UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Modelo de Correção de Caracteres Reconhecidos Através de Visão Computacional

André Mario dos Reis dos Santos

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

André Mario dos Reis dos Santos

Modelo de Correção de Caracteres Reconhecidos Através de Visão Computacional

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Ricardo Rodrigues Ciferri

Versão original

São Carlos

2024

Santos, André
S856m Modelo de Correção de Caracteres Reconhecidos Através de Visão Computacional / André Mario dos Reis dos Santos ; orientador Ricardo Rodrigues Ciferri. – São Carlos, 2024.
50 p. : il. (algumas color.) ; 30 cm.

Monografia (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. OCR. 2. Correção de Erros de OCR. 3. Correção Pós-OCR. 4. Neural Machine Translation. 5. NMT I. Ciferri, Ricardo Rodrigues, orient.

Este trabalho é dedicado a todas as pessoas que fazem do trabalho social um pilar para a construção de uma sociedade mais informada, consciente e unida.

"Todos os modelos estão errados,
mas alguns são úteis."
George Box

RESUMO

SANTOS, A. **Modelo de Correção de Caracteres Reconhecidos Através de Visão Computacional**. 2024. 50 p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

A informatização de dados é um processo essencial para armazenar registros mantidos em papel e permite que o conteúdo destes documentos possa ser manipulado para extrair informações através da computação. Para extrair textos de imagens, existem diversas alternativas para reconhecer os caracteres que compõem a escrita, porém, por mais avançados que estejam estes programas de reconhecimento ótico de caracteres (OCR), falhas podem ocorrer na obtenção das imagens antes do reconhecimento causando a troca ou exclusão de caracteres. Uma proposta para corrigir erros de reconhecimento é apresentada neste trabalho como etapa pós-OCR capaz de detectar e corrigir palavras com erros através de métodos de análise linguística, de tradução estatística (SMT) e tradução com aprendizado de máquina (NMT). A base de dados ICDAR 2019 foi utilizada como amostras de treinamento e avaliação dos modelos. Como resultado da pesquisa este trabalho compila diversas medidas de desempenho para os modelos com destaque para a precisão F1 geral para detecção de erros obtendo a marca de 60,7% e proporção de palavras corrigidas obtendo a margem de 15,71% para o conjunto de avaliação.

Palavras-chave: OCR. Correção de Erros de OCR. Correção Pós-OCR. Neural Machine Translation. NMT.

ABSTRACT

SANTOS, A. **Correction Model For Character Recognition Through Computer Vision**. 2024. 50 p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

The digitization of data is an essential process for storing records kept on paper and allows the content of these documents to be manipulated to extract information through computing. To extract text from images, there are various alternatives for recognizing the characters that make up the writing. However, no matter how advanced these optical character recognition (OCR) programs are, errors can occur in the image acquisition process before recognition, leading to the replacement or omission of characters. A proposal to correct recognition errors is presented in this work as a post-OCR step capable of detecting and correcting misspelled words through linguistic analysis methods, statistical machine translation (SMT), and machine translation with neural networks (NMT). The ICDAR 2019 dataset was used as training samples and for model evaluation. As a result of the research, this work compiles various performance measures for the models, highlighting an overall F1 accuracy for error detection of 60.7% and a word correction rate of 15.71% for the evaluation set.

Keywords: OCR. OCR Error Correction. Post-OCR Correction. Neural Machine Translation. NMT.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

OCR	Optical Character Recognition
AM	Aprendizado de Máquina
RNA	Redes Neurais Artificiais
RNC	Redes Neurais Convolucionais
PLN	Processamento de Linguagem Natural
SMT	Statistical Machine Translation
TA	Tradução Automática
NMT	Neural Machine Translation
CER	Character Error Ratio
WER	Word Error Ratio
MR	Mudança Relativa
RelImp	Relative Improvement
GT	Ground Truth

SUMÁRIO

1 INTRODUÇÃO

Neste capítulo serão apresentadas as motivações e o contexto de aplicação de uma camada de reconstrução de palavras oriundas de um processo de reconhecimento óptico de caracteres. Através do aprendizado de máquina, uma rede neural utilizando processamento de linguagem natural pode reconstituir caracteres que não foram corretamente reconhecidos. Na seção 1.1 será detalhado o estado da arte da visão computacional aplicada a reconhecimento de escrita e como ela vêm sendo aplicada. Na seção 1.2 serão apresentadas as motivações deste trabalho. Na seção 1.3 serão detalhados os objetivos do trabalho. Por fim, na seção 1.4 será resumida a estrutura dos capítulos deste trabalho.

1.1 Contextualização

Apesar da maior integração entre sistemas para circular informações, muitas áreas continuam utilizando papéis impressos como documentos oficiais ou comprovatórios. Para facilitar o armazenamento e tratamento destes documentos, seus controladores normalmente recorrem à digitalização destes papéis, entretanto, para agregar mais valor a estes documentos armazenados apenas como registro, pode ser possível extrair dados importantes contidos neles. A principal técnica utilizada para extrair texto de imagens é chamada de Optical Character Recognition (OCR), que são modelos de reconhecimento de visão computacional capazes de reconhecer caracteres. Esta tecnologia avançou muito e diversos modelos estão disponíveis para uso, porém, nem os mais avançados modelos são capazes de lidar com desafios do trabalho de campo vinculado à informatização de textos impressos (Thanki; Davda; Swaminarayan, 2021). Quando muitos documentos são escaneados em bateladas, é comum que o resultado apresente falhas: a orientação da folha pode acabar desalinhada, o papel pode ter uma dobra sobre os textos, a resolução do escaneamento pode ter sido baixa. Entre tantas possibilidades de falhas, torna-se necessária uma maneira de tratar a digitalização de documentos defeituosos.

1.2 Motivação

As ferramentas de OCR vêm auxiliando tarefas de digitalização de documentos cada vez melhor nos últimos anos, entretanto, muitos dos principais softwares de reconhecimento óptico ainda não são capazes de lidar com problemas corriqueiros associados à tarefas como: má qualidade da digitalização, baixa resolução da digitalização, papel danificado, má qualidade de impressão do material. Para garantir que a digitalização venha a fornecer dados úteis, é preciso adicionar um tratamento para o texto gerado através do OCR para ser devidamente armazenado ou seja processado por ferramentas para extração de informações textuais.

1.3 Objetivo

Este trabalho tem como objetivo desenvolver um modelo de I.A. capaz de reconhecer e corrigir palavras com caracteres errados de documentos impressos que passaram por processos de OCR. Será verificado por fim a aplicabilidade deste modelo como um microsserviço.

1.4 Organização da Monografia

Esta monografia está organizada da seguinte forma: No Capítulo 2 será apresentada a fundamentação teórica dos processos e tecnologias utilizadas. No Capítulo 4 será discutida a metodologia aplicada. O Capítulo 5 traz os resultados obtidos e uma análise sobre os mesmos. Finalmente o Capítulo 6 apresenta as conclusões sobre o trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão aprofundados os principais conceitos associados ao estudo da correção de caracteres utilizando inteligência artificial. Na Seção 2.1 será apresentado o funcionamento do reconhecimento de caracteres. Na Seção 2.2 são abordados os princípios do aprendizado de máquina. A Seção 2.4 explica o que é processamento de linguagem natural e suas ferramentas utilizadas neste trabalho. A Seção 2.5 traz o contexto dos métodos de correção palavras. Na Seção 2.6, as métricas de avaliação são descritas.

2.1 Reconhecimento Óptico de Caracteres

O Reconhecimento Óptico de Caracteres, em inglês, Optical Character Recognition (OCR) é o processo de classificação de padrões visuais contidos em uma imagem digital (Chaudhuri *et al.*, 2017). O processo consiste na segmentação e classificação de trechos de imagens que contém texto escrito com o objetivo de transpor a informação classificada para informação textual capaz de ser armazenada e interpretada em computadores (Cabral, 2021).

Esta área de pesquisa vem aumentando seu uso devido aos bons resultados que as técnicas de OCR apresentaram nos últimos anos. Thanki, Davda e Swaminarayan (2021) mencionam que existe uma demanda crescente para a digitalização de informações sensíveis, escanear e armazenar estes dados como imagens seria custoso, portanto, para resolver este problema, processar estes documentos e armazená-los como informação textual será mais simples e rápido e possibilitaria também a utilização dos dados extraídos para análises futuras.

Pessoas são capazes de reconhecer caracteres com 100% de certeza quando são bem escritos, entretanto não existe ainda algum OCR capaz de se aproximar de tal precisão de reconhecimento (Mori; Nishida; Yamada, 1999). Para Chaudhuri *et al.* (2017), caracteres escritos à mão e impressos podem ser reconhecidos, mas a precisão será diretamente dependente da qualidade dos documentos utilizados e que quanto mais uniforme for a entrada, melhor será o desempenho do sistema de OCR.

No diagrama da Figura 1 é apresentada uma divisão de diferentes escopos para OCR. Com base na divisão proposta por Chaudhuri *et al.* (2017), este trabalho abordará apenas o reconhecimento de caracteres alfanuméricos individuais do alfabeto romano em letras impressas.

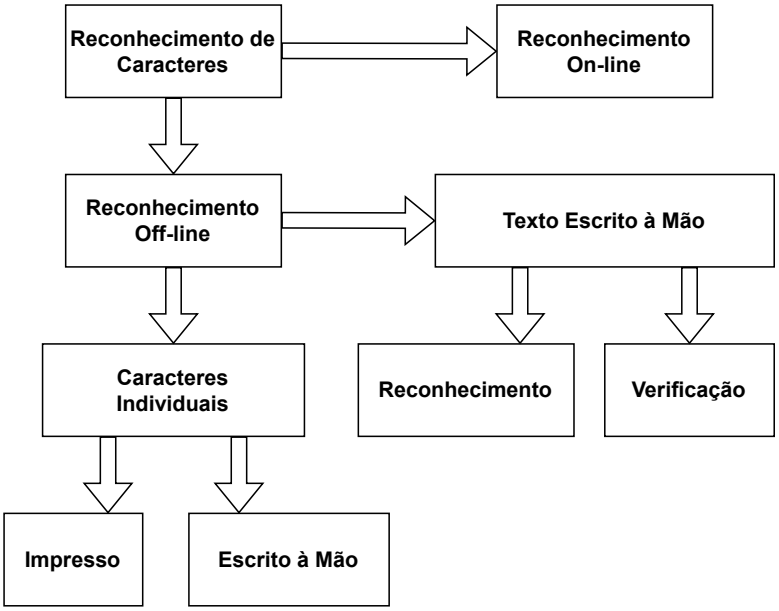

Figura 1 – As Diferentes Áreas do Reconhecimento de Caracteres.

Fonte: (Chaudhuri *et al.*, 2017), tradução livre.

2.2 Aprendizado de Máquina

Aprendizado de Máquina (AM) é um conjunto de técnicas que tem como seus objetivos fazerem com que computadores sejam capazes de aprender sem necessariamente serem programados (Samuel, 1959). O aprendizado, de modo geral, é o resultado de diversas iterações sobre um modelo estatístico probabilista com o objetivo de otimizar um critério de desempenho (Ayodele, 2010), a otimização se dá convergindo os pesos ou coeficientes do modelo para fornecer respostas contidas dentro do escopo dos dados de utilizados como exemplo. (Grebovic *et al.*, 2023).

2.2.1 Redes Neurais Artificiais

Redes neurais artificiais (RNA) compreendem uma gama de topologias de programas baseados no AM envolvendo neurônios (Ayodele, 2010). Os neurônios são uma regra de aprendizado desenvolvida por Rosenblatt (1958). A proposta do algoritmo, chamado de perceptron, era imitar um modelo de ativação das células neuronais, desta forma, cada neurônio artificial pode receber diversas entradas ponderadas que após serem somadas, tem seu resultado inserido em uma função de ativação que determina se o neurônio é ativado ou não. Tomando como exemplo a aplicação inicial do perceptron, o neurônio tinha o objetivo classificar dois grupos de elementos linearmente separáveis.

Com o aprimoramento da capacidade de processamento dos computadores os resultados obtidos por Rosenblatt puderam ser multiplicados em redes de neurônios capazes de receber e produzir sinais para múltiplas camadas de neurônios interligados.

Esta topologia ganhou notoriedade ao obter sucesso em encontrar soluções para problemas com nível de maestria próximos aos de soluções estatísticas para o reconhecimento de padrões (Ayodele, 2010).

2.3 Transformer

Transformer é uma arquitetura de AM desenvolvida para processamento de textos com o objetivo de reduzir custos computacionais através de uma função capaz de calcular o contexto de palavras de uma sentença (Vaswani *et al.*, 2017). Os principais componentes da arquitetura são codificadores posicionais, camadas Múltiplas de Atenção e camadas de realimentação conectadas através de subcamadas de normalização dispostas de modo a gerar uma entrada de dados para um estimador Softmax para o cálculo das probabilidades de relacionamentos para entre tokens codificados (Vaswani *et al.*, 2017) em forma tabular. Esta estrutura pode ser observada na Figura 2.

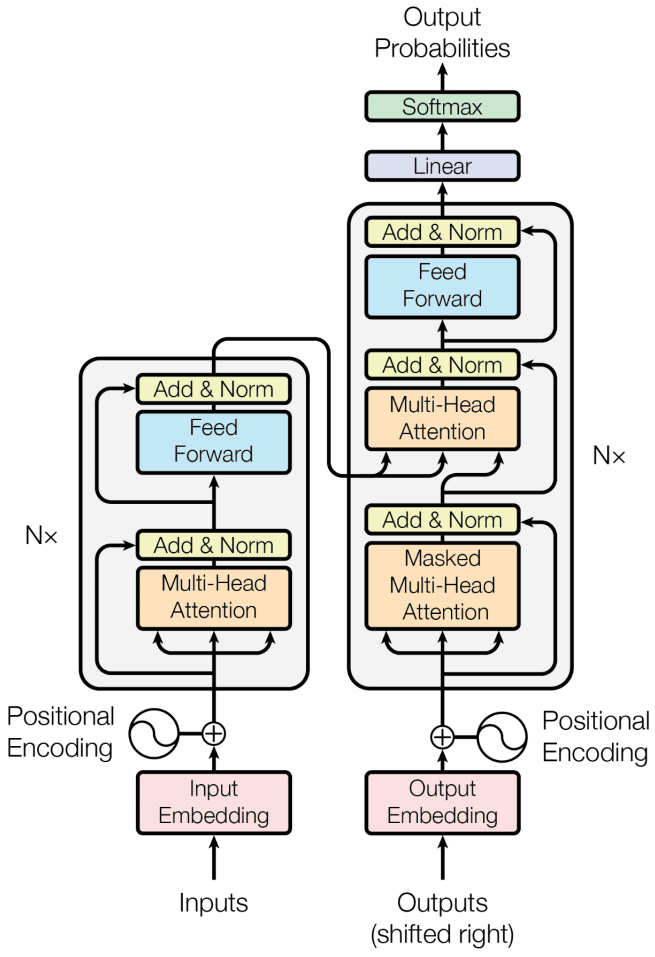

Figura 2 – Arquitetura do Modelo Transformer.

Fonte: (Vaswani *et al.*, 2017).

Este modelo se tornou muito importante no campo de estudo de aprendizado de

máquina voltado à linguagem pois, segundo Merritt (2022), 70% dos artigos do site arXiv sobre IA mencionam o modelo Transformer e por ser uma peça fundamental do modelo BERT, que é um modelos estado da arte de IA voltado à interpretação de textos.

2.4 Processamento de Linguagem Natural

A forma mais comum de processar informações em computadores é através de dados estruturados, que por sua vez, são dados que possuem um formato ou contexto predefinido. Por possuírem uma estrutura fixa, transformar ou extrair informações destes dados é uma tarefa mais previsível quando comparados aos dados não estruturados (Oracle, 2024). O processamento de linguagem natural (PLN) é uma área da computação que estuda métodos para analisar, modelar e entender a linguagem humana (Vajjala *et al.*, 2020). Dentre os diversos métodos e índices de manipulação textual as técnicas utilizadas neste estudo são apresentadas nas seções seguintes.

2.4.1 Segmentação de Sentenças

De modo geral, sentenças de um texto podem ser separadas na presença de pontuações como pontos finais, interrogações ou exclamações, entretanto, pontos presentes em abreviações e endereços ou reticências por exemplo, podem corromper o contexto da sentenças com uma divisão errônea. As principais ferramentas de PLN contam com métodos de segmentação baseadas na detecção de exceções à separação baseada em pontuações (Vajjala *et al.*, 2020).

2.4.2 Análise Léxica

Técnica também conhecida pelo termo em inglês tokenization, é a etapa de atomização do texto em tokens, ou seja, as palavras ou caracteres são divididas com base na separação por espaços e outras regras definidas por cada ferramenta de análise léxica (Vajjala *et al.*, 2020). A biblioteca spaCy por exemplo, segmenta uma sentença como visto na Figura 3, no primeiro passo os tokens são divididos entre os espaços entre as palavras, em seguida, os prefixos são separados depois os sufixos são separados.

2.4.3 Similaridade de Jaccard

A Similaridade de Jaccard é uma medida de proximidade de conjuntos caracterizada pela razão da intersecção pela união dos conjuntos (Chung *et al.*, 2018), como pode ser visto na Equação 2.1. Esta medida é uma forma simples de medir a similaridade entre caracteres de uma palavra.

$$J = \frac{|A \cap B|}{|A \cup B|} \tag{2.1}$$

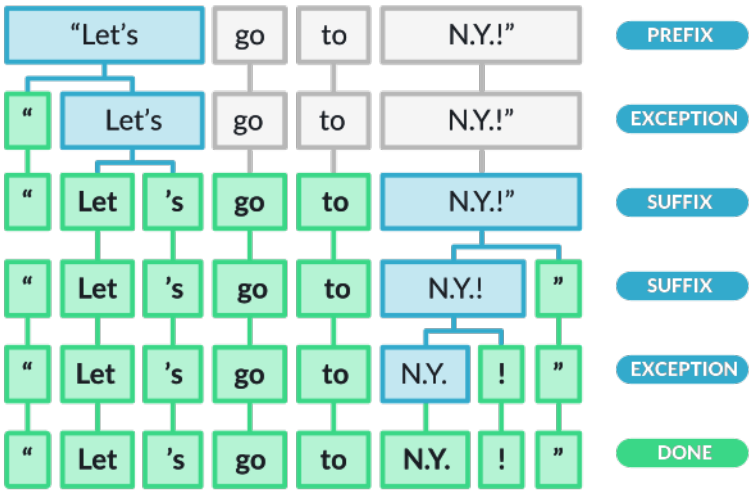

Figura 3 – Separação de tokens através do spaCy nativo.

Fonte: (spaCy, 2024).

2.4.4 Distância de Levenshtein

A distância de Levenshtein é uma forma de medir a diferença entre dois conjuntos através das substituições, adições ou exclusões de itens de um conjunto com o objetivo de se tornar idêntico ao outro conjunto comparado. Para cada operação realizada, o custo para a transformação do conjunto aumenta em uma unidade (Grashchenko, 2024). A definição da distância pode ser observada na Equação 2.2, é possível observar também que a equação é recursiva por definição, onde i e j são a quantidade de itens dos conjuntos A e B respectivamente, desta forma, a distância para dada quantidade de itens é calcula pela menor distância apresentada através da remoção de um dos itens de A, B ou de ambos acrescido de 1. Caso a remoção faça um conjunto não ter itens a quantidade do conjunto que possui mais items é utilizada.

$$lev_{A,B}(i,j) = \begin{cases} max(i,j) & se\ min(i,j) = 0 \\ min \begin{cases} lev_{A,B}(i-1,j) + 1 \\ lev_{A,B}(i,j-1) + 1 \\ lev_{A,B}(i-1,j-1) + 1_{(a \neq b)} \end{cases} & caso\ contrário \end{cases} \tag{2.2}$$

2.5 Correção de Erros

Os métodos de correções textuais podem se dividir em geral em duas categorias: estatísticos e de aprendizado de máquina (Singh; Singh, 2018). Modelos estatísticos são baseados na ocorrência e semelhança entre palavras comparadas a um dicionário conhecido. Modelos baseados em AM se aproximam do modelo estatístico com a diferença de não necessitar de um dicionário definido, as regras de correção são determinadas como resultado

do processo de aprendizado (Amrhein; Clematide, 2018). Um problema citado por (Vajjala *et al.*, 2020) é a correção para sistemas específicos como textos obtidos através de OCR, caracteres podem ser reconhecidos como outros visualmente semelhantes. Através do AM seria possível estabelecer regras de correção de caracteres mais adequadas aos erros oriundos do processo de reconhecimento visual (Amrhein; Clematide, 2018).

2.5.1 Tradução Automática Neural

Desde a antiguidade, estudiosos buscaram formas mais simples de traduzir línguas. Em meados do século *IX* Al-Kindi relacionou a ocorrência de vogais e entre a escrita do árabe e do latim e as combinações entre vogais e as letras adjacentes para criar um sistema de predição de possíveis traduções a partir de algumas palavras conhecidas (DuPont, 2022). O desenvolvimento do estudos de estatística permitiram a criação de um novo ramo no campo da tradução automática (TA), através de análises de frequência de letras, assim como Al-Kindi, e utilizando n-gramas, Booth e Weaver criaram o que seria um dos primeiros métodos de tradução automática estatística, ou em inglês, statistical machine translation (SMT) (DuPont, 2022).

A Tradução Automática Neural utiliza ferramentas de AM para determinar as regras de tradução de um modelo através de exemplos. Devido ao aumento das aplicações de AM para tarefas de PLN, a área da TA observou bons resultados com a divulgação dos primeiros trabalhos utilizando redes neurais e atualmente os modelos baseados em AM começam a ser considerados mais avançados e com melhor precisão do que os métodos estatísticos de tradução mais difundidos (Amrhein; Clematide, 2018). No decorrer deste trabalho serão utilizados o nome e a sigla do processo em inglês: neural machine translation e NMT, por serem mais difundidos.

2.6 Métricas de Avaliação

Para aferir a precisão de acertos e quantificar a melhoria dos tratamentos processados, as métricas utilizadas neste estudo são apresentadas nas seguintes seções.

2.6.1 WER

Uma das formas de medir a assertividade de sistemas de reconhecimento de texto ou fala é a razão de palavras erradas ou word error ratio (WER), os valores obtidos para essa medida podem ser obtidos através da Equação 2.3 que define a medida como a razão das palavra com erros sobre a quantidade total de palavras.

$$WER = \frac{Quantidade\ de\ Palavras\ com\ Erros}{Total\ de\ Palavras} \tag{2.3}$$

2.6.2 CER

Similar à medida WER, a medida da razão de caracteres errados ou character error rate (CER) é utilizada para aferir a quantidade de caracteres errôneos em um texto. A CER é comumente utilizada como medida de qualidade de sistemas de OCR. A Fórmula 2.4 descreve como obter a razão. Na fórmula, S representa a quantidade de caracteres substituídos, I a quantidade de caracteres inseridos ao texto, E a quantidade de caracteres excluídos e T o total de caracteres no texto.

$$CER = \frac{S + I + E}{T} \tag{2.4}$$

2.6.3 Mudança Relativa

A mudança relativa (MR) é uma forma de medir o impacto de uma mudança em um conjunto em relação ao seu estado original. A razão de mudança pode ser obtida através da equação da Fórmula 2.5 onde V' é o conjunto alterado e V o conjunto original (Nguyen *et al.*, 2020a).

$$MR = \frac{V - V'}{V} \tag{2.5}$$

Mais adequada para a comparação entre palavras é a aplicação da MR utilizando a distância de Levenshtein normalizada que pode ser vista na Equação 2.6. Para distinguir a MR da melhora relativa entre distâncias de Levenshtein, neste trabalho será utilizado a nomenclatura em inglês para se referir à melhora relativa, ou em inglês, relative improvement (RelImp).

$$RelImp = \frac{LVN_{original} - LVN_{corrigido}}{LVN_{original}} \tag{2.6}$$

Onde LVN é a distância de Levenshtein normalizada, vide Equação 2.7.

$$LVN = \frac{\sum_{n=1}^{n} p_i * lev(w_i, GTtoken_i)}{N} \tag{2.7}$$

Na equação, p_i é a probabilidade do candidato a correção w_i ser corrigido para o token $GTtoken_i$, n é o número de tokens errados em consideração e N o total de caracteres considerados (Nguyen *et al.*, 2020a).

2.6.4 F1-score

Sistemas de predição e classificação utilizam amplamente a medida de F1, a medida permite avaliar o desempenho de acertos do sistema, entretanto, a medida também considera a ocorrência de casos falsos positivos e falsos negativos permitindo avaliar desvios

de no funcionamento esperado de determinado modelo. o F1-score pode ser obtido através da Fórmula 2.8, onde VP é conjunto de valores verdadeiros-positivos, FN é o conjunto de valores falsos-negativos e FP o conjunto de valores falsos-positivos (Nguyen *et al.*, 2020a).

$$F1 = \frac{2VP}{2VP + FN + FP} \tag{2.8}$$

3 TRABALHOS RELACIONADOS

Neste capítulo serão apresentadas a metodologia de busca de trabalhos relacionados ao escopo do problema levantado por essa monografia e as principais referências obtidas para fundamentar a metodologia a ser aplicada.

3.1 Metodologias de Busca

A busca por trabalhos relacionados foi realizada através da pesquisa em quatro plataformas de publicações científicas: IEEE Xplore, Springer Link, ACM Digital Library e DBLP. Todas as pesquisas foram criadas na língua inglesa pois as plataformas hospedam trabalhos apenas neste idioma. As palavras-chave utilizadas foram: OCR, post, correction e NMT, como as plataformas possuem sintaxes distintas para a utilização de suas buscas avançadas, as frases de pesquisa foram montadas de modo a procurar por conteúdos que contivessem diversas variações de forma de escrita das palavras-chave citadas exceto o termo OCR. As frases de busca criadas para cada plataforma são as seguintes:

- **IEEE Xplore**

  ("Document Title":"OCR") AND ("All Metadata":"correction"OR "All Metadata":"post-correction"OR ("All Metadata":"post"AND "All Metadata":"correction") OR "All Metadata":"post-ocr"OR ("All Metadata":"post"AND "All Metadata":"ocr")) AND ("All Metadata":"NMT"OR ("All Metadata":"neural"AND "All Metadata":"machine"AND "All Metadata":"translation") OR "All Metadata":"neural-machine-translation")

- **Springer link**

  (OCR AND (error OR error-correction OR (error and correction) OR post-ocr OR (and and ocr) AND (NMT OR neural-machine-translation or (neural and machine and translation)))

- **ACM Digital Library**

  OCR AND (error AND correction) OR error-correction OR post-ocr OR (post AND ocr)

- **DBLP**

  OCR error NMT

O resultado das buscas pode ser observado na Tabela 1, foi possível observar poucos resultados exceto para a busca em Springer Link.

Tabela 1 – Resultados da Pesquisa por Trabalhos Relacionados - Fonte: Autoria Própria.

Base de dados	Quantidade de Resultados	Estudos Selecionados
IEEE Xplore	4	1
Springer Link	1179	1
ACM Digital Library	17	1
DBLP	1	1

3.1.1 Amrhein e Clematide (2018)

O trabalho de Amrhein e Clematide (2018) parte do questionamento sobre o desempenho de modelos de tradução automática a nível de caracteres, em específico, se modelos baseados em SMT tem desempenho superior aos modelos baseados em NMT para detecção e correção de erros.

Os pesquisadores utilizaram a base de dados ICDAR2017 (Chiron *et al.*, 2017). O corpus é formado por termos de origens e períodos variados igualmente distribuídos nas línguas inglesa e francesa. Os dados da base são apenas textuais divididos entre o resultado do processo de OCR e o texto esperado, ou no termo utilizado, *ground-tuth* (GT).

Dos principais pontos abordados na metodologia da publicação se destacam: a adoção de modelos de 10-gramas para modelos de SMT, o descarte do uso de redes neurais convolucionais (RNC) para modelos de NMT, uso de uma estrutura de treinamento chamada Nematus devido à possibilidade de ajustar parâmetros, utilização de tokens com e sem erros de OCR na proporção de 75% de tokens com erros e a proposição de um sistema combinado de SMT e NMT para detectar e corrigir erros. Um resumo do sistema proposto pode ser visto na Figura 4.

O resultado do experimento mostrou que, para a base de dados utilizada, o melhor modelo de SMT apresentou melhor desempenho na correção de erros, enquanto o melhor modelo de NMT teve melhor desempenho na detecção de erros. Segundo os autores, eles não foram capazes de encontrar os melhores parâmetros de treinamento e ajuste dos hiperparâmetros do Nematus, portanto existe a possibilidade de aprimorar o processo.

3.1.2 Nguyen *et al.* (2020b)

Nguyen *et al.* (2020b) estudam o uso de BERT, uma variação pré treinada do modelo transformer, para a detecção e correção de textos. A estrutura do modelo proposta consiste em duas etapas: a primeira é um modelo de reconhecimento de entidades adaptado para classificar tokens como válidos e inválidos. Os token são obtidos pelo processo de tokenização WordPiece. A segunda etapa é um modelo baseado na estrutura OpenNMT com ajustes nos tamanhos das camadas ocultas.

Assim como Amrhein e Clematide (2018), os autores utilizaram a fatoração de caracteres para aumentar os dados de treinamentos evidenciando os tipos de fonte de

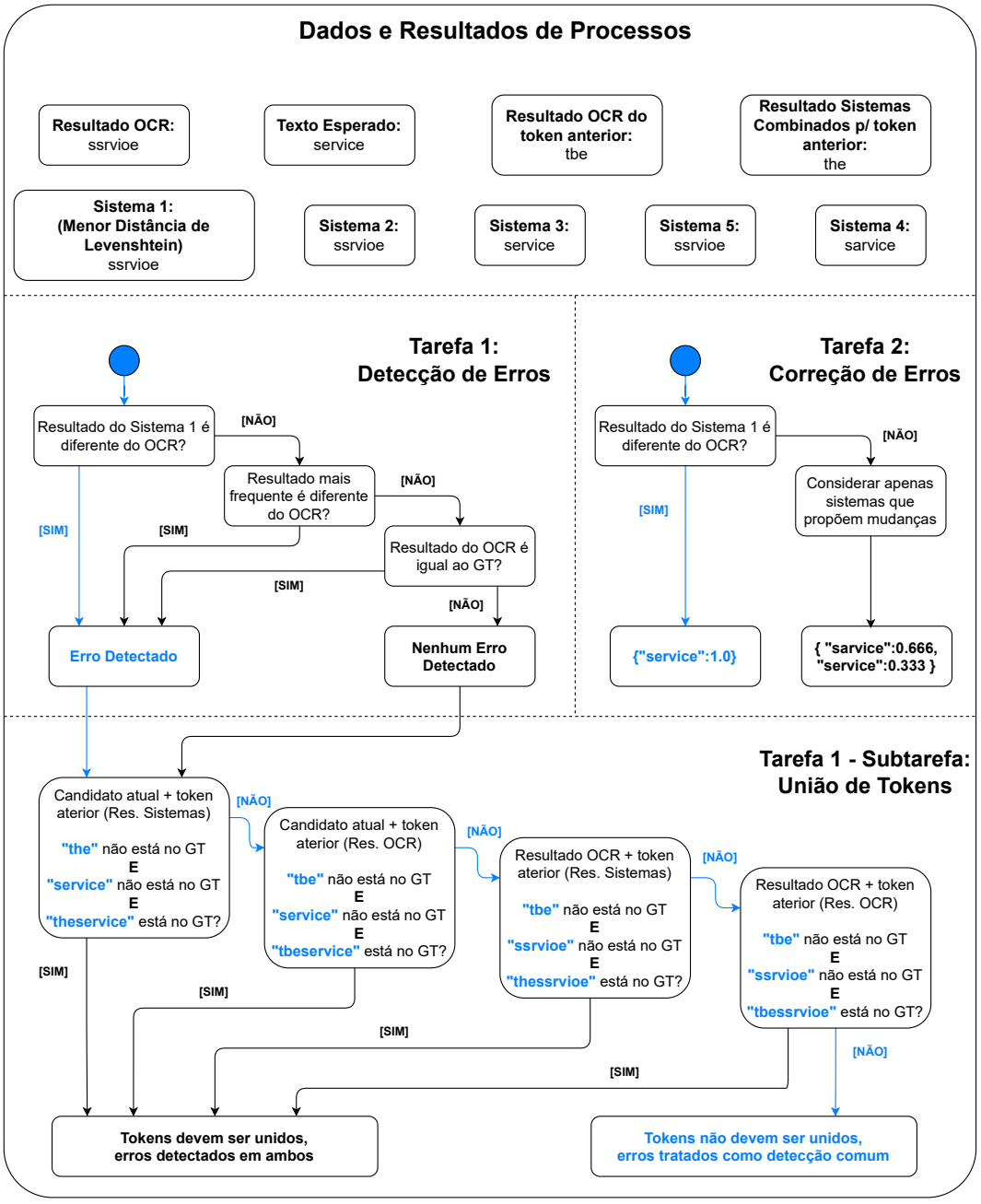

Figura 4 – Diagrama de Sistema de Detecção e Correção.

Fonte: (Amrhein; Clematide, 2018), tradução livre.

dados dos corpus ICDAR2017 que são divididos entre periódicos e monografias e da base ICDAR2019 (Chiron *et al.*, 2017) (Rigaud *et al.*, 2019).

Os resultados do trabalho foram comparados ao desempenho de outros trabalhos sobre as mesmas bases de dados. Para a detecção de erro a métrica comparada foi a pontuação F1. O modelo de Nguyen *et al.* (2020b) teve um desempenho maior que os demais trabalhos para as bases ICDAR2019 e ICDAR 2017 para periódicos e obtendo

desempenho acima da média para monografias. Os resultados da correção de erros foram medidos pela RelImp. O modelo de Nguyen *et al.* (2020b) obteve resultados acima da média quando comparados com os demais trabalhos.

3.1.3 Nguyen *et al.* (2020a)

Diferente dos demais trabalhos relacionados, Nguyen *et al.* (2020a) estudam o uso de algoritmos evolutivos como modelos de detecção e correção de erros através do self-organizing migrating algorithm (SOMA).

O processo de correção de erros é realizado do forma iterativa de modo que um token que é detectado contendo erros é analisado por uma janela deslizante de um ou dois caracteres em busca de padrões de correção que substituirão caracteres do token. Para cada padrão encontrado e substituído um candidato a substituição é criado de modo que a o melhor candidato é avaliado através de uma pontuação de uma soma ponderada que considera a similaridade, frequência de unigramas, bigramas, trigramas e a probabilidade de substituição do candidato. Como resultado, o modelo obtido de detecção teve desempenho dentro da média para monografias da base ICDAR2017 enquanto o modelo de correção teve resultados acima da média para a mesma base.

3.1.4 Mokhtar, Bukhari e Dengel (2018)

Os pesquisadores Mokhtar, Bukhari e Dengel (2018) partem da contestação dos métodos considerados estado da arte para correção de textos obtidos via OCR contra 4 métodos de correção apresentados: a nível de palavra e a nível de caracteres ambos com e sem normalização das sugestões de correção.

O foco dos autores foi voltado para a correção de erros, desta forma, os principais pontos apresentados por sua metodologia foram: o reconhecimento do melhor desempenho de modelos de correção a nível de caracteres e uso da normalização de palavras previstas pelos modelos. A normalização é um processo proposto pelos autores para verificar através da distância de Levenshtein entre os tokens de uma sentença se os tokens sugeridos pelo modelo de correção tem mais proximidade do que a sentença original. As bases de dados utilizadas no estudo abrangem três línguas. Em inglês, as bases foram UNLV e UW-III que são formadas por documentos digitalizados de fontes variadas. Em alemão, a base de dados foi criada a partir da raspagem de textos de páginas web, renderizados como imagens para posteriormente serem processados por OCR. A base em latim que foi construída a com base em livro digitalizado datado do século XV.

Os resultados do estudo diferem dos demais trabalhos relacionados, pelas bases diferentes e pelas métricas observadas, os autores avaliaram o desempenho dos modelos através da razão de erro entre caracteres (CER) e a razão de erro entre as palavras (WER). Para cada idioma analisado, os índices de mudança relativa foram calculados para cada razão,

estas dados foram levantados para todos os modelos analisados. Comparando os resultados entre os modelos, as principais observações percebidas foram: o pior desempenho em geral foi do modelo NMT baseado em palavras sem normalização, os melhores desempenhos em geral foram obtidos pelos modelos de NMT baseados em caracteres. Os desempenhos dos modelos SMT e NMT baseado em palavras normalizado, em geral, apresentaram resultados próximos, o modelo SMT obteve um desempenho levemente menor.

3.2 Relações Entre os Estudos Selecionados

Dentre os trabalhos selecionados, todos tiveram como objetivo aumentar a RelImp para modelos de correção e a precisão F1 dos modelos de detecção, exceto para Mokhtar, Bukhari e Dengel (2018) que não utilizaram métodos de detecção. A maior parte dos estudos utilizaram as bases da competição ICDAR, cada um abordou tipos de soluções distintas para o problema central, para facilitar a comparação das principais características de cada estudo bem como a proposta de pesquisa deste trabalho, um resumo das informações pode ser observado na Tabela 2.

Tabela 2 – Relações Entre Estudos - Fonte: Autoria Própria.

Estudo	Base de Imagens	Língua	Método de Detecção de erros	Método de Correção	Métricas de Detecção	Métricas de Correção
(Amrhein; Clematide, 2018)	ICDAR 2017	Francês e Inglês	NMT e SMT a nível de caracteres	Idem à detecção de erros	F1	RelImp
(Nguyen *et al.*, 2020b)	ICDAR 2017 e ICDAR 2019	Francês, Inglês, Tcheco, Alemão, Polonês, Espanhol, Neerlandês, Eslovaco e Finlandês	Similaridade	BERT ajustado como corretor	F1	RelImp
(Nguyen *et al.*, 2020a)	ICDAR 2017	Francês e Inglês	Verificação de unigrama no corpus	Substituição através do SOMA	F1	RelImp
(Mokhtar; Bukhari; Dengel, 2018)	UNLV e UW-III	Inglês, Alemão e Latim	Não utilizado	OpenNMT como corretor	Não utilizado	CER, WER
Proposta de pesquisa	ICDAR 2019	Francês, Inglês, Tcheco, Alemão, Polonês, Espanhol, Neerlandês, Eslovaco e Finlandês	NMT e SMT a nível de caracteres	Idem à detecção de erros	F1	RelImp

4 METODOLOGIA

Neste capítulo serão descritas as metodologia deste trabalho detalhando as bases de dados e seu tratamento na Seção 4.1. Na Seção 4.2 serão apresentadas os algoritmos e topologias do sistema de detecção de caracteres errôneos, enquanto na Seção 4.3, será abordado como serão realizadas as correções dos erros encontrados pelo sistema. A forma de avaliação do desempenho dos componentes dos sistemas de detecção e correção erros serão detalhados na Seção 4.4.

4.1 Base de Dados

A base de dados selecionada foi a disponibilizada no desafio ICDAR 2019 (Rigaud *et al.*, 2019) pois se aproxima dos trabalhos relacionados citados em 3.1. A base de dados contém 22 milhões de caracteres distribuídos em parágrafos em 10 idiomas diferentes. Cada amostra contém primeiro um identificador do tipo da amostra entre colchetes seguido de um parágrafo de texto. O primeiro parágrafo contém o resultado do texto obtido pelo OCR. O segundo parágrafo contém o texto obtido por OCR alinhado, ou seja, caracteres faltantes em relação ao GT são preenchidos com o caractere @. O último parágrafo contém o texto esperado.

Tabela 3 – Exemplo de Amostra da Base de Dados.

[OCR_toInput] A Misfion are but chips or fhave- ings)do not onely keep our fins lower,but also weigh against the tem poral penalty of those which are in the scale. It may admit a que stion whither it be a more precious Chri stian exercise to do good, or to endure evils : that state is certainly the best in which both are con joy ned, when suf fering many grie vances,we act as much . good
[OCR_aligned] @@@@A Misfion are but chips or fhave- ings)do not onely keep our fins lower,but also weigh against the tem@ poral penalty of those which are in the scale. It may admit a que@ stion whither it be a more precious Chri@ stian exercise to do good, or to endure evils : that state is certainly the best in which both are con joy ned, when suf@ fering many grie@ vances,we act as much . good
[GS_aligned] 92 A Misffon are but chips or shave- ings)do not onely keep our ffns lower,but also weigh against the tem- poral penalty of those which are in the scale. It may admit a que- stion whither it be a more precious Chri- stian exercise to do good, or to endure evils : that state is certainly the best in which both are con@joy@ned, when suf- fering many grie- vances,we act as much@@ good

A razão entre tokens de treinamento contendo ou não erros é um fator importante. Para garantir que o modelo traduza os erros apresentados, será montado um conjunto de treinamento com a razão de 75% de tokens contendo erros e 25% de tokens sem erros (Amrhein; Clematide, 2018).

4.1.1 Tratamento da Base de Dados

Como visto na Seção 4.1 os textos de OCR alinhados possuem caracteres de espaçamentos, para manter a integridade das palavras, os caracteres de espaçamento foram substituídos pelos respectivos caracteres do GT. Com o objetivo de preparar as amostras de treinamento e avaliação, os textos foram divididos em pares de tokens através do tokenizador da biblioteca NLTK. Para calcular as razões de amostras com erro, cada de token de OCR foi classificado como idêntico ou não ao GT.

Como a base de dados é composta por textos em diversas línguas, Amrhein e Clematide (2018) propõem separar e fatorar os caracteres de cada token e tratá-los como uma sentença para o treinamento dos modelos de tradução. Os caracteres foram fatorados adicionando um caractere "|" e um código de duas letras referente à língua original de dado token. Alguns exemplos de fatorações estão descritos na Tabela 4.

Tabela 4 – Exemplo de Conjunto de Dados Fatorados.

OCR	GT	Língua
i|DE h|DE m|DE	i h m	Alemão (DE)
P|FR a|FR r|FR t|FR 1|FR e|FR	P a r t i e	Francês (FR)
S|EN O|EN U|EN R|EN C|EN E|EN	S O U R C E	Inglês (EN)

4.2 Modelo de Detecção de Erros

Assim como proposto por Amrhein e Clematide (2018), o modelo de detecção pode se beneficiar da estrutura de combinação de métodos de SMT junto do modelo de NMT. Um resumo da estrutura proposta por Amrhein e Clematide (2018) pode ser vista na Figura 4. O sistema apresentado utiliza 4 tokens de entrada: OCR, GT, OCR do último e resultado do sistema para o último token. Foram utilizados 5 métodos de correção, sendo o principal, o método baseado na distância de Levenshtein, mais três sistemas de correção baseados em SMT e um modelo de correção NMT. O processamento de texto é dividido em 3 tarefas: detecção de erros, união de tokens e correção de erros. Na primeira etapa, o primeiro passo é verificar se o token OCR é diferente do resultado do método de Levenshtein, caso seja, o erro é detectado, caso contrário, é verificado se os resultados que apresentarem maior frequência são diferentes do token OCR, caso sejam, um erro é detectado, caso contrário, é verificado se o token de OCR é igual ao GT, caso seja, o erro é detectado, caso não seja, nenhum erro é detectado. A correção de erros é a segunda etapa do processo, no caso do token OCR ser diferente do resultado do método de Levenshtein, será escolhido o resultado com a menor distância de Levenshtein dentre os modelos que apresentarem um resultado diferente do token OCR. A terceira etapa é a união de tokens, o processo que se resume em verificar se tanto os tokens de OCR quanto o candidato a correção fazem parte do corpus do GT quando unidos ao último token de OCR ou ao

último resultado de correção. Caso nenhuma das uniões esteja no corpus, os tokens são tratados separadamente, caso contrário, os tokens são unidos.

4.2.1 Detecção por SMT

Os métodos selecionados para a etapa de detecção dos erros por SMT foram alguns dos sistemas utilizados por Amrhein e Clematide (2018). A mínima distância de Levenshtein entre cada token e o corpus Words da biblioteca NLTK atua como primeira etapa de detecção. Para os sistemas de suporte, inicialmente foi planejado utilizar o Moses Toolkit (Koehn *et al.*, 2007), que foi utilizado no sistema original, entretanto, devido a falta de manutenção da ferramenta, sua utilização se tornou inviável, levando substituição deste modelo pelo modelo de tradução IBM1 que é componente da biblioteca NLTK (NLTK, 2023). O último modelo de tradução é baseado na distância mínima de Jaccard comparada com o corpus words da biblioteca NLTK.

4.2.2 Detecção por NMT

O modelo de detecção NMT foi baseado no algoritmo Transformer-NMT. Este programa cria uma esteira de treinamento para um modelo de transformers (Muñoz, 2020). Este modelo permite a alteração de diversos hiperparâmetros. Foi escolhido manter as configurações iniciais do autor, sendo elas: 512 entradas e saídas e 6 camadas profundas para encoders e decoders, 8 camadas paralelas de head, 512 unidades de feedforward e dropout-rate de 0.2. Os parâmetros de treinamento também foram mantidos iguais aos utilizados pelo autor.

4.3 Modelo de Correção de Erros

Mantendo a proposta de um sistema integrado de Amrhein e Clematide (2018). A correção será realizada a partir do cálculo das distâncias de Levenshtein dos resultados obtidos na etapa de classificação, o candidato que apresentar a menor distância será o selecionado.

O diagrama da Figura 5 descreve a estrutura do sistema de detecção e correção proposto para o estudo.

4.4 Avaliação de Desempenho

As principais formas de aferir o desempenho dos modelos obtidos acompanharão os trabalhos citados na Seção 3.2 sendo as métricas: F1 para modelo de detecção e para o modelo de correção a mudança relativa, CER WER e RelImp. Os resultados obtidos serão comparados com os resultados apresentados pelas literaturas levantadas no Capítulo 3.

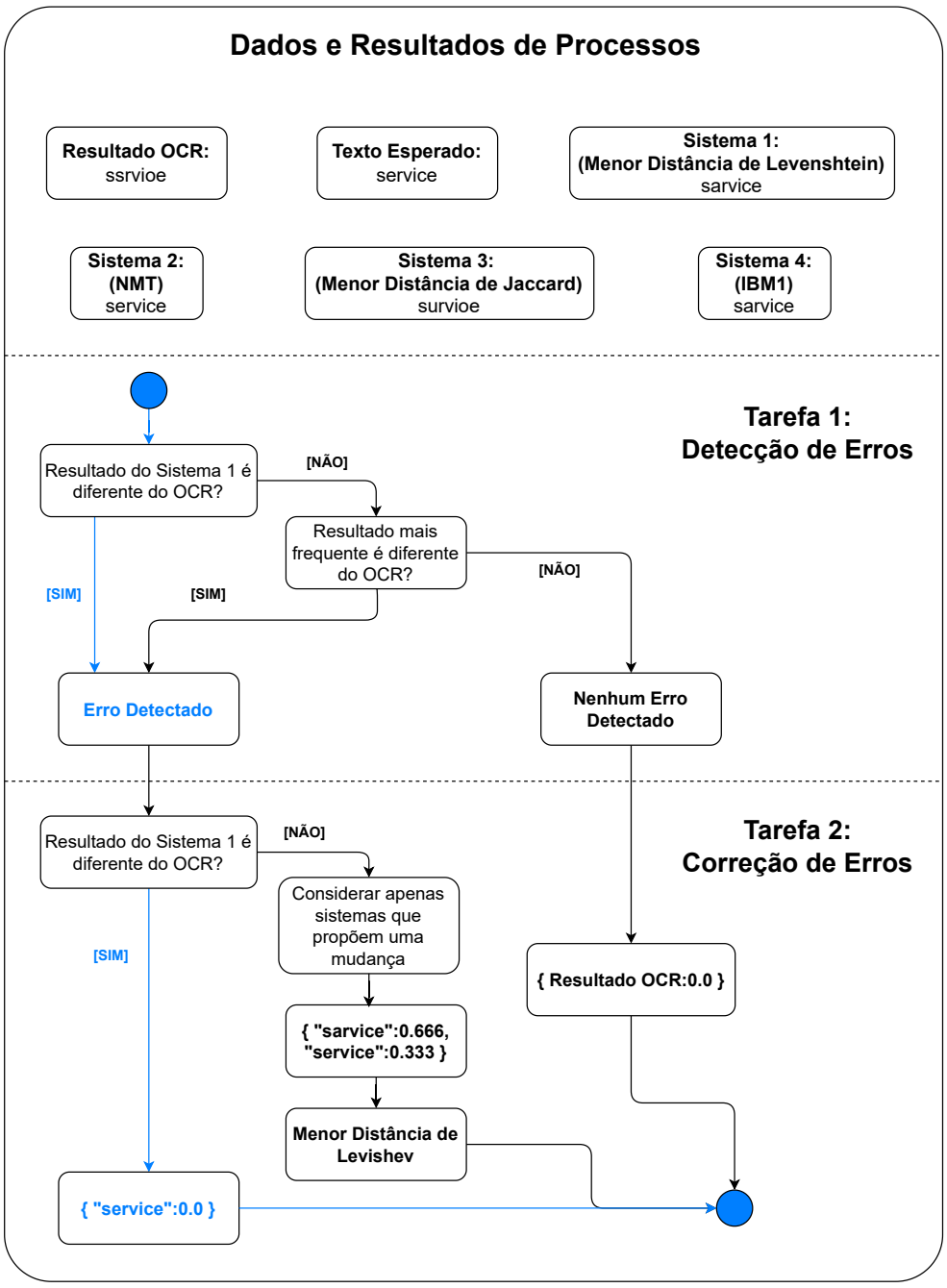

Figura 5 – Proposta de Sistema de Detecção e Correção.

Fonte: Autoria Própria.

5 AVALIAÇÃO EXPERIMENTAL

Este trabalho teve como objetivo construir um sistema de correção de caracteres para textos extraídos de OCR. Os resultados do estudo estão dispostos neste capítulo de modo que na Seção 5.1 são discutidos como foram criados e treinados os modelos SMT. A Seção 2.5.1 trata do treinamento do modelo NMT. A Seção 5.3 traz os resultados dos testes realizados. A Seção 5.4 apresenta característica do sistema utilizado para processar os algoritmos. Para o experimento foi utilizada parte da base dados ICDAR 2019 com cerca de 1.700.000 tokens de treinamento com a proporção de 75% de tokens com algum erro, para a avaliação de resultados foram utilizados 50.000 tokens com a proporção de 50% de tokens com erros.

5.1 Obtenção dos Modelos SMT

Dos quatro modelos utilizados neste trabalho, os modelos de Levenshtein e Jaccard foram implementados a partir da ordenação dos resultados dos cálculos das distâncias entre um token em letras minúsculas comparadas ao corpus words. Para cada palavra corrigida foram selecionados as menores distâncias de Jaccard e Levenshtein para cada modelo respectivamente. O modelo IBM1 foi treinado utilizando as amostras de treino não fatoradas. Ao alimentar o modelo com um token para traduzir, a operação é realizada caractere a caractere através da seleção do caractere que apresentar a maior probabilidade de acerto calculada pelo modelo. Os caracteres traduzidos são armazenados em uma lista e são unidos em uma palavra através da função TreebankWordDetokenizer do NLTK.

5.2 Obtenção do modelo NMT

O modelo que apresentou o treinamento mais complexo entre todos do sistema foi o de NMT. A construção do modelo de Transformer é realizada instanciando as classes das camadas do modelo para que seja possível definir uma função de treinamento utilizando o modelo instanciado junto de um gradiente descendente para atualização dos pesos do modelo e um gerenciador de modelos para armazenamento de versões dos modelos em arquivos. Foi feita uma primeira tentativa de treinamento com base no algoritmo original com dez épocas de treinamento. Devido ao elevado tempo observado para processar as primeiras épocas, o treinamento foi reiniciado com apenas uma época. Com uma época, o processo de treino durou cerca de 3 horas, as especificações de hardware utilizadas estão detalhadas na Seção 5.4.1. Foi observado pelas métricas de treinamento que as perdas apresentaram uma grande variação para as primeiras bateladas de treino e a partir de aproximadamente 15% das bateladas houve uma diminuição muito pequena das perdas. O mesmo comportamento foi observado para a precisão. Grande aumento no início do

processo e pouca variação no restante do treino. A quantidade de épocas selecionada para o treinamento definitivo foi de duas épocas devido à pequena elevação linear observada na Figura 6, o treinamento levou cerca de 9 horas e com base nas métricas apresentadas. Não houve uma melhora significativa nos indicadores que justificasse um processo com mais do dobro de duração de tempo, resultando na diminuição das perdas na ordem de 0,02 pontos e aumento de 0,005 de precisão. Após treinar o modelo de Transformer, o modelo de tradução foi criado. O modelo foi programado para receber uma palavra, separar e fatorar seus caracteres. A sentença criada é processada pelo encoder posicional de entrada, a tabela de itens processados é alimentada ao modelo que processa através de encoding-decoding até atingir o marcador de fim de sentença adicionado. Os resultados para cada caractere são concatenados em um decoder. A tabela armazenada no decoder é processada pelo decoder do corpus de saída para substituir os índices da tabela em caracteres e uní-los em uma palavra.

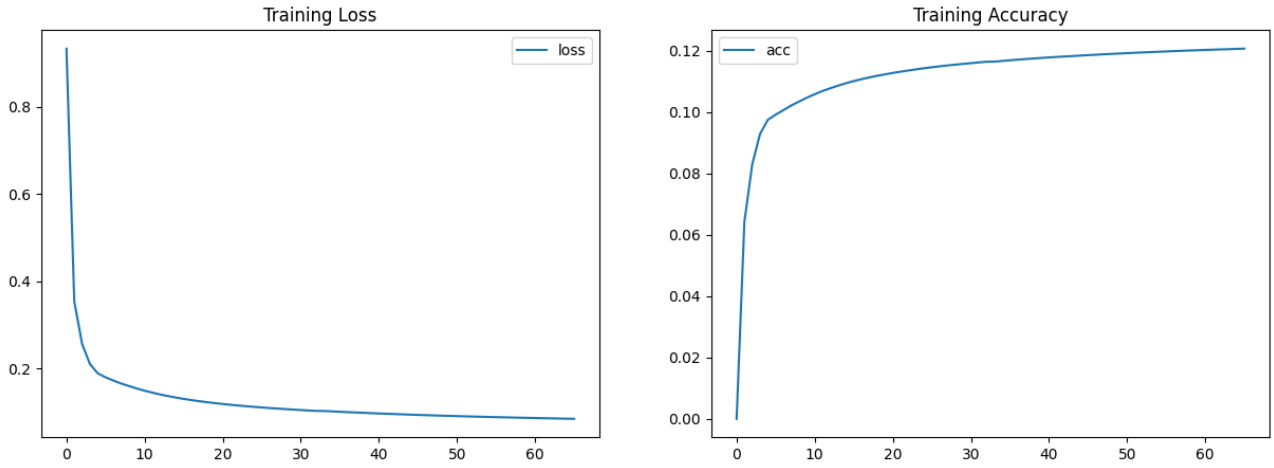

Figura 6 – Evolução do Erro e Precisão Durante Treinamento do Modelo NMT.

Fonte: Autoria Própria.

5.3 Desempenho dos Modelos e Sistema

Devido à estrutura do sistema construído utilizar as palavras traduzidas pelos modelos para realizar tanto as detecções quanto as correções, inicialmente serão analisados os desempenhos de correção para discutir posteriormente os resultados dos modelos como candidatos ao sistema de detecção. O sistema de correção e detecção será referenciado apenas como sistema ao longo do trabalho. Cada modelo foi avaliado com base nas métricas apresentadas na Seção 3.2.

Para verificar a capacidade de correção de palavras foram obtidas as medidas de CER para cada a base de dados original e para os resultados das correções para cada modelo, a Figura 7 demonstra uma comparação entre os resultados de CER obtidos.

Pode se notar que para apenas o modelo NMT foi capaz produzir uma MR positiva, mesmo que pequena na ordem de 4,95% em relação à literatura, o sistema teve a maior MR, evidenciando que o sistema foi capaz de aproveitar os melhores resultados entre os modelos. Comparando os resultados da Figura 7 com os resultados apresentados por Mokhtar, Bukhari e Dengel (2018) que obtiveram como melhor resultado de MR da CER uma redução de 2,71% de caracteres errados, os resultado apresentados foram satisfatórios tanto pelo modelo NMT quanto para o sistema que foi capaz de reduzir cerca de 12% dos caracteres errados.

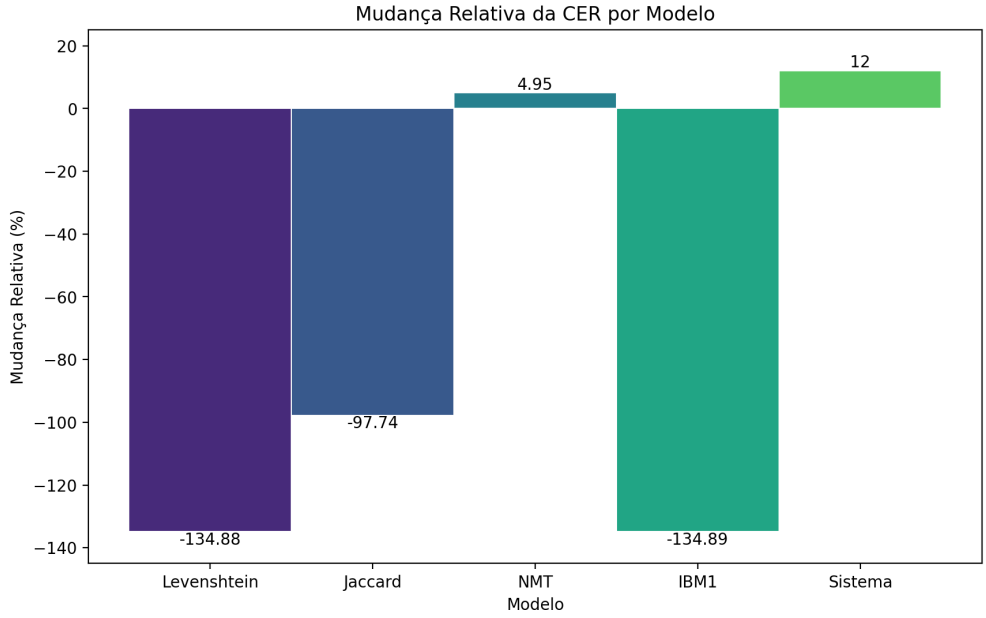

Figura 7 – Mudança Relativa das CERs por Modelo.

Fonte: Autoria Própria.

A Figura 8 evidencia os resultados das MR para as WERs. Pode se observar que assim como as CER, o modelo NMT obteve o melhor resultado entre os modelos e o sistema o melhor resultado no geral. Dentre os estudos relacionados, o melhor resultado de MR das WERs obtidas pelos estudos, foram de uma redução de 25,4% de erros (Mokhtar; Bukhari; Dengel, 2018). Não se pode afirmar que o sistema apresentou um bom resultado mesmo que tenha reduzido as palavras erradas em 15%.

Por conta do sistema construído utilizar todos os modelos na etapa da de detecção, foi possível avaliar a precisão F1 apenas do sistema como um todo, resultando no valor de 60,7%. A precisão resultante do teste está próxima dos resultados apresentados pelos trabalhos relacionados como se pode observar no resumo dos resultados presentes na Tabela 5.

A comparação da efetividade das correções pode ser observada no gráfico da Figura 9 que demonstra as RelImps de cada modelo. Na Figura 9 é possível notar os mesmos comportamentos das MRs das CERs e WERs. O NMT apresentou o melhor resultado

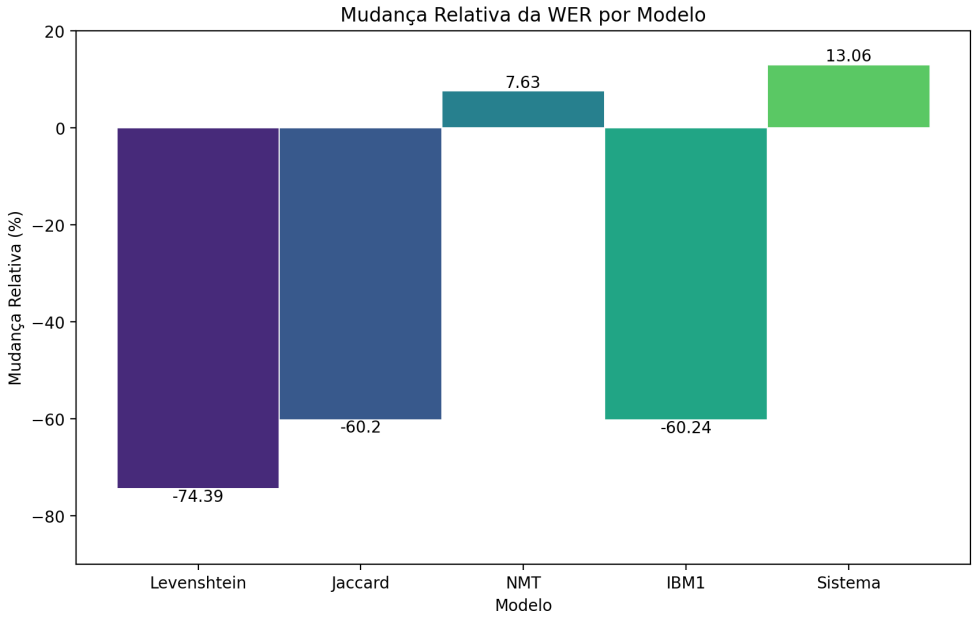

Figura 8 – Mudança Relativa das WERs por Modelo.

Fonte: Autoria Própria.

entre os modelos e o sistema superou todos os modelos atingindo a marca de 15,71%, um resultado ainda abaixo dos resultados apresentados pela literatura que estão resumidos na Tabela 5.

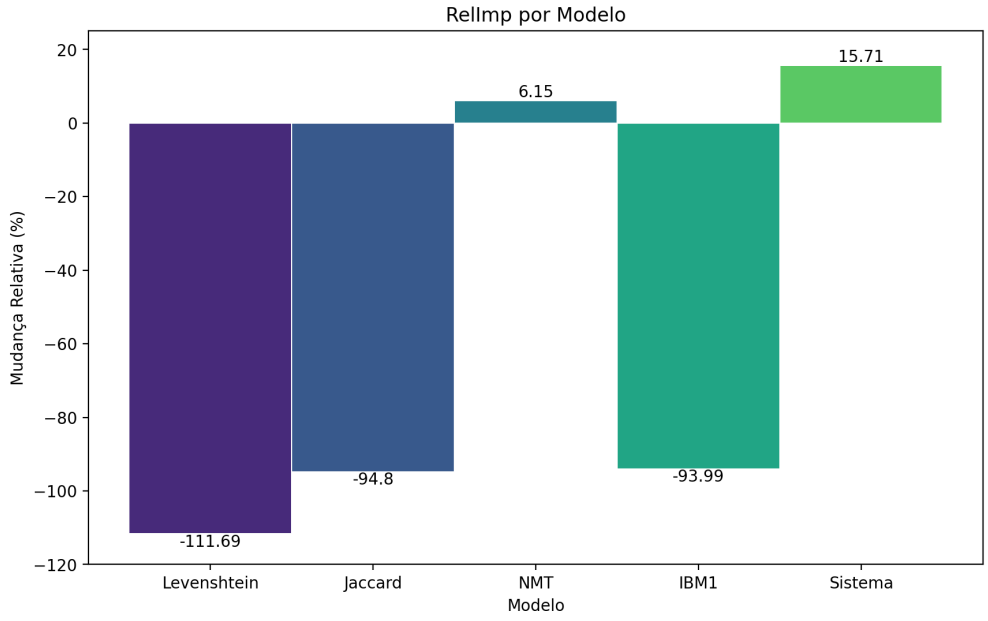

Figura 9 – Melhora Relativa por Modelo.

Fonte: Autoria Própria.

Tabela 5 – Resumo dos Melhores Resultados Experimentais.

Referência	MR CER (%)	MR WER (%)	RelImp (%)	Precisão F1 (%)
(Amrhein; Clematide, 2018)	-	-	43,0	73,0
(Nguyen *et al.*, 2020b)	-	-	27,0	74,0
(Nguyen *et al.*, 2020a)	-	-	33,7	68,9
(Mokhtar; Bukhari; Dengel, 2018)	2,71	29,44	-	-
Resultados Obtidos	12,0	13,06	15,71	60,7

5.4 Caracteristicas Computacionais do Sistema

Algoritmos de treinamento de IA dependem da configuração do ambiente de programação para viabilizar os processos de treinamento e execução (Kim; Deka, 2021). Esta seção a dedicada a apresentar pontos relevantes observados sobre as caracteristicas e consumo de recursos pelo sistema de detecção e correção.

5.4.1 Especificações de Hardware

Todos os códigos foram processados em linguagem python em uma máquina local utilizando o sistema operacional Windows 10. O hardware da máquina consiste de um processador AMD Ryzen 3600x com 6 núcleos de processamento, 16 GB de memória RAM e uma placa de aceleração gráfica Nvidia GTX 1660 com 6 GB de VRAM. Para se adequar à última versão da biblioteca Tensorflow, versão 2.10.0, que suporta o uso de GPUs, o driver CUDA 11.8 foi utilizado (Tensoflow, 2020).

5.4.2 Custo Computacional

Foi observado um alto custo computacional para processar o sistema, partindo da premissa apresentada na Seção 1.3, com o objetivo de criar um microsserviço, o software obtido utiliza grande parte de VRAM disponível, apresenta alto consumo de memória e por conta das bibliotecas e arquivos de bases de dados para uso no modelo NMT, também ocupa considerável espaço de armazenamento em disco. O custo computacional do sistema foi considerado elevado quando comparado ao mesmo sistema sem o uso do tradutor NMT. As informações dos custos computacionais estão detalhadas na Tabela 6

Tabela 6 – Resumo dos Custos Computacionais.

Componente	Quantidade Consumida com Modelo NMT	Quantidade Consumida sem Modelo NMT
RAM	1,5 GB	300 MB
VRAM	1,5 GB	0 B
CPU	10 %	2%
Espaço em disco	6,54 GB	700 MB

6 CONCLUSÕES

Neste capítulo serão apresentados as conclusões dos estudos realizados. O capítulo se divide em 3 seções nas quais a Seção 6.1 traz uma discussão dos resultados de desempenho, a Seção ?? aponta particularidades da base de dados e do desempenho computacional observadas e a Seção 6.2 propõe diferentes abordagens para o estudo da correção pós-ocr.

6.1 Resultados

Através dos dados apresentados na Seção 5.3, pode se concluir que: o sistema foi capaz detectar erros tão bem quanto a literatura atingindo a marca de F1 de 60,7%, porém, com índices de correção inferiores aos levantados pela literatura pois o sistema corrigiu corretamente cerca de 15,71% dos erros da base de dados.

Ao verificar que para todas as métricas levantadas, os modelos SMT apresentaram degradações das RelImps. Os modelos não corrigiram os erros e aumentaram a quantidade de caracteres errados. Um melhor desempenho destes modelos poderia garantir o melhor desempenho geral do sistema. Tanto o modelo Levenshtein quanto o modelo Jaccard utilizaram o corpus words do NLTK. Estes modelos são dependentes do corpus e é possível associar o baixo desempenho destes modelos com a referência que utilizaram. Seria possível utilizar os tokens do GT como referência, isto poderia produzir resultados melhores, entretanto, esta decisão tornaria o modelo muito específico para o tratamento da base de dados utilizada e menos genérico. A proposta inicial da pesquisa foi de obter um sistema de uso genérico.

O modelo IBM1 foi alinhado através do GT e não produziu bons resultados. A estratégia de tradução caractere a caractere não parece ter produzido bons resultados para esse modelo.

O modelo que obteve o melhor desempenho foi em geral foi o NMT, apesar do resultado positivo, é relevante considerar a relação entre a qualidade das correções do modelo e seu custo computacional associado. Como o estado da arte dos algoritmos de NMT estão baseados em transformers (Nguyen *et al.*, 2020b), faz sentido recorrer à esta estrutura quando se utiliza uma estratégia de tradução para a correção de erros. Entretanto, o custo de operação do modelo NMT levanta dúvidas sobre esta ser a melhor estratégia para um modelo de correção pós-OCR.

Observando que apesar do baixo desempenho dos modelos, a estratégia elaborada por Amrhein e Clematide (2018) é efetiva pois foi capaz de gerar um índice positivo de RelImp mesmo com modelos apresentando altos graus de degradação dos candidatos.

Sobre conjunto de dados utilizado pela literatura, a base foi planejada para uso

em competições, ela contém boa quantidade de amostras e erros conhecidos para correção de OCR. Porém, assim como Amrhein e Clematide (2018) e Nguyen *et al.* (2020b) reconheceram que as bases ICDAR2017 e ICDAR2019 são muito específicas. Atingir bons desempenhos na avaliação destas bases não garantem um bom modelo para uso geral. Por exemplo, as bases ICDAR não contém textos originados de formulários, os quais geralmente não possuem parágrafos mas informações dispersas, que após o OCR, são unidas em uma lista ou bloco de texto com pouco ou nenhum contexto entre si.

O trabalho de Amrhein e Clematide (2018) citam o uso de ferramentas como o Moses Toolkit e Nematus e Mokhtar, Bukhari e Dengel (2018) utilizaram OpenNMT, houveram esforços para reproduzir os experimentos com estes programas, porém estes projetos se encontram sem suporte e com códigos defasados impossibilitando o funcionamento de funções como uso da GPU ou do software por completo.

Importante pontuar que se não existirem limitações de hardware para a utilização do sistema de detecção e correção construído neste estudo é possível utilizá-lo e obter a margem mencionada anteriormente de capacidade de correção de erros. Porém, se limitações de hardware ou de ambiente de uso existirem, este modelo não se mostra como uma ferramenta viável para correção, devido ao seu custo de operação e baixo desempenho.

6.2 Trabalhos Futuros

Ainda utilizando a estrutura de múltiplos modelos, o sistema se beneficiaria virtualmente de qualquer tipo de modelo linguístico de correção, assim, algumas abordagens diferentes poderiam ser adicionadas para verificar a capacidade de correção de erros como modelos de linguagem massivos como ChatGPT, Llama ou Falcon.

Outra possibilidade de estudo seria avaliar a diferença de desempenhos de uma base de dados especializada contra uma base mais ampla de linguagem. A base especializada seria composta de palavras e caracteres com erros mais comuns de resultados de OCR. A base ampla seria mais próxima das bases ICDAR com mais amostras.

Caso utilizado um mecanismo de OCR permita obter a confiança do reconhecimento dos caracteres, seria possível elaborar um modelo não-semântico capaz de detectar erros ou utilizar as confianças de predição para aprimorar a precisão dos modelos de correção a nível de caracteres.

REFERÊNCIAS

AMRHEIN, C.; CLEMATIDE, S. Supervised ocr error detection and correction using statistical and neural machine translation methods. **Journal for Language Technology and Computational Linguistics (JLCL)**, v. 33, n. 1, p. 49–76, 2018.

AYODELE, T. O. **New Advances in Machine Learning**. Intech, 2010. (Introduction to Machine Learning). ISBN 9789533070346. Disponível em: http://gen.lib.rus.ec/book/index.php?md5=d7f984a43345b615e2f1959a8b9c0f6f.

CABRAL, M. O. **Detecção de Postagens Com Informações Falsas Sobre A Pandemia Do Covid-19 Na Rede Social Instagram**. Out 2021. Dissertação (Mestrado) — Future Computing Systems, São Carlos - SP, Out 2021.

CHAUDHURI, A. *et al.* **Optical Character Recognition Systems for Different Languages with Soft Computing**. 1. ed. Springer International Publishing, 2017. (Studies in Fuzziness and Soft Computing 352). ISBN 9783319502519. Disponível em: http://gen.lib.rus.ec/book/index.php?md5=5DA893626335E8C883A3E55E7C76D7BE.

CHIRON, G. *et al.* Icdar2017 competition on post-ocr text correction. *In*: **Proceedings of the 14th International Conference on Document Analysis and Recognition (2017)**. [*S.l.: s.n.*], 2017. p. 1423–1428.

CHUNG, N. C. *et al.* Jaccard/tanimoto similarity test and estimation methods for biological presence-absence data. **International Symposium on Bioinformatics Research and Applications**, v. 8, p. 210–229, 2018.

DUPONT, Q. **The Cryptological Origins Of Machine Translation**. 2022. Dinsponível em: http://amodern.net/article/cryptological-origins-machine-translation/. Acesso em: 14 de Abril de 2024.

GRASHCHENKO, S. Levenshtein distance computation. **Amodern**, 2024. Acesso em: 10 de Agosto de 2024.

GREBOVIC, M. *et al.* Machine learning models for statistical analysis. **Int. Arab J. Inf. Technol.**, v. 20, p. 505–514, 2023. Disponível em: https://api.semanticscholar.org/CorpusID:258839284.

KIM, S.; DEKA, G. C. **Hardware Accelerator Systems for Artificial Intelligence and Machine Learning**. Elsevier, 2021. v. 122. (Advance in Computers, v. 122). ISBN 978012823123. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=AggWEAAAQBAJ&oi=fnd&pg=PP1&dq=computing+artificial+intelligence+hardware&ots=xLRiwc1WzC&sig=_7KgwxLl-kazTaN_6NG3sWDYpSo#v=onepage&q=computing%20artificial%20intelligence%20hardware&f=false.

KOEHN, P. *et al.* **Moses: Open Source Toolkit for Statistical Machine Translation**. 2007. http://www2.statmt.org/moses/index.php?n=Main.HomePage. Acesso em: 06 de Março de 2024.

MERRITT, R. O que é um modelo transformer? **Nvidia**, 2022. Acesso em: 13 de Agosto de 2024.

MOKHTAR, K.; BUKHARI, S. S.; DENGEL, A. Ocr error correction: State-of-the-art vs an nmt based approach. **IAPR International Workshop on Document Analysis Systems**, 2018.

MORI, S.; NISHIDA, H.; YAMADA, H. **Optical Character Recognition**. Wiley, 1999. (Wiley Series in Microwave and Optical Engineering). ISBN 9780471308195. Disponível em: http://gen.lib.rus.ec/book/index.php?md5=d7f984a43345b615e2f1959a8b9c0f6f.

MUÑOZ, S. E. **Transformer-NMT**. 2020. https://github.com/edumunozsala/Transformer-NMT. Acesso em: 06 de Março de 2024.

NGUYEN, Q. *et al.* Ocr error correction using correction patterns and self-organizing migrating algorithm. **Springer Nature**, 2020.

NGUYEN, T. T. H. *et al.* Neural machine translation with bert for post-ocr error detection and correction. **Joint Conference on Digital Libraries (JCDL)**, 2020.

NLTK. **NLTK Documentation**. 2023. https://www.nltk.org/. Acesso em: 01 de Março de 2024.

ORACLE. **Tipos de dados Estruturados versus não Estruturados**. 2024. Dinsponível em: https://www.oracle.com/br/big-data/structured-vs-unstructured-data. Acesso em: 27 de Fevereiro de 2024.

RIGAUD, C. *et al.* Icdar 2019 competition on post-ocr text correction. *In*: **Proceedings of the 15th International Conference on Document Analysis and Recognition (2019)**. [*S.l.: s.n.*], 2019.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, 1958.

SAMUEL, A. L. Some studires in machine learning using the game of checkers. **IBM Journal of Research and Development**, v. 3, p. 210–229, 1959.

SINGH, S.; SINGH, S. Supervised ocr error detection and correction using statistical and neural machine translation methods. **2nd International conference on Electronics, Communication and Aerospace Technology (ICECA 2018)**, 2018.

SPACY. **Tokenization**. 2024. Dinsponível em: https://spacy.io/usage/linguistic-features#tokenization. Acesso em: 13 de Março de 2024.

TENSOFLOW. **GPU Support**. 2020. https://www.tensorflow.org/install/gpu?hl=pt-br. Acesso em: 06 de Março de 2024.

THANKI, J. D.; DAVDA, P. D.; SWAMINARAYAN, P. A review on ocr technology. **Journal of Emerging Technologies and Innovative Research (JETIR)**, v. 8, n. 4, 2021.

VAJJALA, S. *et al.* **Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems**. O'Reilly Media, Inc., 2020. ISBN 9781492054054. Disponível em: http://gen.lib.rus.ec/book/index.php?md5=E95D950B81B7CDA404070077EEF3BB62.

VASWANI, A. *et al.* Attention is all you need. **31st Conference on Neural Information Processing Systems NIPS**, 2017.