluche et al. (2017)	Inglês	14.30	41.17
Este trabalho (2023)	Português	9.10	27.81

Table 4: Junção do IAM Dataset com HCAO Dataset.

Autor	CER(%)	WER(%)
Este trabalho	3.74	10.42

6. RESULTADOS

A Tabela 3 mostra os resultados de um estudo comparativo com trabalhos da literatura utilizando o *IAM Dataset*, onde foi feito uma comparação das taxas CER e WER obtidas utilizando apenas esse *dataset*.

Neste trabalho realizou-se uma média de três experimentos para calcular os dados informados, com uma divisão 95/5, ou seja, 95% dos dados foram usados para treinamento e 5% foram para validação.

Não foi possível implementar os modelos da literatura utilizados no comparativo, então foi usado o mesmo *dataset* para termos de comparação e a mesma divisão dos dados treino e validação, analisando os resultados fornecidos pelos autores.

CER (*Character Error Rate* - Taxa de Erro de Caractere) é baseado no conceito da distância de *levenshtein*, onde é contado o número mínimo de operações por caracteres necessários para transformar o (*ground truth*) na palavra da saída da arquitetura. Esta taxa é calculada pela seguinte equação 3:

$$CER = (S + D + I)/N \tag{3}$$

onde S é o número de substituições, D é o número de remoções, I é o número de Inserções e N o número de caracteres do texto de referência (*ground truth*).

Já o WER (*Word Error Rate* - Taxa de Erro de Palavra) é aplicável na transcrição de parágrafos e frases de palavras com significado. Sua equação é idêntica ao CER, mudando apenas que o WER opera no nível da palavra, ou seja, substituições, remoções e inserções são feitas baseadas na palavra inteira, ao invés de caracteres por caracteres.

Adicionalmente, realizou-se os cálculos de CER e WER para o modelo treinado com a mistura do *IAM Dataset* e o *HCAO Dataset*, onde foi obtida uma melhoria moderada, pois embora haja uma visível melhoria nas taxas, em experimentos com palavras com fontes mais complexas, os resultados tendem a ser ruins devido ao enviesamento dos dados oriundos do *dataset*. Na Tabela 4 são apresentados os resultados.

Os trabalhos da literatura possuem certas variações da arquitetura, incluindo propostas diferentes na decodificação das palavras, onde são experimentados desde decodificação simples pela saída bruta da arquitetura pelo CTC até modelos de linguagem treinados. Porém, todas as arquiteturas são do tipo CRNN.

7. CONCLUSÃO

Este trabalho propôs uma metodologia para reconhecer textos manuscritos em imagens de redações, escritas por humanos. A proposta é facilitar o envio da redação para sistemas de correção automática, auxiliando o profissional em suas atividades de correção, além de incentivar o candidato a escrever a redação em papel.

Ao final, é possível a construção de *datasets* de textos para utilização em modelos de NLP para realização de correção automática de texto, pois com essa metodologia proposta, é possível a extração de textos em imagens de redações de sistemas de correção manual, já que muitos professores e escolas guardam redações de seus alunos na forma de imagens.

Esta metodologia apresentou resultados moderados para um escopo específico, porém é possível obter melhores resultados, realizando algumas mudanças na arquitetura proposta, onde ao invés de usar uma CNN para detectar características da imagem de entrada poderia ser utilizado curvas de *bézier* para extração direta das curvas da palavra e inserção na LSTM, pois tal técnica possibilita a obtenção de pontos de curva da escrita sem precisar treinar uma rede neural convolucional para extrair caracteristicas sobre as curvas, otimizando então os resultados.

Outra melhoria seria a troca da LSTM por uma arquitetura *Transformer*, para realizar o processamento dos dados de entrada utilizando o conceito de atenção, proposta por essa arquitetura.

Em teoria seria possível obter uma melhoria na saída da arquitetura desenvolvida, pois o *Transformer* calcularia a probabilidade da sequência dos caracteres da palavra baseado em contexto de palavras reais, melhorando a utilização do decodificador na saída da arquitetura.

8. REFERÊNCIAS

[1] G. Axler and L. Wolf. Toward a dataset-agnostic word segmentation method. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2635–2639. IEEE, 2018.

[2] S. C. B. d. Barros. Estudo do desempenho de candidatos à ufrn na prova de redação do enem no período de 2013 a 2016. Master's thesis, Brasil, 2019.

[3] R. M. Bozinovic and S. N. Srihari. Off-line cursive script word recognition. *IEEE Transactions on pattern analysis and machine intelligence*, 11(1):68–83, 1989.

[4] V. Carbune, P. Gonnet, T. Deselaers, H. A. Rowley, A. Daryin, M. Calvo, L.-L. Wang, D. Keysers, S. Feuz, and P. Gervais. Fast multi-language lstm-based online handwriting recognition. *International Journal on Document Analysis and Recognition (IJDAR)*, 23(2):89–102, 2020.

[5] A. Fischer, V. Frinken, A. Fornés, and H. Bunke. Transcription alignment of latin manuscripts using hidden markov models. In *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, pages 29–36, 2011.

[6] A. Gupta, M. Srivastava, and C. Mahanta. Offline handwritten character recognition using neural network. In *2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pages 102–107. IEEE, 2011.

[7] E. INEP. Enem 2023. *https://abre.ai/iOT6*, 1:1, 2021.

[8] E. INEP. Microdados enem 2023. *https://abre.ai/iOT3*, 1:1, 2021.

[9] E. INEP. Painéis enem. *https://abre.ai/hAJI*, 1:1, 2021.

[10] F. Kleber, S. Fiel, M. Diem, and R. Sablatnig. Cvl-database: An off-line database for writer retrieval, writer identification and word spotting. In *2013 12th international conference on document analysis and recognition*, pages 560–564. IEEE, 2013.

[11] J. C. Marinho, R. T. Anchiêta, and R. S. Moura. Essay-br: a brazilian corpus of essays. *arXiv preprint arXiv:2105.09081*, 2021.

[12] J. C. Marinho, F. Cordeiro, R. T. Anchiêta, and R. S. Moura. Automated essay scoring: An approach based on enem competencies. In *Anais do XIX Encontro Nacional de Inteligência Artificial e Computacional*, pages 49–60. SBC, 2022.

[13] U.-V. Marti and H. Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5:39–46, 2002.

[14] R. Parthiban, R. Ezhilarasi, and D. Saravanan. Optical character recognition for english handwritten text using recurrent neural network. In *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, pages 1–5. IEEE, 2020.

[15] PROPOR'24. Propor'24 competition on automatic essay scoring of portuguese narrative essays. *https://abre.ai/iO5L*, 1:1, 2021.

[16] F. Sampaio, R. Moura, and K. Aires. Uso de reconhecimento Óptico de caracteres para extração de textos em imagens de redações. In *Anais do XVI Encontro Unificado de Computação do Piauí*, pages 57–64, Porto Alegre, RS, Brasil, 2023. SBC.

[17] J. A. Sánchez, V. Romero, A. H. Toselli, and E. Vidal. Icfhr2014 competition on handwritten text recognition on transcriptorium datasets (htrts). In *2014 14th International Conference on Frontiers in Handwriting Recognition*, pages 785–790. IEEE, 2014.

[18] J. A. Sanchez, V. Romero, A. H. Toselli, and E. Vidal. Icfhr2016 competition on handwritten text recognition on the read dataset. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 630–635. IEEE, 2016.

[19] H. Scheidl, S. Fiel, and R. Sablatnig. Word beam search: A connectionist temporal classification decoding algorithm. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 253–258. IEEE, 2018.

[20] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. East: an efficient and accurate scene text detector. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5551–5560, 2017.