UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
CURSO DE GRADUAÇÃO EM ENGENHARIA ELETRÔNICA

Yuri Nunes Pagani

Extração e Classificação de Textos-chave de Artigos Acadêmicos Utilizando Modelo de Reconhecimento Óptico de Caracteres

Florianópolis
2023

Yuri Nunes Pagani

Extração e Classificação de Textos-chave de Artigos Acadêmicos Utilizando Modelo de Reconhecimento Óptico de Caracteres

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia Eletrônica do Centro Tecnológico da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia Eletrônica.
Orientador: Prof. Eduardo Luiz Ortiz Batista, Dr.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Pagani, Yuri Nunes
Extração e Classificação de Textos-chave de Artigos Acadêmicos Utilizando Modelo de Reconhecimento Óptico de Caracteres / Yuri Nunes Pagani ; orientador, Eduardo Luiz Ortiz Batista, 2023.
62 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia Eletrônica, Florianópolis, 2023.

Inclui referências.

1. Engenharia Eletrônica. 2. Extração de textos. 3. Classificação de textos. 4. Indexação de textos. 5. Aprendizado de máquina. I. Batista, Eduardo Luiz Ortiz. II. Universidade Federal de Santa Catarina. Graduação em Engenharia Eletrônica. III. Título.

Yuri Nunes Pagani

Extração e Classificação de Textos-chave de Artigos Acadêmicos Utilizando Modelo de Reconhecimento Óptico de Caracteres

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Bacharel em Engenharia Eletrônica" e aprovado em sua forma final pelo Curso de Graduação em Engenharia Eletrônica.

Florianópolis, 15 de Dezembro de 2023.

Documento assinado digitalmente
Fernando Rangel de Sousa
Data: 20/12/2023 17:27:21-0300
CPF: ***.649.114-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Fernando Rangel de Sousa, Dr.
Coordenador do Curso

Banca Examinadora:

Documento assinado digitalmente
Eduardo Luiz Ortiz Batista
Data: 20/12/2023 11:46:55-0300
CPF: ***.521.889-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Eduardo Luiz Ortiz Batista, Dr.
Universidade Federal de Santa Catarina
Orientador

Documento assinado digitalmente
Richard Demo Souza
Data: 20/12/2023 11:47:32-0300
CPF: ***.267.379-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Richard Demo Souza, Dr.
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Danilo Silva
Data: 20/12/2023 14:24:27-0300
CPF: ***.557.924-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Danilo Silva, Dr.
Universidade Federal de Santa Catarina

Dedico este trabalho aos meus amados pais e ao meu querido irmão, cujo amor e apoio foram a luz constante nas trilhas desafiadoras da jornada acadêmica.

AGRADECIMENTOS

Quero expressar meu profundo agradecimento aos meus pais pelo amor e dedicação incansáveis que me ofereceram ao longo de toda a minha vida. Agradeço também ao meu irmão, cujo caráter íntegro e amável sempre serviu como um dos maiores exemplos de humanidade para mim. Às minhas amizades, gratidão por iluminarem essa jornada desafiadora, indicando o caminho do equilíbrio. E, especialmente, à minha companheira, que me traz alegria diariamente e demonstra disposição para sacrificar tudo pelo nosso bem. Cada um de vocês contribuiu de maneira inestimável para a pessoa que sou hoje. Obrigado pelo apoio, amor e inspiração constante.

"O presente contém todo o passado."
(Antônio Gramsci)

RESUMO

O armazenamento e preservação do conteúdo de documentos escritos é muito caro para sociedade, visto a importância no acúmulo e construção de um conhecimento base para as mais diversas áreas do saber. O presente trabalho trata de apresentar um processo para realizar a extração de textos chaves de artigos científicos escaneados, ou digitalizados, através de modelos de aprendizado de máquina voltados para classificação de imagens e Reconhecimento Óptico de Caracteres (OCR). Para tal foi necessário realizar o processamento das imagens dos artigos, treinamento de modelo de classificação para localização dos textos-chave e extração dos textos-chave correspondentes as áreas classificadas através de modelo de OCR. Alguns resultados, como acurácia de mais de 70% para extração de textos-chave como título e resumo, demonstra que a abordagem utilizada para definir o processo como um todo é promissora.

Palavras-chave: Extração de textos. Classificação de textos. Indexação de textos. Aprendizado de máquina.

ABSTRACT

Storing and preserving the content of written documents is very crucial for society, given the importance of accumulating and building a comprehension base for the most diverse areas of knowledge. This work aims to present a process for extracting key texts from scanned or digitized scientific articles through machine learning models aimed at image classification and Optical Character Recognition (OCR). This process aims to facilitate the indexing of articles in order to help recognize the content covered by them. For this it was necessary to perform the processing of the images of the articles, classification model training for location of key texts and extraction of key texts corresponding to the areas classified through an OCR model. Some results, such as accuracy of more than 70% for key text extraction such as title and abstract, demonstrates that the approach used to define the process as a whole is promising.

Keywords: Text extraction. Text classification. Text indexing. Machine learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CNNs	Redes Neurais Convolucionais
IoU	Intersection over Union
LSTM	Long Short-term Memory
OCR	Reconhecimento Óptico de Caracteres
R-CNN	Rede Neural Convolucional Baseada em Região
ReLU	Rectified Linear Unit
RNAs	Redes Neurais Artificiais
RNN	Rede Neural Recorrente
SBrT	Simpósios Brasileiros de Telecomunicações
SBRT	Sociedade Brasileira de Telecomunicação

SUMÁRIO

1 INTRODUÇÃO

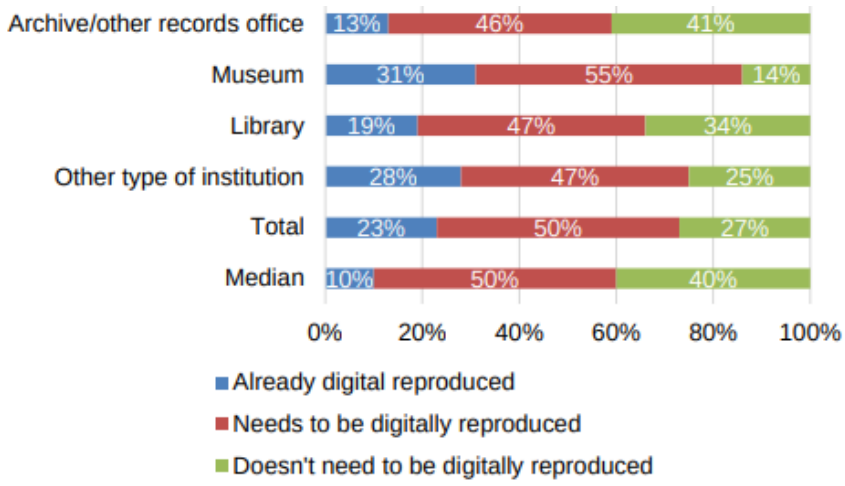
Ao longo da trajetória da humanidade, o armazenamento e registro de informações através de jornais, revistas, publicações científicas e outros meios sempre desempenharam um papel fundamental. Essa acumulação material não apenas representa a manifestação tangível do conhecimento de uma sociedade, mas também serve como testemunho do acúmulo de sabedoria ao longo de períodos históricos específicos. Durante muito tempo, a transcrição manual de livros, jornais e *papers* para formatos digitais foi a principal abordagem para preservar o conhecimento de maneira duradoura. Contudo, esse método demandava um esforço considerável e, por vezes, tornava-se impraticável.

Com a ascensão dos computadores e do armazenamento eletrônico, tornou-se imperativo migrar esses documentos do meio físico para o digital. No entanto, esse processo, marcado por uma quantidade exorbitante de trabalho manual, apresentava desafios significativos. No contexto atual, o progresso tecnológico na área de inteligência artificial emerge como um divisor de águas, eliminando a necessidade de trabalho manual na transcrição e classificação desses documentos. O advento de modelos pré-treinados possibilita a automação eficiente dessas tarefas, proporcionando uma maneira inovadora e eficaz de lidar com o vasto acervo de informações existente (NUNDLOLL *et al.*, 2022). logo, a inteligência artificial, em particular, contribui para a transformação do processo de transcrição e classificação de documentos, marcando uma era em que a tecnologia se torna a principal aliada na preservação do conhecimento.

Paralelamente aos avanços tecnológicos, enfrentamos um desafio considerável relacionado à vasta quantidade de arquivos históricos ainda não digitalizados em todo o mundo. Estima-se que milhões de documentos significativos permanecem inacessíveis em seus formatos físicos, impedindo um acesso eficiente e abrangente ao conhecimento que contêm. De acordo com o inquérito de digitalização da União Europeia de 2015 a cerca de 1000 instituições responsáveis pelo patrimônio cultural revela que, em média, 23% das coleções europeias foram digitalizadas até agora, com os depósitos de arquivos a terem a menor percentagem (13%), atrás das bibliotecas (19%) e dos museus (31%), sendo que, documentos textuais representam aproximadamente 83% desse patrimônio (NAUTA; HEUVEL; TEUNISSE, 2015). Ou seja, muitos documentos relevantes para a história da humanidade ainda aguardam sua transição para o meio digital.

Essa lacuna digital não apenas representa uma perda potencial de patrimônio cultural, mas também implica desafios substanciais para o avanço contínuo do conhecimento. A não classificação e indexação desses arquivos históricos impedem a sua utilização efetiva, resultando em um cenário no qual valiosas informações permanecem subutilizadas. A falta de acesso rápido e organizado a esses recursos históricos pode prejudicar seriamente a pesquisa acadêmica, a compreensão da evolução da sociedade e a formulação de políticas informadas.

Figura 1 – Percentagem dos patrimônios analógicos que já foram reproduzidos digitalmente.



Fonte: (NAUTA; HEUVEL; TEUNISSE, 2015)

Neste contexto, a automação por meio de modelos de inteligência artificial surge como uma solução promissora para superar esse obstáculo. A capacidade de transcrever, classificar e indexar automaticamente documentos históricos não apenas acelera o processo, mas também amplia significativamente o alcance e a eficácia das investigações acadêmicas. À medida que exploramos essa interseção entre a tecnologia emergente e o patrimônio histórico não digitalizado, torna-se claro que a adoção de abordagens inovadoras é crucial para desbloquear o vasto potencial desses registros, alimentando assim o contínuo desenvolvimento do conhecimento humano.

1.1 OBJETIVO

1.1.1 Objetivo Geral

O propósito fundamental desse trabalho consiste em conceber uma metodologia avançada para a extração de dados essenciais provenientes de artigos científicos que foram previamente escaneados, utilizando como conjunto de dados de referência os anais dos Simpósios Brasileiros de Telecomunicações (SBrT) realizados entre os anos de 1983 e 1998.

De maneira concisa, almeja-se a coleta de informações cruciais, tais como título, autor, resumo e palavras-chave de cada artigo contido nesses simpósios. Esta abordagem tem como intuito otimizar e agilizar o processo de inserção desses dados no sistema de registro da respectiva organização. A concretização desse objetivo assegurará não apenas a eficiência operacional, mas também a preservação meticulosa da rica história e contribuições científicas desses simpósios. O empenho na digitalização e classificação destes artigos emerge como um meio estratégico para garantir o acesso facilitado a esses

registros, perpetuando assim o valioso legado científico desses eventos ao longo do tempo.

1.1.2 Objetivos Específicos

Considerando o desenvolvimento do trabalho e o objetivo geral apresentado, destacam-se os seguintes objetivos específicos:

- Realizar tratamento do conjunto de dados, como: renomeação de arquivos, reestruturação de diretórios, exclusão de arquivos não necessários e etc.
- Pesquisar sobre mecanismos de transformação e processamento de imagens
- Definir pré-processamento do conjunto de dados, como: binarização, remoção de ruído, configuração de escala e etc.
- Pesquisar e definir melhor modelo disponível para classificação e rotulação de imagens voltado para documentos textuais.
- Pesquisar e definir melhor modelo disponível para extração de textos de imagens.
- Deliberar métricas de acurácia para modelos de classificação de imagens e extração de textos de imagens.
- Organizar processo de ponta-a-ponta. Ou seja, desde o dado bruto até os textos classificados e extraídos.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 MODELO DE CLASSIFICAÇÃO E DETECÇÃO DE OBJETOS

Os modelos de aprendizado de máquina destinados à classificação e detecção de objetos representam um marco significativo na capacidade dos sistemas computacionais em interpretar e compreender informações visuais, assemelhando-se muito com a percepção humana. Esses modelos têm evoluído substancialmente, sendo impulsionados, em grande parte, pelo advento das Redes Neurais Convolucionais (CNNs), conhecidas por sua eficácia na extração de características complexas de imagens.

A classificação e detecção em imagens envolve a atribuição de rótulos, ou categorias, a uma imagem com base em seu conteúdo visual. As CNNs emergiram como arquiteturas essenciais, capazes de aprender padrões hierárquicos e representações significantes, resultando em capacidade de generalização para novas imagens.

2.1.1 Rede Neural Convolucional

CNNs se assemelham muito com as tradicionais Redes Neurais Artificiais (RNAs), uma vez que ambas são compostas por neurônios que passam por processos de auto-otimização durante o aprendizado.

Cada neurônio em uma CNN ainda realiza operações fundamentais, incluindo uma combinação linear de entradas seguida por uma função de ativação não linear, semelhante à base de muitas RNAs convencionais. O que as difere de uma clássica RNA é que os neurônios que compõem as camadas da CNN são compostos por neurônios organizados em três dimensões, a dimensionalidade espacial da entrada (altura e largura) e a profundidade (O'SHEA; NASH, 2015).

2.1.1.1 Arquitetura Típica

Uma arquitetura típica de CNNs pode ser divida nos seguintes componentes:

1. Camada de Convolução: Camada onde são aplicados filtros (também chamados de kernels) à entrada da imagem para detectar padrões específicos, como bordas, texturas ou formas. Esses filtros deslizam pela imagem, realizando operações de convolução que resultam em mapas de características.

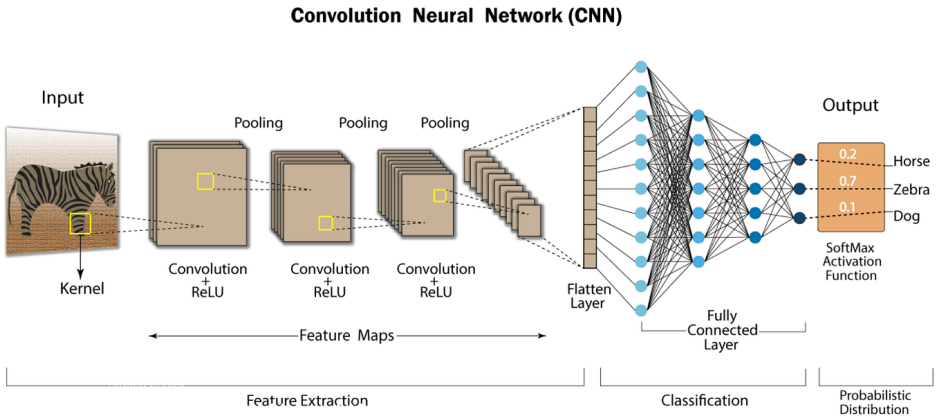
2. Camada de Ativação: Após a convolução, é comum aplicar uma função de ativação, geralmente a ReLU (Rectified Linear Unit), para introduzir não linearidades na rede. Ajudando a rede a aprender relações mais complexas nos dados.

3. Camada de *Pooling*: Camada onde ocorre a redução da dimensionalidade espacial dos mapas de características, diminuindo a quantidade de parâmetros e, assim, a

carga computacional. A operação mais comum é o *max pooling*, onde o valor máximo em uma região é mantido.

4. Camada Totalmente Conectada: Camada também presente em RNAs, ela conecta todos os neurônios da camada anterior a todos os neurônios da próxima camada. Esta camada é responsável por combinar as características aprendidas pelas camadas convolucionais e a de *pooling* para realizar uma previsão.

Figura 2 – Exemplo de arquitetura padrão para CNNs.



Fonte: (ALI *et al.*, 2023)

2.1.1.1.1 Camada de Convolução

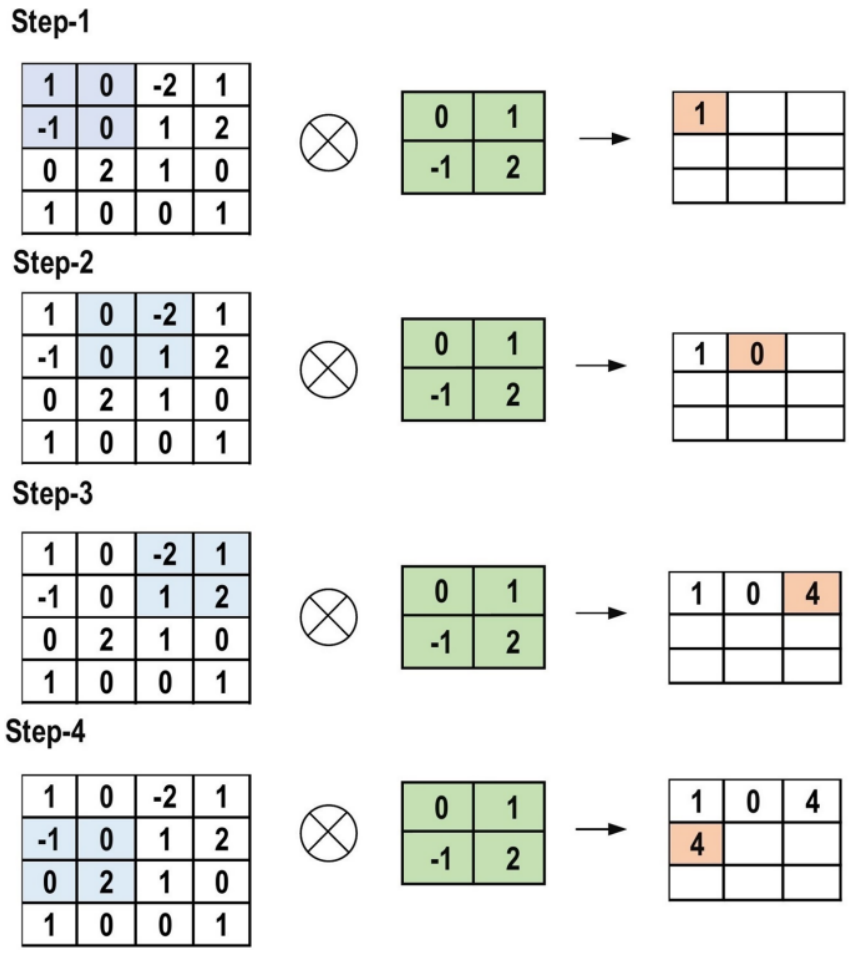
Consiste em uma coleção de filtros convolucionais (os *kernels*), onde, a imagem de entrada, expressa como vetores N-dimensionais, é convolvida com estes filtros para gerar o mapa de características na saída.

- Definição dos *kernels*: Uma matrix de valores discretos descreve o *kernel*, onde cada valor é chamado de peso do *kernel*. No início do processo de treinamento da CNN, esses pesos são inicializados com números aleatórios. Métodos como a inicialização Xavier ou He (XU; WANG, 2022) têm sido amplamente adotados para proporcionar uma inicialização eficaz dos pesos, facilitando a convergência do processo de treinamento. Durante cada época de treino, os pesos do *kernel* são ajustados com base nas informações contidas nos dados de treinamento. Esse processo de ajuste contínuo permite que o *kernel* aprenda a destacar características relevantes e distintivas na imagem (ALZUBAIDI *et al.*, 2021).

- Operação Convolucional: Primeiro, o *kernel* desliza sobre imagem horizontal e verticalmente. Durante isso, o produto escalar entre a imagem de entrada e o *kernel*

é determinado, onde seus os valores correspondentes são multiplicados e depois somados para criar um único valor escalar. Todo o processo é então repetido até que não seja possível deslizamento adicional. Logo, os valores calculados dos produtos escalares geram o mapa de características na saída (ALI *et al.*, 2023).

Figura 3 – Exemplo do processo de convolução.



Fonte: (ALZUBAIDI *et al.*, 2021)

2.1.1.1.2 Camada de Ativação

Camada responsável por introduzir não linearidades na rede, permitindo que esta aprenda padrões mais complexos e representações não lineares dos dados. Uma função de ativação é aplicada aos resultados obtidos após as operações de convolução ou *pooling* em cada neurônio da rede.

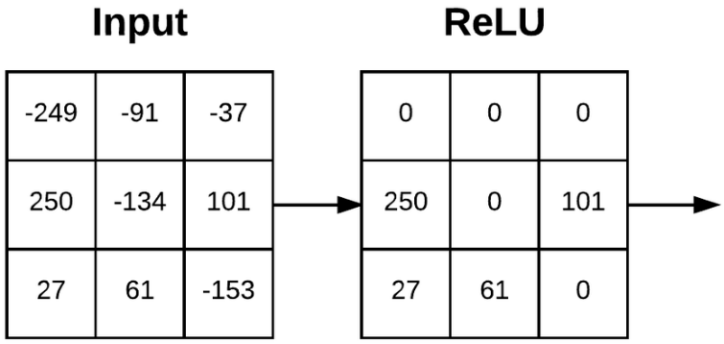
A função de ativação mais comum e amplamente utilizada nas CNNs é a função Rectified Linear Unit (ReLU). A ReLU opera de maneira simples: para qualquer valor de

entrada positivo, ela retorna o próprio valor, enquanto para valores negativos, ela retorna zero (BYERS; SHETA, 2023). A expressão matemática da ReLU é dada por:

$$f(x) = max(0, x) \tag{1}$$

A ReLU melhora as redes neurais acelerando seu treinamento pois o cálculo do gradiente é muito simples. Mas, em contrapartida, isso pode gerar um custo de 'neurônios mortos' dado a não ativação inicial.

Figura 4 – Exemplo do uso da ReLU.



Fonte: (BYERS; SHETA, 2023)

2.1.1.1.3 Camada de Pooling

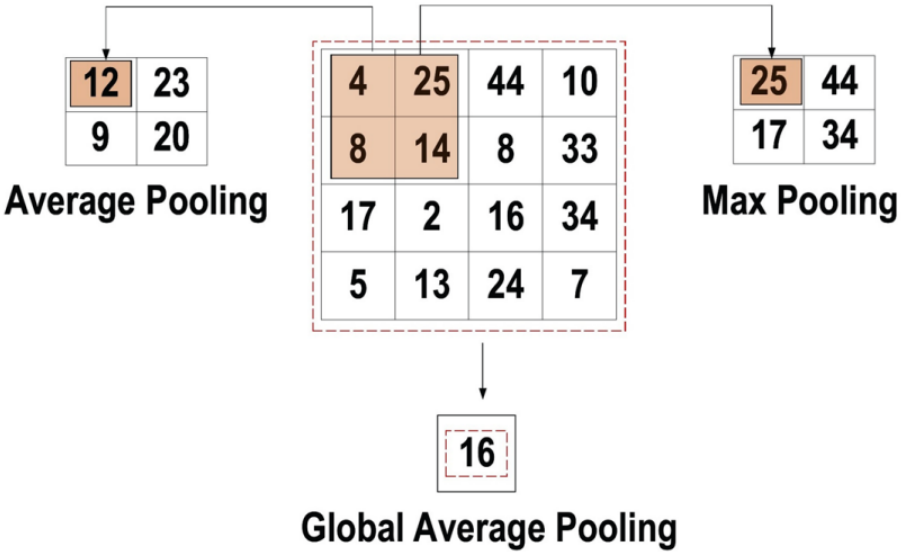
Camada que desempenha a função *sub-sampling* dos mapas de características gerados pelas operações convolucionais. Esses mapas de características, obtidos através das convoluções, frequentemente têm dimensões consideráveis, e a camada de *pooling* é responsável por reduzi-los, criando mapas de características menores. Ao mesmo tempo, essa camada mantém a maioria das informações dominantes em cada etapa do estágio de agrupamento, garantindo que não ocorra perda significativa de informações. Essa abordagem não apenas diminui o número de parâmetros, mas também reduz a complexidade computacional do modelo (ALI *et al.*, 2023).

Vários tipos de métodos de pooling estão disponíveis para utilização, são alguns deles:

- Agrupamento Máximo (*Max Pooling*): Mantém o valor máximo em uma região específica, destacando a característica mais proeminente.
- Agrupamento Mínimo (*Min Pooling*): Preserva o valor mínimo em uma região, enfatizando características distintas.
- Agrupamento Médio (*Average Pooling*): Calcula a média dos valores em uma região, proporcionando uma visão mais suavizada das características.

- Agrupamento Global Médio (*Global Average Pooling*): Realiza a média global sobre todo o mapa de características, resultando em um único valor para cada canal, reduzindo ainda mais a dimensionalidade.

Figura 5 – Exemplo do processo de *pooling*.



Fonte: (ALZUBAIDI *et al.*, 2021)

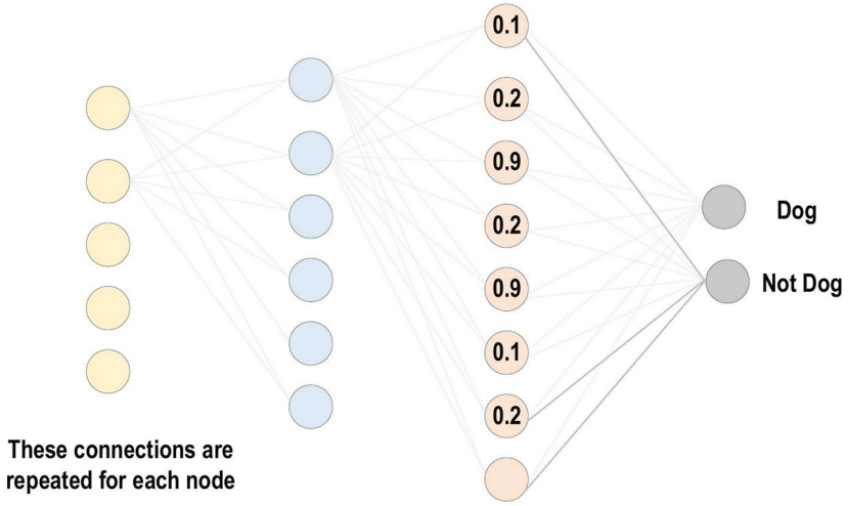
2.1.1.1.4 Camada Totalmente Conectada

A camada totalmente conectada costuma ser a última camada de uma CNN, onde cada neurônio dessa camada estabelece conexões com todos os neurônios da camada anterior. A entrada de uma camada totalmente conectada é um vetor que vem da camada convolucional ou da camada de *pooling* após o *flattening*[1] (O'SHEA; NASH, 2015). É estabelecido uma rede densa que captura relacionamentos de alto nível, facilitando a incorporação de mapas de características locais com características globais extraídas.

Por fim, o resultado dessa camada é utilizado como entrada para calcular o erro previsto criado nas amostras de treinamento no modelo CNN. Este erro revela a diferença entre a saída real e a prevista. Então, através de um algoritmo de *backpropagation* é feito o calculo do gradiente da função de perda em relação aos pesos da rede. Sendo esse gradiente responsável para determinar o ajuste dos pesos, buscando minimizar o erro.

[1] flattening é o processo utilizado para converter todas as matrizes bidimensionais resultantes de mapas de características em um único vetor linear longo e contínuo.

Figura 6 – Exemplo de camada totalmente conectada.



Fonte: (ALZUBAIDI *et al.*, 2021)

2.1.2 Rede Neural Convolucional Baseada em Região

A Rede Neural Convolucional Baseada em Região (R-CNN) foi desenvolvida para abordar especificamente o desafio da detecção de objetos em imagens, um problema que as CNNs tradicionais têm dificuldade em resolver eficientemente (GIRSHICK *et al.*, 2014).

As CNNs tradicionais foram inicialmente projetadas para tarefas de classificação de imagens, onde a tarefa é identificar a categoria geral da imagem. No entanto, para a detecção de objetos, é necessário não apenas classificar a imagem, mas também localizar e delimitar as posições dos objetos presentes. Além do mais, CNNs não possuem um mecanismo incorporado para sugerir automaticamente as regiões da imagem onde objetos podem estar presentes. Isso exige abordagens adicionais, como a varredura de regiões fixas ou o uso de técnicas externas para propostas de regiões de interesse.

Dada essas limitações, a R-CNN introduz uma abordagem inovadora para a detecção de objetos:

- Regiões de Interesse: A R-CNN utiliza algoritmos de geração de propostas de regiões para identificar candidatos promissores onde objetos podem estar presentes na imagem. Isso reduz significativamente a área a ser processada pela rede.

- Extração de Atributos por Região: As regiões propostas são recortadas e redimensionadas para serem processadas pela CNN, garantindo que cada região seja tratada individualmente. Isso permite a extração de características específicas de cada região.

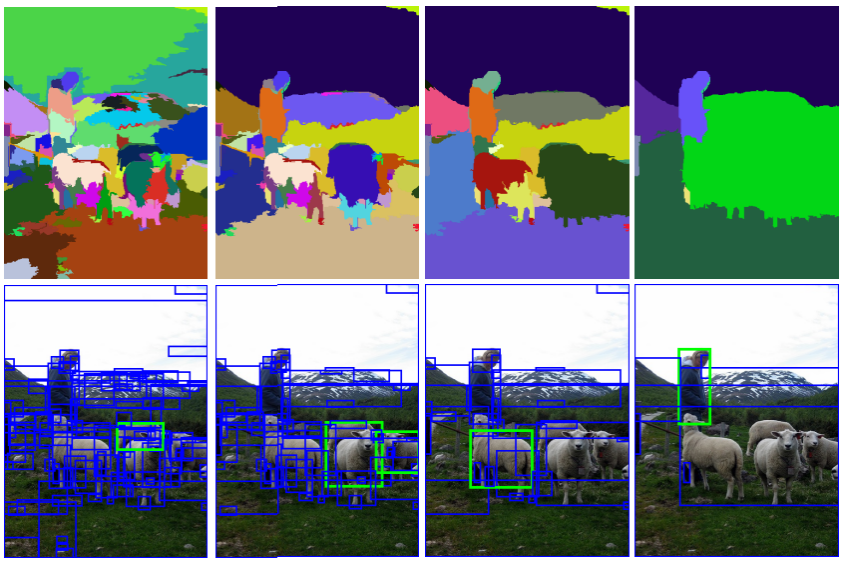
- Classificação e Refinamento: As características extraídas são então utilizadas para classificar a região proposta e, ao mesmo tempo, refinam a caixa delimitadora (*bounding box*) que envolve o objeto, melhorando a precisão da localização.

2.1.2.1 Regiões de Interesse

Para definição das regiões de interesse é utilizado o algoritmo *Selective Search* (UIJLINGS *et al.*, 2013) sobre as imagens de entrada. O *Selective Search* é um método de geração de propostas de regiões que visa encontrar regiões promissoras que possam conter objetos.

A abordagem do *Selective Search* é baseada em uma combinação de diferentes critérios de similaridade entre os pixels. Onde, primeiramente, a imagem é segmentada em regiões homogêneas baseadas em critérios como cor, textura, intensidade e compactação. Gerando um conjunto inicial de segmentos ou superpixels. Esses segmentos então são agrupados hierarquicamente com base em critérios de similaridade, formando regiões maiores que representam potenciais objetos. A partir do agrupamento hierárquico, o algoritmo cria uma coleção de propostas de regiões em diferentes escalas e resoluções. Essas propostas representam regiões candidatas que podem conter objetos. Então, essas propostas de regiões são ordenadas com base em uma medida de similaridade que considera a probabilidade de duas regiões conterem um objeto em comum. Com isso, As melhores propostas de regiões são então utilizadas como entradas para a R-CNN. Por fim, cada proposta é extraída da imagem original, redimensionada para um tamanho fixo e passada pela rede para extração de características e classificação.

Figura 7 – *Selective Search* aplicado para diferentes escalas.



Fonte: (UIJLINGS *et al.*, 2013)

2.1.2.2 Extração de Atributos por Região

As regiões propostas são submetidas a um tratamento individualizado pela CNN. Esse procedimento é essencial para garantir que cada região seja processada de maneira

adequada, permitindo a extração de características específicas que são fundamentais para a detecção de objetos.

Para garantir consistência e compatibilidade com a arquitetura da CNN, cada região recortada é redimensionada para um tamanho fixo, que é tipicamente a dimensão de entrada esperada pela rede. Isso padroniza as dimensões das regiões para facilitar o processamento subsequente.

As regiões redimensionadas são passadas através das camadas convolucionais da CNN. Nesse processo, a rede identifica padrões, texturas, bordas e outras características específicas presentes em cada região.

As representações resultantes das regiões, após o processo de extração de características, são transformadas em um vetor de características. Esse vetor é então utilizado para duas tarefas principais: classificação e regressão. A classificação determina a probabilidade da região conter um objeto de uma determinada classe, enquanto a regressão ajusta as coordenadas da caixa delimitadora que envolve o objeto, refinando sua localização.

2.1.2.3 Classificação e Refinamento

Dado o vetor de características criado na saída da CNN. Então é feito a classificação utilizando um classificador como o *Softmax* para determinar a probabilidade da região conter um objeto de uma determinada classe. Também, utilizando o vetor, é efetuada a regressão para ajustar as coordenadas da caixa delimitadora que envolve o objeto, refinando sua localização.

2.1.2.3.1 Softmax

O classificador *Softmax* é uma função de ativação que converte as saídas da CNN, conhecidas também como logits[2], em probabilidades associadas a diferentes classes (BISHOP, 1995). Ele opera atribuindo uma distribuição de probabilidade a cada classe possível.

A função *Softmax* é definida da seguinte forma para uma classe i:

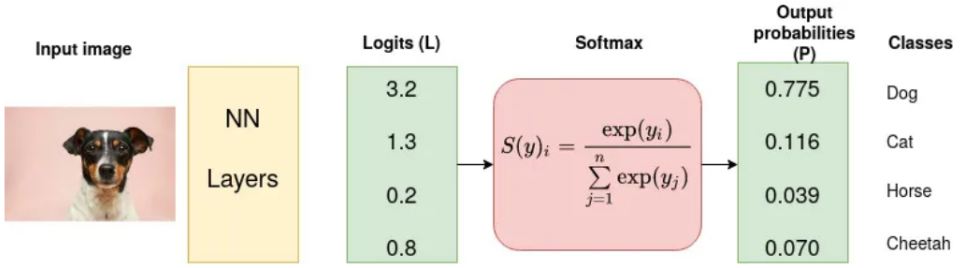
$$Softmax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}} \tag{2}$$

A partir dos logits, a função *Softmax* calcula a probabilidade associada a cada classe. Cada valor de logit é transformado em uma probabilidade positiva. Os logits são elevados à exponenciação (usando a função exponencial e^x), ampliando as diferenças entre os valores. Em seguida, as probabilidades resultantes são normalizadas, dividindo cada probabilidade pela soma de todas as probabilidades. Esse processo garante que a soma de todas as probabilidades seja igual a 1. Em suma, a classe com a maior probabilidade

[2] logits é uma forma de denominar os dados brutos da camada final de um modelo de aprendizagem profunda.

normalizada é escolhida como a classe prevista pela rede. Isso significa que a classe com a maior probabilidade é tratada como a classe mais provável dada a entrada.

Figura 8 – Classificação de imagem utilizando *Softmax*.



Fonte: (KOECH, 2020)

2.1.2.3.2 *Regressão de Bounding Box*

Em geral, as funções custo utilizadas para regressão de *bounding box* se fazem uso do fator de Intersection over Union (IoU), também conhecida como Jaccard Index. Essa métrica avalia a sobreposição entre duas regiões, como caixas delimitadoras (*bounding boxes*) (REZATOFIGHI *et al.*, 2019).

A fórmula do IoU é dada pela razão entre a área de interseção e a área da união entre duas regiões. Suponha que tenhamos duas caixas delimitadoras, A referente a caixa predita, e B a caixa verdadeira. Logo, IoU é:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

Então para os possíveis valores de IoU temos:

- $IoU = 0$: Não há sobreposição entre as duas regiões.
- $0 < IoU < 1$: Existe alguma sobreposição entre as regiões.
- $IoU = 1$: As regiões são idênticas ou completamente sobrepostas.

2.2 MODELO DE RECONHECIMENTO ÓPTICO DE CARACTERES

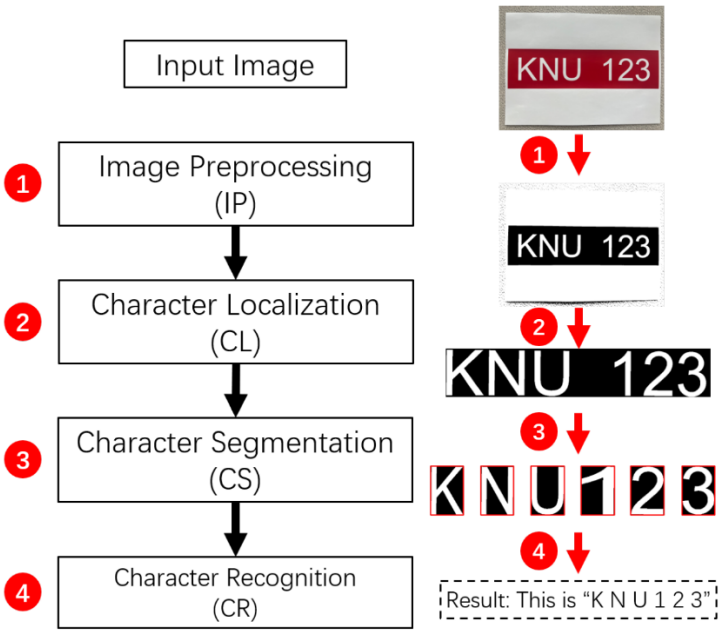
Modelos de OCR realizam a conversão de diferentes tipos de documentos, como documentos impressos, manuscritos ou imagens, em texto editável. Em outras palavras, OCRs convertem uma imagem bidimensional contendo texto, seja impresso ou manuscrito, de sua forma visual para um formato de texto compreensível por máquinas. A atividade realizada pelo OCR é determinado por diversos subprocessos, como:

- Pré-processamento da imagem.
- Localização de texto.
- Segmentação de caracteres.
- Reconhecimento de caracteres.

Ao longo de aproximadamente duas décadas, os OCRs desempenharam um papel significativo na automação da entrada de texto em sistemas computacionais. No entanto, durante esse período, os modelos convencionais de OCR enfrentaram limitações notáveis, incapazes de lidar eficientemente com uma variedade extensa de fontes e formatos de página (MEMON *et al.*, 2020). Esta incapacidade estendia-se a tipos com espaçamento proporcional, fontes de impressora a laser e até mesmo diversas fontes de máquinas de escrever, restringindo assim sua aplicabilidade. Consequentemente, o impacto do OCR convencional na conversão total de documentos para formato digital foi apenas marginal.

Os modelos de OCR de nova geração, por outro lado, superaram essas limitações ao incorporar avanços recentes na área de aprendizagem profunda. Através da utilização de modelos como Long Short-term Memory (LSTM) e extensos conjuntos de dados disponíveis publicamente, os OCRs alcançaram níveis de precisão de última geração em suas tarefas específicas. Essas melhorias representam um marco significativo no avanço da tecnologia OCR, destacando seu potencial transformador na automatização de processos de entrada de texto em ambientes computacionais.

Figura 9 – Processos realizados por um modelo de OCR.



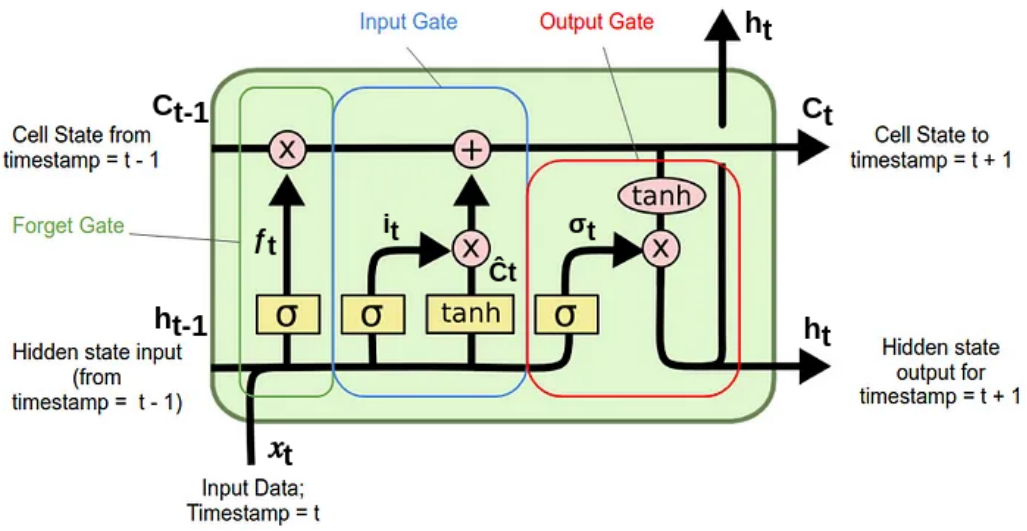
Fonte: (YU; KIM; CHOI, 2023)

2.2.1 Long Short-term Memory

A LSTM é um tipo de Rede Neural Recorrente (RNN) capaz de lidar com o Problema de Dissipação do Gradiente (HOCHREITER, 1998). Logo, apta a manipular e processar dados sequenciais. Uma rede LSTM consiste em uma série de células de memórias, cada uma das quais possui um conjunto de portas (portas de entrada, saída e portas de esquecimento) que controlam o fluxo de informações para dentro e para fora da célula. As portas são usadas para esquecer ou reter seletivamente informações dos intervalos de tempo anteriores, permitindo ao LSTM manter dependências de longo prazo nos dados de entrada (OLAH, 2015).

- Porta de Entrada (*Input Gate*): Determina quais novas informações devem ser adicionadas à célula de memória. A saída dessa porta é um vetor de valores candidatos a serem adicionados.
- Porta de Saída (*Output Gate*): Regula qual parte do estado interno da célula de memória será a saída da célula. A saída passa por uma função de ativação para produzir a saída final da unidade de memória.
- Porta de Esquecimento (*Forget Gate*): Decide quais informações na célula de memória devem ser descartadas ou mantidas. A saída dessa porta varia entre 0 e 1, onde 0 significa 'esquecer tudo' e 1 significa 'lembrar tudo'.

Figura 10 – Estrutura de uma célula do LSTM.



Fonte: (J., 2020)

O processo de uma célula do LSTM pode ser dado por:

1. O primeiro passo é decidir quais informações serão descartadas do estado da célula. Esta decisão é tomada por uma camada sigmoide (*Forget Gate*). Ele analisa h_{t-1} e x_t e gera um número entre 0 e 1 para cada valor no estado da célula C_{t-1}.

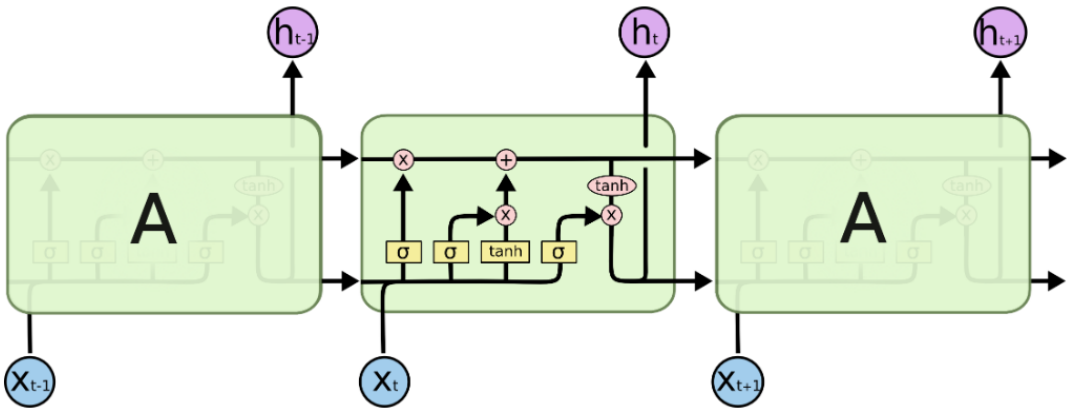
2. Na próxima etapa então é decidido quais novas informações serão armazenados no estado da célula C_t. Isto ocorre em dois momentos. Primeiro, uma camada sigmoide (*Input Gate*) decide quais valores iremos atualizar. Ao mesmo tempo, uma camada *tanh* cria um vetor de novos valores candidatos, $\hat{C}_t$, que podem ser adicionados ao estado.

3. Nesse momento é feito atualização do antigo estado da célula, C_{t-1}, para o novo estado da célula C_t. Para isso, é executado a multiplicação o estado antigo por f_t, a fim de 'esquecer' as características decididas anteriormente. Então a isso é somado $i_t * \hat{C}_t$. Gerando os valores para o novo estado.

4. Finalmente, baseado no estado da célula, é determinado a saída, a qual pode ser considerada uma versão filtrada de C_t. Primeiro, C_t passa por uma camada (*Output Gate*) que decide quais partes do estado da célula serão propagados. Em seguida, o estado da célula é transformado por uma camada *tanh* a fim de limitar valores entre -1 e 1. Então é feita a multiplicação da saída da camada *tanh* pela saída da porta sigmoide, de modo a produzir a saída da célula h_t.

Como dito, as LSTMs acabam sendo muito úteis para problemas de OCR dada a capacidade de capturar informações contextuais de longo prazo. Onde, em OCR, a compreensão de relações entre caracteres em palavras ou frases pode abranger uma extensão significativa.

Figura 11 – Representação de múltiplas células interconectadas de uma LSTM.



Fonte: (J., 2020)

2.3 TREINAMENTO DE MODELOS PRÉ-TREINADOS

2.3.1 Aprendizagem por Transferência

A Aprendizagem por Transferência (*Transfer Learning*) é a capacidade de transferir o conhecimento de um modelo sobre domínio no qual foi treinado para outro domínio, onde, geralmente, os dados são escassos ou é necessário um treinamento rápido (ZHUANG *et al.*, 2020). Logo, define-se como o processo de utilizar um modelo pré-treinado e adaptá-lo a um novo conjunto de dados diferente, transferindo ou reaproveitando as características aprendidas do domínio original.

Esse processo de aprendizado geralmente envolve usar os pesos pré-treinados nas primeiras camadas, que geralmente são gerais para muitos conjuntos de dados, inicializar as últimas camadas aleatoriamente e treiná-las para fins de classificação.

2.3.1.1 Ajuste Fino

O juste Fino (*Fine Tuning*) é uma abordagem de *Transfer Learning*, onde um modelo pré-treinado em uma tarefa geral é refinado para desempenhar tarefas mais específicas ou relacionadas. Ao invés de treinar um modelo do zero para cada tarefa, *Fine-Tuning* permite que aproveitemos o conhecimento adquirido por um modelo em uma tarefa inicial e o adaptemos para atender aos requisitos de uma nova tarefa, muitas vezes mais especializada(FU *et al.*, 2022).

O processo que define, em geral, o *Fine Tuning* é:

- Dado o modelo pré-treinado, as camadas pré-treinadas são "congeladas", mantendo os conhecimentos iniciais adquiridos da tarefa relacionada.
- Camadas adicionais são inseridas no topo do modelo para realizar a tarefa desejada.
- As novas camadas são treinadas com dados rotulados específicos da nova tarefa.
- Após algumas épocas de treinamento das novas camadas, pode-se optar por descongelar algumas ou todas as camadas pré-treinadas.
- Os pesos dessas camadas são então ajustados (*Fine Tuning*) em conjunto com as camadas adicionadas para melhor se adaptarem à nova tarefa.
- Comumente é realizada a redução da taxa de aprendizado ao ajustar as camadas pré-treinadas para evitar grandes atualizações que possam comprometer o conhecimento prévio.

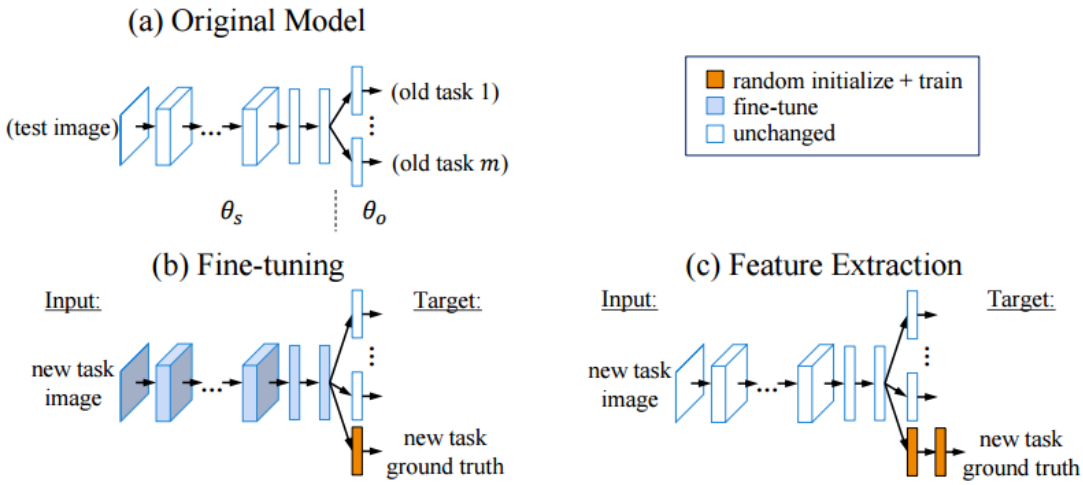
2.3.1.2 Extração de Características

Na abordagem de Extração de Características (*Feature Extraction*), o modelo pré-treinado é utilizado como uma máquina para a extração de características. Isso significa que

as camadas pré-treinadas do modelo são mantidas fixas, e apenas as camadas específicas relacionadas ao novo domínio são adicionadas ou modificadas (LI; HOIEM, 2017).

Nesse processo, as camadas iniciais, que aprendem características genéricas em tarefas amplas, são preservadas. A saída dessas camadas servem como um conjunto de características que são então alimentadas a um novo conjunto de camadas adicionadas, geralmente específicas para a nova tarefa.

Figura 12 – Comparação entre os diferentes tipos de *Transfer Learning*.



Fonte: (LI; HOIEM, 2017)

3 DESENVOLVIMENTO

Este capítulo tem como objetivo apresentar os procedimentos realizados para a obtenção de um processo completo para extração de textos-chave de artigos científicos, tais como: título, autor, resumo e palavras-chave. Aqui será demonstrado o conjunto de dados, pré-processamento e todas as abordagens para classificação e extração dos textos específicos. No geral, o trabalho foi em sua totalidade desenvolvido utilizando a linguagem *Python*, dada sua versatilidade para execução das mais diversas tarefas.

3.1 CONJUNTO DE DADOS

O conjunto de dados de interesse, no qual se faz necessária a extração dos textos-chave, é composto por artigos científicos apresentados nos Simpósios Brasileiros de Telecomunicação. Esse conjunto específico abrange todos os artigos dos simpósios realizados no período de 1983 a 1998.

Esses artigos constituem uma parte significativa da história da disseminação do conhecimento sobre telecomunicações e temas correlatos no Brasil. Para a Sociedade Brasileira de Telecomunicação (SBRT), essas publicações possuem um valor excepcional, uma vez que uma de suas principais missões consiste na difusão do conhecimento na área das telecomunicações por meio de eventos, reuniões, publicações, cursos e outras atividades técnico-científicas.

3.1.1 Obtenção dos Dados

Os artigos, em sua totalidade, foram disponibilizados integralmente em um diretório online hospedado no *Google Drive*. Contudo, a obtenção desses documentos variou de ano para ano, envolvendo diferentes métodos, como a extração de dados de CD-ROMs e disquetes, bem como o simples *upload* direto de pastas contendo os arquivos para o diretório no *Google Drive.*

3.1.2 Estrutura dos Dados

Os dados brutos, representados pelos artigos, já apresentavam uma organização inicial ao serem estruturados em repositórios que os segregavam por ano. Entretanto, cada repositório anual exibia uma estrutura distinta de sub-repositórios, os quais continham uma variedade de arquivos além dos próprios artigos. Essa diversidade de sub-repositórios decorreu diretamente do método específico de obtenção dos artigos para cada edição do simpósio ao longo dos anos. Para ilustrar a organização inicial, A Figura 13 apresenta exemplos da organização dos arquivos para dois anos distintos.

Quanto aos artigos em si, a maioria estava armazenada como arquivos PDF individualizados, carecendo de uma nomenclatura clara e pré-definida. A exceção foi observada

em um ano específico, no qual todos os artigos foram compilados em um único PDF, assemelhando-se a um livro referente ao simpósio. Esse PDF não apenas continha os artigos, mas também proporcionava um contexto abrangente sobre os acontecimentos durante o evento.

(a) Repositório dos dados do ano de 1992.

(b) Repositório dos dados do ano de 1997.

Figura 13 – Comparação da estrutura dos repositórios de dados brutos para diferentes anos.

Fonte: Autor

3.2 PRÉ-PROCESSAMENTO DOS DADOS

Antes de iniciar o tratamento dos dados, foi essencial reorganizar e limpar os diretórios para facilitar a manipulação dos artigos em fases subsequentes.

3.2.1 Reestruturação do Diretório

A fim de simplificar a manipulação dos arquivos de dados, o primeiro passo consistiu em eliminar a complexa estrutura de sub-repositórios associada a cada ano. Consequentemente, realizou-se um processo de achatamento, no qual todos os arquivos de dados foram consolidados na raiz do repositório. O código usado para fazer tal achatamento está apresentado no Algoritmo 3.1.

Algoritmo 3.1 – Código para achatamento de estrutura de sub-repositórios

```
import os
import shutil

def flat_dir(dir: str):
    for dirpath, _, filenames in os.walk(dir, topdown=False):
        for filename in filenames:
            i = 0
            source = os.path.join(dirpath, filename)
            target = os.path.join(dir, filename)

            while os.path.exists(target):
                i += 1
                file_parts = os.path.splitext(os.path.basename(filename))
                target = os.path.join(dir, f"{file_parts[0]}_{i}_{file_parts[1]}")
```

```
        shutil.move(source, target)

        if dirpath != dir:
            os.rmdir(dirpath)
```

Fonte: Autor

3.2.2 Remoção de Arquivos

Considerando que o repositório, após o processo de achatamento, inclui não apenas os arquivos PDF referentes aos artigos, mas também outros arquivos de metadados, foi necessário realizar a exclusão de todos os arquivos que não possuíam a formatação PDF. Isso foi feito com o objetivo de assegurar que o conjunto de dados consistisse exclusivamente em arquivos representativos de artigos. O código usado para essa tarefa está apresentando no Algoritmo 3.2. No caso de arquivos residuais de metadados que, por acaso, estavam formatados em PDF, a remoção foi efetuada manualmente.

Algoritmo 3.2 – Código para remoção de arquivos que não sejam de formatação PDF

```
import os

def remove_metadata_files(dir: str)
    for filename in os.listdir(dir):
        if not filename.lower().endswith('.pdf'):
            os.remove(os.path.join(parent, filename))
```

Fonte: Autor

3.2.3 Renomeação de Arquivos

Para simplificar a manipulação e identificação de cada arquivo de artigo, estabeleceu-se um padrão de nomenclatura, o qual é composto por SBrT_<ano do simpósio>_<numeração do artigo>. O código utilizado para renomear os artigos está apresentado no Algoritmo 3.3.

Algoritmo 3.3 – Código para renomeação dos arquivos seguindo a nomenclatura proposta.

```
import os

def rename_files(dir: str)
    symposium_year = dir.split("_")[-1]

    for i, filename in enumerate(os.listdir(dir)):
        new_filename = f"SBrT_{symposium_year}_{i}.pdf"
        os.replace(f"{dir}/{filename}", f"{dir}/{new_filename}")
```

Fonte: Autor

3.3 PROCESSAMENTO DOS DADOS

O processamento dos dados, nesse caso imagens de artigos, é de suma importância para a obtenção da máxima acurácia. Antes de submeter imagens a um OCR, é fundamental realizar procedimentos que melhorem a qualidade, clareza e legibilidade do conteúdo visual. Esse processamento é essencial para garantir resultados precisos e eficientes durante a extração de texto. Diversos desafios, como ruídos, distorções, iluminação inadequada e variações de contraste, podem comprometer a performance do OCR (MEMON *et al.*, 2020). Portanto, adotar estratégias de processamento de imagens se mostra indispensável para otimizar a qualidade do reconhecimento de caracteres.

As principais formas para processamento das imagens nesse contexto são:

- Filtragem de Ruídos: Utilização de filtros para remover ruídos indesejados da imagem.
- Aprimoramento de Contraste: Utilização de filtros para remover ruídos indesejados da imagem.
- Binarização: Conversão da imagem para uma representação binária, facilitando a segmentação entre o texto e o fundo.
- Correção de Iluminação: Conversão da imagem para uma representação binária, facilitando a segmentação entre o texto e o fundo.
- Normalização de Tamanho e Orientação: Padronização das dimensões e orientação das imagens para evitar distorções durante o OCR.

Ao empregar estas técnicas de processamento de imagens antes da execução do OCR, é possível otimizar a precisão e eficácia na extração de texto, contribuindo para a qualidade e confiabilidade do reconhecimento óptico de caracteres. Para o atual conjunto de dados pré-processado foi realizado somente o processamento através da binarização e filtragem de ruído.

Ao final do processamento é esperado obter arquivos de imagem, em formatação PNG, referentes a primeira pagina de cada artigo, pagina a qual apresenta todos os textos-chave, binarizado e filtrado obedecendo a estrutura estabelecida durante o pré-processamento.

3.3.1 Binarização

A binarização é um processo que transforma uma imagem em tons de cinza em uma imagem binária, onde os pixels são representados por apenas dois valores distintos, preto e branco. O objetivo é simplificar a análise da imagem, destacando as regiões de interesse e facilitando a detecção de objetos ou padrões.

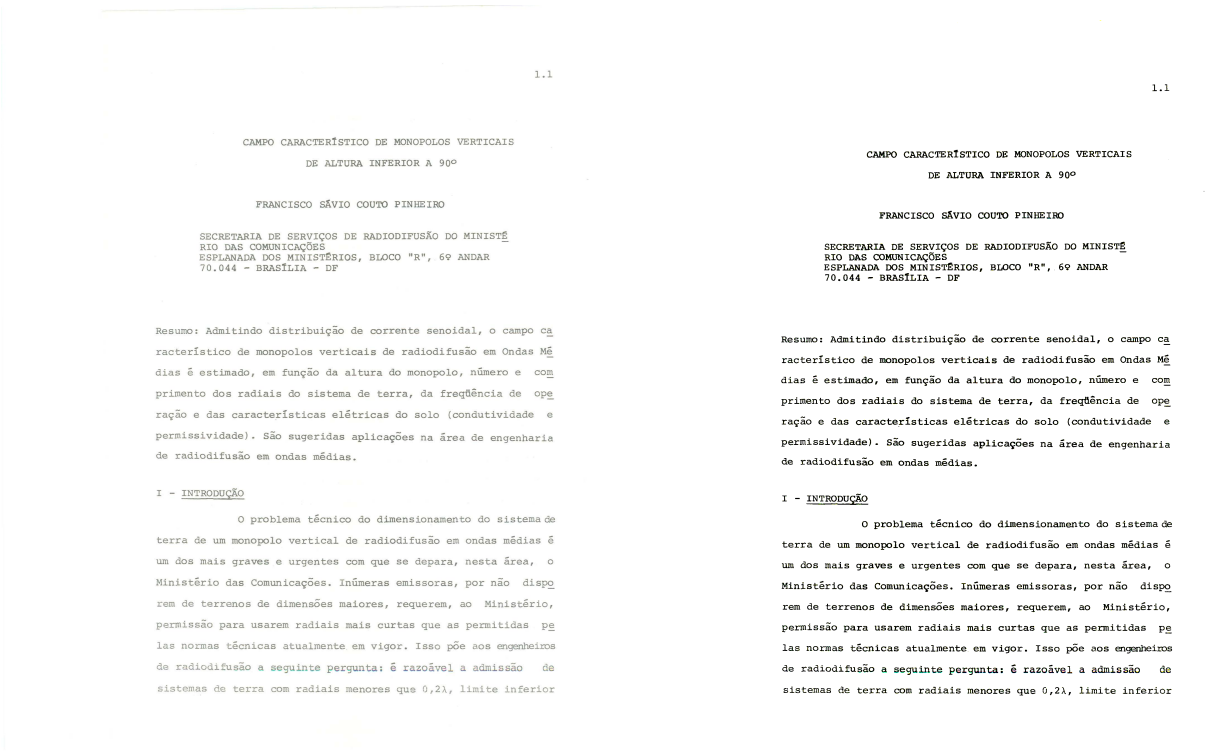
Inicialmente, a imagem original é convertida para uma imagem em tons de cinza. Isso é feito para unificar a representação de intensidade de cor, tornando-a mais simples de ser processada. Então, um valor de limiar é determinado para separar os pixels em duas categorias: aqueles abaixo do limiar são atribuídos à cor preta, enquanto os pixels acima do limiar são atribuídos à cor branca. A escolha do limiar pode ser fixa, global para toda a imagem, ou variável, adaptando-se localmente a diferentes regiões da imagem. Por fim, cada pixel na imagem em tons de cinza é comparado com o valor do limiar. Sendo o resultado final a imagem binária, composta apenas por pixels pretos e brancos, onde os objetos de interesse são destacados em branco sobre um fundo preto ou vice-versa. O código utilizado no presente trabalho para binarização está ilustrado no Algoritmo 3.4. Tal código baseia-se no uso do módulo cv2 do Python. A Figura 14 mostra o resultado obtido com a aplicação do processo de binarização em um dos artigos do dataset.

Algoritmo 3.4 – Código utilizado para realizar a binarização das imagens.

```
import cv2

def binarization(image):
  gray = cv2.cvtColor(cv2_img, cv2.COLOR_BGR2GRAY)
  _, image_binarized = cv2.threshold(image, 210, 255, cv2.THRESH_BINARY)
  return image_binarized
```

Fonte: Autor



(a) Artigo sem tratamento. (b) Artigo binarizado.

Figura 14 – Processo de binarização

Fonte: Autor

3.3.2 Filtragem de Ruídos

A filtragem de ruídos em uma imagem é um processo utilizado para reduzir ou remover padrões indesejados e aleatórios que podem comprometer a qualidade e a interpretação da imagem. Ruídos podem se manifestar de diversas formas, como granulações, pixels defeituosos, interferências elétricas, entre outros.

Para o presente conjunto de dados, após o processo de binarização, é realizado uma filtragem customizada, onde ocorre as seguintes operações morfológicas:

- Dilatação: A dilatação é usada para aumentar a região de pixels brancos em uma imagem. Ela envolve a aplicação de um kernel à imagem de entrada onde se pelo menos um pixel na vizinhança é branco, o pixel central (ponto âncora) na imagem de saída torna-se branco. Logo resultando em uma expansão ou dilatação das regiões brancas na imagem.

- Erosão: A erosão é usada para diminuir a região de pixels brancos em uma imagem. De forma diametralmente oposta a dilatação, na erosão ocorre que se todos os pixels na vizinhança são brancos, o pixel central (ponto âncora) na imagem de saída permanece branco; caso contrário, torna-se preto. Logo resultando em uma redução ou erosão das regiões brancas na imagem.

- Fechamento: Operação utilizada para fechar pequenos buracos e suavizar contornos em objetos de uma imagem binária. O fechamento é frequentemente usado para melhorar a segmentação de objetos, especialmente quando esses objetos estão muito próximos uns dos outros.

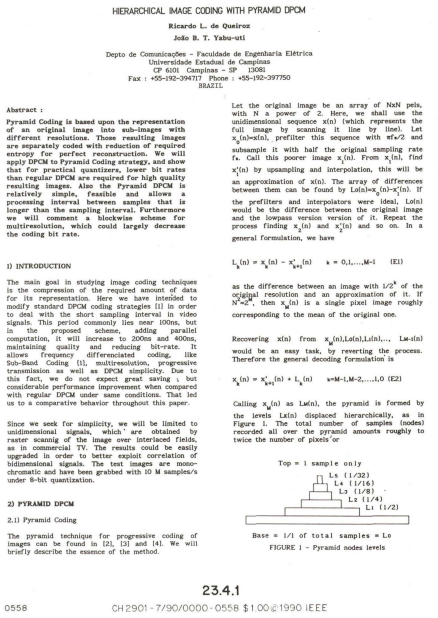
O código utilizado para realizar a filtragem customizada é ilustrado no Algoritmo 3.5. A Figura 15 traz um exemplo do processo de binarização e remoção de ruído para um artigo do conjunto de dados.

Algoritmo 3.5 – Código utilizado para realizar a filtragem de ruídos das imagens.

```
import numpy as np
import cv2

def noise_removal(image):
    kernel = np.ones((2, 2), np.uint8)
    image = cv2.dilate(image, kernel, iterations=2)
    image = cv2.erode(image, kernel, iterations=1)
    image = cv2.morphologyEx(image, cv2.MORPH_CLOSE, kernel)
    image_filtered = cv2.medianBlur(image, 3)
    return image_filtered
```

Fonte: Autor



HIERARCHICAL IMAGE CODING WITH PYRAMID DPCM

Ricardo L. de Queiroz
João B. T. Yabu-uti

Depto de Comunicações - Faculdade de Engenharia Elétrica
Universidade Estadual de Campinas
CP 6101 Campinas - SP 13081
Fax : +55-192-394717 Phone : +55-192-397750
BRAZIL

Abstract :

Pyramid Coding is based upon the representation of an original image into sub-images with different resolutions. Those resulting images are separately coded with reduction of required entropy for perfect reconstruction. We will apply DPCM to Pyramid Coding strategy, and show that for practical quantizers, lower bit rates than regular DPCM are required for high quality resulting images. Also the Pyramid DPCM is relatively simple, feasible and allows a processing interval between samples that is longer than the sampling interval. Furthermore we will comment a blockwise scheme for multiresolution, which could largely decrease the coding bit rate.

1) INTRODUCTION

The main goal in studying image coding techniques is the compression of the required amount of data for its representation. Here we have intended to modify standard DPCM coding strategies [1] in order to deal with the short sampling interval in video signals. This period commonly lies near 100ns, but in the proposed scheme, adding parallel computation, it will increase to 200ns and 400ns, maintaining quality and reducing bit-rate. It allows frequency differenciated coding, like Sub-Band Coding [1], multiresolution, progressive transmission as well as DPCM simplicity. Due to this fact, we do not expect great saving , but considerable performance improvement when compared with regular DPCM under same conditions. That led us to a comparative behavior throughout this paper.

Since we seek for simplicity, we will be limited to unidimensional signals, which are obtained by raster scannig of the image over interlaced fields, as in commercial TV. The results could be easily upgraded in order to better exploit correlation of bidimensional signals. The test images are monochromatic and have been grabbed with 10 M samples/s under 8-bit quantization.

2) PYRAMID DPCM

2.1) Pyramid Coding

The pyramid technique for progressive coding of images can be found in [2], [3] and [4]. We will briefly describe the essence of the method.

Let the original image be an array of NxN pels, with N a power of 2. Here, we shall use the unidimensional sequence x(n) (which represents the full image by scanning it line by line). Let $x_0(n)=x(n)$, prefilter this sequence with $\pi f_s/2$ and subsample it with half the original sampling rate f_s. Call this poorer image $x_1(n)$. From $x_1(n)$, find $x_1^*(n)$ by upsampling and interpolation, this will be an approximation of x(n). The array of differences between them can be found by $L_0(n)=x_0(n)-x_1^*(n)$. If the prefilters and interpolators were ideal, $L_0(n)$ would be the difference between the original image and the lowpass version version of it. Repeat the process finding $x_2(n)$ and $x_2^*(n)$ and so on. In a general formulation, we have

$$L_k(n) = x_k(n) - x_{k+1}^*(n) \qquad k = 0,1,\ldots,M-1 \qquad (E1)$$

as the difference between an image with $1/2^k$ of the original resolution and an approximation of it. If $N^2=2^M$, then $x_M(n)$ is a single pixel image roughly corresponding to the mean of the original one.

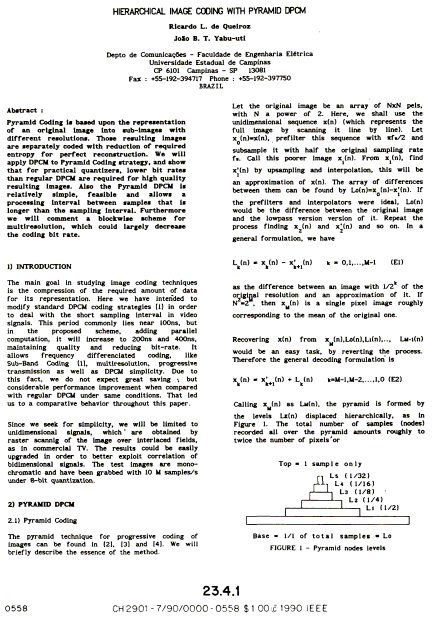
Recovering x(n) from $x_M(n),L_M(n),L_{M-1}(n),\ldots,\ L_{M-1}(n)$ would be an easy task, by reverting the process. Therefore the general decoding formulation is

$$x_k(n) = x_{k+1}^*(n) + L_k(n) \qquad k=M-1,M-2,\ldots,1,0 \quad (E2)$$

Calling $x_0(n)$ as $L_M(n)$, the pyramid is formed by the levels $L_k(n)$ displaced hierarchically, as in Figure 1. The total number of samples (nodes) recorded all over the pyramid amounts roughly to twice the number of pixels or

FIGURE 1 - Pyramid nodes levels

0558 CH2901-7/90/0000-0558 $1.00©1990 IEEE



HIERARCHICAL IMAGE CODING WITH PYRAMID DPCM

Ricardo L. de Queiroz
João B. T. Yabu-uti

Depto de Comunicações - Faculdade de Engenharia Elétrica
Universidade Estadual de Campinas
CP 6101 Campinas - SP 13081
Fax : +55-192-394717 Phone : +55-192-397750
BRAZIL

Abstract :

Pyramid Coding is based upon the representation of an original image into sub-images with different resolutions. Those resulting images are separately coded with reduction of required entropy for perfect reconstruction. We will apply DPCM to Pyramid Coding strategy, and show that for practical quantizers, lower bit rates than regular DPCM are required for high quality resulting images. Also the Pyramid DPCM is relatively simple, feasible and allows a processing interval between samples that is longer than the sampling interval. Furthermore we will comment a blockwise scheme for multiresolution, which could largely decrease the coding bit rate.

1) INTRODUCTION

The main goal in studying image coding techniques is the compression of the required amount of data for its representation. Here we have intended to modify standard DPCM coding strategies [1] in order to deal with the short sampling interval in video signals. This period commonly lies near 100ns, but in the proposed scheme, adding parallel computation, it will increase to 200ns and 400ns, maintaining quality and reducing bit-rate. It allows frequency differenciated coding, like Sub-Band Coding [1], multiresolution, progressive transmission as well as DPCM simplicity. Due to this fact, we do not expect great saving , but considerable performance improvement when compared with regular DPCM under same conditions. That led us to a comparative behavior throughout this paper.

Since we seek for simplicity, we will be limited to unidimensional signals, which are obtained by raster scannig of the image over interlaced fields, as in commercial TV. The results could be easily upgraded in order to better exploit correlation of bidimensional signals. The test images are monochromatic and have been grabbed with 10 M samples/s under 8-bit quantization.

2) PYRAMID DPCM

2.1) Pyramid Coding

The pyramid technique for progressive coding of images can be found in [2], [3] and [4]. We will briefly describe the essence of the method.

Let the original image be an array of NxN pels, with N a power of 2. Here, we shall use the unidimensional sequence x(n) (which represents the full image by scanning it line by line). Let $x_0(n)=x(n)$, prefilter this sequence with $\pi f_s/2$ and subsample it with half the original sampling rate f_s. Call this poorer image $x_1(n)$. From $x_1(n)$, find $x_1^*(n)$ by upsampling and interpolation, this will be an approximation of x(n). The array of differences between them can be found by $L_0(n)=x_0(n)-x_1^*(n)$. If the prefilters and interpolators were ideal, $L_0(n)$ would be the difference between the original image and the lowpass version version of it. Repeat the process finding $x_2(n)$ and $x_2^*(n)$ and so on. In a general formulation, we have

$$L_k(n) = x_k(n) - x_{k+1}^*(n) \qquad k = 0,1,\ldots,M-1 \qquad (E1)$$

as the difference between an image with $1/2^k$ of the original resolution and an approximation of it. If $N^2=2^M$, then $x_M(n)$ is a single pixel image roughly corresponding to the mean of the original one.

Recovering x(n) from $x_M(n),L_M(n),L_{M-1}(n),\ldots,\ L_{M-1}(n)$ would be an easy task, by reverting the process. Therefore the general decoding formulation is

$$x_k(n) = x_{k+1}^*(n) + L_k(n) \qquad k=M-1,M-2,\ldots,1,0 \quad (E2)$$

Calling $x_0(n)$ as $L_M(n)$, the pyramid is formed by the levels $L_k(n)$ displaced hierarchically, as in Figure 1. The total number of samples (nodes) recorded all over the pyramid amounts roughly to twice the number of pixels or

Top = 1 sample only
L5 (1/32)
L4 (1/16)
L3 (1/8)
L2 (1/4)
L1 (1/2)
Base = 1/1 of total samples = L0

FIGURE 1 - Pyramid nodes levels

23.4.1

0558 CH2901-7/90/0000-0558 $1.00©1990 IEEE

(a) Artigo sem tratamento. (b) Artigo Binarizado e Filtrado.

Figura 15 – Processo de Filtragem após Binarização

Fonte: Autor

3.4 CLASSIFICAÇÃO DOS ARTIGOS

3.4.1 Abordagem OCR Tesseract

O Tesseract é uma ferramenta OCR (Reconhecimento Óptico de Caracteres) de código aberto que foi originalmente desenvolvida pela Hewlett-Packard nos anos 80 e posteriormente adquirida pelo Google. Desde então, o Tesseract passou por significativas melhorias e atualizações, tornando-se uma dos modelos de OCR mais amplamente utilizadas (SMITH, 2007).

A robustez desse modelo é evidenciada pela sua capacidade de lidar com uma variedade de fontes, tamanhos e estilos de texto em imagens. Ele é projetado para ser eficaz em diferentes cenários, desde documentos digitalizados de alta qualidade até imagens com baixa resolução, com ruídos ou distorções.

O Tesseract possui uma funcionalidade essencial que é a capacidade de classificar regiões com texto em uma imagem. Isso significa que ele pode identificar e delimitar áreas onde há texto, proporcionando uma abordagem eficaz para a segmentação de documentos com fim de extrair o texto.

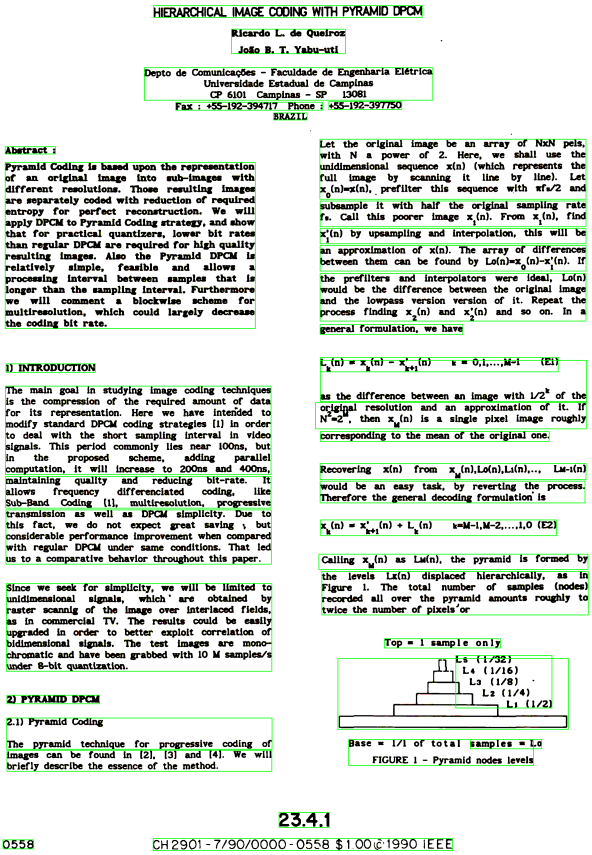
Dada essa capacidade de classificar regiões, é possível através de sua interface extrair não somente os textos finais, mas também as coordenadas, na imagem, para as caixas delimitadores desses textos. O Tesseract então fornece essas coordenadas em uma lista ordenada pela aparição desses blocos de textos, de cima para baixo, na imagem. A

Figura 16 apresenta os *bounding boxes* definidos pelo Tesseract.

Logo, através desse abordagem, uma vez definida as *bounding boxes*, são atribuídos rótulos a elas baseados em suas posições, onde, no geral:

- Título: Como sempre presente no topo das páginas, é atribuído a primeira caixa delimitadora.
- Autor: Disponível abaixo do titulo, logo é atribuído a segunda caixa delimitadora.
- Resumo: Em geral é atribuído a terceira ou quarta caixa delimitadora.
- Palavras-chave: Sempre abaixo do resumo, logo é atribuído a caixa delimitadora seguinte a do resumo.

Figura 16 – *Bounding Boxes* definidas pelo modelo Tesseract.



Fonte: Autor

Esse procedimento, inicialmente se mostrou muito promissor, pois, se torna extremamente prático ser possível utilizar o mesmo modelo que faz extração do texto da

imagem para também classificar rotulando que tipo de texto esta sendo extraído. No entanto, durante a iteração de replicar esse processo por todo o conjunto de dados, foi possível notar falhas fundamentais na sua concepção enquanto metodologia, dentre as quais podemos destacar:

- Para qualquer alteração no padrão de organização das informações no artigo era necessário adicionar uma exceção e trata-lá como caso especial.
- Por vezes a definição das *bounding boxes* pelo Tesseract era imprecisa, de forma a gerar exceções desnecessárias a regra estabelecida.
- Dado o número grande de exceções, foi necessário desenvolver um *parser* dos textos para verificar se correspondiam ao rótulo atribuído. Para rótulos como resumo e palavras-chave essa verificação era simples, visto que todo artigo apresenta palavras âncoras como 'Resumo', 'Sumário', 'Abstract', 'Palavra-chave' ou 'Keywords'. Já para rótulos como título e autor, se faz necessário uma análise do conteúdo do texto para minimamente determinar se poderia se encaixar em um dos rótulos citados.

Prontamente, essa abordagem foi descontinuada, visto sua alta complexidade e especificação, favorecendo a abordagem por um modelo específico para classificação e rotulação.

3.4.2 Abordagem Faster R-CNN

O Faster R-CNN, é uma evolução sobre os modelos de R-CNN. Ele introduz uma arquitetura eficiente para localizar e classificar objetos em imagens. Sua proposta inovadora integra um mecanismo de geração de propostas (*Region Proposal Network*) diretamente na arquitetura da rede neural (REN *et al.*, 2015), otimizando assim o processo de detecção de objetos em comparação com seus predecessores.

Utilizando um modelo pré-treinado de Faster R-CNN, foi realizado um treinamento baseado em *transfer learning* para gerar um modelo robustos que atendesse as necessidades de classificação do conjunto de dados em questão. O modelo selecionado foi o **faster_rcnn_R_50_FPN_3x**, o qual apresenta as seguintes características:

- Utiliza como CNN principal a ResNet. Essa CNN é conhecida por sua capacidade de lidar com arquiteturas profundas, alcançando centenas de camadas. Além de introduzir a inovação das conexões de atalho (*Skip Connections*). Essas conexões de atalho permitem que a informação do início da rede seja transmitida diretamente para camadas mais profundas.
- Treinado sobre $\frac{1}{50}$ do *dataset* PubLayNet. Onde o PubLayNet é um grande conjunto de dados de imagens de documentos, em sua maioria artigos acadêmicos da área da

saúde, cujo layout é anotado com caixas delimitadoras e segmentações poligonais. A fonte dos documentos é o PubMed Central Open Access Subset.

3.4.2.1 Conjunto de Dados de Treinamento

O conjunto de dados utilizado para o treinamento de *transfer learning* é composto por uma fração do conjunto alvo. Onde foram anotados com coordenadas de *bounding box* cerca de 30 artigos por ano, totalizando 480 artigos anotados para o treinamento, o que refere-se a aproximadamente 25% do total do conjunto de dados.

As anotações foram feitas de forma manual utilizando o software LabelStudio, o qual disponibiliza uma interface gráfica (veja Figura 17) para interagir com a imagem e definir graficamente a região da caixa delimitadora por rótulo. No LabelStudio, para o conjunto de dados de treinamento, foram definidos 4 rótulos possíveis: Paper Title, Author, Abstract e Keywords.

Após as anotações feitas, exporta-se elas para um arquivo JSON seguindo a formatação COCO, a qual é muito utilizada como estrutura de arquivos de anotação voltados para treinamento de modelos de aprendizado de máquina.

A formatação COCO apresenta 3 principais parâmetros:

- "images": Lista de objetos JSON que representam cada imagem anotada. Cada objeto apresenta os seguintes parâmetros:
  - "id": Um identificador exclusivo para a imagem.
  - "width": A largura da imagem em pixels.
  - "height": A altura da imagem em pixels.
  - "file_name": O caminho até o arquivo da imagem no sistema de arquivo.
- "categories": Lista de objetos JSON onde cada um representa um rótulo possível. Cada objeto apresenta os seguintes parâmetros:
  - "id": Um identificador único para o rótulo.
  - "name": Nome do rótulo.
- "annotations": Lista de objetos JSON onde cada um representa uma anotação de um rótulo em uma imagem do *dataset*. Cada objeto apresenta os seguintes parâmetros:
  - "id": Identificador único para a anotação.
  - "image_id": Identificador único da imagem que contém o objeto anotado.
  - "category_id": Identificador único do rótulo do objeto anotado.
  - "bbox": Lista de quatro números que representam a caixa delimitadora do objeto anotado no formato (x, y, largura, altura), onde (x, y) é o canto superior esquerdo da caixa delimitadora.

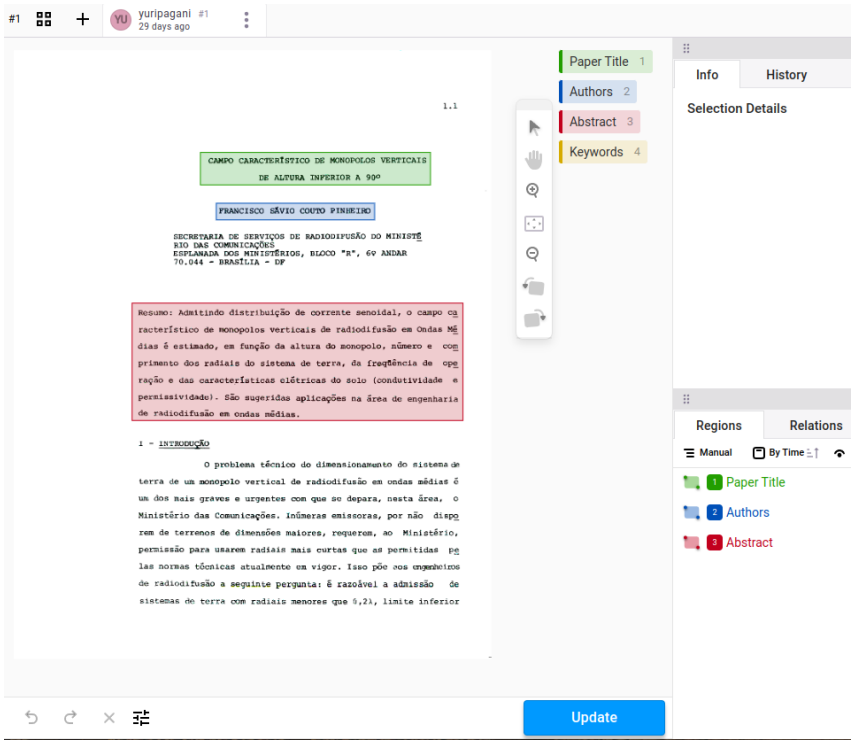
– “area”: A área da caixa delimitadora em pixels quadrados.

Algoritmo 3.6 – Representação da formatação COCO no conjunto utilizado.

```
{
    "images": [
        {"width": 2492, "height": 3177, "id": 0, "file_name": "images/SBrT_1983_001.png
            "},
        {"width": 2489, "height": 3175, "id": 1, "file_name": "images/SBrT_1983_002.png"}
    ],
    "categories": [
        {"id": 0, "name": "Abstract"},
        {"id": 1, "name": "Authors"},
        {"id": 2, "name": "Keywords"},
        {"id": 3, "name": "Paper Title"}
    ],
    "annotations": [
        {
            "id": 0,
            "image_id": 0,
            "category_id": 3,
            "bbox": [972.1967, 533.8401, 1197.8852, 169.2663],
            "area": 202761.7153319
        }
    ]
}
```

Fonte: Autor

Figura 17 – Anotação de um artigo do *dataset* feito através do LabelStudio.



Fonte: Autor

3.4.2.2 Treinamento do Modelo

O Treinamento para o modelo pré-treinado foi feito utilizando o LayoutParser. Módulo da linguagem Python que serve como um kit de ferramentas unificado para análise de imagens de documentos com base em aprendizado profundo (SHEN *et al.*, 2021).

Através do LayoutParser, é feita a interação com o Detectron2, que é um framework de código aberto para detecção e segmentação de objetos em imagens desenvolvido pelo Facebook AI Research(WU *et al.*, 2019). O Detectron2 é construído sobre o PyTorch, outro pacote da linguagem Python específico para aprendizado de máquina, e fornece uma arquitetura modular e flexível para treinar e implementar modelos de visão computacional.

Para que seja realizado o treinamento são necessários alguns pré-requisitos:

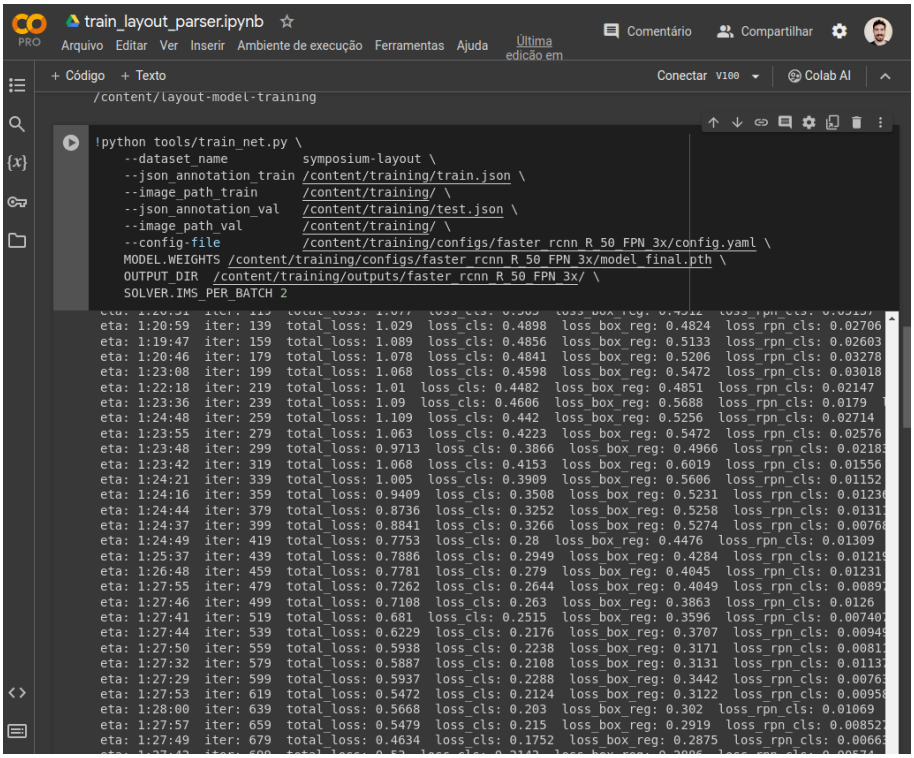
- Arquivos referente ao conjunto de dados de treinamento.
- Arquivo de anotações no formato COCO relacionado ao conjunto de dados de treinamento.
- Arquivo de pesos do modelo pré-treinado no formato PTH. (Este arquivo é conhecido como dicionário de estado PyTorch, o qual é um dicionário Python, que contém o estado de um modelo PyTorch, incluindo os pesos, tendências e outros parâmetros do modelo.)
- Arquivo de configuração do modelo pré-treinado no formato YAML.

Antes do próprio treino, o conjunto de dados foi dividido em duas partes distintas, com o propósito de designar uma porção para o treinamento do modelo e outra para sua validação. Para realizar essa divisão, um *script* Python foi empregado, o qual gerou dois arquivos no formato COCO, com base nas 480 imagens anotadas disponíveis. O primeiro arquivo compreende as anotações de 80% das imagens, sendo destinado ao treinamento do modelo, enquanto o segundo contém as anotações correspondentes aos 20% restantes das imagens, destinando-se à validação do modelo.

Dada a aquisição de todos os arquivos e configurações necessárias, foi feito o treinamento do modelo utilizando uma máquina virtual com GPU V100 do *Google Colab*.

Ao concluir o processo de treinamento (representado na Figura 18), são gerados dois arquivos essenciais para o novo modelo. Um desses arquivos armazena os pesos obtidos durante o treinamento no formato PTH, enquanto o outro contém as configurações do modelo em um formato YAML. A combinação desses dois arquivos oferece a definição completa do modelo final, pronto para ser empregado na classificação do conjunto de dados-alvo.

Figura 18 – Processo de treinamento utilizando o ambiente *Google Colab*.



Fonte: Autor

3.4.2.3 Classificação

Para realizar classificações de artigos utilizando o modelo treinado, o LayoutParser disponibiliza uma interface que facilita a interação e carregamento do modelo por meio de seu arquivo de configuração e pesos. Com isso, é possível empregar o modelo de maneira fácil e eficiente em qualquer arquivo de imagem, exigindo apenas um desenvolvimento mínimo de código. O Algoritmo 3.7 ilustra o código utilizado para interação com o modelo treinado.

Após executar a classificação com o modelo carregado, o LayoutParser fornece como saída um objeto *layout* que inclui todas as informações referentes à classificação e detecção de elementos na imagem. Dessa forma, torna-se possível selecionar os rótulos de interesse e realizar recortes específicos com base nessas informações, gerando imagens com somente o texto de interesse presente.

O LayoutParser também oferece uma função denominada *draw_box*, que facilita a visualização dos rótulos classificados e de suas respectivas regiões na imagem original. Essa funcionalidade torna o processo de verificação visual dos resultados bastante simples e acessível. A Figura 19 mostra um artigo com as caixas delimitadores e seus rótulos correspondentes determinados através do modelo treinado utilizando código presente no Algoritmo 3.7.

Algoritmo 3.7 – Código de carregamento e uso do modelo treinado.

```
import layoutparser as lp

BASE_PATH = "/content/gdrive/MyDrive/TCC"


def classifier(image):
    custom_label_map = {
        0: "Abstract",
        1: "Authors",
        2: "Keywords",
        3: "Paper Title"
    }
    model = lp.Detectron2LayoutModel(
        config_path = f"{BASE_PATH}/training/outputs/faster_rcnn_R_50_FPN_3x/config.yaml"
            ,
        model_path = f"{BASE_PATH}/training/outputs/faster_rcnn_R_50_FPN_3x/model_final.
            pth",
        extra_config = ["MODEL.ROI_HEADS.SCORE_THRESH_TEST", 0.75],
        label_map=custom_label_map
    )
    layout = model.detect(image)
    lp.draw_box(
        image,
        layout,
        box_width=7,
        id_font_size=22,
        box_alpha=0.3,
        show_element_type=True
    )
    return layout
```

Fonte: Autor

Figura 19 – Artigo rotulado utilizando modelo treinado.

Paper Title
MISTURA DE QUATRO ONDAS EM FIBRA ÓPTICA COM COMPRIMENTO DE ONDA DE ZERO DE DISPERSÃO VARIÁVEL AO LONGO DO COMPRIMENTO DA FIBRA

Authors
P.B. Harboe e J.R. Souza

Centro de Estudos em Telecomunicações - CETUC
Pontifícia Universidade Católica do Rio de Janeiro - PUC/Rio
Rua Marquês de São Vicente 225, Rio de Janeiro - RJ, cep:22453-900

Authors
Telefone: (021) 5299450
FAX: (021) 2945748
Correio Eletrônico: paula@mamao.cetuc.puc-rio.br
jrsouza@cetuc.puc-rio.br

Abstract
Resumo
Neste trabalho, o fenômeno de mistura de quatro ondas em fibra óptica é analisado considerando-se que o comprimento de onda de zero de dispersão cromática da fibra é variável ao longo de seu comprimento. Os resultados obtidos indicam que uma nova frequência óptica é eficientemente gerada para combinações particulares de comprimento de onda dos sinais de entrada, embora a condição de casamento de fase não seja completamente satisfeita.

Abstract
Abstract
This paper presents an analysis of the four-wave mixing phenomenon in single mode optical fibres considering that the zero dispersion wavelength varies along the fibre. The results indicate that new optical frequencies are efficiently generated for specific combinations of the input signal wavelengths, even when the phase matching condition is not totally satisfied.

1. Introdução

O aprimoramento da tecnologia de fibras ópticas de silica fundida, o desenvolvimento de lasers semicondutores de alta potência óptica e a utilização de técnicas de deteção coerente possibilitam, por um lado, um aumento significativo na taxa de transmissão de dados embora, por outro lado, levam a uma limitação operacional nos sistemas de comunicação óptica (SCO) avançados, devido a efeitos não lineares na fibra [1]. As não linearidades ópticas resultam em interferência, distorção e atenuação excessiva dos sinais ópticos, degradando o desempenho dos sistemas. Seus efeitos podem ser minimizados através de um projeto adequado do sistema.

Para explorar a ampla largura de banda oferecida pelas fibras ópticas, um grande número de canais, em diferentes comprimentos de onda são multiplexados em uma mesma fibra (SCO/WDM) [2]. Neste contexto, o processo de mistura de quatro ondas (FWM) é o fenômeno não linear dominante e constitui objeto de estudo neste trabalho.

A mistura de quatro ondas é um efeito não linear de terceira ordem onde duas ou mais ondas ópticas, de diferentes comprimentos de onda, se misturam para produzir outras ondas ópticas em diferentes comprimentos de onda. Em SCO/WDM com espaçamento uniforme entre canais, as novas ondas ópticas geradas podem coincidir com algumas das frequências dos canais, resultando em forte crosstalk (transferência de potência entre canais). Seu impacto é mais severo em sistemas WDM coerentes nos quais o espaçamento entre canais é tipicamente de 1 a 10 GHz.

A eficiência de geração da nova onda óptica depende fortemente de que a condição de casamento de fase seja satisfeita, para o que a dispersão cromática da fibra é um parâmetro relevante. Recentemente,

47

Fonte: Autor

3.5 EXTRAÇÃO DOS TEXTOS

Após obter os rótulos e suas caixas delimitadoras correspondentes, procede-se com o recorte de cada elemento, adicionando margens conforme necessário para criar sub-imagens. Essas sub-imagens são então fornecidas ao modelo Tesseract para a extração do texto.

O LayoutParser também disponibiliza uma interface que facilita a interação com o modelo Tesseract, tornando o código necessário para a extração de texto mais intuitivo. São especificadas as línguas inglês e português para o Tesseract, uma vez que os artigos podem estar redigidos em ambas essas línguas. O Algoritmo 3.8 ilustra o código utilizado no processo de extração de todas as classes de textos-chave.

Algoritmo 3.8 – Código para extração de texto dada as sub-imagens que cada rótulo.

```
import layoutparser as lp

def extract_text(image, model):
    def get_block_text(block, image):
        segment_image = block.pad(left=5, right=5, top=5, bottom=5).crop_image(image)
        ocr_agent = lp.TesseractAgent(languages="eng+por")
        text = ocr_agent.detect(segment_image)
        return text

    paper_data = {
      "title": None,
      "authors": None,
      "abstract": None,
      "keywords": None
    }
    layout = model.detect(image)
    paper_title_box = lp.Layout([b for b in layout if b.type=="Paper Title"])
    authors_boxes = lp.Layout([b for b in layout if b.type=="Authors"])
    abstract_boxes = lp.Layout([b for b in layout if b.type=="Abstract"])
    keyword_box = lp.Layout([b for b in layout if b.type=="Keyword"])

    # ———— Title PARSER ————
    for block in paper_title_box:
        paper_data["title"] = get_block_text(block, image).replace("\n\x0c", "")

    # ———— Authors PARSER ————
    authors_list = list()
    for block in authors_boxes:
        authors_list.append(get_block_text(block, image).replace("\n\x0c", ""))
    paper_data["authors"] = authors_list

    # ———— Abstract PARSER ————
    abstract_list = list()
    for block in abstract_boxes:
        abstract_list.append(get_block_text(block, image).replace("\n\x0c", ""))
    paper_data["abstract"] = abstract_list[0]

    # ———— Keywords PARSER ————
    for block in keyword_box:
        paper_data["keywords"] = get_block_text(block, image).replace("\n\x0c", "")
    return paper_data
```

Fonte: Autor

Figura 20 – Texto extraído de cada *bounding box* presente em 19

```
---- Paper Title ----
MISTURA DE QUATRO ONDAS EM FIBRA OPTICA
COM COMPRIMENTO DE ONDA DE ZERO DE DISPERSAO
VARIAVEL AO LONGO DO COMPRIMENTO DA FIBRA

---- Abstract ----
Resumo

Neste trabalho, o fendmeno de mistura de quatro ondas em fibra Optica é analisado considerando-se
que o comprimento de onda de zero de dispers4o cromatica da fibra é variavel ao longo de seu
comprimento. Os resultados obtidos indicam que uma nova frequéncia optica é eficientemente gerada
para combinages particulares de comprimento de onda dos sinais de entrada, embora a condi¢ao de
casamento de fase nao seja completamente satisfeita.

---- Authors ----
PB. Harboe e J.R. Souza

---- Abstract ----
Abstract

This paper presents an analysis of the four-wave mixing phenomenon in single mode optical fibres
considering that the zero dispersion wavelength varies along the fibre. The results indicate that new
optical frequencies are efficiently generated for specific combinations of the input signal wavelengths,
even when the phase matching condition is not totally satisfied.

---- Authors ----
Telefone: (021): 5299450
```

Fonte: Autor

4 RESULTADOS

4.1 MÉTRICAS

A avaliação do desempenho dos modelos utilizados é crucial para garantir a precisão na extração de texto dos artigos. Diversas métricas são empregadas para medir a acurácia desses modelos, sendo que, para o modelo de classificação, as métricas já são são tipicamente dadas no próprio treinamento do mesmo. Logo, são apresentadas algumas métricas fundamentais para OCR, incluindo *Word Error Rate* (WER), *Character Error Rate* (CER), Distância de Levenshtein e Distância de Jaro-Winkler.

Para calcular as métricas mencionadas, foi imprescindível realizar a transcrição manual de um conjunto de artigos, estabelecendo-os como resultados esperados. Como parte desse processo, 10 artigos por ano foram transcritos manualmente, totalizando cerca de 160 artigos transcritos no conjunto completo. Essas transcrições foram documentadas em arquivos YAML, organizados de acordo com o ano correspondente.

4.1.1 Word Error Rate

A WER é uma métrica que quantifica a discrepância entre as palavras reconhecidas pelo modelo e as palavras reais. Ela considera inserções, exclusões e substituições de palavras, fornecendo uma visão global da precisão do OCR em nível de palavra. A fórmula para calcular a Word Error Rate é dada por:

$$WER = \frac{S + D + I}{N} \times 100 \tag{4}$$

Onde: S é o número de palavras substituídas; D é o número de palavras deletadas; I é o número de palavras inseridas; N é o número total de palavras na transcrição de referência.

4.1.2 Character Error Rate

Ao contrário da WER, a CER opera no nível de caractere, medindo a taxa de erros de caracteres entre o texto previsto e o texto de referência. Ela é valiosa para identificar imprecisões específicas nos caracteres extraídos. De forma semelhante a WER, a formula para CER é dada por:

$$CER = \frac{S + D + I}{N} \times 100 \tag{5}$$

Onde: S é o número de caracteres substituídas; D é o número de caracteres deletadas; I é o número de caracteres inseridas; N é o número total de caracteres na transcrição de referência.

4.1.3 Distância de Levenshtein

É uma medida quantitativa da diferença entre duas sequências de caracteres. Ela calcula o número mínimo de operações a nível de caracteres (inserções, exclusões, substituições) necessárias para converter uma sequência na outra. Quanto menor a distância, maior é a similaridade entre as sequências. A fórmula para distância de Levenshtein é dada por:

$$Lev_{a,b}(i,j) = \begin{cases} max(i,j) & \text{se } \ min(i,j) = 0, \\ min \begin{cases} Lev_{a,b}(i-1,j) + 1 \\ Lev_{a,b}(i,j-1) + 1 \\ Lev_{a,b}(i-,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{senão} \end{cases} \tag{6}$$

Sendo a e b as sequencias de caracteres e i e j os correspondentes índices de caractere para cada sequencia.

4.1.4 Distância de Jaro-Winkler

Métrica de similaridade entre duas sequências de caracteres, levando em conta o comprimento comum inicial das sequências. Ela atribui pesos maiores a correspondências próximas ao início das sequências, refletindo a probabilidade de erro em OCR. A fórmula para distância de Jaro-Winkler é:

$$JaroWinkler(a,b) = Jaro(a,b) + (\lambda \times P \times L \times (1 - Jaro(a,b)) \tag{7}$$

Onde: $Jaro(a,b)$ é a é a medida de similaridade de Jaro entre as sequencias a e b; λ é um fator de escala; P é o comprimento do prefixo comum entre a e b; L é o comprimento do prefixo comum após o ajuste para o fator λ.

$$Jaro(a,b) = \frac{m}{max(\text{comprimento } \ a, \text{comprimento } \ b)} \tag{8}$$

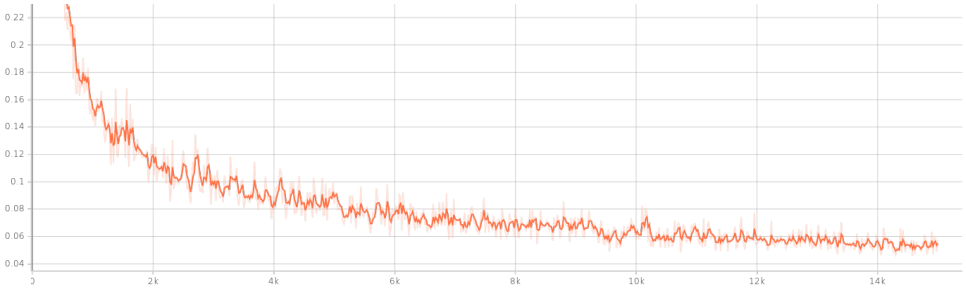
Sendo m o número de caracteres que coincidem, contados uma vez, até a metade do comprimento da sequência mais curta.

4.2 CLASSIFICADOR

O treinamento do modelo de classificação foi conduzido ao longo de aproximadamente 2 horas e 30 minutos. Durante todo esse período, não foram observadas discrepâncias significativas ao otimizar as funções custo associadas, seja para a acurácia de rotulação ou para a acurácia na determinação de caixas delimitadoras. O processo transcorreu de maneira consistente, demonstrando estabilidade e coerência nas métricas de avaliação ao longo do treinamento.

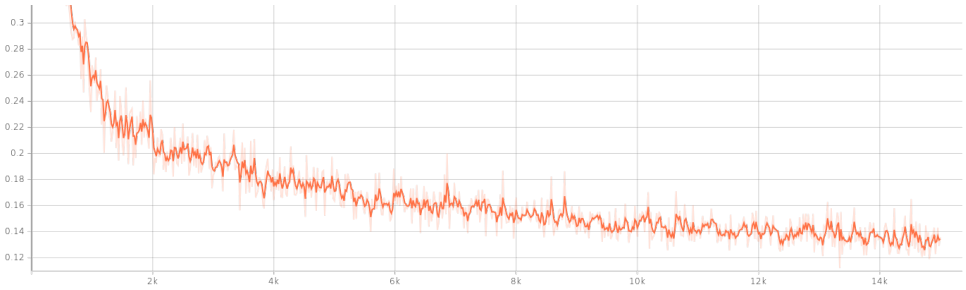
Ao fim do treinamento, obteve-se os valores para acurácia média apresentados na Tabela 2.

Figura 21 – Valor da função custo (*Binary Cross Entropy*) para rotulação durante o período das N iterações.



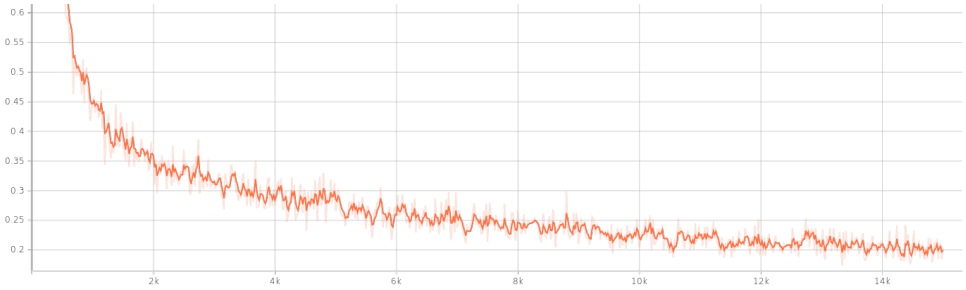
Fonte: Autor

Figura 22 – Valor da função custo (*L1*) para delimitação de *bounding box* durante o período das N iterações.



Fonte: Autor

Figura 23 – Valor da função custo total durante o período das N iterações.



Fonte: Autor

Título	Autor	Resumo	Palavras-Chave
71,8%	49,3%	72,2%	0%

Tabela 2 – Acurácia média por classe.

Fonte: Autor

Nota-se que os rótulos referentes ao título e resumo apresentaram as acurácias mais

elevadas. Esse fenômeno pode ser atribuído à sua posição relativamente estável dentro da estrutura do artigo, com poucas variações. Além disso, esses elementos ocupam uma parte significativa da página do artigo. O título, consistentemente localizado na parte superior e central da página, é sempre o primeiro elemento visível. Por outro lado, o resumo, geralmente um parágrafo extenso, encontra-se frequentemente na região central da página, entre a introdução e os autores, ou como o primeiro parágrafo na coluna à esquerda de texto. Essa consistência na posição desses elementos contribui para a obtenção de acurácias mais altas durante o processo de classificação.

A baixa acurácia observada para o rótulo de autor era antecipada. Esses elementos são frequentemente pequenos e exibem uma ampla variação em termos de posição e organização dentro do artigo. Essa diversidade dificulta a delimitação precisa da região contendo exclusivamente o nome ou nomes dos autores do artigo.

Para o rótulo de palavras-chave, também era esperada uma acurácia próxima de zero. Isso se deve ao fato de que no conjunto de dados de treinamento, apenas dois artigos tinham palavras-chave anotadas. Além disso, ao considerar o conjunto de dados transcritos, menos de 10 artigos incluíam informações de palavras-chave. Essa escassez de exemplos no conjunto de treinamento tornou dificultoso para o modelo aprender efetivamente a identificar e rotular corretamente as palavras-chave nos artigos.Na Tabela 3 é demonstrado o erro de incompatibilidade[1], o qual corrobora com o levantamento da acurácia de classificação até então.

Ano	Título	Autor	Resumo	Palavras-Chave
1983	0	0	2	10
1984	0	0	0	10
1985	0	0	0	10
1986	0	0	0	10
1987	0	0	0	10
1988	0	0	0	10
1989	0	0	1	10
1990	0	0	0	10
1991	0	1	0	10
1992	0	1	1	10
1993	0	0	0	10
1994	0	0	0	10
1995	0	0	1	10
1996	0	0	0	10
1997	0	0	0	10
1998	0	2	1	10

Tabela 3 – Quantidade de erros de incompatibilidade entre artigos transcritos e a classificação do modelo.

Fonte: Autor

[1] É considerado um erro de incompatibilidade quando é identificado um texto-chave onde não há, ou então, quando não é identificado um texto-chave mas ele esta presente.

4.3 OCR

Primeiramente, antes de qualquer analise criteriosa de acurácia, é interessante realizar o levantamento do tempo de processamento necessário para a realização da extração dos textos-chave para todos os artigos. Os tempos obtidos estão descritos na Tabela 4.

Ano	Tempo de Processamento	Total de Artigos	Tempo Médio por Artigo
1983	291s	64	4,54s
1984	342s	74	4,62s
1985	151s	40	3.77s
1986	207s	45	4.60s
1987	380s	68	5.58s
1988	328s	58	5.65s
1989	560s	115	4.86s
1990	520s	116	4.48s
1991	558s	93	6.00s
1992	640s	123	5.20s
1993	862s	153	5.63s
1994	548s	102	5.37s
1995	655s	128	5.11s
1996	851s	150	5.67s
1997	934s	128	7.29s
1998	987s	144	6.85s

Tabela 4 – Tempo de execução para o processo completo sobre o *dataset* alvo.

Fonte: Autor

No contexto do conjunto de dados alvo, composto por aproximadamente 1601 artigos, o tempo total de execução do processo, incluindo classificação e extração, foi de 8314 segundos, equivalente a cerca de 2 horas e 20 minutos. Isso resulta em uma média de processamento de aproximadamente 5,19 segundos por artigo.

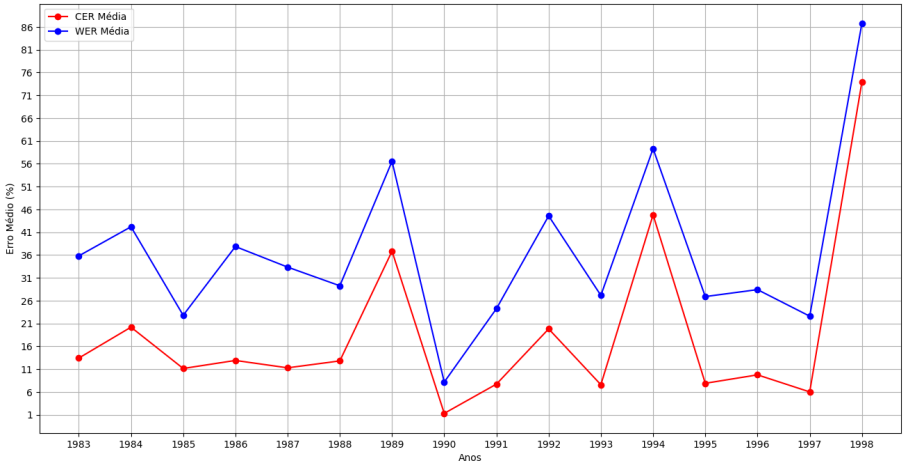
Para efeito de comparação, ao realizar a transcrição dos 160 artigos, o autor registrou uma média de aproximadamente 4 minutos por artigo. A uma taxa constante, seriam necessárias mais de 100 horas de esforço humano para concluir a transcrição do conjunto de dados alvo.

4.3.1 Título

Quanto ao título, observa-se que as porcentagens de erro de palavras e caracteres estão diretamente relacionadas à sua natureza, caracterizada pela presença de um número limitado de palavras e caracteres. Em termos gerais, o classificador consegue identificar consistentemente a região do título. Além disso, é comum que o título seja apresentado em uma fonte maior em comparação com o restante do texto no artigo. Essa distinção visual contribui para a localização precisa do título. No entanto, a baixa quantidade de palavras e caracteres no título amplifica as porcentagens de erro quando ocorrem discrepâncias,

visto que cada erro tem um impacto maior em relação ao tamanho total do título. As curvas apresentadas na Figura 24 mostram o CER e WER de título obtidos para cada ano do evento.

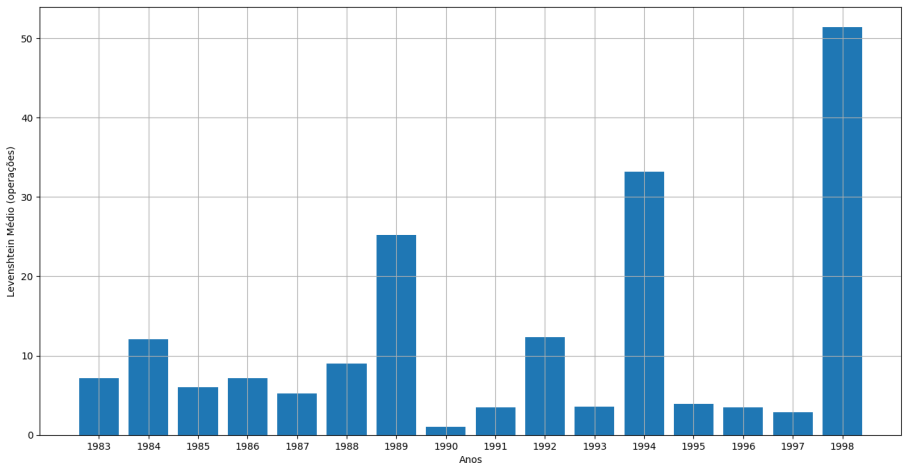
Figura 24 – Porcentagem média de CER e WER para o rótulo de título por ano.



Fonte: Autor

Confirma-se que as métricas CER e WER podem estar amplificadas, especialmente ao considerarmos o número de operações necessárias para transformar o texto extraído no texto de validação, dado apresentado na Figura 25 que mostra a distância de Levenshtein para o título. Nota-se que, para vários anos, o número de operações é inferior a 10, indicando um alto grau de semelhança entre os textos, uma vez que, por exemplo, a média de caracteres para o título no conjunto de dados transcritos é 83 caracteres. Essa constatação sugere que, embora as métricas possam indicar uma certa taxa de erro, a natureza desses erros pode ser relativamente pequena em termos absolutos, sugerindo uma boa concordância entre o texto extraído e o texto de validação. Portanto, a interpretação das métricas CER e WER deve ser feita considerando o contexto e a escala das operações de correção necessárias.

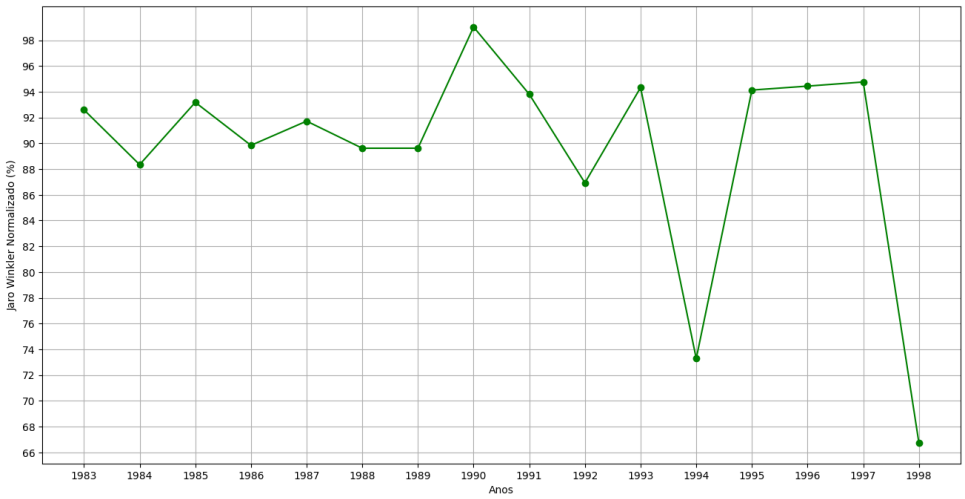
Figura 25 – Distância de Levenshtein para rótulo de título por ano.



Fonte: Autor

A observação de que os textos, em geral, exibem um alto grau de semelhança é reforçada ao calcular a Distância de Jaro-Winkler (apresentado na Figura 26 para o título), uma métrica projetada para avaliar precisamente o nível de semelhança entre dois textos. O fato de que a distância calculada é relativamente pequena destaca a proximidade entre os textos em consideração. Tal distância é normalizada para variar entre 0 e 1, onde 0 representa uma diferença total entre os textos, e 1 indica que os textos são idênticos.

Figura 26 – Distância de Jaro-Winkler normalizada para rótulo de título por ano.



Fonte: Autor

Podemos observar na Figura 26 que nos anos de 1994 e 1998 há uma discrepância nos resultados, atribuída principalmente ao processo de digitalização dos artigos desses anos. No caso de 1994, os caracteres são pequenos e estão muito próximos, o que dificulta

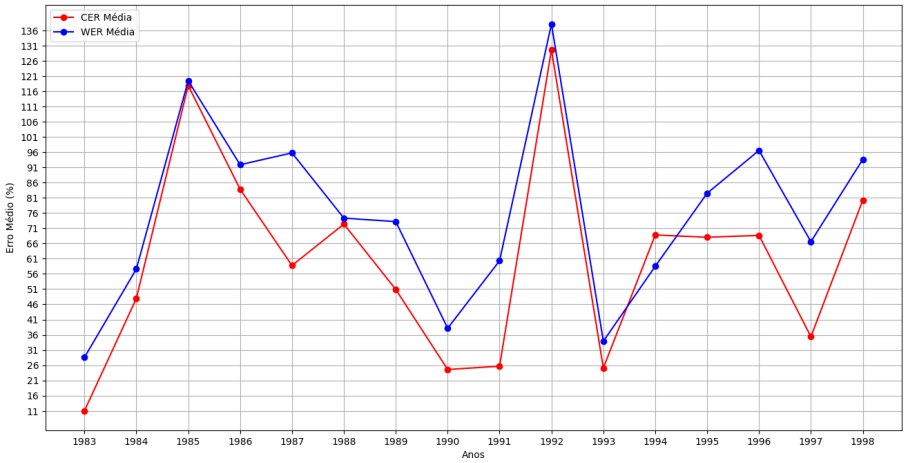
a legibilidade. Em relação a 1998, os artigos foram escaneados de um livro referente ao simpósio, resultando em uma digitalização que não atingiu a melhor qualidade possível. Essas condições adversas na digitalização impactam diretamente na qualidade do texto extraído, contribuindo para as discrepâncias observadas nos resultados desses anos específicos.

4.3.2 Autor

No contexto do rótulo de autor, observa-se uma situação agravante do que ocorre no rótulo de título. Isso acontece, em parte, devido ao fato de que, em geral, os nomes de autores tendem a ter menos palavras e caracteres do que os títulos. Além disso, o classificador apresenta uma acurácia inferior na identificação de regiões contendo nomes de autores, tornando mais provável a classificação equivocada dessas regiões.

Essa afirmação pode ser corroborada pelo Figura 27 que exibe as métricas CER e WER. Os valores acima de 100% nesse gráfico só podem ser justificados por meio da comparação de textos totalmente distintos, ou seja, o numero de operações (inserção, deleção e substituição) é maior que o numero total de caracteres presente no texto transcrito.

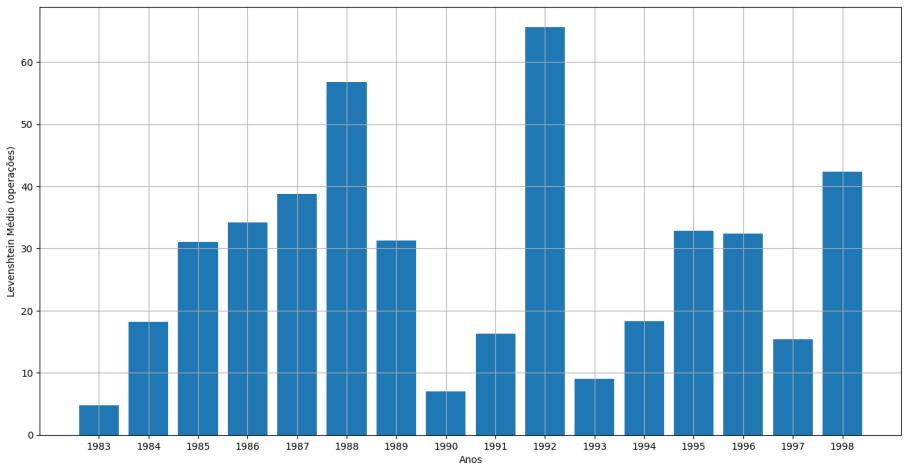
Figura 27 – Porcentagem média de CER e WER para o rótulo de autor por ano.



Fonte: Autor

Os valores da distância de Levenshtein (veja Figura 28) podem parecer baixos devido ao reduzido número de caracteres associados aos nomes dos autores. Para os valores mais elevados, como no caso do ano de 1992, isso sugere uma possível classificação incorreta da região, resultando na necessidade de todas as operações possíveis para gerar o texto corrigido.

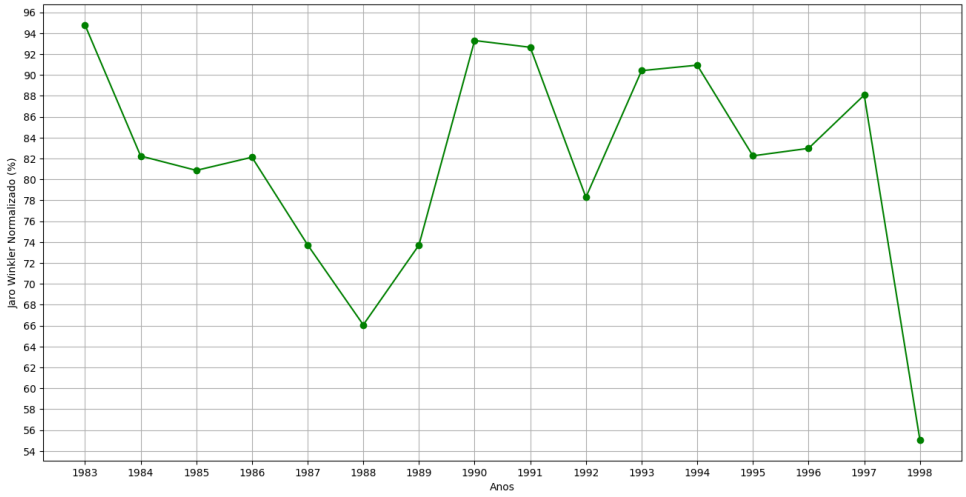
Figura 28 – Distância de Levenshtein para rótulo de autor por ano.



Fonte: Autor

Verifica-se na Figura 29 que, em comparação com o título, há uma média menor de semelhança para os textos que contêm nomes de autores. Isso se deve principalmente à dificuldade na classificação precisa da região que abrange os nomes dos autores. A complexidade associada à identificação correta dessas regiões resulta em uma menor concordância entre os textos extraídos e os textos de validação.

Figura 29 – Distância de Jaro-Winkler normalizada para rótulo de autor por ano.



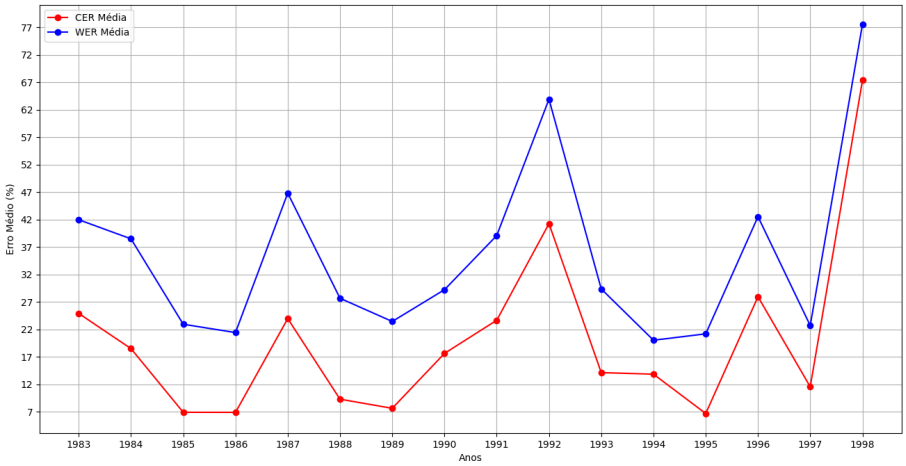
Fonte: Autor

4.3.3 Resumo

Para o rótulo de resumo, observamos na Figura 30 valores semelhantes de CER e WER médio em comparação com o rótulo de título. No entanto, neste caso, os resumos

apresentam uma quantidade muito maior de caracteres e palavras. Isso torna a CER (Character Error Rate) e a WER (Word Error Rate) métricas mais confiáveis para determinar o grau de paridade entre os textos.

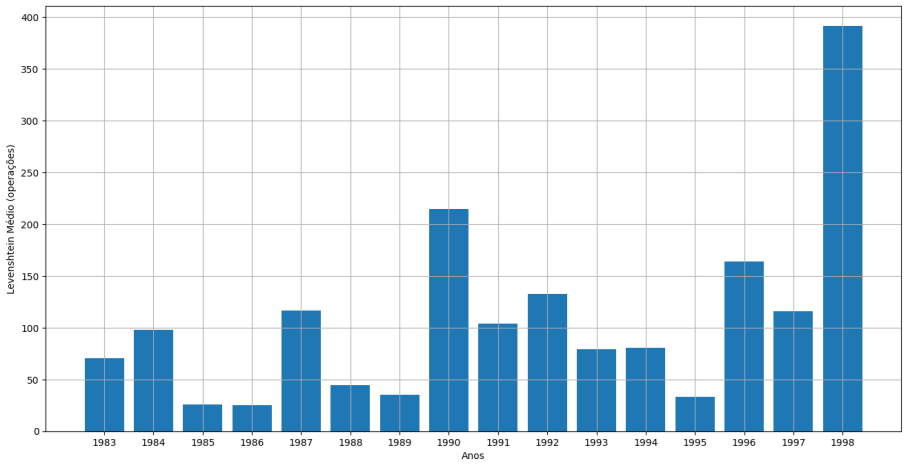
Figura 30 – Porcentagem média de CER e WER para o rótulo de resumo por ano.



Fonte: Autor

Ao observar os valores da distância de Levenshtein, apresentados na Figura 31, nota-se uma média significativamente maior de operações necessárias para o rótulo de resumos. Isso ocorre devido à natureza expansiva dos resumos, que contêm um grande número de caracteres. Portanto, para valores de Levenshtein abaixo de 50 operações, podemos considerar isso como um indicador de um alto grau de semelhança entre os textos.

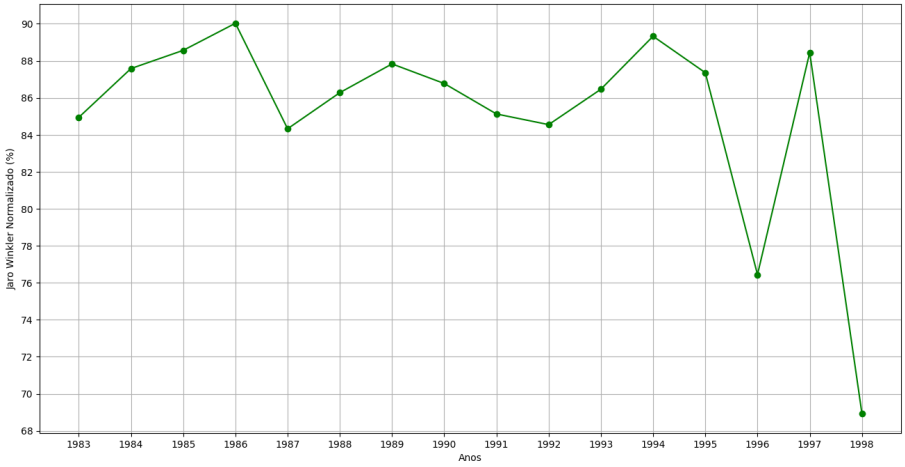
Figura 31 – Distância de Levenshtein para rótulo de resumo por ano.



Fonte: Autor

No rótulo de resumos, através da curva ilustrada na Figura 32, observamos a maior média de semelhança entre os textos. Isso pode ser atribuído à alta acurácia do classificador na identificação da região que contém os resumos. Além disso, a quantidade substancial de palavras nos resumos contribui para diluir o impacto de erros individuais, favorecendo uma maior concordância entre os textos extraídos e os textos de validação. Essa combinação de precisão na classificação e a densidade de palavras nos resumos resulta em uma métrica mais alta de semelhança em comparação com outros rótulos.

Figura 32 – Distância de Jaro-Winkler normalizada para rótulo de resumo por ano.



Fonte: Autor

4.3.4 Palavras-chave

Devido à baixa ocorrência do rótulo de palavras-chave no conjunto de dados alvo e à sua ausência durante o treinamento do modelo de classificação, resultando em um erro total para esse rótulo, não foi possível obter métricas de resultado para as palavras-chave. A escassez de exemplos durante o treinamento tornou desafiador para o modelo aprender efetivamente a identificar e rotular corretamente as palavras-chave nos artigos. Como resultado, não há informações disponíveis para avaliar o desempenho do modelo nesse aspecto específico.

5 CONCLUSÃO

Ao longo deste trabalho, foi explorada a aplicação de modelos de Classificação e Detecção de Objetos e de Reconhecimento Óptico de Caracteres (OCR) na tarefa de extrair e classificar textos-chave de artigos acadêmicos. Os resultados obtidos revelam uma variedade de desafios e sucessos, influenciados por fatores como a qualidade da digitalização, o tamanho e a estrutura dos elementos textuais.

Observou-se que o modelo Faster R-CNN demonstrou maior eficácia na classificação de títulos e resumos, onde a consistência na posição e formatação desses elementos facilitou a identificação precisa. No entanto, houve obstáculos significativos ao lidar com nomes de autores e palavras-chave, destacando a sensibilidade do modelo a características específicas da estrutura dos documentos.

A análise das métricas, como a Distância de Levenshtein, CER e WER, proporcionou uma visão detalhada da qualidade das predições do modelo. Notou-se que, para resumos, a quantidade substancial de caracteres contribuiu para métricas mais confiáveis, enquanto a baixa ocorrência de palavras-chave dificultou a avaliação correspondente.

Em comparação com o título, a média de semelhança para os textos que contêm nomes de autores mostrou-se menor, evidenciando a complexidade associada à classificação precisa dessas regiões. As dificuldades aumentaram em anos específicos devido a desafios na digitalização.

Esses resultados ressaltam a importância da qualidade dos dados de treinamento, bem como a necessidade contínua de adaptação do modelo para lidar com características específicas de diferentes conjuntos de dados. Em última análise, a automação da transcrição e classificação de documentos acadêmicos oferece uma perspectiva inovadora, alinhada com a transformação digital necessária para superar os desafios do acesso eficiente ao vasto acervo de informações históricas. Contudo, é crucial considerar a diversidade e complexidade desses documentos para garantir a robustez e precisão do processo de extração e classificação.

REFERÊNCIAS

ALI, Amjad *et al.* Demystifying CNN with Mathematical Insights: A Prelude with Application to an AI-based Sustainable Solution for Diabetic Retinopathy Diagnosis, set. 2023. DOI: 10.21203/rs.3.rs-3338196/v1.

ALZUBAIDI, Laith *et al.* Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. **Journal of big Data**, Springer, v. 8, p. 1–74, 2021.

BISHOP, Christopher M. Neural networks for pattern recognition. **Oxford university press**, 1995.

BYERS, Benjamin; SHETA, Alaa. Design of Convolutional Neural Networks for Fish Recognition and Tracking. v. 22, p. 1–9, set. 2023.

FU, Zihao *et al.* **On the Effectiveness of Parameter-Efficient Fine-Tuning**. [*S.l.*: *s.n.*], 2022. arXiv: 2211.15583 [cs.CL].

GIRSHICK, Ross *et al.* **Rich feature hierarchies for accurate object detection and semantic segmentation**. [*S.l.*: *s.n.*], 2014. arXiv: 1311.2524 [cs.CV].

HOCHREITER, Sepp. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. **International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems**, v. 6, p. 107–116, abr. 1998. DOI: 10.1142/S0218488598000094.

J., Ryan T. J. **LSTMs Explained: A Complete, Technically Accurate, Conceptual Guide with Keras**. [*S.l.*: *s.n.*], 2020. https://shorturl.at/ckuIY. Accessed: Date Accessed.

KOECH, Kiprono Elijah. **Softmax Activation Function — How It Actually Works**. [*S.l.*: *s.n.*], 2020. https://towardsdatascience.com/softmax-activation-function-how-it-actually-works-d292d335bd78. Accessed: Date Accessed.

LI, Zhizhong; HOIEM, Derek. **Learning without Forgetting**. [*S.l.*: *s.n.*], 2017. arXiv: 1606.09282 [cs.CV].

MEMON, Jamshed *et al.* Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). **IEEE Access**, p. 1–1, jul. 2020. DOI: 10.1109/ACCESS.2020.3012542.

NAUTA, Gerhard Jan; HEUVEL, Wietske van den; TEUNISSE, S. **Survey report on digitisation in european cultural heritage institutions 2015**. [*S.l.*]: Europeana, 2015.

NUNDLOLL, Vatsala *et al.* Automating the extraction of information from a historical text and building a linked data model for the domain of ecology and conservation science. **Heliyon**, v. 8, e10710, out. 2022. DOI: 10.1016/j.heliyon.2022.e10710.

O'SHEA, Keiron; NASH, Ryan. An introduction to convolutional neural networks. **arXiv preprint arXiv:1511.08458**, 2015.

OLAH, Christopher. **Understanding LSTM Networks**. [*S.l.*: *s.n.*], 2015. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: Date Accessed.

REN, Shaoqing *et al.* **Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks**. [*S.l.*: *s.n.*], 2015. P. 91–99.

REZATOFIGHI, Hamid *et al.* **Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression**. [*S.l.*: *s.n.*], 2019. arXiv: 1902.09630 [cs.CV].

SHEN, Zejiang *et al.* LayoutParser: A Unified Toolkit for Deep Learning Based Document Image Analysis. **arXiv preprint arXiv:2103.15348**, 2021.

SMITH, R. An Overview of the Tesseract OCR Engine. *In:* p. 629–633. DOI: 10.1109/ICDAR.2007.4376991.

UIJLINGS, Jasper *et al.* Selective Search for Object Recognition. **International Journal of Computer Vision**, v. 104, p. 154–171, set. 2013. DOI: 10.1007/s11263-013-0620-5.

WU, Yuxin *et al.* **Detectron2**. [*S.l.*: *s.n.*], 2019. https://github.com/facebookresearch/detectron2.

XU, Chunyu; WANG, Hong. Research on a Convolution Kernel Initialization Method for Speeding Up the Convergence of CNN. **Applied Sciences**, v. 12, p. 633, jan. 2022. DOI: 10.3390/app12020633.

YU, Ke; KIM, Minguk; CHOI, Jun Rim. Memory-Tree Based Design of Optical Character Recognition in FPGA. **Electronics**, v. 12, n. 3, 2023. ISSN 2079-9292. DOI: 10.3390/electronics12030754. Disponível em: https://www.mdpi.com/2079-9292/12/3/754.

ZHUANG, Fuzhen *et al.* **A Comprehensive Survey on Transfer Learning**. [*S.l.*: *s.n.*], 2020. arXiv: 1911.02685 [cs.LG].