Uma abordagem baseada em Aprendizagem de Máquina e Grafos para Segmentação de Páginas

Ana Lúcia Lima Marreiros Maia

Tese apresentada ao
Instituto de Matemática e Estatística
da Universidade de São Paulo
para obtenção do título de
Doutora em Ciências

Programa: Ciência da Computação
Orientadora: Nina S. T. Hirata

Durante o desenvolvimento deste trabalho a autora recebeu auxílio financeiro
da Universidade Estadual de Feira de Santana

São Paulo
Maio de 2023

Uma abordagem baseada em Aprendizagem de Máquina e Grafos para Segmentação de Páginas

Ana Lúcia Lima Marreiros Maia

Esta versão da tese contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 31 de Maio de 2023.

Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão julgadora:

Profª. Drª. Nina S. T. Hirata (orientadora) – IME-USP
Prof. Dr. Byron Leite Dantas Bezerra – UPE
Prof. Dr. David Menotti Gomes – UFPR
Profª. Drª. Fátima Nelsizeuma Sombra de Medeiros – UFC
Prof. Dr. Roberto Marcondes Cesar Junior – IME-USP

Aos meus pais, ***João Luiz e Lúcia****,*
os responsáveis por tudo o que sou e conquistei.

"Ensina à criança o caminho que ela deve seguir;
mesmo quando envelhecer, dele não há de se afastar."
(Provérbios 22:6)

Agradecimentos

É muito difícil começar a escrever estes agradecimentos, porque durante este longo caminho no meu Doutorado, eu sou grata a muitas pessoas.

Eu seria injusta se não começasse estes agradecimentos pela minha orientadora **Nina Hirata**, que além de tudo mais foi um porto seguro. Ela nunca desistiu de mim, mesmo quando eu mesma me desacreditava. Meu desejo como professora é um dia poder ser para os meus alunos um pouquinho do que a Nina é para mim: um exemplo de pessoa e profissional. Tenho muito orgulho de ser sua orientanda! Nunca terei palavras o suficiente para agradecer TUDO o que você fez por mim. Muito obrigada!

Meus Pais, **Zinhão e Lucinha** que, mesmo com toda a simplicidade, sempre me ensinaram os valores mais preciosos pra mim: fé, caráter, honestidade e muitos outros que nem consigo listar. Eu sempre serei devedora a vocês de tudo o que sou, porque vocês nunca mediram esforços para nos proporcionar a melhor educação, e nunca deixaram faltar nada para nós (eu e minhas irmãs). Nada mesmo, e muitas vezes, até hoje, abdicando de vocês mesmos por nós. Muito obrigada por todo o Amor, orações e tudo mais!

Por falar nas minhas irmãs, elas são tão diferentes e me completam cada uma do seu jeitinho, uma mais com o coração e a outra mais com a razão. A **Paula**, com seu jeitinho sensível e preocupada, está sempre torcendo por mim, vibrando a cada pedacinho de vitória, e mesmo distante eu sempre a sinto perto. Quando pequena era meu ídolo, depois crescemos, tivemos nossas diferenças e nos reaproximamos de uma maneira tão profunda e verdadeira, com a minha ida a São Paulo para cursar o doutorado, que nos tornamos inseparáveis. Eu amo fazer parte da sua vida e amo que você sempre fez e faz parte da minha. Ela é o coração, muito embora muitas vezes me dê doses muito boas de razão. A **Patty**, que sempre teve a fama de ser mais quieta e fechada, sempre me fez sentir muito amada, com as suas palavras sempre certeiras. Parece que ela consegue enxergar através de mim, e mesmo morando a muitos quilômetros de distância, adivinha quando eu preciso daquela exata. Ela é a razão, muito embora, consigamos passar horas e horas, sem perceber, rindo e falando sobre os assuntos mais variados, sem contar nas inúmeras dicas de filmes

de romance água com açúcar que adoramos. Amo cada uma de vocês do jeitinho que vocês são! Muito obrigada por serem sempre a minha força!

O que falar das minhas três pedrinhas preciosas, minha Pipoquinha **Amanda**, meu Príncipe Holandês **Liam** e minha Pequena **Lara**? A Amanda eu acompanho desde a notícia da gravidez da Paula: gravidez, nascimento e desde que eu olhei pra você ainda na maternidade, há apenas algumas horinhas do seu nascimento. Você trouxe um colorido muito especial pra minha vida e me fez experimentar um amor que eu nunca tinha sentido. Amor que me fez aprender a fazer artesanato para deixar a sua festinha de aniversário mais bonita, amor que sempre me faz esquecer o cansaço e me jogar no chão pra brincar com você, amor que não tem medida. A titia ama você demais e sempre estará aqui pra você! O Liam e a Lara moram do outro lado do oceano e, mesmo não tendo tido tanto convívio com eles, eu sinto o mesmo amor e sempre que posso tento me fazer presente em ligações de vídeo, em que eu aprendo palavras em holandês pra conversar com o Liam Isso nem é necessário, porque o Liam é a doçura em forma de criança, ele que já aprendeu que a parte brasileira da família mora longe, nos faz participar da vidinha dele trazendo tudo próximo ao telefone para que não deixemos de participar. E quando ele me chama de Tia Ana com o sotaque mais lindo do mundo? O coração da Titia derrete. A Pequena Lara que pude conhecer e conviver um pouquinho há pouco tempo. Que menina linda e cheia de personalidade. Não deixem ela com fome e sono que ela mostra toda a potência dos seus pulmões. Ela ainda não consegue se expressar em palavras do alto dos seus 6 meses de vida, mas sempre sorri quando eu falo com ela. Será ilusão de titia coruja? Muito obrigada a vocês três pelas cores que vocês trouxeram pra minha vida, que muitas vezes me fizeram esquecer os períodos cinzentos. Amo vocês, meus Pequenos!

Thiago, meu companheiro nesta e em muitas jornadas, muitas difíceis, nesses recém completados vinte anos de que nos vimos pela primeira vez. Meu parceiro, meu melhor amigo, meu Amor. Eu costumo dizer que ele é meu maior fã, porque sempre me coloca num lugar de admiração que eu não sei se mereço. Mas que ele me faz acreditar, porque uma coisa que o Thiago sempre foi é verdadeiro, justo e leal. Mesmo com todas as adversidades enfrentadas, sinto que no fundo sempre tentamos honrar o trecho de uma música que colocamos no nosso convite de casamento: "Aqui ou noutro lugar, que pode ser feio ou bonito, se nós estivermos juntos, haverá um céu azul...". Às vezes com algumas nuvenzinhas, né? Muito obrigada por sempre querer o melhor para mim e para nós!

Só as pessoas mais sortudas têm sogros como os meus, **Seu Paulinho e Dona Bete**, que são como pais pra mim, às vezes causando até ciúmes no Thiago. Eu sinto o amor e a preocupação que eles têm por mim diariamente. Espero que eles também sintam o tanto que são amados por mim e o tanto que sou orgulhosa de ser a nora mais querida deles (e

não é porque sou a única, tá?). Eles são pessoas maravilhosas e exemplos pra mim de força e fé. Já perdi a conta de quantas novenas, trezenas e orações avulsas foram oferecidas a mim pela minha sogra. Muito obrigada pelo Amor e orações.

Quero fazer mais um agradecimento especial a pessoas muito queridas na minha vida que mesmo de longe estão sempre rezando e torcendo muito por mim. Minhas lindas Amigas **Bia**, **Deise**, **Gra**, **Katy** e **Patty Parreira**, que honram como ninguém a palavra amiga, pois sei que sempre estarão aqui por mim e eu por elas, custe o que custar e estejamos onde estivermos. Amo muito vocês! Minha querida **"tia"Alice**, que sempre esteve presente na minha vida de uma maneira única, com conversas deliciosas, orações, sem contar a hospitalidade que só ela sabe oferecer. Minha madrinha **"tia"Tânia**, que amo e não vejo tanto quanto gostaria, mas de quem sei estar sempre no coração, como ela no meu. Meus **"tios"Reis** (*in memoriam*) **e Célia** que sempre foram muito mais que apenas vizinhos, e sim família, e a quem eu tenho uma gratidão profunda por tudo o que sempre fizeram por mim e pela minha família. E a todos aqueles que também fazem parte da minha vida e sempre torceram por mim, mas não foram citados nominalmente aqui, pois este texto ficaria mais longo ainda, meu mais sincero muito obrigada!

Agradeço aos meus colegas e professores do IME, tão importantes nessa etapa da minha vida profissional, mas em especial ao **Frank**, que além de companheiro de laboratório, foi um amigo, uma inspiração como pesquisador e parceiro de trabalho no início dessa pesquisa. Muito sucesso sempre e muito obrigada pela amizade e parceria!

A meus alunos, que foram muitas vezes prejudicados, por conta das minhas atividades no doutorado, eu agradeço pela compreensão e paciência. Espero que vocês saibam do amor que eu tenho pela minha profissão e o apreço que eu tenho por cada um de vocês. Tudo isso é também por vocês! Muito obrigada por fazerem eu me sentir realizada com todo o carinho que vocês dedicam a mim. Muito obrigada mesmo!

Por fim, gostaria de agradecer à Universidade Estadual de Feira de Santana pelo apoio financeiro durante o período que estive em São Paulo, e a todos os meus colegas professores da Área de Informática e funcionários do Departamento e Colegiado que sempre me ajudaram, seja me deixando passar na frente na fila da licença-prêmio, seja ensinando em meu lugar quando as coisas apertavam muito ou dando outros tipos de apoio. Muito obrigada!

Resumo

Ana Lúcia Lima Marreiros Maia. **Uma abordagem baseada em Aprendizagem de Máquina e Grafos para Segmentação de Páginas**. Tese (Doutorado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2023.

Muitos documentos originalmente gerados em papel são digitalizados para possibilitar sua preservação ou para agilizar seu processamento por meio de ferramentas computacionais. Consultar documentos em bancos de dados de imagens ou extrair informações de interesse de imagens de documentos requer a análise do conteúdo da imagem. Em particular, uma etapa crítica nesta análise é a análise lógica de leiaute, que consiste em detectar os componentes da página e identificar suas funções lógicas. A análise lógica de leiaute permite estabelecer as relações entre os componentes e determinar informações importantes, como a ordem de leitura. Uma etapa fundamental na análise lógica de leiaute é detectar e classificar essas componentes de página, como blocos de texto, figuras e tabelas, problema conhecido como segmentação de página. Nesta tese, propomos um método que segue uma abordagem *bottom-up*, combinando modelagem de grafos e técnicas de aprendizado de máquina, para o problema de segmentação de páginas. O método proposto consiste em um *pipeline* no qual algumas etapas estratégicas são implementadas por meio de algoritmos de aprendizado de máquina. Como os algoritmos de aprendizado de máquina são treináveis a partir de dados, o método proposto pode ser facilmente adaptado a conjuntos de documentos com diferentes características, desde que os dados de treinamento estejam disponíveis. Esta tese também discute um procedimento experimental para otimizar o *pipeline*. Os experimentos utilizaram imagens de documentos (revistas e artigos científicos) do *PRIMA Layout Analysis Dataset*, com leiautes diversificados e complexos. Os resultados experimentais demonstram o potencial do método proposto.

Palavras-chave: Imagem de documento. Leiaute de página de documento. Segmentação de imagem. Grafo de adjacência. Rede neural convolucional.

Abstract

Ana Lúcia Lima Marreiros Maia. **A machine learning and graph based approach to page segmentation**. Thesis (Doctorate). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2023.

Many documents originally generated on paper are digitized to enable their preservation or to streamline their processing through computational tools. Querying documents in image databases or extracting information of interest from document images requires the analysis of image content. In particular, a critical step in this analysis is the logical layout analysis, which consists of detecting page components and identifying their logical function. Logical layout analysis enables establishing the relationships between the page components and determining important information, such as the reading order. A fundamental step in logical layout analysis is detecting and classifying these page components, such as blocks of text, figures, and tables, a problem known as page segmentation. In this thesis, we propose a method that follows a bottom-up approach, combining graph modeling and machine learning techniques, for the page segmentation problem. The proposed method consists of a pipeline in which some strategic steps are implemented through machine learning algorithms. Since machine learning algorithms are trainable from data, the proposed method can be easily adapted to document sets with different characteristics as long as training data is available. This thesis also discusses an experimental procedure to optimize the pipeline. The experiments used document images (magazines and scientific papers) from PRImA Layout Analysis Dataset, with diverse and complex layouts. The experimental results demonstrate the potential of the proposed method.

Keywords: Document image. Document page layout. Image segmentation. Adjacency graph. Convolutional neural network.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Documentos são instrumentos comumente utilizados para registro de fatos, eventos, informações ou conhecimentos desde séculos atrás. Estamos envoltos por diferentes tipos de documentações e registros em papel tais como livros, revistas, jornais, panfletos, cartazes, anotações, documentos legais, entre outros.

Com o avanço da tecnologia de informação, muitos registros já são realizados diretamente no formato digital, dando origem aos chamados *born digital documents*[1]. No entanto, o volume de documentos arquivados e ainda sendo produzidos em papel é grande e cada vez mais conteúdos originalmente disponibilizados apenas em papel estão sendo digitalizados e disponibilizados em plataformas digitais. Livros ou documentos raros que antes podiam ser encontrados apenas em algumas bibliotecas específicas, agora estão sendo disponibilizados na forma digital em bibliotecas ou arquivos digitais.

Apesar do grande volume de material digital e ferramentas computacionais de busca disponíveis, encontrar conteúdos de interesse ainda é uma tarefa difícil pois a maior parte do material digitalizado está na forma de imagens (fotografias) de páginas de documentos. Em geral, as buscas por esses materiais podem ser feitas por palavras-chave, época de publicação, ou outros metadados. Em se tratando por exemplo de livros, a forma de busca muitas vezes não é muito diferente de buscar livros/documentos em uma biblioteca. Para localizar informações de interesse, após livros relacionados serem encontrados, precisamos ainda folhear cada um deles e examinar seu conteúdo em busca das informações de interesse.

Desta forma, a interpretação automática de conteúdos de documentos a partir das imagens de suas páginas é uma tarefa fundamental para facilitar uma indexação mais flexível e ampla de documentos, a extração de informações de interesse, ou ainda para diversos outros propósitos.

Existe uma vasta literatura relacionada ao processamento e análise de imagens de documentos. Os métodos propostos lidam desde processamentos básicos como filtragem de ruído ou binarização de imagens, detecção de texto e reconhecimento de caracteres, até determinação da ordem de leitura de uma página (Doermann e Tombre, 2014).

[1] Wikipedia: Born-digital

Para tarefas como determinação da ordem de leitura ou interpretação de conteúdo, a análise de leiaute é um importante passo de processamento. Em análise de leiaute, busca-se determinar quais são os elementos constituintes de uma página e qual a relação entre eles (Dengel e Shafait, 2014). Por exemplo, alguns componentes comumente presentes na maioria dos documentos são blocos de texto, figuras (fotografias ou diagramas), tabelas, e número da página. O conjunto de componentes define o leiaute físico. Assim, a análise de leiaute físico preocupa-se em identificar os componentes de página, associando a eles rótulos de categorias genéricas. Já a análise lógica engloba a análise física e busca adicionalmente determinar a função lógica de cada componente assim como as relações espaciais e hierárquicas entre eles. Por exemplo, um bloco de texto pode ser um título, um parágrafo, uma legenda de figura, entre outras possibilidades. Ou ainda, caracteres localizados um ao lado do outro formam palavras que por sua vez formam uma linha de texto, e uma sequência de linhas forma um parágrafo, que formam blocos de texto; uma figura é composta por uma parte gráfica e uma legenda, e essa mesma figura costuma ser referenciada em algum bloco de texto. A partir de informações que permitem entender a lógica do leiaute, torna-se possível por exemplo estabelecer a ordem correta de leitura.

A utilização de técnicas de aprendizado de máquina na área de imagens de documentos tem origem já nos inícios da era da computação, destacando-se notadamente os OCRs (reconhecimento óptico de caracteres). Para efeitos de contextualização desta tese, podemos dividir os métodos baseados em aprendizado de máquina em antes e depois do surgimento das modernas redes neurais, técnicas conhecidas como *deep learning* (Goodfellow *et al.*, 2016). Antes da emergência do chamado *deep learning*, grande parte do esforço no emprego de técnicas de aprendizado de máquina era dedicada à extração de características das imagens (Binmakhashen e Mahmoud, 2019). Os algoritmos de aprendizado de máquina eram então treinados a partir dessas características. Em geral, o conjunto de características que culminam em bons resultados são altamente dependentes do conjunto de imagens. Desta forma, características que resultam em bom desempenho para um conjunto de imagens geralmente não apresentam o mesmo desempenho para outros conjuntos de imagens. Esse quadro foi significativamente alterado com o surgimento de *deep learning*. As técnicas de *deep learning* incorporam a extração de características no próprio processo de treinamento dos algoritmos. Desta forma, elas são totalmente baseadas em dados, e facilmente aplicáveis a novos conjuntos de dados (desde que tenhamos dados suficientes para o treinamento).

Embora *deep learning* tenha vindo a público em 2012, quando o modelo AlexNet (Krizhevsky *et al.*, 2012) venceu a competição ILSVRC[2] (*ImageNet Large Scale Visual Recognition Challenge*), ele passou a ser a técnica dominante na área de Visão Computacional em torno ou depois de 2014. Na área de processamento de documentos, tornou-se mais popular apenas alguns anos mais tarde, apesar de em 2007, (Liwicki *et al.*, 2007) ter trazido um olhar para redes neurais profundas, quando seu modelo venceu uma competição de reconhecimento de manuscritos com redes RNN e LSTM na *International Conference on Document Analysis and Recognition*.

[2] ILSVRC

1.1 Objetivos

O tema tratado nesta tese é a segmentação de componentes de página, isto é, a detecção de todos os componentes tais como blocos de texto, figuras, tabelas, linhas separadoras, entre outros que compõem uma página de documento. Como já mencionado, este é um passo importante para a análise de leiaute da página que, por sua vez, é importante para a interpretação do conteúdo de uma página, justificando a relevância desse problema.

Quando esta tese começou a ser desenvolvida, publicações com a utilização de técnicas de *deep learning* para processamento e análise de imagens de documentos eram praticamente inexistentes. De fato, também eram escassas as publicações relacionadas à segmentação de componentes de página utilizando técnicas de aprendizado de máquina. As publicações existentes em geral consideravam cenários restritos, estabelecendo um conjunto limitado de componentes de interesse, tais como texto, figuras, tabelas ou expressões matemáticas. Além disso, as bases de dados anotados eram também limitadas em termos de número de imagens, variedade de documentos, ou tipos de anotações.

Diante desse cenário, nosso entendimento foi de que para avançar o estado-da-arte nessa área, métodos gerais para a segmentação de páginas em suas partes constituintes, na granularidade de interesse, precisariam ser desenvolvidos. Motivado por essa constatação, estabelecemos um objetivo principal: desenvolver métodos para a segmentação de componentes de página que não fossem muito restritivos quanto ao leiaute da página nem aos tipos e formatos dos componentes de página. Em outras palavras, o método não deveria pressupor um conjunto limitado de componentes de página possíveis nem uma forma fixa sobre como os mesmos se encontram arranjados na página (por exemplo, textos da esquerda para a direita ou página organizada em duas colunas). Em vez disso, gostaríamos que o método a ser desenvolvido pudesse idealmente "aprender" essas informações a partir de imagens de treinamento.

1.2 Método proposto e contribuições

Nesta tese, propomos um método dividido em etapas que parte de uma super-segmentação da imagem e em seguida realiza o agrupamento dos segmentos atômicos em regiões (segmentos) maiores que correspondem aos componentes de página de interesse, que serão, enfim rotulados. Para que essa abordagem seja flexível, de forma a ser igualmente aplicável a famílias de documentos com características distintas em termos de leiaute e componentes constituintes, utilizamos técnicas de aprendizado de máquina em pontos estratégicos. Especificamente, utilizamos técnicas de aprendizado de máquina para classificar os segmentos atômicos quanto ao tipo de componente de página ao qual eles pertencem, e também para decidir se segmentos vizinhos fazem parte de um mesmo componente de página ou não. A relação de vizinhança é representada por um grafo, que inicialmente conecta todos os segmentos atômicos e que posteriormente sofre partições (através de uma classificação das arestas), em que os seguimentos atômicos que pertencem à mesma parte recebem um rótulo único e formam um determinado componente de página. Portanto, o método proposto se apoia em duas modelagens importantes: emprego de técnicas de aprendizado de máquina e de grafos.

Para viabilizar a investigação do método proposto, escolhemos componentes conexas de imagens binarizadas como os segmentos primitivos, redes neurais convolucionais para a classificação dos segmentos primitivos, e a triangulação de Delaunay para a criação de grafos de adjacência. Atributos diversos foram associados aos nós (segmentos primitivos) e arestas do grafo, e algoritmos de aprendizado de máquina foram treinados para determinar se arestas devem ser mantidas ou removidas. Ao final, as componentes conexas do grafo resultante são consideradas como os componentes de página, os quais são ainda em seguida classificados quanto ao tipo de componente.

Concretamente, o método proposto resulta em um *pipeline* com passos de processamento sequenciais. A otimização de um *pipeline* não é uma tarefa trivial, pois os ajustes em um determinado passo afetam os passos subsequentes. Nesta tese apresentamos discussões sobre possíveis formas de realizar a otimização do *pipeline* e adotamos um esquema de otimização sequencial reversa, no qual a otimização é realizada do último para o primeiro passo.

Este esquema foi testado no conjunto de imagens do *PRImA Layout Analysis Dataset* (Antonacopoulos, Bridson *et al.*, 2009), que consiste em uma base de dados de imagens de revistas e artigos científicos, variando a configuração de alguns parâmetros nos passos que utilizam algoritmos de aprendizado de máquina. Os resultados mostram que o método apresenta um bom desempenho na segmentação de páginas com leiautes mais regulares e tem potencial para realizar a segmentação de páginas com leiautes mais complexos.

Vale ressaltar que a proposta é que cada etapa do método possa facilmente ser ajustado para incorporar diferentes técnicas de aprendizado de máquina, diferentes técnicas de construção de grafos, diferentes conjuntos de características etc. Dessa forma, testamos algumas configurações com o intuito de validar o modelo proposto e a técnica de otimização de pipeline proposta.

Assim, os resultados obtidos indicam que o método proposto é promissor pois atende a proposta de ser um processo completo para Segmentação de Páginas, considerando todas as características dos documentos a serem segmentados, sem impor restrições quanto ao tipo de componentes de páginas a serem considerados, língua, leiaute.

1.3 Organização deste texto

O restante desta tese está organizado da seguinte forma. No capítulo 2, descrevemos o problema de Segmentação de Página, desde a sua definição, principais conceitos e trabalhos relacionados, e desafios identificados. Vale notar que atualmente observa-se a utilização predominante de técnicas de *deep learning* em processamento de imagens de forma geral, inclusive em relação a imagens de documentos. Dado que nesta tese utilizamos técnicas de *deep learning* em apenas parte do método proposto, nesta seção apresentamos também um panorama dos desdobramentos mais recentes na área relacionados ao emprego de técnicas de *deep learning* no problema de segmentação de página ou análise de leiaute. No capítulo 3 apresentamos a abordagem proposta, detalhando o método flexível brevemente exposto acima para o problema de segmentação de página. Em particular, o método descrito pode ser visto como uma instanciação da abordagem proposta, implementada como um

pipeline com módulos adaptáveis. No capítulo 4 detalhamos os experimentos que foram desenhados para avaliar o efeito de determinadas escolhas nos vários módulos adaptáveis do método proposto. No capítulo 5 apresentamos e discutimos os resultados experimentais. Finalmente, no capítulo 6 apresentamos as considerações finais deste trabalho.

Capítulo 2

O problema de segmentação de página

Neste capítulo discorremos sobre o problema de Segmentação de Páginas. Inicialmente caracterizamos o problema, descrevemos em seguida conceitos básicos importantes para o entendimento das estratégias e métodos comumente utilizados para esse problema. Apresentamos também uma abstração da estratégia *bottom-up*, que servirá de base para o desenvolvimento da abordagem proposta nesta tese. Ao final do capítulo apresentamos um panorama do emprego de técnicas de Deep Learning para o problema.

2.1 Segmentação de Página

O problema de Segmentação de Página consiste em particionar a imagem de um documento em regiões (ou segmentos) homogêneas que correspondem a o que chamamos de componentes de página (Kise, 2014). Alguns dos principais componentes de página são blocos de texto, figuras, gráficos, tabelas, e fórmulas (Dengel e Shafait, 2014). Exemplos desses elementos são exibidos na Figura 2.1.

A disposição espacial desses componentes define o que é conhecido por leiaute (físico ou geométrico) da página. De fato, os documentos podem apresentar leiautes variados (Kise, 2014). Podemos dividir os tipos de documentos existentes em quatro categorias principais: leiaute retangular, leiaute Manhattan, leiaute não-Manhattan e leiaute com sobreposição. A figura 2.2 ilustra esses diferentes tipos de leiaute.

O leiaute retangular apresenta todos os componentes de página circunscritas em retângulos não sobrepostos, e cujos lados são paralelos às bordas da página (Figura 2.2a). Muitos artigos científicos e livros apresentam este tipo de leiaute.

No leiaute Manhattan, os componentes de página estão circunscritos em polígonos não sobrepostos, com lados também paralelos às bordas da página (Figura 2.2b). Muitos jornais e revistas possuem este tipo de leiaute. É fácil ver que o leiaute retangular é um caso particular do leiaute Manhattan.

O leiaute não-Manhattan também apresenta os componentes de página circunscritos

Figura 2.1: *Exemplos de elementos que compõem a página de um documento. Imagens originais extraídas da base de dados PRIMA (Antonacopoulos, Pletschacher et al., 2009).*

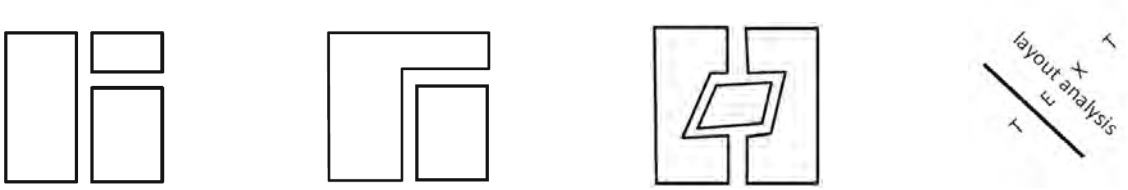


Figura 2.2: *Ilustração de tipos de leiaute. (a) leiaute retangular, (b) leiaute Manhattan, (c) leiaute não-Manhattan e (d) leiaute com sobreposição. Fonte: Kise, 2014.*

em formas geométricas não sobrepostas, porém elas podem assumir qualquer formato (Figura 2.2c).

Os três tipos de leiaute apresentados cobrem a maioria dos documentos. Entretanto, vale citar um tipo particular de leiaute em que um componente de página pode se sobrepor a outro (Figura 2.2d).

Para ilustrar essa grande variedade de leiautes, na Figura 2.3 podemos ver exemplos de imagens de revistas com diferentes leiautes e com diferentes componentes de página.

A segmentação de páginas é uma etapa no processo de interpretação de documentos (Dengel e Shafait, 2014). A interpretação de imagens de documentos requer uma análise lógica do leiaute do documento. Essa análise lógica requer informações sobre o leiaute físico da página, que por sua vez é formada pelos componentes de página, resultantes da segmentação de página. O leiaute físico pode ser representado por meio do conjunto de componentes de página, associados cada um a informações geométricas e categóricas que identificam o tipo de componente (bloco de texto, figura, etc). Na análise lógica busca-se organizar o conteúdo da página em termos desses componentes, de forma a facilitar a

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

Figura 2.3: *Exemplos de imagens de revistas com diferentes leiautes e componentes de página. Imagens originais extraídas da base de dados ICDAR-2009 (Antonacopoulos, Pletschacher et al., 2009).*

extração de informações que possam ser entendidas pelos humanos ou para colocar o conteúdo em um formato adequado para o processamento computacional. Por exemplo, um componente do tipo bloco de texto pode ser associado a rótulos lógicos tais como título, parágrafo, nota de rodapé, etc.

Outra situação em que a segmentação de páginas pode ser usada é na classificação ou busca de documentos. De acordo com Marinai, 2014, a categoria de vários documentos pode ser identificada a partir de seu leiaute, sem necessariamente examinar o conteúdo textual do documento. Enquanto abordagens iniciais para a classificação ou busca de documentos consistiam em associar um descritor global formado por um conjunto de características extraídas das imagens de documentos, abordagens mais recentes passaram a representar as páginas por meio de um conjunto de zonas e então associar descritores para cada zona. Desta forma, a classificação de uma imagem de documento ou a busca baseada em similaridade pode levar em conta a similaridade entre os conjuntos de zonas dos documentos. Nesse contexto, os componentes de página resultantes do processo de segmentação de página podem ser vistos como as possíveis zonas representantes da página.

2.2 Abordagens para segmentação de página

Ao longo dos anos, várias abordagens foram desenvolvidas para a segmentação de página. Kise, 2014 apresenta uma introdução ao problema e cobre métodos publicados até 2007. Já Eskenazi *et al.*, 2017 cobrem os métodos publicados a partir de 2008. Em contraste a métodos heurísticos ou baseados em regras, com o passar dos anos passou-se a observar cada vez mais abordagens que utilizam técnicas de aprendizado de máquina. Mais recentemente há uma intensificação de utilização de técnicas de *deep learning*. Um artigo de revisão mais recente (Binmakhashen e Mahmoud, 2019) dedica uma seção para discutir o assunto. Um panorama sobre a utilização de técnicas de *deep learning* no problema de segmentação de página é apresentado posteriormente no capítulo 2.6 desta tese. Nesta seção nos restringimos ao contexto anterior ao emprego de técnicas de *deep learning*.

Em geral, o problema de segmentação de páginas é abordado principalmente por meio de duas estratégias: *bottom-up* e *top-down*. Na estratégia *top-down* os documentos são iterativamente divididos em seções até formarem os componentes de página desejados (Ha *et al.*, 1995b; Ha *et al.*, 1995a). Na estratégia *bottom-up*, que é a mais comumente usada, parte-se da segmentação das imagens em unidades menores (primitivas) que são agrupadas para formar os componentes de página.

Os primeiros métodos consideravam imagens em tons de cinza e exploravam basicamente o *foreground* ou o *background*. Entre as abordagens *top-down*, um método bastante conhecido é o corte XY recursivo (Ha *et al.*, 1995b), que funciona bem para leiautes retangulares. Entre as abordagens *bottom-up*, em relação ao *foreground*, o princípio comum explorado consiste em agrupar as primitivas até a formação dos componentes de página. Já no caso de *background*, o princípio consiste em agrupar primitivas no fundo de forma a destacar as bordas externas dos componentes de página. As primitivas usadas para a análise de *foreground* são pixels, superpixels ou componentes conexas e as usadas para a análise de *background* são retângulos brancos (isto é, retângulos maximais inteiramente contidos no *background*).

À medida que componentes de página e leiautes complexos passam a ser tratados, começam a surgir métodos baseados em aprendizado (*learning-based methods*), assim como abordagens híbridas que combinam estratégias *bottom-up* e *top-down* (Binmakhashen e Mahmoud, 2019). Embora os termos análise de leiaute físico e análise lógica de leiaute ainda sejam bastante utilizados, a fronteira entre elas é vaga. Com o avanço das técnicas, segmentações de componentes de página com nível lógico mais diversificado estão se tornando viáveis. Por exemplo, se antes a detecção de um parágrafo consistia primeiramente na detecção de linhas e em seguida o agrupamento de múltiplas linhas, atualmente parágrafos são detectados diretamente. A mesma situação ocorre com a detecção de uma tabela. A distinção de um bloco de texto como sendo título ou parágrafo é outro exemplo de análise lógica que está começando a ser tratada no contexto de segmentação de página.

Embora esteja fora do escopo deste trabalho, vale mencionar que o leiaute de uma página de documento costuma ser representado por estruturas do tipo árvore ou então por grafos. No caso de leiautes retangulares são usadas as estruturas do tipo árvore, refletindo diretamente a organização hierárquica do conteúdo de uma página. Já os grafos são utilizados para representar leiautes mais complexos. Por exemplo, Marinai, 2014 cita as ARGs (*Attributed Relational Graphs*) nos quais os nós representam os componentes lógicos, aos quais podem ser associadas características dos respectivos componentes, e as arestas representam relações entre os componentes. Então, a partir dessas estruturas passa a ser possível realizar buscas, verificar similaridade entre documentos, e outras finalidades.

O problema de segmentação de página já vem sendo bastante discutido ao longo dos anos. Entretanto, as abordagens propostas na literatura ainda exploram escopos restritos do problema (Kise, 2014; Eskenazi *et al.*, 2017; Binmakhashen e Mahmoud, 2019). Dentre as restrições que identificamos, destacamos os seguintes:

- **Restrição de leiaute:** a maioria dos métodos concentra seus experimentos em um tipo de leiaute de documentos. Os métodos em geral não apresentam bons resultados para leiautes complexos como o não-Manhattan ou simplesmente não podem ser

estendidos para outros tipos de leiaute. Como pode ser visto no capítulo 2.6, mesmo as abordagens mais recentes baseadas em técnicas de *deep learning* consideram predominantemente os documentos com leiaute retangular.

- **Restrição de categoria de documentos:** em geral, os trabalhos existentes são desenvolvidos para uma categoria específica de documentos como artigos científicos, documentos históricos, manuscritos, formulários.
- **Restrição de idioma:** documentos com idiomas latinos são os mais abordados nos trabalhos de análise de leiaute. Os métodos existentes, em geral, raramente exploram sua extensão para aplicação em documentos em idiomas não latinos. Os trabalhos que propõem essa extensão, em geral não apresentam resultados satisfatórios (Tran, Oh *et al.*, 2017).
- **Restrição de componentes de página:** comumente, os trabalhos existentes exploram um subconjunto, muitas vezes muito restrito, do vasto conjunto de componentes de página possíveis. O subconjunto mais explorado até recentemente consiste em apenas duas categorias: texto e não-texto. Da mesma forma, alguns trabalhos focam na detecção de um tipo específico de componente de página (por exemplo, tabelas (Shafait e Smith, 2010; Kasar *et al.*, 2013), ou fórmulas matemáticas (Anitei *et al.*, 2021)).

2.3 Tipos de primitivas em abordagens *bottom-up*

Conforme mencionado anteriormente, as estratégias *bottom-up* consistem em dividir a imagem em um conjunto de unidades primitivas e em seguida agrupá-las de forma que cada grupo corresponda a um componente de página. Essa mesma ideia é bastante utilizada em segmentação semântica de imagens em geral. A imagem é primeiramente super segmentada, e em seguida diferentes métodos são usados para fazer a fusão de segmentos menores até ser alcançado um segmento que corresponde a uma região de interesse com alguma semântica associada.

O tipo de primitivas utilizadas é determinante na formação dos componentes de página, pois as estratégias *bottom-up* pressupõem que as primitivas não são divisíveis. Além disso, uma vez que essas primitivas precisarão ser manuseadas, a quantidade delas pode também tornar-se crítica. Desta forma, a escolha do tipo de primitiva a ser utilizada dependerá da aplicação e poderá ter influência direta na eficiência e qualidade dos resultados.

As primitivas comumente utilizadas são pixels, superpixels e componentes conexas, e são explicadas a seguir.

2.3.1 Pixels e Superpixels

A menor granularidade possível quando pensamos em uma super segmentação da imagem corresponde à partição da imagem em seu conjunto de pixels. Embora essa partição seja direta, a quantidade de primitivas atômicas pode facilmente alcançar o número de milhões. Por outro lado, a partição nessa granularidade é a mais segura caso exista a necessidade de delineação de contornos detalhados e precisos.

O termo superpixels foi introduzido por Ren e Malik, 2003, que propuseram a partição da imagem em agrupamentos maximais de pixels adjacentes com cor e outras características similares. Superpixels são comumente utilizados em problemas de segmentação semântica de imagens, mas foram também explorados em alguns trabalhos de Segmentação de Páginas de documentos históricos (Cohen *et al.*, 2013; Mehri *et al.*, 2015; Chen, Liu *et al.*, 2016).

Existem diferentes métodos de segmentação em superpixels, cada um com suas vantagens e desvantagens e que podem ser adequados a diferentes tipos de problema (Wang *et al.*, 2017).

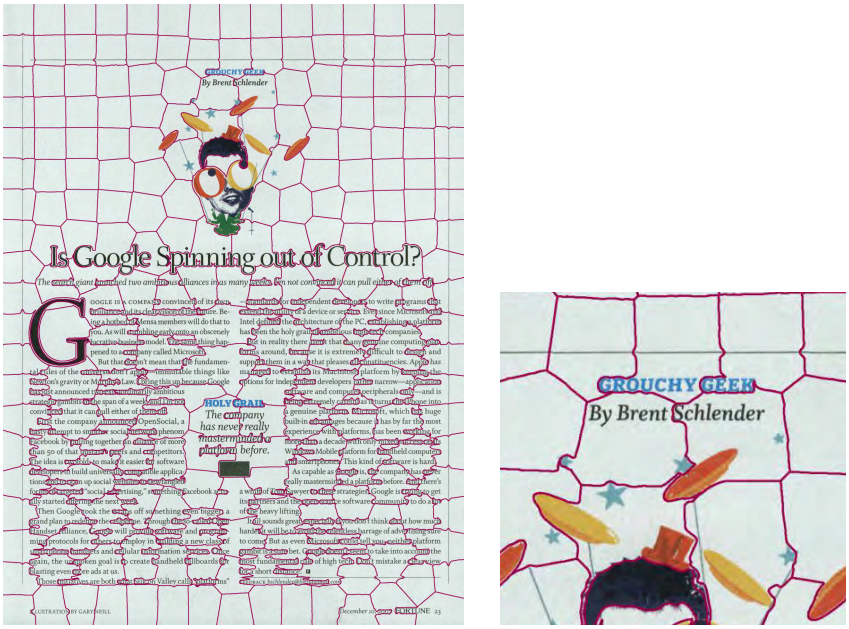
Segundo Achanta *et al.*, 2012, que introduziu o bem conhecido método SLIC, não há uma abordagem ou método geral que seja adequado a todos os tipos de aplicação. Entretanto, algumas propriedades são geralmente desejáveis, como:

1. Superpixels devem aderir bem aos contornos dos objetos na imagem;
2. Quando utilizados como uma etapa de pré-processamento para reduzir a complexidade computacional, os superpixels devem ser simples, permitir eficiência no uso de memória e na sua computação.
3. Quando utilizados para segmentação, os superpixels devem aumentar tanto a velocidade quanto a qualidade dos resultados.

No contexto de processamento de imagens de documento, superpixels são mais adequados para documentos em cores, nos quais a separação entre *foreground* e *background* é uma tarefa difícil.

Um exemplo de segmentação em superpixels usando o algoritmo SLIC (Achanta *et al.*, 2012) para uma imagem de documento é apresentado na Figura 2.4a. Pode-se notar que enquanto há uma certa aderência do contorno das regiões aos contornos de parte de alguns objetos na imagem, de forma geral não há aderência aos contornos dos componentes de páginas. No detalhe destacado na figura 2.4b, elementos pertencentes a diferentes componentes de página estão contidos em um mesmo superpixel. Aceitando que superpixels são indivisíveis, jamais poderíamos encontrar uma delimitação precisa desses componentes de página.

É possível que melhores segmentações possam ser obtidas por meio de ajuste de alguns parâmetros. Porém, dada a variabilidade das características de imagens de documentos, esse ajuste poderá ser específico para cada imagem, o que inviabilizaria a sua utilização em um processo automatizado.



(a) **(b)**

Figura 2.4: *À esquerda, exemplo de uma segmentação de imagem de um documento utilizando o algoritmo SLIC. À direita, detalhe ilustrando um superpixel contendo partes de dois componentes de página (texto e imagem) distintos.*

2.3.2 Componentes Conexas

Em documentos, uma boa parte dos pixels corresponde ao fundo e portanto não existe a necessidade desses pixels serem tratados. Documentos, principalmente aqueles formados predominantemente por texto, podem ser reduzidos a um conjunto de objetos que tipicamente correspondem a caracteres ou parte de caracteres. Essa simplificação da imagem pode ser obtida por meio de técnicas de binarização. Uma técnica de binarização bem conhecida é a binarização de Otsu (Otsu, 1979).

A figura 2.5 mostra uma imagem binária contendo cinco objetos. Também é comum o uso dos termos *foreground* para se referir aos objetos (em branco na figura) e *background* para se referir ao fundo da imagem (em preto na figura).

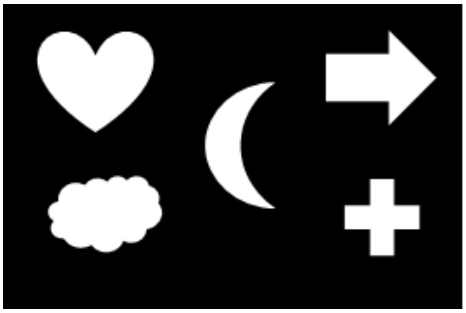


Figura 2.5: *Exemplo de uma imagem binária com cinco objetos (em branco).*

Esses objetos presentes em imagens binárias são comumente denominados de componentes conexas (CC)[1]. Para uma definição precisa de CC, convém modelar uma imagem como grafo. Uma imagem binária pode ser modelada por um grafo da seguinte forma. Pontos do *foreground* formam o conjunto de vértices e um certo vértice p é conectado a outro vértice q se q é um dos 8 vizinhos adjacentes a p.

Um grafo é dito conexo se, para qualquer par de vértices (u, v), existe um caminho com extremos u e v. Uma componente conexa de um grafo G é qualquer subgrafo conexo maximal de G (Feofiloff *et al.*, 2011).

Assim, CC em imagens binárias correspondem exatamente às componentes conexas do grafo subjacente. De forma informal, são os agrupamentos maximais de pontos adjacentes do *foreground*.

Para a identificação de CCs em imagens binárias, utiliza-se um algoritmo de rotulação de CCs. Tal algoritmo visa a computar e identificar cada uma das CC em uma imagem. Todos os pixels de uma CC recebem um rótulo único que identifica a componente, e estas são numeradas de 1 a N, com N correspondendo à quantidade total de componentes. Quando há interesse em se manipular ou analisar as CCs, informações que as identificam e permitem acesso rápido aos seus pixels são em geral armazenadas em uma estrutura de dados adequada. Um algoritmo bem conhecido para a rotulação de componentes conexas consiste em atribuir um novo rótulo a um pixel ainda não rotulado do *foreground* e então

[1] No contexto deste trabalho, utilizaremos a palavra componente como substantivo feminino quando se tratar de componentes conexas e como substantivo masculino quando se tratar de componentes de página

propagar esse rótulo sucessivamente a todos os pixels adjacentes, até não existirem mais pixels não rotulados na adjacência. O processo é finalizado quando todos os pixels do *foreground* estiverem rotulados. O algoritmo pode ser visto, por exemplo, em P. Soille, 2003.

A figura 2.6 mostra o resultado da rotulação das componentes conexas da figura 2.5, na qual cada cor representa um rótulo e consequentemente identifica uma componente individualmente.

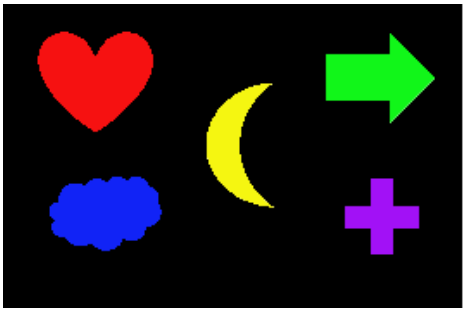


Figura 2.6: *Imagem da Figura 2.5 com as componentes rotuladas. Cada cor representa uma componente.*

As componentes conexas possuem inúmeras características que podem ser exploradas. Por exemplo, os tamanhos predominantes dos caracteres que compõem uma região de texto são similares. Muitos trabalhos de análise de documentos explorando componentes conexas vem sendo apresentados ao longo dos anos (Bukhari *et al.*, 2010; Chen, Yin *et al.*, 2013; Le *et al.*, 2015; Tran, Na *et al.*, 2016; Tran, Oh *et al.*, 2017; Arenas *et al.*, 2018; C. Ma *et al.*, 2023).

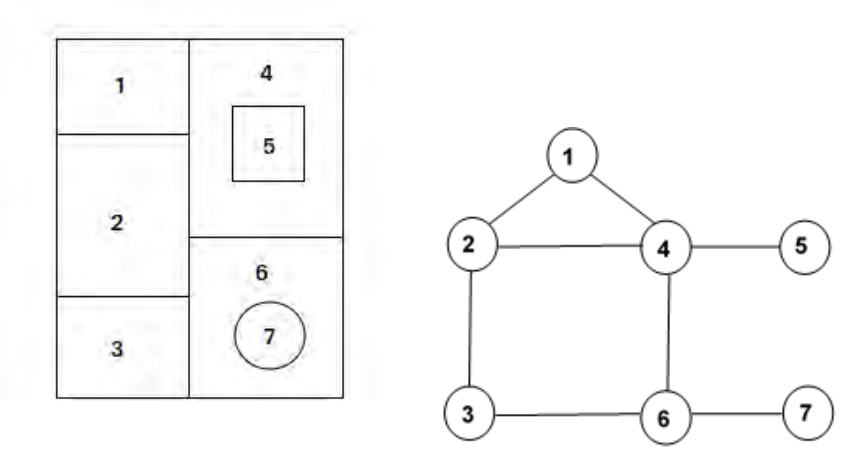
2.4 Agrupamento de primitivas

Na seção anterior citamos pixels, superpixels e CCs como possíveis tipos de primitivas. As primitivas que compõem uma imagem precisam ser agrupadas de forma que as regiões finais resultantes tenham algum significado semântico, como é o caso das componentes de página. Independentemente do tipo de primitiva, em geral o princípio utilizado para o agrupamento consiste em fundir primitivas adjacentes com características similares (tais como cor ou tamanho), de forma que a região final seja homogênea em relação às características consideradas.

2.4.1 Representação de relações de adjacência

Para representar a relação de adjacência, uma estrutura comumente utilizada é o grafo de adjacência de regiões (RAG, do inglês *Region Adjacency Graph*). Introduzida por Rosenfeld (Rosenfeld, 1974), um RAG é um grafo planar, em que os vértices correspondem às regiões e dois vértices estão conectados se as regiões correspondentes aos vértices são adjacentes entre si. A figura 2.7a mostra um exemplo com sete regiões, numeradas 1 a 7, e a figura 2.7b representa o correspondente RAG.

RAGs são facilmente construídas a partir de primitivas do tipo pixel ou superpixel. Quando as primitivas consideradas são as componentes conexas, outros critérios podem



(a) **(b)**

Figura 2.7: *Exemplo da construção de um grafo de adjacência de regiões. (a) Imagem original particionada (regiões numeradas). (b) Correspondente grafo de adjacências de regiões. No grafo, cada vértice corresponde a uma região, identificada pelo número.*

ser usados para estabelecer adjacências no grafo. Por exemplo, pode-se definir um fator ϵ, e adicionar arestas entre vértices cujos correspondentes CCs possuem distância menor que ϵ. As distâncias podem ser definidas de diferentes formas: poderia ser simplesmente a distância Euclidiana entre os centroides das CCs, ou a distância restrita ao eixo horizontal ou vertical (úteis, por exemplo, para representar o espaçamento entre caracteres consecutivos em uma frase ou o espaçamento entre duas linhas consecutivas), dentre outras.

A seguir descrevemos três métodos usados para o estabelecimento de adjacência entre componentes conexas (Kise, 2014).

O primeiro método, árvore geradora mínima, considera a distância Euclidiana como medida de distância entre os vértices, para calcular a árvore geradora de menor custo. Métodos eficientes de construção de árvores geradoras mínimas, como Kruskal e Prim, estão disponíveis em diversas bibliotecas de manipulação de grafos. Esse método assume que componentes conexas que fazem parte de uma mesma região de interesse (por exemplo, caracteres em um bloco de texto) estão relativamente mais próximas umas às outras do que em relação a componentes conexas em outra região de interesse (por exemplo, outro bloco de texto ou um diagrama, por exemplo). Os componentes de página, as regiões de interesse, podem então ser vistos como subárvores da árvore geradora quando certas arestas (que conectam diferentes componentes de página) são removidas. A figura 2.8 ilustra uma árvore geradora mínima (em vermelho) para um pequeno bloco de texto, em que cada letra corresponde a um vértice.

O segundo método baseia-se no conhecido método de k-vizinhos mais próximos (k-NN, *k-nearest neighbors*). Dado k, uma aresta entre cada vértice e seus k vizinhos mais próximos é acrescentada, considerando a distância Euclidiana, por exemplo. Uma desvantagem desse método diz respeito à estimativa de k, que pode depender do leiaute da página e pode comprometer a tarefa de agrupamento de regiões, pois uma má estimativa de k pode incluir arestas indevidas ou deixar de incluir arestas importantes.

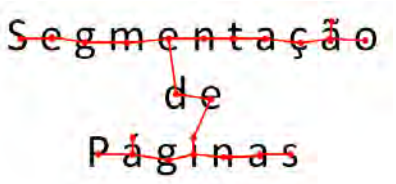


Figura 2.8: *Ilustração de uma Árvore Geradora Mínima para um bloco de texto..*

O terceiro método utiliza a chamada triangulação de Delaunay (Fortune, 1997). Seja $P = \{P_1, P_2, \dots, P_n\}$ um conjunto de pontos chamados de geradores, e seja $d(P_i, P_j)$ a distância entre os pontos P_i e P_j, para i, j em $1, 2, ..., n$. No caso de imagens de documentos binarizados, esses pontos geradores podem ser, por exemplo, os centroides das componentes conexas.

Uma região de Voronoi $V(P_i)$ de um ponto P_i é definida da seguinte forma:

$$V(P_i) = \{Q \,|\, d(Q, P_i) \leq d(Q, P_j), \forall j \neq i\}$$

Então, o diagrama de Voronoi $V(P)$ é gerado a partir do conjunto de pontos P e é definido como um conjunto de regiões de Voronoi:

$$V(P) = \{V(P_i), \dots, V(P_n)\}$$

As bordas das regiões de Voronoi são chamadas de arestas de Voronoi.

O grafo dual do diagrama de Voronoi é chamado de triangulação de Delaunay. Os vértices da triangulação de Delaunay são os chamados geradores, e as arestas conectam pares de vértices cujas regiões de Voronoi compartilham arestas de Voronoi. Um exemplo de um diagrama de Voronoi e a sua correspondente triangulação de Delaunay é mostrado na figura 2.9.

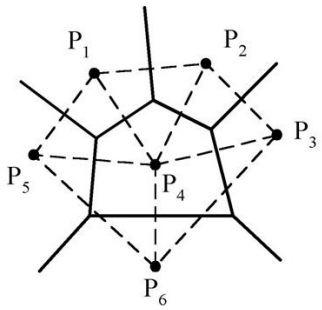


Figura 2.9: *Ilustração do diagrama de Voronoi e a sua correspondente triangulação de Delaunay. A triangulação está representada por linhas tracejadas.*

Considerando que os centroides das componentes conexas são os vértices de um grafo, as arestas obtidas a partir da triangulação de Delaunay para a imagem da figura 2.5 é apresentada na figura 2.10.

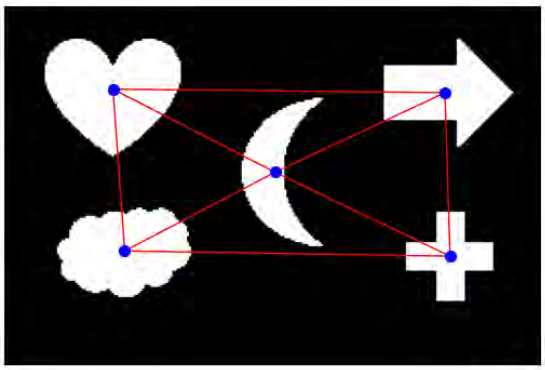


Figura 2.10: *Ilustração da triangulação de Delaunay para a imagem da figura 2.5.*

2.4.2 Cálculo dos agrupamentos

Por um lado, a estrutura do RAG é útil para registrar as adjacências no domínio espacial, necessárias para garantir que os agrupamentos de primitivas formem regiões conexas. Por outro lado, precisamos de informações adicionais para estabelecer os limites de cada componente de página (isto é, quais são exatamente as primitivas que constituem cada um dos componentes de página).

Em geral, existem características diversas que, ao menos para a interpretação humana, permitem o delineamento do contorno dos componentes de interesse. Por exemplo, ao olhar rapidamente em uma página de documento, facilmente identificamos texto, figuras, tabelas. Com um exame um pouco mais cuidadoso conseguimos também distinguir um parágrafo, uma subseção, um título, e assim por diante. Embora a interpretação humana seja dependente de contexto, podemos citar homogeneidade de cor ou tamanho, padrões texturais, ou espaçamentos, como exemplos de características que podem ajudar a identificar os componentes de página.

Um grande desafio para o cálculo computacional de agrupamentos é justamente fazer a caracterização adequada das primitivas, de modo que primitivas que constituem um único componente de página sejam caracterizadas de forma similar entre elas, e ao mesmo tempo de forma distinta em relação às demais primitivas.

Em problemas de cálculo de agrupamentos têm-se, portanto, um conjunto de objetos com características associadas a eles, e alguma métrica de similaridade baseada nessas características. Desta forma, podemos decidir agrupar ou não duas ou mais primitivas com base na medida de similaridade entre elas.

Em segmentação de imagens, existem algumas abordagens bem conhecidas para o cálculo de agrupamentos, tais como *clustering*, crescimento de regiões ou corte em grafos. Uma revisão desses algoritmos está além do escopo desta tese.

No caso específico em que as primitivas são componentes conexas e a relação de adjacência é representada por um RAG construído por meio da triangulação de Delaunay,

uma possível forma para o cálculo de agrupamentos consiste na remoção de determinadas arestas do RAG, de forma que as componentes conexas resultantes no grafo correspondam exatamente aos componentes de página.

2.5 Abstração da estratégia *bottom-up*

De forma resumida, a estratégia *bottom-up* pode ser pensada em termos da divisão da imagem em um conjunto de primitivas, o estabelecimento de adjacência e similaridade entre as primitivas, e o cálculo de agrupamentos de primitivas. Esse processo é flexível quando às primitivas a serem usadas, às relações de adjacência a serem estabelecidas, e aos critérios para o cálculo de agrupamentos.

Desta forma, podemos esquematizar o processo de forma abstrata conforme ilustrado na figura 2.11. Na primeira etapa, "Cálculo e caracterização de primitivas", a imagem é dividida em primitivas, e em seguida características diversas podem ser associadas a essas primitivas. Esta é uma etapa importante pois a escolha e descrição das primitivas afeta a etapa subsequente. Na segunda etapa, "Adjacência e agrupamento de primitivas", o estabelecimento da relação de adjacência depende do tipo de primitivas e pode depender também de sua descrição. Analogamente, no cálculo de agrupamentos, idealmente gostaríamos que cada agrupamento corresponda exatamente a um componente de página. Para isso, tanto as características que descrevem as primitivas assim como estabelecem a adjacência entre elas devem ser escolhidas de forma a não excluir a possibilidade de formação dos agrupamentos de interesse.

No próximo capítulo será apresentado o método proposto neste trabalho, que é baseado no esquema acima e faz uso de técnicas de aprendizado de máquina em alguns pontos chave, de forma a tornar o método mais flexível e robusto.

2.6 Emprego de técnicas de *Deep learning*

Nesta seção apresentamos um panorama sobre o uso de técnicas de *deep learning* na área de análise de leiaute de documentos e/ou segmentação de páginas de documentos. Este panorama é baseado em um conjunto de artigos, não exaustivos, que avaliamos serem suficientemente representativos. Ao final, discutimos as tendências observadas, tecendo relações com o método proposto nesta tese quando pertinente.

2.6.1 Preliminares

O artigo de Eskenazi *et al.*, 2017 apresenta uma revisão da literatura sobre algoritmos para segmentação de página, com foco no período a partir de 2008. Segundo os autores, outros artigos já haviam apresentado revisões da literatura da área cobrindo períodos anteriores a 2008, a exemplo do artigo de Kise, 2014. Embora Eskenazi *et al.*, 2017 façam menção a redes neurais, não há menção a *deep learning*.

Um outro artigo mais recente (Binmakhashen e Mahmoud, 2019) descreve inicialmente um arcabouço de análise de leiaute de documentos. A revisão da área é feita

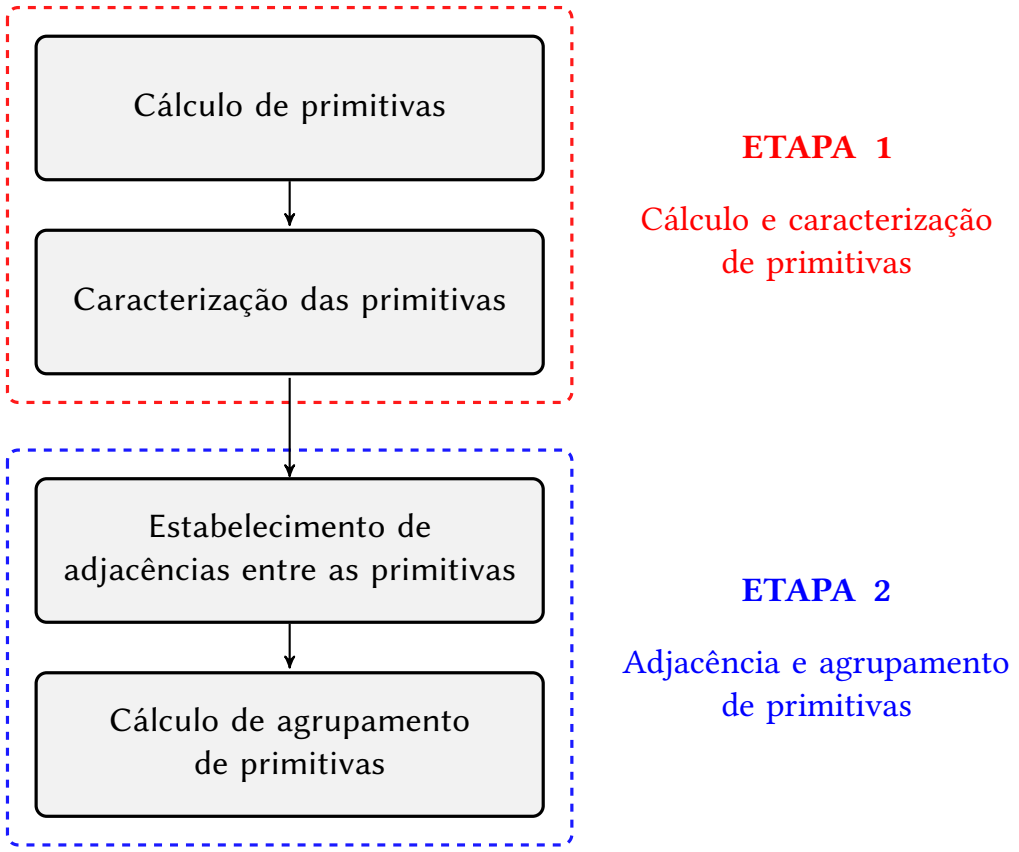


Figura 2.11: *Diagrama representando uma abstração de estratégias bottom-up para segmentação de páginas.*

então apoiando-se nesse arcabouço. O arcabouço descrito consiste em cinco etapas principais: (1) pré-processamento (binarização, correção de inclinação, melhoria de imagem), (2) estimativa de parâmetros (de modelos ou daqueles intrínsecos aos dados, tais como espaçamento entre linhas), (3) tipos de estratégia (*bottom-up*, *top-down* ou híbrido), (4) pós-processamento para melhorar os resultados, (5) análise de desempenho. As etapas de (2) a (4) compreendem a análise de leiaute propriamente dita.

Esse segundo artigo contém uma seção dedicada aos métodos baseados em *deep learning*. São citados alguns artigos que empregam redes totalmente convolucionais (por exemplo, variantes de FCN (Long *et al.*, 2015) ou de U-Net (Ronneberger *et al.*, 2015)). Os autores apontam também que, devido aos erros de classificação, em geral é comum uma etapa de pós-processamento no qual a segmentação gerada pelas redes é melhorada por meio, por exemplo, de filtragem morfológica. Essas abordagens são discutidas, no contexto do arcabouço considerado, dentro de uma seção que revisa as estratégias do tipo *bottom-up*, uma vez que as redes convolucionais essencialmente realizam classificação de pixels.

De fato, as redes totalmente convolucionais foram projetadas para fazer mapeamentos de imagem a imagem. Elas são utilizadas para segmentação semântica de imagens, uma vez que são capazes de fazer uma classificação densa, simultânea, de todos os pixels. Portanto é natural que as primeiras abordagens usando técnicas de *deep learning* para a segmentação de imagens de documentos tenha utilizado esse tipo de rede.

Com base nos dois artigos de revisão citados acima, podemos entender que o emprego de técnicas de *deep learning* em segmentação de componentes de página ou análise de leiaute de páginas de documentos teve início por volta de 2017 em diante.

2.6.2 Um panorama

Nesta seção, revisamos alguns artigos recentes que utilizam primariamente técnicas de *deep learning* para abordar o problema de segmentação de imagens de páginas de documentos. O problema de segmentação de componentes de página pode ser visto como um caso particular de segmentação semântica. Portanto, não é surpreendente que os primeiros trabalhos na área usando *deep learning* sigam a abordagem de segmentação semântica. Além disso, de forma similar ao observado em outros tipos de problemas na área de Visão Computacional, observamos uma evolução gradativa quanto ao escopo do problema e técnicas utilizadas. O restante desta seção traz, portanto, uma amostra ilustrativa dessa evolução. Optamos por enfatizar as principais ideias e características dos métodos propostos, sem a preocupação de incluir detalhes completos quanto aos modelos de redes neurais utilizadas, avaliações realizadas, ou resultados obtidos.

Lee *et al.*, 2019 utilizam uma combinação de uma rede do tipo U-Net com outro módulo denominado TML (*Trainable Multiplicative Layer*). Os autores argumentam que embora as redes convolucionais (base da U-Net) sejam capazes de extrair características locais das imagens, elas não são capazes de extrair relações de co-ocorrência, comuns para a descrição de texturas. Uma vez que regiões de texto podem ser reconhecidas baseado em aspectos texturais, os autores propõem o uso de camadas TML que são treináveis e úteis para a detecção de padrões de co-ocorrência nas imagens. Experimentos consideram um problema de classificação binária texto/não-texto, no qual componentes como fórmulas e tabelas são unificados em uma classe geral "texto". Resultados reportados mostram que a U-Net com a modificação proposta apresenta desempenho superior em relação à U-Net com sua arquitetura original.

Zou e J. Ma, 2021 consideram o problema de segmentação de páginas em quatro tipos de componentes de página (texto, tabela, figura e fórmula), além de *background*, em documentos em inglês e em chinês. Algumas redes conhecidas para segmentação semântica são utilizadas e, segundo os autores, a DeepLab v3+ é a que apresentou melhores resultados. Uma vez que o *ground-truth* das bases de dados utilizadas nos experimentos consistem em retângulos envoltórios, as componentes conexas maximais com classificação homogênea são delimitadas por retângulos envoltórios e a cada um deles é atribuída o rótulo de classe com maior escore. Esse processo pode resultar em retângulos sobrepostos (por conta de erros de classificação de pixels). Portanto, uma heurística que elimina as sobreposições, priorizando a preservação integral dos retângulos com escore maior, é aplicada para produzir o resultado final consistindo em conjuntos de retângulos sem sobreposição dois a dois.

Tanto os artigos de Lee *et al.*, 2019 com de Zou e J. Ma, 2021, comentados acima, abordam o problema de segmentação de componentes de página como um problema de classificação de pixels e utilizam redes que são comumente usadas em problemas de segmentação semântica. O artigo de Zou e J. Ma, 2021 representa um exemplo de trabalho que aplica um pós-processamento para melhorar o resultado gerado pela rede de

segmentação semântica.

Um segundo grupo de artigos são aqueles que abordam o problema de segmentação de componentes de página como um problema de detecção de objetos. Em particular, essa abordagem é bastante natural para a segmentação de páginas com leiaute retangular.

Li *et al.*, 2021 observam que apesar de redes comumente usadas para detecção de objetos estarem sendo empregadas com relativo sucesso na detecção de componentes de página, ainda se observa dificuldades para uma classificação mais refinada, como por exemplo para discriminar algumas classes como lista, tabela, texto e título. A partir da observação de que diferentes componentes possuem características diferentes, os autores propõem a combinação de características visuais de baixo e alto níveis com características textuais extraídas para cada retângulo detectado. Essas características são usadas conjuntamente para gerar uma predição. Com essa abordagem, os autores obtêm resultados melhores do que os obtidos apenas com redes de detecção de objetos.

Biswas *et al.*, 2021 formulam o problema de segmentação de componentes de página como um problema de segmentação de instâncias. Segmentação de instâncias, diferentemente da segmentação semântica, preocupa-se não apenas com a atribuição de rótulo de classe correta para cada pixel da imagem, mas também em identificar individualmente as múltiplas ocorrências de instâncias de cada tipo de objeto. Por exemplo, uma página de documento pode conter múltiplos componentes do tipo bloco de texto. Em segmentação semântica, todos os pixels correspondentes a esses blocos serão rotulados como texto. Já em segmentação de instâncias, cada bloco é identificado individualmente. Em princípio, quando os blocos de texto estão suficientemente separados, as abordagens baseadas em segmentação semântica ou em detecção de objetos tendem a gerar resultados similares. Porém, quando o leiaute é do tipo não-retangular, delimitar os componentes por retângulos envoltórios pode resultar em sobreposição dos componentes. A segmentação de instâncias ajuda a contornar esse problema uma vez que o resultado final é a classificação dos pixels. Uma diferença fundamental é que as redes que realizam a segmentação de instâncias são treinadas considerando-se os retângulos envoltórios das regiões de interesse. Assim, essas técnicas incorporam a habilidade de detectar as instâncias de interesse (guiados pelos retângulos envoltórios) e ao mesmo tempo fazer a segmentação precisa dos contornos de cada componente de página por meio de classificação de pixels. Os resultados apresentados comparam o desempenho de redes de segmentação de instâncias com redes de detecção de objetos, e mostram a superioridade das primeiras.

Markewich *et al.*, 2022 também exploram conjuntamente as redes de segmentação semântica e as informações dos retângulos envoltórios. Diferentemente de Biswas *et al.*, 2021 que formulam o problema como um problema de segmentação de instâncias, Markewich *et al.*, 2022 propõem uma variação da rede U-Net, que é treinada usando uma função de perda que inclui um termo de regressão sobre as coordenadas do retângulo envoltório, favorecendo a localização de componentes.

Por fim, Luo *et al.*, 2022 concentram-se explicitamente no problema de análise lógica dos componentes de página. Observam que os métodos usados para a segmentação de página utilizam geralmente apenas características visuais e ignoram outras como as relações entre os componentes ou os elementos contextuais. Ainda segundo os autores, outros trabalhos que tentam fazer uma classificação lógica dos componentes estão começando

a combinar características visuais e textuais dos componentes. Motivados por essa observação, propõem a utilização de características sintáticas, semânticas, de densidade, e de aparência, juntamente com *Graph Convolutional Networks* (GCN). Primeiramente são construídos grafos para cada categoria de características considerada. Nesses grafos, os vértices correspondem aos componentes de página com as respectivas características computadas a partir deles. Esses grafos são então processados por GCNs que aplicam convoluções (sobre grafos) e atualizam a representação dos nós, levando em conta os nós vizinhos. Em seguida, essas representações atualizadas são combinadas usando *max pooling* e concatenação e então classificadas usando uma rede neural convencional. Em um dos experimentos descritos, são apresentados resultados referentes à classificação das seguintes categorias lógicas: Abstract, Author, Caption, Date, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table, Title.

2.6.3 Discussão

Apresentamos inicialmente um quadro-resumo (tabela 2.1) que sintetiza algumas informações que ilustram a evolução observada nos artigos brevemente discutidos na seção anterior. A tabela indica, para cada artigo, as categorias de componentes de página considerados, a abordagem principal, e as bases de dados utilizadas na parte experimental. A partir da seção anterior e esse quadro-resumo, podemos observar que, de fato, percebe-se diferentes eixos de evolução.

Artigo	Componentes de página	Abordagem	Datasets
Lee *et al.*, 2019	texto/não-texto	Classificação de pixel	ICDAR2017 dataset, PRimA Layout Analysis (PRimA) dataset
Zou e J. Ma, 2021	texto, tabela, figura, fórmula, *background*	classificação de pixel	POD, private
Li *et al.*, 2021	texto, título, lista, figura, tabela	detecção de objetos	PubLayNet
Biswas *et al.*, 2021		Instance segmentation	PubLayNet, HJDataset
Markewich *et al.*, 2022		pixel segmentation + BB	DAD and PubLayNet
Luo *et al.*, 2022	múltiplos	GNN	PubLayNet, FUNSD, and DocBank

Tabela 2.1: *Quadro resumo dos artigos examinados na seção anterior*

Primeiramente, em relação às abordagens utilizadas, inicialmente eram empregadas predominantemente as redes de segmentação semântica diretamente no problema de segmentação de componentes de página (Lee *et al.*, 2019; Zou e J. Ma, 2021). Em seguida passaram a ser empregadas também as redes de detecção de objetos (Li *et al.*, 2021). Surgem então as abordagens que exploram de forma combinada a capacidade de localização de objetos das redes de detecção de objetos e a capacidade de delineação de contornos precisos das redes de segmentação semântica (Biswas *et al.*, 2021; Markewich *et al.*, 2022). Mais recentemente, passam a ser utilizadas também as redes que exploram a modelagem por meio de grafos (Luo *et al.*, 2022).

Em outro eixo, observa-se também uma ampliação na quantidade de categorias de componentes de página considerados. São consideradas por exemplo texto, tabela, figura, fórmula, e *background*, e mais recentemente categorias lógicas como título, nota de rodapé, parágrafo, etc. De fato, nota-se uma transição de análise de leiaute físico para análise lógica de leiaute, indicando uma fronteira vaga entre essas duas caracterizações. A análise lógica de leiaute depende de informações extras, e não apenas da imagem. Não à toa, os resultados apresentados nessa linha parecem estar em sua maioria concentrados em componentes lógicos que são do tipo texto (título, legenda de figura, nota de rodapé, resumo, nome de autor, etc), já que existem técnicas bem-sucedidas para extração de informação sintática e semântica de textos.

Essa ampliação na quantidade de categorias de componentes de página reflete e é refletida em um terceiro eixo que é o aumento de bases de dados, públicas ou não, utilizadas nos trabalhos mais recentes. Algumas dessas são listadas na tabela 2.1. Consideramos inicialmente a possibilidade de compilar uma lista com todos as bases de dados públicas com as respectivas referências bibliográficas e *links* para os respectivos repositórios, porém notamos que alguns *links* indicados nos artigos estão quebrados e, portanto, decidimos não elaborar tal lista. Informações adicionais sobre as bases de dados listadas podem ser obtidas nos artigos que fazem uso delas.

Uma constatação importante é que os componentes de página considerados são, em sua maioria, todos delimitados por retângulos. Em algumas bases de dados estão disponíveis além dos retângulos envoltórios, máscaras de segmentação (em nível de pixels). Os artigos citados na seção anterior não incluem componentes delimitados por polígonos de formatos arbitrários. Em contraste, a base de dados utilizada neste trabalho contém componentes delimitados por polígonos com formatos arbitrários.

Outra constatação é a de que aparentemente a distância entre a análise de leiaute físico e lógico está diminuindo. No entanto, por enquanto as abordagens que ilustram essa aproximação estão restritas ao contexto de documentos com estruturas do tipo Manhattan, quando não retangulares. Por exemplo, artigos científicos enquadram-se nesse contexto. Fundamentalmente esses dois tipos de análises não são disjuntos, uma vez que a análise lógica de leiaute pode ser vista como dependente do leiaute físico. Alternativamente, a primeira pode ser vista como um problema de detecção de componentes de página com categorias mais refinadas (por exemplo, em vez de simplesmente “texto”, um componente pode ser “título”, “nota de rodapé”, “legenda”, “elemento de uma tabela”, entre outros). Acreditamos que à medida que ocorram avanços das técnicas quanto à capacidade de incorporar informações diversas tais como relações entre os componentes, características

globais ou até externas, a análise de leiaute não precisará ser dividida em dois tipos.

Por fim, destacamos a utilização de grafos. Em nosso trabalho utilizamos grafos para detectar os componentes de página. Os elementos primitivos que constituem um componente de página são as componentes conexas, obtidos após a binarização da imagem de documento. Para não depender de binarização de imagens, uma possível abordagem seria considerar regiões geradas pelas redes de segmentação semântica (ou eventualmente uma super-segmentação dessas regiões) como as primitivas a serem usadas em nosso método. Além disso, de forma similar ao realizado em Luo *et al.*, 2022 (isto é, utilização de grafos para a classificação lógica dos componentes de página), características adicionais poderiam ser adicionadas à modelagem dos grafos em nosso método. Outra possibilidade é o uso de redes neurais sobre grafos para otimizar a detecção dos componentes de página.

Apesar do nosso método não utilizar técnicas de deep learning de ponta a ponta, veremos no capítulo a seguir que a proposta do nosso método descreve um pipeline que permite a utilização de diferentes técnicas em diferentes etapas do processo. Isso pode favorecer alguns pontos que ainda são críticos no uso de técnicas de deep learning como o custo computacional, e a necessidade de grande volume de dados de entrada.

Capítulo 3

Método Proposto

Neste capítulo detalhamos o método proposto neste trabalho. Apresentamos inicialmente a estrutura geral do método proposto e nas seções subsequentes detalhamos os passos que compõem o método.

3.1 Visão geral

O ponto de partida do método proposto é o esquema abstrato apresentado no final do capítulo anterior (figura 2.11). Por conveniência, relembramos como a abstração apresentada lá pode ser expressa em termos de uma sequência de passos:

1. Divisão da imagem em um conjunto de primitivas
2. Caracterização das primitivas (por exemplo, associação de características tais como informações geométricas, cores, entre outros)
3. Estabelecimento de adjacência entre as primitivas
4. Agrupamento de primitivas, de forma que cada grupo corresponda a um componente de página

Consideramos que esse esquema abstrato abarca várias possibilidades com respeito aos elementos em cada um dos passos. Por exemplo, as primitivas podem ser superpixels ou componentes conexas, e o estabelecimento de relação de adjacência entre primitivas pode ser baseada em diferentes critérios. Analogamente, diferentes características podem ser associadas às primitivas.

A abordagem proposta neste trabalho visa explorar essa flexibilidade de escolhas por meio do emprego de técnicas de aprendizado de máquina. Uma vez que os algoritmos de aprendizado de máquina são treinados usando amostras de dados, é esperado que o método resultante possa ser aplicado a diferentes cenários quanto aos tipos de componentes de página, leiaute ou documentos.

Especificamente, o método proposto consiste em uma sequência de passos na qual parte deles é baseado em técnicas de aprendizado de máquina, enquanto outros são mantidos fixos.

O tipo de primitiva escolhido são as componentes conexas, por entendermos que elas funcionam bem para uma variedade de tipos de documentos. A caracterização das primitivas inclui, além de características geométricas, uma pré-classificação das componentes conexas. A relação de adjacência entre as componentes conexas é representada em um grafo, e o cálculo de agrupamentos consiste em particionamento do grafo por meio da remoção de certas arestas. Em relação ao esquema abstrato do capítulo anterior (figura 2.11), a especialização decorrente dessas escolhas pode ser esquematizada em um diagrama similar conforme mostrado na figura 3.1.

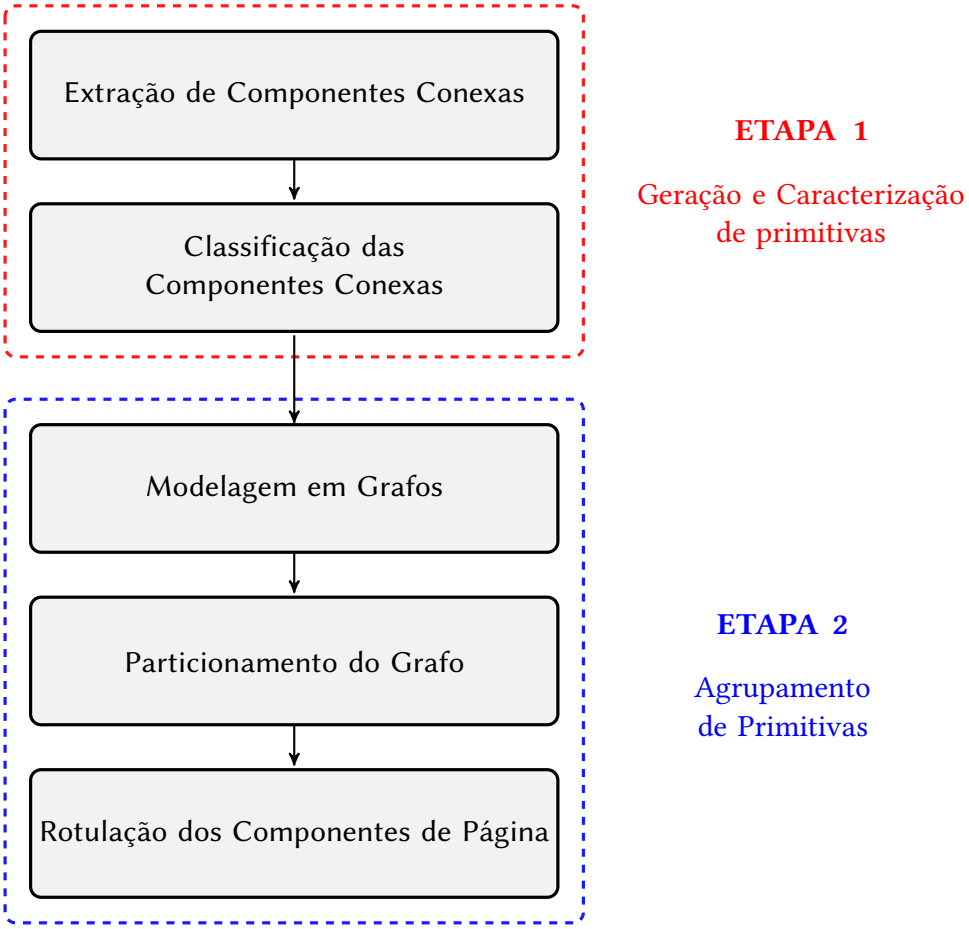


Figura 3.1: *Diagrama do método proposto de Segmentação de Páginas.*

O esquema acima é mostrado com maiores detalhes na forma de um *pipeline* na figura 3.2. Nesse *pipeline* destacamos em vermelho os três passos (classificação de componentes conexas, classificação de arestas e atribuição de rótulos aos componentes de página) que são flexíveis e podem ser ajustados de acordo com as especificidades dos documentos a serem segmentados. Os demais passos mostrados em cor preta são aqueles que, nesta tese, serão mantidos fixos.

Cada um desses passos é descrito nas seções subsequentes. Para facilitar a leitura, as descrições são organizadas em três subseções: (a) **componentes conexas**, que descreve como as componentes conexas são calculadas e quais características são associadas a elas, (b) **grafo de componentes**, que descreve a construção do grafo de componentes conexas

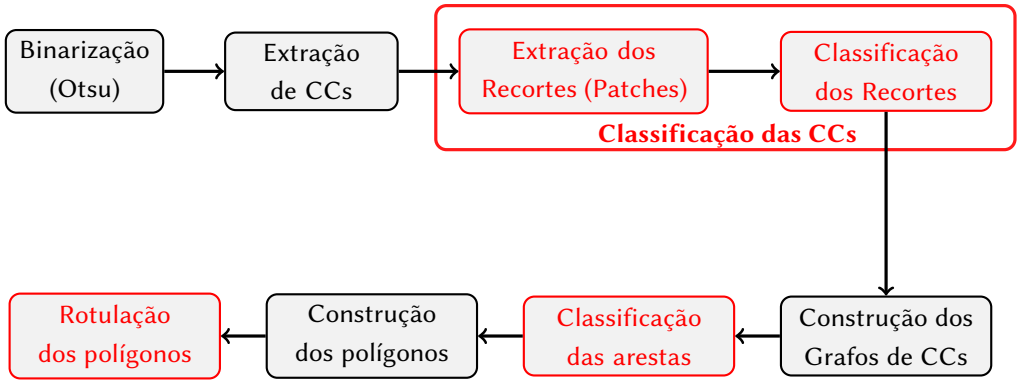


Figura 3.2: *Passo a passo do processo de Segmentação de Páginas. Estão destacados em vermelho os passos que podem ser configurados baseados nos dados. Os demais passos são mantidos fixos neste trabalho.*

e o processo de particionamento do grafo em componentes de página, e (c) **componentes de página**, que descreve a delimitação do contorno e classificação dos componentes de página detectados.

3.2 Componentes conexas

A primeira etapa do método visa o cálculo e caracterização das componentes conexas. Isso envolve um passo de binarização da imagem, a partir da qual são calculadas as componentes conexas, e a associação de características (*features*, em inglês) a cada componente conexa. Adicionalmente, também associamos às componentes conexas as predições geradas por meio de um classificador. Esses passos estão descritos a seguir.

3.2.1 Binarização de imagens e rotulação de componentes conexas

A binarização das imagens é realizada aplicando-se o algoritmo de Otsu (Otsu, 1979), que calcula automaticamente um limiar de binarização das imagens originais, porém em escalas de cinza. Um exemplo do resultado de binarização pode ser visto na figura 3.3. Salientamos que outros métodos de binarização, inclusive as baseadas em aprendizado de máquina, podem ser utilizadas. No entanto, neste trabalho optamos por tratar a binarização como um pré-processamento fixo.

Após a binarização, componentes conexas são rotuladas (conforme explicação na seção 2.3.2).

3.2.2 Características geométricas

No passo de caracterização das componentes conexas, parte das informações discriminativas associadas são as seguintes características:

- *Altura:* Razão entre altura da caixa delimitadora da componente conexa e a altura do documento em pixels.

(a) **(b)**

Figura 3.3: *Exemplo do resultado da aplicação da binarização de uma imagem de documento, utilizando o limiar de Otsu.(a) Imagem original em RGB. (b) Imagem binarizada.*

- *Largura:* Razão entre largura da caixa delimitadora da componente conexa e a largura do documento em pixels.
- *Razão de Aspecto:* Largura dividida pela altura
- *Alongamento:* Razão entre a altura e a largura, computada como (3.1)

$$\min(\text{altura}, \text{largura})/\max(\text{altura}, \text{largura}). \tag{3.1}$$

- *Solidez:* Área da componente conexa (em pixels) dividida pela área do seu fecho convexo,
- *Área:* Número de pixels na componente conexa.
- *Extensão:* Razão entre os pixels da componente conexa e o total de pixels da caixa delimitadora. Computada como (3.2)

$$area/(altura * largura) \tag{3.2}$$

Essas características são interessantes no contexto de Segmentação de Páginas. Por exemplo, a altura e largura de caracteres num mesmo parágrafo tendem a ser semelhantes em termos de tamanho; já os separadores costumam ter uma das dimensões (altura ou largura) com comprimento grande e a outra dimensão com comprimento bem pequeno; figuras, por sua vez, tendem a resultar em componentes conexas com grandes áreas

ou formatos diversos e irregulares. Assim, essas características podem ser úteis para discriminar a qual componente de página cada componente conexa pertence.

3.2.3 Escores de classificação

Um segundo conjunto de características associadas às componentes conexas são os escores de classificação (valores entre 0 e 1) gerados por meio de um classificador. Quanto maior o valor em relação a um tipo de componente de página, maior é a confiança do classificador de que a componente conexa é daquela classe. É importante notar que em uma situação ideal, o classificador seria capaz de fazer predições 100% corretas, e o problema de segmentação de páginas estaria resolvido. No entanto, na prática essas predições dificilmente serão 100% corretas. Ao associarmos esses escores às componentes conexas, é esperado que eles possam ser úteis nos passos subsequentes nos quais predições sobre informações de mais alto nível são realizados.

No processo de classificação de componentes conexas, diversas características podem ser exploradas, incluindo informações da componente em si e do contexto ao redor da componente. Em nosso método, propomos o uso de Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN), por conta dos bons resultados obtidos em problemas de classificação de imagens em diversos contextos.

As CNNs são em geral aplicadas em imagens de tamanho fixo. Dado que as componentes conexas possuem tamanhos variados, utilizamos uma ideia inspirada no trabalho de Bukhari *et al.*, 2010 para a normalização dos tamanhos. No caso, os autores tratam o problema de separação texto/não-texto por meio da classificação de componentes conexas.

Preparação de dados

Apesar de estarmos usando o termo "classificação de componentes conexas", efetivamente o que será classificado de fato é uma pequena região da imagem original, centrada justamente sobre a posição na qual a componente conexa está localizada. Assim, por exemplo, enquanto a componente conexa correspondente a um caractere é apenas uma máscara binária com o formato do caractere, a região na imagem centrada nesse caractere não apenas conterá os dados originais do conjunto de pixels que formam a componente conexa, mas terá também dados da imagem no entorno do caractere. Ou seja, a componente conexa poderá ser classificada tendo-se um "retrato" dela e de seu entorno.

Seja $N \times N$ o tamanho da imagem a ser utilizada na classificação de componentes conexas. Isso significa que, não importa o tamanho da componente conexa, as imagens de todas elas precisam ser ajustadas para $N \times N$.

Para lidar com componentes conexas de tamanhos distintos e ao mesmo tempo contemplar o entorno delas, definimos um segundo tamanho $M \times M$ ($M < N$). Então, para cada componente conexa é gerada uma imagem de tamanho $N \times N$, de tal forma que a componente conexa esteja inteiramente inserida em uma região quadrada de tamanho $M \times M$, centrada na região de tamanho $N \times N$.

Quando a altura e a largura da componente conexa não são maiores que M, essa

imagem é simplesmente um recorte (*patch*) extraído da imagem original, com tamanho $N \times N$ e centrado no centroide da componente conexa. Quando ou a largura ou a altura da componente conexa é maior que M, então um fator de escalonamento é calculado de forma que a componente conexa de escala reduzida (sem distorção) fique inteiramente contida num quadrado $M \times M$. O tamanho do recorte, neste caso, é definido de tal forma que, ao se aplicar a mesma redução de escala, ele resulte em uma imagem de tamanho $N \times N$.

Na figura 3.4, temos alguns exemplos de recortes, com os valores de M e N fixados em 8 e 40, respectivamente. Na imagem da página completa temos os recortes em seu tamanho original e ao redor da imagem temos os recortes redimensionados para o tamanho 40×40. Note que o recorte em torno da componente conexa localizada na parte central da página (na parte inferior da figura que está no topo da página) é bem maior que os demais recortes, justamente porque a componente conexa tem um tamanho bem maior que a de outras componentes conexas destacadas. Após a alteração de escala adequada, as imagens geradas a partir do recorte para o treinamento da rede (exemplos destacados nas laterais da figura da página) possuem o tamanho $N \times N$ e de forma que a componente conexa em questão fique confinada a uma região central de tamanho $M \times M$.

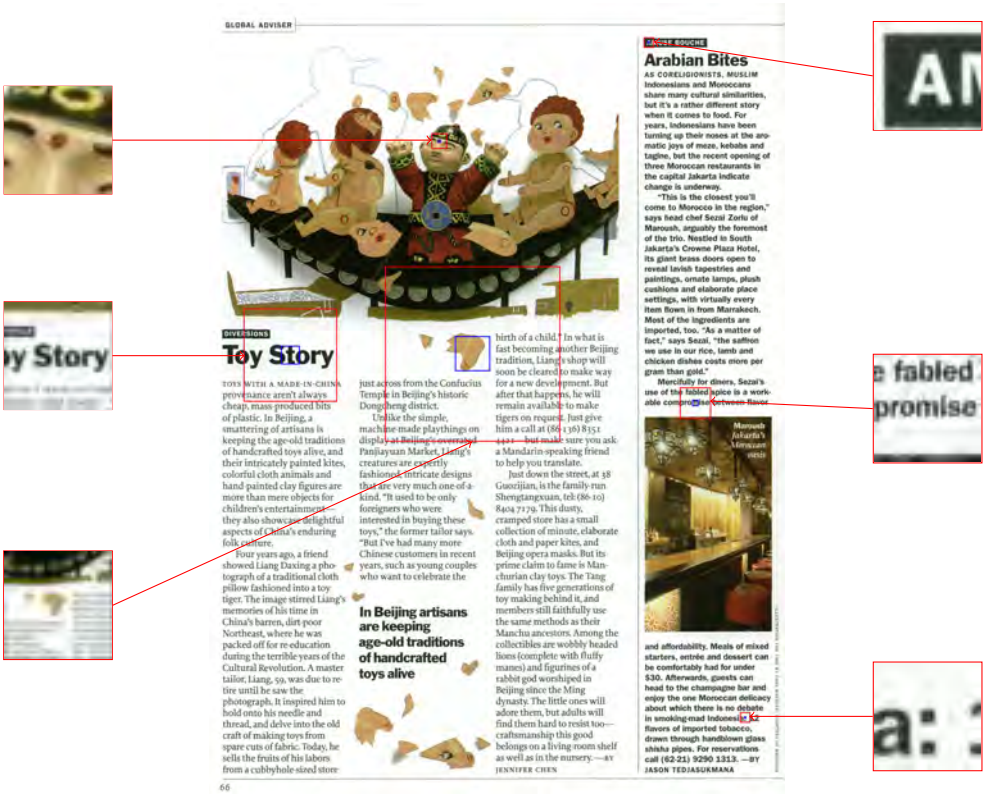


Figura 3.4: *Exemplos de recortes centrados em componentes conexas da imagem, escalados de forma que resultem em imagens de tamanho* 40×40 *e tal que as componentes conexas estejam dentro de uma região* 8×8*, centrada na imagem* 40×40.

Destacamos que, em relação ao trabalho de Bukhari *et al.*, 2010, uma importante diferença está na forma como os recortes são feitos. Eles fixam o tamanho do recorte em função da altura e largura da componente, o que na nossa modelagem corresponde a fixar o valor de M e N. Em contraste, em nossa modelagem a razão entre M e N pode

ser alterada, contemplando mais ou menos contexto ao redor da componente conexa. A forma de fazer o recorte, com o redimensionamento da componente conexa e proposta neste trabalho, foi preliminarmente aplicada sobre o problema de segmentação de páginas em texto/não-texto e foi publicado em um artigo (Julca-Aguilar *et al.*, 2017).

Treinamento do classificador

Para o treinamento do classificador, cada componente conexa (ou recorte) é rotulado com a classe do componente de página a qual ela pertence. Para tanto, é realizada uma interseção entre os polígonos delimitadores de cada componente de página e a imagem de componentes conexas. Todas as componentes conexas cobertas por um determinado polígono recebem o rótulo de classe do componente de página correspondente.

De posse dos pares consistindo dos recortes em torno das componentes conexas mais o rótulo de classe, a CNN pode ser treinada e em seguida usada para fazer predições para novas componentes conexas.

Conforme já mencionado anteriormente, uma vez que na estratégia *bottom-up* os erros na base podem ser propagados para os níveis superiores, é interessante manter todas as interpretações possíveis. Neste sentido, o uso de CNNs é adequado já que a forma padrão de treinamento de redes desse tipo para tarefas de classificação geram escores $\hat{P}(y = k|X)$ para cada uma das possíveis classes $y = k$ de componentes de página.

Na parte experimental detalhamos a arquitetura de CNN utilizada, assim como a escolha de valores dos hiperparâmetros, e o treinamento.

3.3 Grafo de componentes

Na segunda etapa do método, o passo inicial é a construção de um grafo, cuja estrutura estabelece a relação de adjacência entre as componentes conexas. Em seguida, um algoritmo para remoção de arestas é aplicado visando a obtenção de subgrafos que correspondem aos componentes de página.

3.3.1 Relação de adjacência

Para representar as relações de adjacência, por não haver sobreposição entre as componentes conexas, propomos a construção do diagrama de Voronoi e a consequente triangulação de Delaunay, descrita na seção 2.4.1. Um exemplo de um grafo, construído a partir da triangulação de Delaunay para a imagem de um documento, pode ser visto na figura 3.5.

Além da adjacência, explicitamente representada pela estrutura do grafo, associamos aos nós a caracterização das respectivas componentes conexas, a saber:

- informações geométricas da componente conexa (altura, largura, razão de aspecto, alongamento, solidez, área e extensão), conforme descrito anteriormente na seção 3.2.2
- coordenadas do centroide da componente conexa

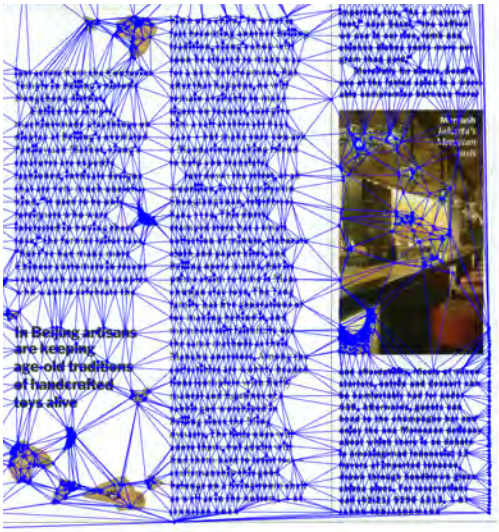


Figura 3.5: *Ilustração do grafo construído a partir da triangulação de Delaunay para um recorte da imagem do documento da figura 3.4. Os vértices estão localizados nos centroides de cada componente conexa.*

- escores relativos a cada uma das K classes de componentes de página, gerados pelo classificador de componentes conexas (seção 3.2.3)

Adicionalmente, para cada aresta (u, v) associamos as seguintes informações:

- a distância euclidiana entre os vértices u e v (mais especificamente, entre os centroides das respectivas CCs)
- o ângulo formado entre a aresta (u, v) e o eixo x no plano cartesiano

3.3.2 Classificação de arestas

A tarefa de agrupamento das componentes conexas visa particionar o grafo de tal forma que cada parte corresponda a um componente de página. Isso significa que as arestas que conectam componentes conexas pertencentes a diferentes componentes de página precisam ser removidas. Logo, a tarefa de agrupamento de componentes conexas pode ser tratada como uma tarefa de classificação de arestas quanto à permanência ou não delas no grafo. A figura 3.6 mostra em vermelho quais são as arestas da figura 3.5 que devem ser removidas, para que os subgrafos conexos resultantes (figura 3.7) correspondam aos componentes de página.

Uma observação importante em relação ao problema de classificação binária das arestas diz respeito à quantidade de cada tipo de aresta e a importância de cada um dos resultados obtidos. Supondo que a classificação positiva diz respeito às arestas que desejamos manter e a classificação negativa diz respeito às arestas que desejamos remover, é fácil observar que o uso de RAG na construção do conjunto de arestas, geralmente gera uma alta quantidade de arestas positivas, ou seja que conectam componentes conexas pertencentes a um mesmo componente de página (por exemplo caracteres em um mesmo parágrafo), e uma quantidade

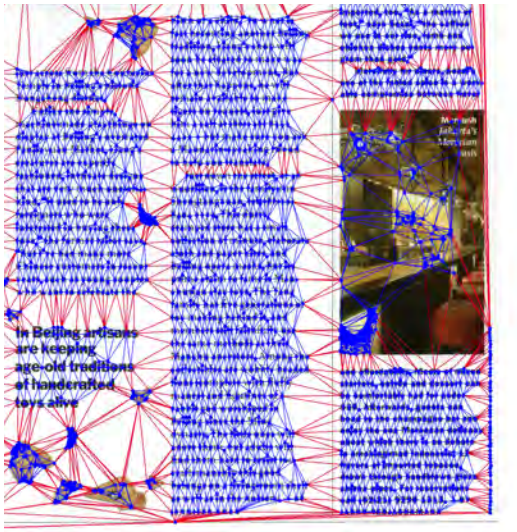


Figura 3.6: *Ilustração das arestas a serem removidas (em vermelho) no recorte da imagem do documento da figura 3.5 de forma que os subgrafos resultantes (em azul) correspondam aos componentes de página.*

bem menor de arestas negativas, que conectam componentes conexas pertencentes a componentes de página distintos. Dessa forma, o treinamento de classificadores tende a favorecer a classificação de arestas positivas. Entretanto, no contexto do nosso problema, alguns falso negativos não importam enquanto os falso positivos certamente afetarão o resultado negativamente. Por exemplo, na figura 3.6 vemos muitas arestas conectando caracteres que fazem parte do mesmo componente de página e, mesmo que muitas destas arestas sejam removidas, o resultado da segmentação não será afetado contanto que os caracteres continuem conectados. Em contraste, uma aresta falso positiva pode manter agrupados vértices que pertencem a componentes de página distintas. No exemplo da figura 3.6, qualquer aresta que conecta um caractere a uma parte de uma figura deve ser removida, caso contrário o texto e a figura farão parte do mesmo agrupamento. O problema do desbalanceamento das classes pode ser contornado treinando os classificadores utilizando pesos no cálculo da função custo, que foi utilizado em nosso método.

3.4 Componentes de página

Após o grafo ser particionado em subgrafos, temos os agrupamentos de componentes conexas. Na parte final do *pipeline*, resta delimitar a região na imagem correspondente a cada um desses agrupamentos e também determinar qual é o tipo de componente de página subjacente.

3.4.1 Polígonos envoltórios

A região na imagem correspondente a cada agrupamentos de componentes conexas pode ser delimitada por um polígono envoltório.

A estratégia mais conhecida para a construção de polígonos envoltórios é o fecho

Figura 3.7: *Ilustração dos subgrafos resultantes após a classificação das arestas para recorte da imagem do documento da figura 3.5.*

convexo. Em geometria, o fecho convexo de um conjunto de pontos em $\mathbb{R}^2$ é definido como o menor polígono convexo que contém todos os pontos, ou seja, cada ponto do conjunto é um vértice do polígono, ou está contido no polígono. Este é um problema fundamental da Geometria Computacional e possui diversos algoritmos bem conhecidos como o Algoritmo do Embrulho (Jarvis, 1973), *QuickHull* (baseado no algoritmo *Quicksort)*, MergeHull (baseado no algoritmo *Mergesort*), o Algoritmo de Graham (Graham, 1972). Os polígonos obtidos pelo fecho convexo de um conjunto de pontos não forma uma figura geométrica totalmente ajustada ao conjunto de pontos, justamente por ser formado pelos pontos, não internos, extremos do conjunto, conforme podemos ver na figura 3.8.

No contexto do nosso problema de segmentação de páginas, os polígonos que formam os componentes de página não são necessariamente convexos, o que pode ocasionar uma má formação dos componentes de página. Dessa forma, a construção de polígonos não convexos é mais adequada ao problema. Para a construção desses polígonos propomos a utilização do algoritmo *alpha-shape*.

Em Geometria Computacional, o conceito de *alpha-shape* (Edelsbrunner *et al.*, 1983), é uma generalização do Fecho Convexo e oferece uma forma mais ajustada ao conjunto de pontos. Dado um conjunto finito de pontos, uma família de formas podem ser derivadas da Triangulação de Delaunay do conjunto de pontos e um parâmetro real alfa (do inglês *alpha*), controla o nível de ajuste desejado. Formalmente, podemos definir o *alpha-shape* da seguinte forma.

Seja um disco generalizado de raio $1/\alpha$ definido conforme segue:

$$\begin{cases} \text{Se } \alpha > 0, \text{ é um disco fechado de raio } 1/\alpha \\ \text{Se } \alpha == 0, \text{ é um semiplano fechado} \\ \text{Se } \alpha < 0, \text{ é o complemento um disco fechado de raio } -1/\alpha \end{cases}$$

Dessa forma, dado um conjunto de pontos e um valor específico de α, uma aresta do *alpha-shape* é desenhada entre dois pontos P_i e P_j do conjunto, sempre que existir um disco generalizado de raio $1/\alpha$ que não contém qualquer outro ponto do conjunto e que tenha a propriedade de conter P_i e P_j em sua borda. Vale ressaltar que quando o valor de α é igual a 0, o *alpha-shape* do conjunto de pontos é equivalente ao Fecho Convexo. Um exemplo de *alpha-shape* e fecho convexo para o mesmo conjunto de pontos pode ser visto na Figura 3.8.

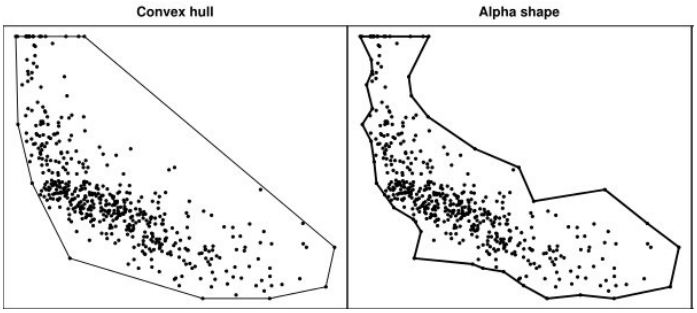


Figura 3.8: *Exemplo de fecho convexo e alpha-shape para o mesmo conjunto de pontos. (Fonte: Wikipedia)*

3.4.2 Rotulação

Por fim, o último passo consiste em rotular cada um dos polígonos (ou componentes de página preditas). A figura 3.9 apresenta os polígonos envoltórios dos componentes de página, bem como a classificação destas representadas por cores distintas, ou seja, cada cor representa um tipo de componente de página.

Para a atribuição de rótulo a cada componente de página, definidos pelos polígonos construídos, podem ser utilizadas as características associadas às componentes conexas pertencentes ao componente de página. Em particular, podem ser utilizados os escores gerados pelo classificador de componentes conexas. Na parte experimental deste trabalho avaliamos alguns critérios de votação que exploram os escores.

Figura 3.9: *Ilustração dos polígonos envoltórios dos componentes de página da imagem do documento da figura 3.7. Cada cor representa um tipo de componente.*

Capítulo 4

Metodologia Experimental

Neste capítulo serão apresentados os métodos experimentais e configurações utilizadas nos experimentos realizados para consolidação do modelo proposto. Inicialmente apresentaremos a base de dados escolhida para os experimentos, bem como suas principais características. Em seguida, apresentaremos a metodologia aplicada aos experimentos, uma vez que o método proposto estabelece que cada passo do processo possa ser configurado e remodelado para se ajustar a diversas aplicações. Especificamente, consideramos como avaliar configurações distintas de cada passo, dado que alterações na configuração de um passo podem afetar a melhor configuração de passos subsequentes, e propomos uma forma para evitar o teste de todas as configurações possíveis. Por fim apresentamos as métricas para a avaliação dos resultados.

4.1 Base de Dados

Uma das poucas bases conhecidas para o problema de segmentação de página até recentemente, com anotações estruturais, é a base de dados *PRImA Layout Analysis Dataset* (Antonacopoulos, Bridson *et al.*, 2009). Subconjuntos dessa base de dados têm sido utilizados, nos últimos anos, em diferentes edições da competição *Page Segmentation and Layout Analysis* da conferência ICDAR (*International Conference on Document Analysis and Recognition*), uma das principais conferências da área de análise de documentos. De acordo com os autores, todas as imagens foram escaneadas em uma resolução de 300 dpi e em cores (24-bits) e contém imagens de revistas e artigos científicos na proporção de 7 imagens de revistas para 1 imagem de artigo científico.

A base de dados completa é uma base privada entretanto, um subconjunto desta, composto por 305 imagens de revistas e artigos científicos, juntamente com seus respectivos *ground-truths*, estão acessíveis mediante requisição[1]. Essas 305 imagens não estão divididas explicitamente em conjunto de treinamento e teste. Desta forma, decidimos utilizar 75 imagens deste conjunto que foram utilizadas para avaliação da competição *Page Segmentation and Layout Analysis* de 2017 Clausner, Antonacopoulos *et al.*, 2017

[1] Disponível a partir de http://www.primaresearch.org/datasets/Layout_Analysis

experimentos efetuados utilizamos os mesmos conjuntos de treinamento (184 imagens), validação (46 imagens) e teste (75 imagens).

Cada imagem da base é armazenada no formato *TIFF* (*Tag Image File Format*) e é acompanhada de um arquivo *xml* contendo informações sobre a imagem, tais como, tamanho da imagem, coordenadas do polígono envoltório de cada componente de página, a rotulação dos componentes de página divididos em 10 classes: Outros (*Other*), Texto (*Text*), Diagrama (*Chart*), Elementos Gráficos (*Graphic*), Imagem (*Image*), Fórmulas Matemáticas (*Maths*), Ruído (*Noise*), Separador (*Separator*), Tabela (*Table*) e Desenho de linha (*Line Drawing*). A classe *Other* é a classe destinada aos itens não rotulados.

Para a imagem da figura 4.1, nomeada na base de dados como *00000707.tif*, temos no cabeçalho do arquivo xml informações gerais sobre a imagem como seu tamanho, criador, e data (Lista 4.1).

Figura 4.1: *Exemplo de imagem da base de dados PRImA Layout Analysis Dataset.*

```
<?xml version="1.0" encoding="UTF-8"?>
<PcGts xmlns="http://schema.primaresearch.org/PAGE/gts/pagecontent
    /2010-03-19" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
    schemaLocation="http://schema.primaresearch.org/PAGE/gts/pagecontent
    /2010-03-19 http://schema.primaresearch.org/PAGE/gts/pagecontent
    /2010-03-19/pagecontent.xsd" pcGtsId="pc-00000707">
  <Metadata>
  <Creator>PRImA Research</Creator>
  <Created>2009-06-18T10:33:42</Created>
  <LastChange>2010-04-23T15:18:50</LastChange></Metadata>
  <Page imageFilename="00000707.tif" imageWidth="2295" imageHeight="3219">
```

Programa 4.1: *Cabeçalho do arquivo xml para a imagem da figura 4.1*

As coordenadas do polígono envoltório do componente de página “tabela”, contida na imagem da figura 4.1, são apresentadas ponto a ponto conforme o trecho do arquivo xml correspondente na lista 4.2. Também estão listadas informações adicionais sobre este

componente de página tais como, número de linhas e colunas, existência das linhas de separação na tabela, dentre outras.

```
<TableRegion id="r19" rows="7" columns="6" lineColour="black" orientation="
    0.00000" lineSeparators="true" bgColour="white" embText="true">
  <Coords points= "608,2510 864,2510 864,2511 1191,2511 1191,2510 1579,2510
      1579,2511 1766,2511 1766,2510 1843,2510 1843,2511 1848,2511 1848,2512
      1849,2512 1849,2532 1850,2532 1850,2794 1851,2794 1851,3046 1757,3046
      757,3047 1580,3047 1580,3046 997,3046 997,3047 65,3047 865,3046
      425,3046 425,2999 424,2999 424,2746 423,2746 423,2512 424,2512 424,2511
       608,2511"/>
</TableRegion>
```

Programa 4.2: *Coordenadas do polígono envoltório da tabela contida no arquivo xml para a imagem da figura 4.1*

Diferentes informações são apresentadas dependendo do tipo de componente de página, como por exemplo, para componentes textuais, além da subclasse, também são apresentadas informações sobre cor do texto e do fundo, orientação de leitura, e língua principal do texto.

Outros dados importantes da base de dados referem-se às quantidades de amostras de cada tipo de componente de página, bem como suas distribuições nas imagens dos conjuntos de treinamento e validação. Para ilustrar tais características da base de dados, no contexto do nosso problema de Segmentação de Páginas, apresentamos a seguir algumas informações estatísticas sobre as imagens da base de dados e as divisões feitas (treinamento e validação). As tabelas 4.1 e 4.2 ilustram, para os conjuntos de treinamento e validação respectivamente, a quantidade de ocorrências dos componentes de página na base de dados, o percentual correspondente, a média de componentes de páginas por tipo e a distribuição dos componentes de páginas nas imagens.

Componentes de Página	**Quantidade Total**	**Percentual**	**Média por página**	**#Imagens que possuem o componente**
Text	3204	81.03%	17.413	184
Chart	10	0.25%	0.054	7
Graphic	136	3.44%	0.739	89
Image	174	4.40%	0.946	82
Maths	8	0.20%	0.043	4
Noise	18	0.46%	0.098	17
Separator	380	9.61%	2.065	115
Table	22	0.56%	0.120	11
Line Drawing	2	0.05%	0.011	2

Tabela 4.1: *Estatísticas do conjunto de treinamento*

Um dos grandes desafios na segmentação desse tipo de imagens são os diferentes leiautes existentes no conjunto de dados, misturando leiautes complexos com leiautes bastante simples, incluindo elementos textuais com variados estilos, fontes, cores e tamanhos, exemplificados na figura 4.2.

Componentes de Página	Quantidade Total	Percentual	Média por página	#Imagens que possuem o componente
Text	760	82.70%	16.52	46
Chart	1	0.11%	0.022	1
Graphic	28	3.05%	0.609	23
Image	39	4.24%	0.848	20
Maths	1	0.11%	0.022	1
Noise	4	0.44%	0.087	4
Separator	81	8.81%	1.761	27
Table	3	0.33%	0.065	2
Line Drawing	2	0.22%	0.043	2

Tabela 4.2: *Estatísticas do conjunto de validação*

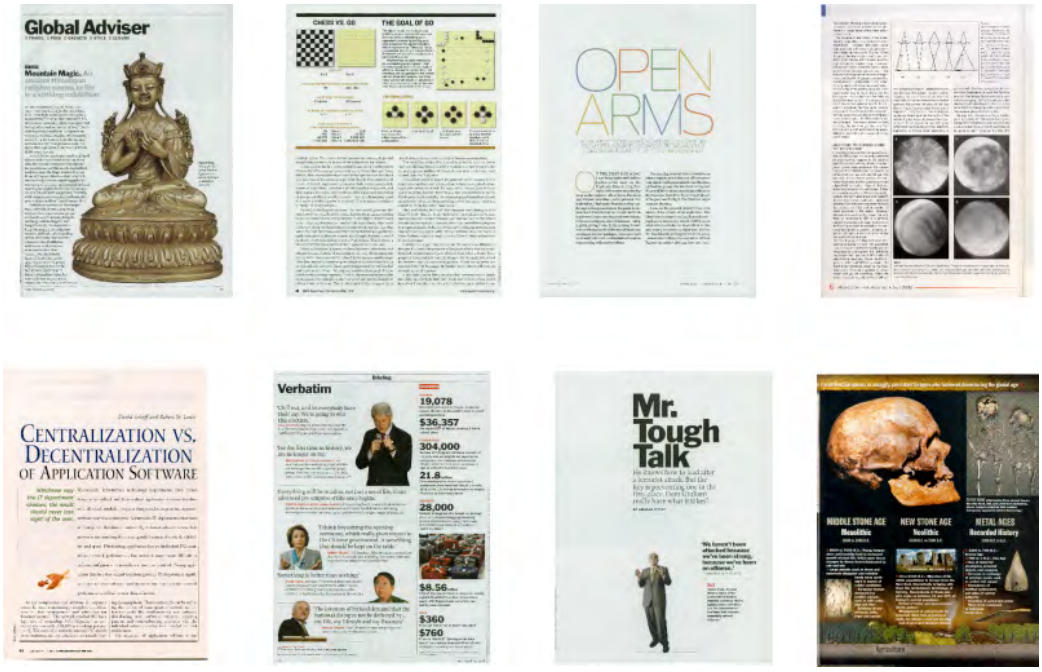


Figura 4.2: *Ilustração dos diferentes leiautes das imagens da base de dados utilizada.*

Apesar das diferenças explícitas existentes, algumas características geométricas, espaciais e contextuais podem ser exploradas pois são, em geral, comuns a componentes de páginas de mesmo tipo. Por exemplo, em geral, caracteres textuais encontram-se rodeados de outros caracteres textuais, independente da fonte, cor ou tamanho. Uma outra característica, que em geral se observa em componentes textuais, é a disposição linear destes. Dessa forma, diversos atributos e características das componentes conexas e dos componentes de página podem ser exploradas com o objetivo de obter uma boa classificação das componentes conexas e posteriormente o agrupamento destas para formar os componentes de página.

4.2 Desenho experimental

O método proposto descrito no capítulo anterior é reapresentado na figura 3.2, acrescido com os tipos de dados (em azul) que fluem de um passo para o outro. Conforme já mencionado, os três passos do *pipeline* destacados em vermelho são configuráveis. Em particular, os dois primeiros são baseados em aprendizado de máquina e portanto adaptáveis às especificidades (tais como tipo de documento, tipo de leiaute, tipos de componentes de página de interesse).

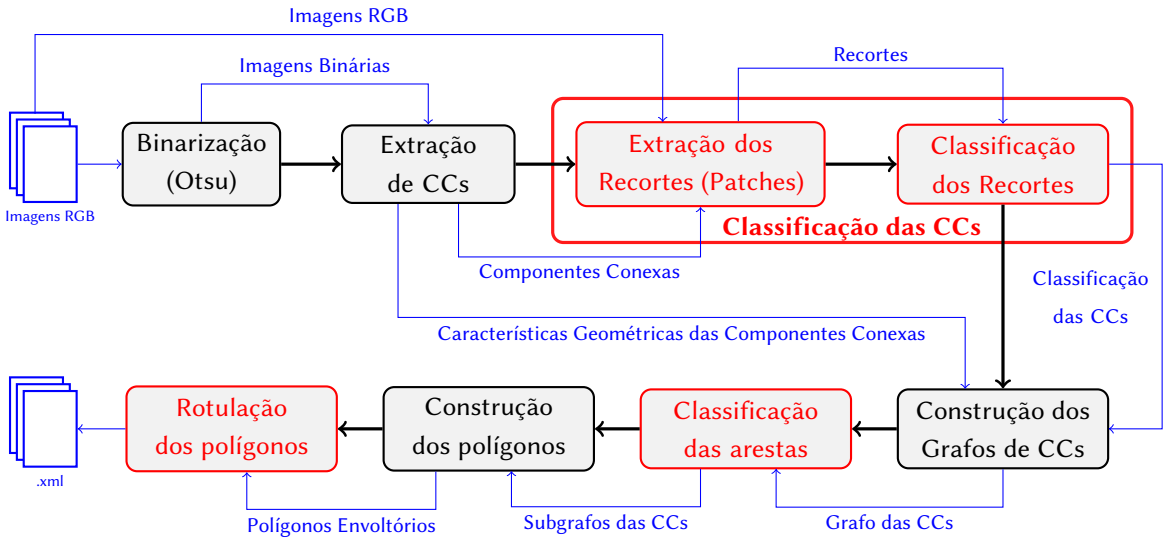


Figura 4.3: *Passo a passo do processo de Segmentação de Páginas. Este diagrama é o mesmo da figura 3.2, porém acrescido de informações sobre o tipo e fluxo de dados (em azul). Em vermelho estão os passos configuráveis.*

4.2.1 Otimização de pipelines de processamento

O método proposto contém três passos que podem ser otimizados de acordo com a aplicação-alvo, destacados em vermelho na figura 4.3. Para esclarecer o que estamos querendo dizer quando mencionamos otimização, tomemos como exemplo a classificação de componentes conexas. Conforme já descrito no capítulo anterior, propomos o uso de CNNs para realizar a classificação. Existem várias configurações que podem ser testadas

apenas neste passo. Por exemplo, podemos testar a arquitetura da CNN, o balanceamento de classes, o tamanho do recorte a ser utilizado, entre outros.

Uma questão importante quanto a *pipelines* de processamento diz respeito à sua otimização. Podemos assumir que sempre é possível fazer a otimização de um passo, individualmente? Além disso, a otimização de passos, realizada de forma individual, leva necessariamente à otimização do *pipeline* como um todo? Discutimos a seguir algumas possibilidades e as implicações associadas. Essa discussão servirá para embasar de que forma os experimentos foram planejados para a avaliação de certas configurações que julgamos interessantes em cada um dos três passos acima.

Otimização individual de cada passo A otimização de um passo específico em um *pipeline* só seria viável caso exista uma forma para avaliar o desempenho do algoritmo executado naquele passo. Considerando que cada passo processa um certo tipo de dado e gera outro dado na saída, isso implica que para cada possível dado de entrada deveríamos saber qual a saída correta (ou ideal).

Nem sempre os dados de saída esperados estão disponíveis (ou sequer são conhecidos). Para exemplificar essa situação, consideremos o primeiro passo do nosso pipeline que é a binarização da imagem. Embora em nosso método estejamos fixando o algoritmo usado nesse passo, vamos supor que desejamos otimizar esse passo experimentando diversos algoritmos de binarização. Porém, não é razoável que uma binarização ideal esteja disponível para cada imagem de entrada de forma a possibilitar a avaliação dos algoritmos. Nessa situação, a otimização automatizada deste passo de forma isolada é inconveniente, quase impraticável. O que é possível fazer, eventualmente, é avaliar como diferentes algoritmos neste passo afetam o resultado mais adiante no *pipeline*.

Outro problema na otimização individual relaciona-se ao fato de a propagação de erros ao longo do *pipeline* não ser considerada. Tomemos como exemplo o passo de classificação de arestas. As instâncias de grafos de entrada para avaliar o classificador de arestas deveriam ser consistentes com os tipos de grafos que seriam observados quando o *pipeline* fosse usado para processar uma imagem real.

Otimização sequencial Uma forma mais organizada consiste em se fazer a otimização individual dos passos de forma sequencial. Por exemplo, supondo que o *pipeline* já esteja otimizado até o passo de construção do grafo, o passo de classificação de arestas pode ser otimizado usando-se como dados de entrada aqueles gerados pela parte já otimizada do *pipeline*. Desta forma, leva-se em conta a propagação de erro. Em consequência, o classificador de arestas pode eventualmente aprender a ignorar ou corrigir os erros do passo anterior.

Um potencial problema quando fazemos a otimização desta forma é que, devido ao fato de os dados de treinamento serem limitados, corremos o risco de fazer a otimização de um passo baseado nos mesmos dados usados para a otimização dos passos anteriores. Essa situação pode não refletir a propagação de erros que seria observada para dados novos. Em outras palavras, haveria uma tendência de *overfitting* do *pipeline* aos dados de treinamento.

Além disso, da mesma forma que no geral um passo qualquer não pode ser otimizado individualmente, pela inexistência de dados para a avaliação de forma isolada, a otimização sequencial também não é possível quando os dados de saída de cada passo não estão disponíveis. Nestas situações, para possibilitar a otimização de um passo qualquer, leva-se em conta o resultado na ponta de saída do *pipeline*. Mas, para isso ser possível, alguma escolha deve ser feita preliminarmente para os passos posteriores. Voltaremos a este ponto mais adiante.

Otimização exaustiva A forma para garantir a melhor otimização do *pipeline* como um todo é, de forma geral, testar todas as combinações possíveis. Por exemplo, suponha que o *pipeline* possui dois passos, sendo 5 opções de algoritmos para o primeiro passo e 2 para o segundo passo. Teríamos um total de 10 combinações possíveis. Em contraste, para a otimização sequencial temos 5 possibilidades no primeiro passo, e após fixado um deles, temos mais dois no segundo passo, totalizando 7 combinações a serem testadas. Essa diferença tende a aumentar quanto maior o número de passos e de opções por passo.

Em geral a otimização exaustiva não é computacionalmente viável por demandar muito tempo de processamento. Por outro lado, a otimização sequencial não garante uma otimização ótima do *pipeline*.

4.2.2 Avaliação sequencial reversa

Baseado na discussão anterior, nesta seção descrevemos uma forma para avaliar diferentes configurações em cada passo, sem depender de uma otimização exaustiva. Ela é baseada em otimização sequencial, porém em ordem reversa.

Seja o *pipeline* mostrado na figura 4.4, referente a um método qualquer composto de três passos (A, B e C) passíveis de otimização.

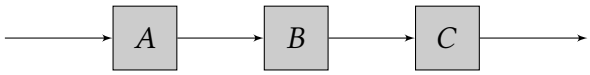


Figura 4.4: Pipeline *geral composto de três etapas.*

Considerando que estamos interessados no desempenho do *pipeline* no seu ponto final, e que optamos por uma otimização sequencial, precisamos escolher uma configuração inicial A_0, B_0 e C_0 para os passos A, B e C respectivamente.

Tal configuração inicial pode ser estabelecida manualmente em alguns casos, ou empiricamente após testar algumas configurações. Por exemplo, para o passo de classificação de componentes conexas em nosso caso, poderíamos utilizar algum modelo CNN padrão.

Suponha que para o passo A desejamos efetuar testes de três configurações distintas (A_1, A_2 e A_3), para o passo B quatro configurações (B_1, B_2, B_3 e B_4) e para o passo C duas configurações (C_1 e C_2).

Na avaliação sequencial, primeiramente seriam testadas as configurações A_1, A_2 e A_3 para o passo A, fixando-se as configurações B_0 e C_0 para os passos B e C, respectivamente.

Na avaliação sequencial reversa, primeiramente seriam testadas as configurações do passo C, enquanto a dos passos anteriores seriam mantidas fixas.

Em nosso entendimento, ao otimizar do último passo para o primeiro, de certa forma estamos conferindo a cada passo do *pipeline* a capacidade de corrigir erros cometidos nos passos anteriores. Em contraste, ao se otimizar do primeiro passo para o último, estamos favorecendo uma melhor otimização dos passos iniciais. Em situações de dados abundantes, é possível que as duas formas de otimização sejam equivalentes. No entanto, no caso do problema tratado neste trabalho, temos situações de dados escassos. Isso, em particular, implica que não podemos nos dar ao luxo de usar subconjuntos distintos para o treinamento de cada um dos passos. Desta forma, o treinamento do *pipeline* tende ao *overfitting*.

Baseado nesse entendimento, optamos por adotar a avaliação sequencial reversa para os nossos experimentos. Dessa forma, retomando o exemplo acima, seriam realizados inicialmente os dois testes do passo C em adição à configuração C_0, mantendo as configurações base para os passos anteriores, conforme a figura 4.5.

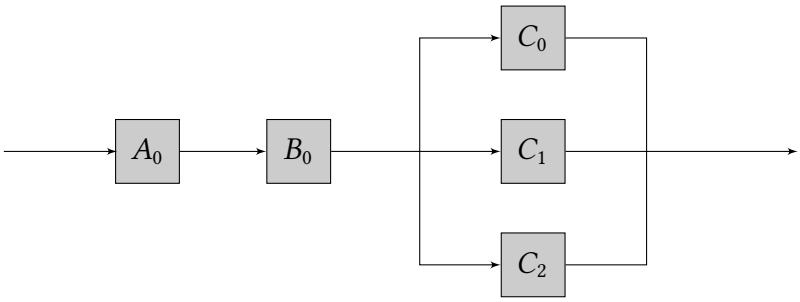


Figura 4.5: *Conjunto de testes para o passo C.*

Seja C^* a configuração que tenha obtido o melhor resultado dentre todas as configurações testadas para o passo C. Dessa forma, seguimos para os testes do passo B, mantendo, a partir de então, C^* como a configuração do passo C, conforme mostrado na figura 4.6.

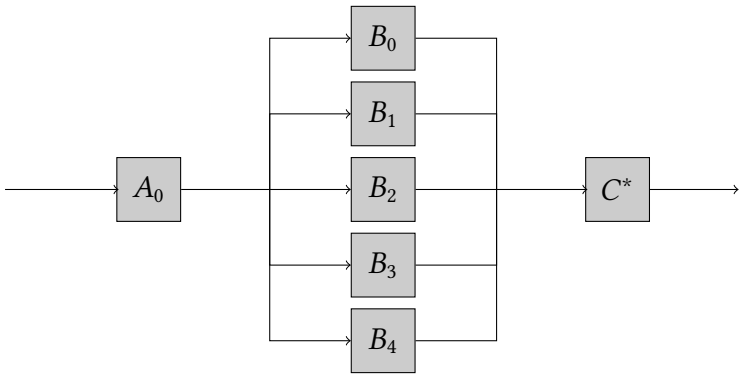


Figura 4.6: *Conjunto de testes para o passo B, mantendo a melhor configuração C^* para o passo C (figura 4.5).*

Da mesma forma, seja B^* o melhor resultado deste conjunto de testes, a última bateria de testes será feita considerando as três configurações de A a serem testadas em adição à configuração base A_0, e mantendo-se as melhores configurações B^* e C^* para os passos B e C, respectivamente, conforme a figura 4.7.

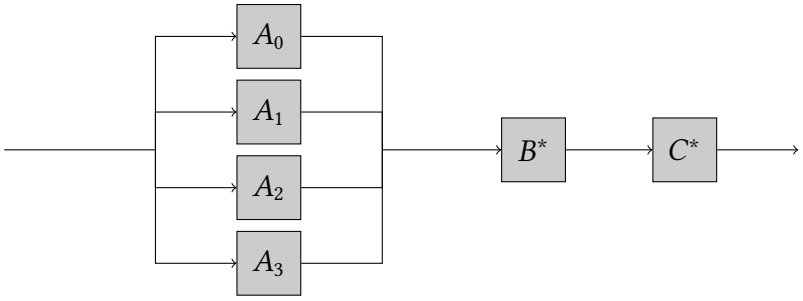


Figura 4.7: *Conjunto de testes para o passo A, a partir do melhor resultado dos testes para os passos C e B (Figura 4.6).*

Ao final dos testes para o passo A, supondo que A^* denota a configuração ótima, teremos a configuração final do *pipeline* (ou seja, do método proposto), conforme mostrado na figura 4.8.

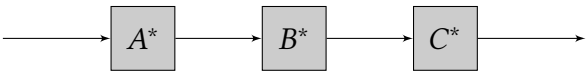


Figura 4.8: *Configuração final do* pipeline *da figura 4.4.*

4.2.3 Experimento Base

Como a metodologia sequencial reversa exige uma configuração inicial, descreveremos aqui esse experimento base, que consiste em três etapas conforme ilustrado na figura 4.9. Para essa configuração base inicial do *pipeline*, baseamo-nos em resultados preliminares publicados em Julca-Aguilar *et al.* (2017) e Maia *et al.* (2018).

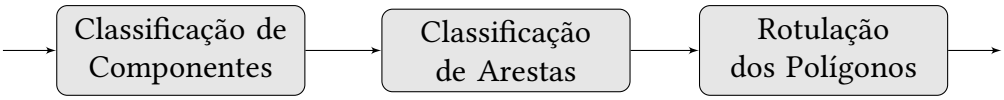


Figura 4.9: *Ilustração das etapas iniciais (base) do pipeline.*

Classificação das Componentes Conexas No passo de classificação de componentes conexas, utilizamos uma CNN treinada a partir de recortes (patches) extraídos da imagem RGB, seguindo o método descrito na seção 3.2.3. Cada recorte contém a imagem da componente conexa, juntamente com informações de contexto ao seu redor. O tamanho deste recorte foi fixado em 40×40 com a caixa delimitadora de cada componente ajustada ao tamanho de 8×8 centrada no recorte (40×8), baseado em nossos estudos preliminares apresentados em Julca-Aguilar *et al.* (2017), em que testamos alguns tamanhos de componentes centrados em um patch de tamanho 40×40

A arquitetura da rede utilizada consiste em duas camadas convolucionais, sendo cada uma delas seguida por ativação *ReLU* e *max-pooling*, seguidas por duas camadas totalmente conectadas com 1024 nós e a camada de saída com ativação *softmax*. Nas camadas convolucionais o kernel tem tamanho 3×3, sendo 32 filtros na primeira camada e 64 filtros na segunda camada. Aplicamos a regularização *dropout* com a taxa de 0.5 na penúltima

camada e utilizamos o algoritmo Adam (Kingma e Ba, 2014) para a otimização da função de custo. A taxa de aprendizado (*learning rate*) foi fixada em 1×10^{-4}.

Classificação das arestas Para a etapa de classificação de arestas, a partir do grafo de componentes construído, um classificador binário para decidir quais arestas serão mantidas (positivas) e quais arestas serão apagadas (negativas) é utilizado. Para tal, treinamos uma rede neural perceptron multicamadas padrão com taxa de aprendizado fixada em 0.001, e fator de desbalanceamento fixado em 0.07 para compensar o maior número de arestas positivas em relação à quantidade de arestas negativas, taxas estas obtidas no estudo prévio apresentado em Maia *et al.* (2018).

As 34 características de cada aresta utilizadas nesta classificação podem ser divididas em duas categorias:

- Informações geométricas (**GEOM**): são as características das componentes conexas (vértices) que formam a aresta descritas na seção 3.2 – altura, largura, razão de aspecto, alongamento, solidez, área e extensão – além do comprimento da aresta (distância euclidiana entre os vértices) e o ângulo formado pela aresta com o eixo X, totalizando 16 características;
- Resultado da classificação das componentes conexas (**CLASS**): vetor contendo os escores de cada classe oriundos da classificação de arestas, formando um total de 18 características, 9 para cada um dos vértices que formam a aresta (excetuando a classe Outros que corresponde a ausência de classificação).

Rotulação dos Polígonos A partir do conjunto de subgrafos resultantes da classificação de arestas, construímos os polígonos envoltórios de cada um dos subgrafos utilizando o algoritmo do *alpha-shape*. Por fim, para uma classificação de cada um dos componentes de página utilizamos a estratégia de votação majoritária (**MAJOR**), em que cada componente conexa contribui para a votação com um voto para a classe com o maior escore obtido na classificação. A classe com maior quantidade de votos, é a classe atribuída ao componente de página.

Dessa forma, para ilustrar a configuração inicial do pipeline, utilizamos o diagrama da figura 4.10 em que apresentamos as configurações base de cada umas das três etapas a serem avaliadas.

Figura 4.10: *Configuração inicial dos experimentos.*

4.3 Métricas de avaliação de desempenho

Existem várias métricas comumente utilizadas em segmentação de imagens. No contexto de segmentação de páginas, a mais comumente utilizada é o IoU (do inglês *Intersection over Union*) que adotamos neste trabalho. De forma informal, essa métrica mede o grau de

sobreposição entre dois segmentos e é definida como a razão entre a área da intersecção dos dois segmentos e a área da união dos dois segmentos. Caso a intersecção seja pequena, esse valor tende a ser pequeno e quando há sobreposição perfeita, esse valor é exatamente 1.

Mais formalmente, seja R uma região alvo (um componente de página) e $\hat{R}$ a região segmentada, ambas delimitadas por um polígono. Podemos definir

- TP (verdadeiros positivos): pixels que estão tanto em R como em $\hat{R}$
- TN (verdadeiros negativos): pixels que não estão em nenhuma das duas regiões
- FP (falsos positivos): pixels que não estão em R mas estão em $\hat{R}$
- FN (falsos negativos): pixels que estão em R mas não estão em $\hat{R}$

A métrica IoU é então definida por:

$$IoU = \frac{TP}{FP + TP + FN} \tag{4.1}$$

Ela captura a similaridade e sobreposição entre a região predita e a região alvo.

Quando há várias classes de componentes de página, esta medida pode ser calculada para cada classe e em seguida a média das medidas por classe, chamada de IOU médio (*MeanIoU*).

Adicionalmente, no nosso método temos também os resultados intermediários de classificação, das componentes conexas e das arestas. Em problemas de classificação binária, podemos estabelecer que uma das classes é a positiva e a outra é a negativa. A partir disso, podemos calcular de forma análoga ao acima os valores TP, TN, FP e FN.

Para avaliar o desempenho de classificadores binários, supondo que N é quantidade de amostras sobre as quais o classificador será avaliado, podemos definir as métricas Acurácia (Accuracy), Precisão (Precision), Revocação (Recall), da seguinte forma.

A acurácia diz respeito à fração de respostas certas em relação a todos os elementos classificados, que pode ser representada pela fórmula 4.2.

$$Acc = \frac{TP + TN}{N} \tag{4.2}$$

A precisão visa a calcular a proporção de predições positivas que são corretas, que pode ser definida pela fórmula 4.3.

$$Prec = \frac{TP}{TP + FP} \tag{4.3}$$

A revocação visa obter a proporção de positivos verdadeiros que foram identificados corretamente. Matematicamente, a revocação é definida conforme a fórmula 4.4.

$$Rec = \frac{TP}{TP + FN} \tag{4.4}$$

A medida geral de avaliação do pipeline é o IOU médio (*MeanIoU*) que é uma métrica global de avaliação do resultado.

4.4 Ferramentas

O método proposto foi implementado em `Python`, usando a biblioteca `Keras/Tensorflow` para os classificadores, `NetworkX` para a construção do RAG e `Scikit Image` para o processamento das imagens.

Capítulo 5

Resultados

Neste capítulo são descritos os detalhes dos experimentos realizados para validar o processo descrito pelo pipeline, além do métodos de avaliação reversa proposto e apresentar os resultados obtidos. Vale ressaltar que o propósito não era necessariamente obter resultados competitivos em relação ao estado da arte, pois se trata de um método que propõe a flexibilidade e a modularidade acima dos resultados, neste primeiro momento, uma vez que a proposta é que cada passo do processo possa ser refinado de maneira a se tornar competitivo. Entretanto, ainda assim, uma comparação com os resultados da competição de onde foi obtida a base de dados utilizada, foi feita. Ao final são apresentadas as discussões dos resultados, além dessa comparação com os resultados da competição.

Os experimentos seguem a metodologia descrita no capítulo anterior (capítulo 4). Recordando brevemente, os experimentos foram desenhados para avaliar diferentes configurações em três passos do método proposto (ver seção 4.2). Por conveniência, o diagrama que destaca esses passos (classificação de componentes conexas, classificação de arestas e atribuição de rótulos aos componentes de página), anteriormente mostrado na figura 3.2, é reproduzido abaixo na figura 5.1.

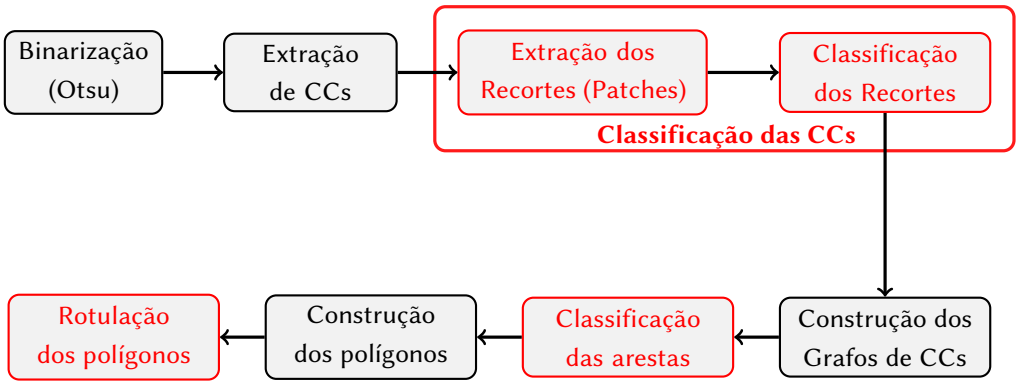


Figura 5.1: *Passos do método proposto, com os passos a serem otimizados destacados em vermelho. Note que este diagrama é o mesmo da figura 3.2, reproduzido aqui por conveniência.*

As imagens utilizadas nos experimentos são de um subconjunto do *PRImA Layout Analysis Dataset*, conforme detalhado na seção 4.1. São 230 imagens para treinamento e

validação, e 75 imagens para teste. Para avaliar os resultados de cada experimento utilizamos as métricas IOU Médio para a avaliação da classificação dos componentes de página e Acurácia, Precisão e Revocação para a avaliação da classificação de componentes conexas e arestas (ver seção 4.3). Seguindo a metodologia experimental descrita na seção 4.2.2, estabelecemos primeiramente a configuração inicial do *pipeline*. Em seguida, configurações distintas para cada um dos três passos são avaliados sobre o conjunto de validação, em ordem sequencial reversa. Ao final, as configurações que correspondem ao melhor resultado são escolhidas para compor a configuração final do *pipeline*. Essa configuração final é avaliada então sobre o conjunto de teste. Os detalhes estão apresentados nas seções subsequentes.

5.1 Experimento Base

Conforme descrito na seção 4.2.3, utilizamos a configuração apresentada na figura 5.2 (aqui repetida), como inicial para avaliação sequencial reversa do conjunto de experimentos apresentados seguir. Assim como também descrito na seção 4.2.3, na etapa de classificação de componentes conexas utilizamos recortes de tamanho 40×40 com o componente de tamanho 8×8, na etapa de classificação de arestas utilizamos características geométricas e os escores de classificação das componentes conexas e, por fim, na etapa de rotulação dos polígonos utilizamos a votação majoritária.

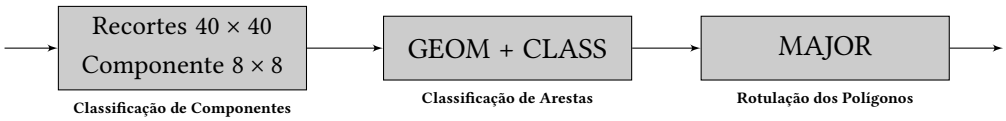


Figura 5.2: *Configuração inicial dos experimentos.*

Classificação das Componentes Conexas O conjunto de treinamento para a classificação de componentes possui um total de 943440 componentes conexas. A quantidade e percentual de componentes conexas por classe são apresentadas na Tabela 5.1. Apesar do grande desbalanceamento, optamos por não utilizar pesos para a função custo, para avaliar posteriormente o impacto desse desbalanceamento na segmentação de páginas.

	Quantidade de amostras	**Percentual**
Other	2179	0.231%
Text	716631	75.959%
Chart	12645	1.340%
Graphic	13218	1.401%
Image	165948	17.590%
Maths	319	0.034%
Noise	24	0.003%
Separator	14288	1.514%
Table	18072	1.916%
Line Drawing	116	0.012%

Tabela 5.1: *Quantidade e Percentual de Componentes Conexas por Classe no conjunto de treinamento.*

O resultado dessa etapa pode ser observada na matriz de confusão (Tabela 5.2), onde as linhas representam o *ground-truth* e as colunas representam os valores preditos.

	Other	**Text**	**Chart**	**Graphic**	**Image**	**Maths**	**Noise**	**Separator**	**Table**	**Line Drawing**
Other	8	18	0	1	262	0	0	6	0	0
Text	0	177902	1	14	1853	0	0	89	74	0
Chart	0	116	0	0	4	0	0	207	0	0
Graphic	0	214	431	92	999	0	0	2	176	0
Image	0	2552	298	458	55014	0	0	670	64	0
Maths	0	13	0	0	0	0	0	0	0	0
Noise	0	7	0	0	0	0	0	0	0	0
Separator	1	54	0	5	432	0	0	2015	2	0
Table	0	18	142	16	119	0	0	1	881	0
Line Drawing	0	307	0	25	84	0	0	0	22	0

Tabela 5.2: *Matriz de confusão (sobre o conjunto de validação) do classificador de componentes conexas.*

Vale notar que as classes Noise, Math e Line Drawing não tiveram predições, por conta da baixa quantidade de amostras.

A acurácia média foi de 96.03 no conjunto de validação e as métricas de precisão e revocação para cada classe é apresentada na tabela 5.3.

Classe	**Precisão**	**Revocação**
Other	0.89	0.03
Text	0.98	0.99
Chart	0.00	0.00
Graphic	0.15	0.05
Image	0.94	0.93
Maths	0.00	0.00
Noise	0.00	0.00
Separator	0.67	0.80
Table	0.72	0.75
Line Drawing	0.00	0.00

Tabela 5.3: *Resultado da Classificação de Componentes Conexas para o Caso Base.*

Classificação das arestas Para a etapa de classificação de arestas, obtivemos uma acurácia de 91.16 e os resultados de precisão e revocação são os apresentados na tabela 5.4.

Classe	**Precisão**	**Revocação**
Apagar	0.44917795	0.75811215
Não Apagar	0.9790844	0.9241259

Tabela 5.4: *Resultado da Classificação de Arestas para o Caso Base.*

Rotulação dos Polígonos Para a etapa de rotulação dos polígonos (componentes de página), os valores do IOU médio, maior IOU, menor IOU e desvio padrão estão apresentados na Tabela5.5.

Medida IOU	Valor
Média	0.8138
Desvio Padrão	0.0957
Menor valor	0.455
Maior valor	0.9605

Tabela 5.5: *Resultado da medida IOU da Classificação de Arestas para o experimento base.*

O IoU Médio, que obteve o valor de 0.8138 será a medida a ser utilizada como base para as próximas etapas dos experimentos.

5.2 Otimização do Pipeline

Para a sequência dos experimentos, conforme discutido anteriormente na seção 4.2.2, a partir desse caso base inicial, faremos a avaliação sequencial reversa, iniciando-se na etapa final do processo e seguir voltando até o início do processo, mantendo sempre a melhor configuração a cada etapa. Dessa forma, a primeira etapa crítica avaliada foi a atribuição dos rótulos aos componentes de página, seguida da classificação de arestas e por fim a classificação de componentes conexas. Utilizamos a medida *MeanIOU* para determinar a melhor configuração.

5.2.1 Atribuição dos rótulos aos componentes de página

Na atribuição dos rótulos aos componentes de página, temos uma estratégia de votação em que os votos de cada componente conexa interna ao polígono são computados. Foram escolhidas quatro estratégias para esta votação:

- Votação Majoritária (**MAJOR**): votação utilizada no caso base inicial, em que se considera a classe com maior escore para ser o voto de cada componente conexa e atribui-se ao componente de página a classe com a maior quantidade de votos;
- Votação por escores de classificação (**SCORE**): cada componente conexa contribui na votação com o vetor de escores gerado pelo classificador de componentes conexas. Os vetores de todas as componentes são somados e a classe com o maior valor obtido será o rótulo atribuído ao polígono;
- Votação a partir da soma dos escores das classes utilizando a área da componente como peso (**SCR_AREA**): semelhante à votação por escore, porém cada componente contribui com o seu vetor multiplicado pela área da componente;
- Votação a partir da soma dos escores das classes utilizando valores empíricos como peso (**SCR_WEIGHT**): semelhante à votação por escore, porém cada componente contribui com o seu vetor multiplicado por um valor obtido empiricamente.

A primeira estratégia trata-se da utilizada no caso base, conforme ilustrado no diagrama da Figura 5.3, em que a configuração base está em verde e as demais configurações a serem testadas estão em vermelho.

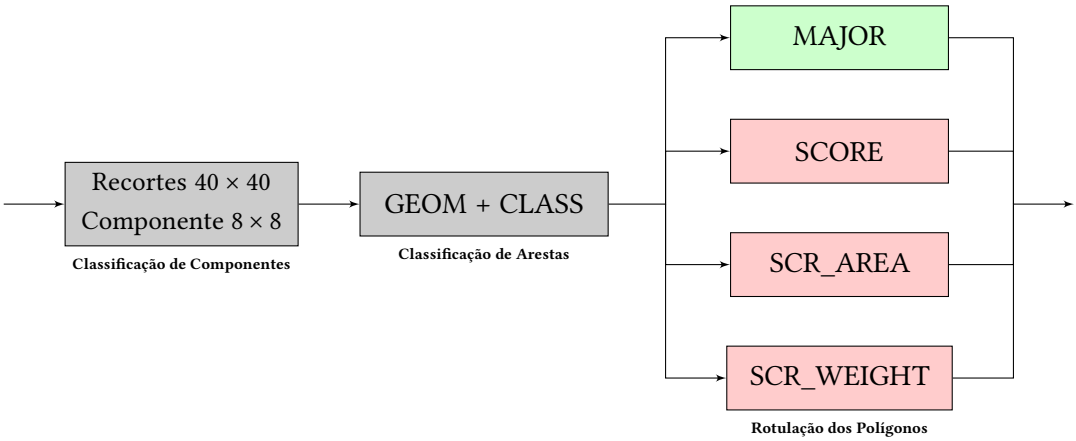


Figura 5.3: *Pipeline dos experimentos da etapa de Rotulação de Polígonos. A configuração em verde representa a configuração base e as configurações em vermelho representam as configurações a serem testadas.*

Os valores médios da medida MeanIoU para cada experimento são apresentados na tabela 5.6.

Experimento	Média IOU
MAJOR	0.8138
SCORE	**0.8208**
SCR_AREA	0.7430
SCR_PESO	0.8199

Tabela 5.6: *Resultado da Média IOU para os Experimentos de Atribuição de Rótulos a Componentes de Página.*

A estratégia que obteve o melhor resultado em relação à Média IOU foi a atribuição dos rótulos aos componentes de página utilizando a soma dos escores. Esse resultado possivelmente se deu por conta de que, algumas vezes, os resultados da classificação das componentes conexas não é dado por uma diferença grande entre os escores, quando há uma certa indeterminação da classe. Considerando, pois, os escores, a classe preterida também contribui de maneira expressiva para votação final.

Os dados de média do IOU, valores mínimos, máximos e desvio padrão da medida IOU do caso base utilizando Votação Majoritária e Votação por Escore podem ser vistos na Tabela 5.7. Apesar do menor valor de IOU ter diminuído, no computo geral da média, obtivemos uma melhora.

Um comparativo mais detalhado entre as duas abordagens pode ser visualizado no gráfico da Figura 5.4, onde apresentamos os valores do IOU para cada imagem do conjunto de validação tanto da configuração base (em azul) quanto da configuração ótima para esta etapa (em vermelho).

Medida IOU	Votação Majoritária	Votação por Escore
Média	0.8138	0.8208
Desvio Padrão	0.0957	0.0832
Menor valor	0.455	0.3231
Maior valor	0.9605	0.9605

Tabela 5.7: *Resultado da medida IOU da Classificação de Arestas para os experimentos de Votação Majoritária e Votação por Escore.*

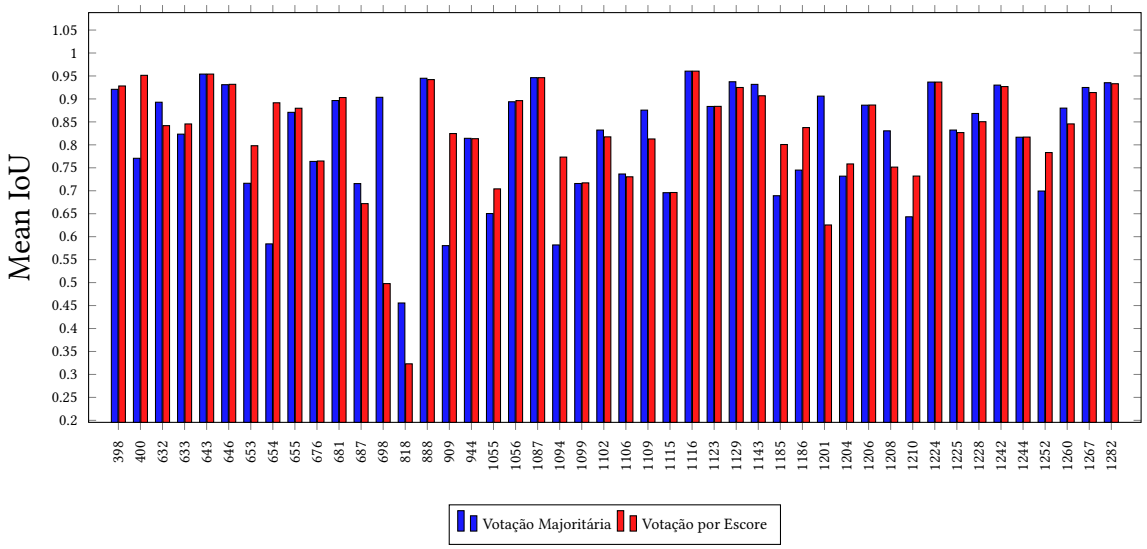


Figura 5.4: *Gráfico comparativo por imagem da medida IoU, por imagem, após experimentos de Rotulação de Polígonos entre o caso base desta etapa (azul) e a configuração ótima (vermelho). Os valores do eixo X representam os identificadores das imagens na base de dados.*

Apesar do resultado utilizando votação por escore ter sido pior que a de votação majoritária para algumas imagens, o menor valor também ter sido pior, como obtivemos um aumento na medida *MeanIOU*, nos experimentos seguintes utilizamos a Votação por Escore como configuração ótima para a etapa de Rotulação dos Polígonos, ilustrado no diagrama da Figura 5.5 pela cor azul e o valor do *MeanIOU* para esta configuração, que servirá de valor base para a próxima etapa, também destacado na figura. As configurações ainda a serem testadas aparecem em cinza no diagrama.

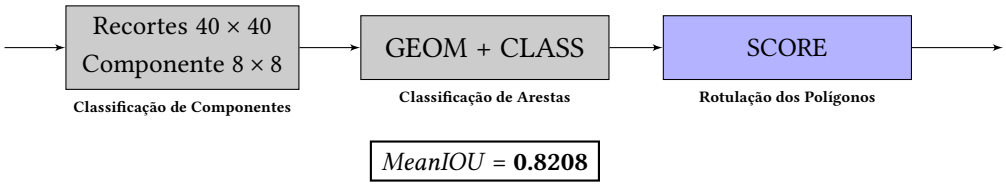


Figura 5.5: *Configuração do pipeline e valor do* MeanIOU *após os experimentos para a etapa de Rotulação de Polígonos. A configuração em azul corresponde à ótima para esta etapa.*

5.2.2 Classificação de arestas

Nesta etapa, uma série de experimentos foram efetuados utilizando algumas combinações das seguintes características:

- Informações geométricas (**GEOM**): são as características geométricas das componentes conexas (16 características) utilizadas no caso base;
- Resultado da classificação das componentes conexas (**CLASS**): vetor contendo os escores de cada classe oriundos da classificação de arestas (18 características), também utilizado no caso base;
- Informações de cores dos vértices e da vizinhança (**COLOR**): valor médio de cada um dos canais de cores (RGB) dos pixels das componentes conexas (vértices) e o valor médio das cores (RGB) dos pixels da vizinhança dos vértices, considerando os recortes (patches) obtidos na etapa de classificação das componentes conexas (12 características);
- Distância entre os vértices e seus adjacentes (**DIST**): média e desvio padrão das distâncias euclidianas entre os pares de vértices formados por cada uma das extremidades da aresta e seus adjacentes (4 características);
- Informações de ângulos entre as extremidades da aresta e suas arestas adjacentes (**ANGLE**): média e desvio padrão dos ângulos formados entre as arestas incidentes em cada uma das extremidades da aresta a ser classificada (4 características);
- Informações de graus dos vértices e seus adjacentes (**DEGREE**): média dos graus dos vértices das extremidades da aresta e seus adjacentes (2 características);

As configurações a serem testadas nesta etapa estão apresentadas no diagrama da Figura 5.6 em que o caso base para esta etapa aparece em verde, as configurações a serem testadas estão em vermelho e a configuração ótima já escolhida aparece em azul.

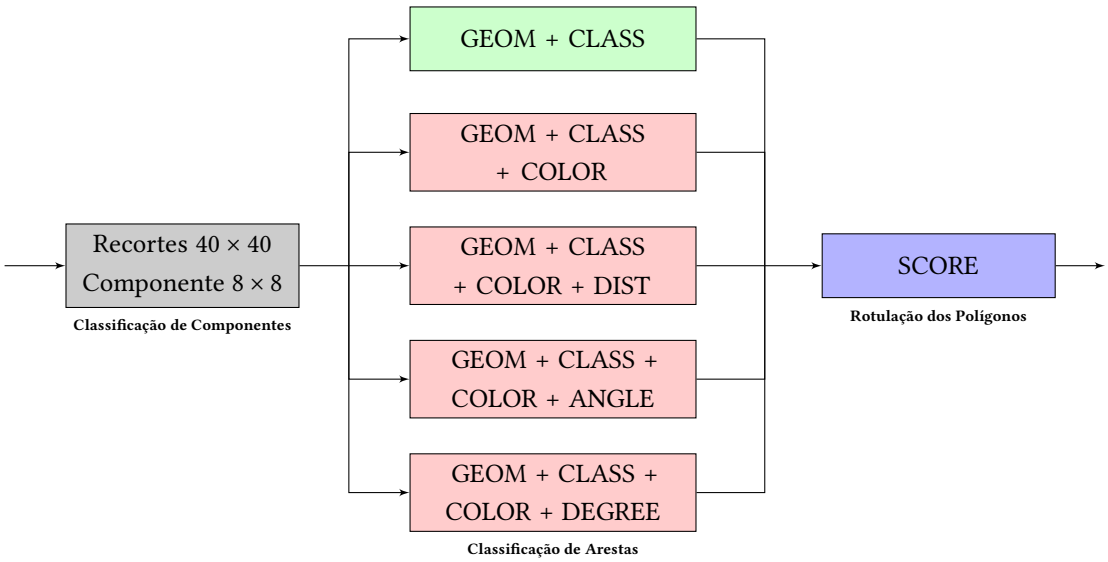


Figura 5.6: *Pipeline dos experimentos da etapa de Classificação de Arestas. A configuração em verde representa a configuração base, as configurações em vermelho representam as configurações a serem testadas e as configuração em azul representa as configuração ótima.*

O primeiro experimento tratou-se da incorporação de informações de cores, considerando que em geral elementos pertencentes ao mesmo componente de página tendem a

ter cores semelhantes e também compartilharem boa parte dos vizinhos. Os resultados de precisão e revocação desse experimento estão apresentados na Tabela 5.8. A acurácia média obtida foi de 89.97.

Classe	Precisão	Revocação
Apagar	0.41551885	0.80777542
Não Apagar	0.98300208	0.9072663

Tabela 5.8: *Resultado da Classificação de Arestas para o experimento incorporando informações sobre cores dos vértices e da vizinhança.*

Embora a acurácia da classificação de arestas, exclusivamente, tenha sido inferior ao valor de 91.16 do caso base, isso não necessariamente reflete uma piora no resultado final, por conta da importância das arestas classificadas como falsos positivos em detrimento à classificação das arestas falso negativas, como discutido na seção 3.3.2. Sendo assim, como estamos usando para esta e demais configurações a medida *MeanIOU* para avaliar o impacto dessas configurações no processo como um todo, verificamos que para esta medida, obtivemos um resultado de 0.8246.

O segundo experimento tratou da inclusão das informações de distância às informações geométricas, de escore e cor das componentes. Para este experimento obtivemos uma acurácia de 85.77 e as informações de Precisão e Revocação podem ser vistas na Tabela 5.9. A medida de IOU para este experimento foi de 0.8323.

Classe	Precisão	Revocação
Apagar	0.32868881	0.84998379
Não Apagar	0.98593608	0.8583173

Tabela 5.9: *Resultado da Classificação de Arestas para o experimento incorporando informações de cores dos vértices e da vizinhança e distância entre os vértices e seus adjacentes.*

O experimento seguinte considerou uma ordenação, em sentido horário, dos vizinhos de cada vértice das extremidades da aresta. A partir desta ordenação, calculou-se a média e o desvio padrão dos ângulos formados entre as arestas de cada extremidade e dos seus vizinhos consecutivos na ordenação e incorporou-se ao vetor de características das arestas, a média e o desvio padrão desses valores. A Figura 5.7 ilustra essa situação, considerando u como uma das extremidades da aresta a ser classificada e $v_1, v_2, ..., v_n$, cada um de seus vértices vizinhos ordenados em sentido horário. Os ângulos considerados para o cálculo da média e desvio padrão para o vértice u são os $\alpha_1, \alpha_1, ..., \alpha_{n-1}$.

O resultado deste experimento, com acurácia de 86.0568 e *MeanIOU* de 0.8393, pode ser visto na Tabela 5.10.

Classe	Precisão	Revocação
Apagar	0.329021811	0.81578853
Não Apagar	0.9829011	0.86422228

Tabela 5.10: *Resultado da Classificação de Arestas para o experimento incorporando informações de cores dos vértices e da vizinhança e ângulos entre as extremidades da aresta e suas arestas adjacentes.*

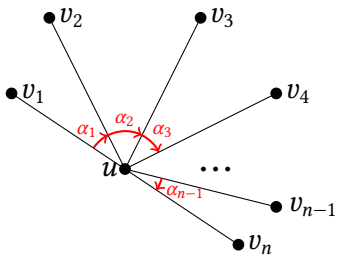


Figura 5.7: *Ilustração dos ângulos a serem considerados para o cálculo dos parâmetros do vetor de características.*

Por fim, o último experimento para esta etapa foi o acréscimo das informações da média dos graus dos vértices das extremidades das arestas e seus adjacentes, obtendo-se os resultados de acurácia de 86.9111 e *MeanIOU* de 0.8572. Os valores de Precisão e Revocação estão apresentados na Tabela 5.11.

Classe	Precisão	Revocação
Apagar	0.34502564	0.81776929
Não Apagar	0.98325479	0.87330166

Tabela 5.11: *Resultado da Classificação de Arestas para o experimento incorporando informações sobre cores dos vértices e da vizinhança e grau entre os vértices e seus adjacentes.*

Os resultados de *MeanIOU* de todas as configurações testadas, sumarizados, estão apresentados na Tabela 5.12 e a configuração ótima para esta etapa é a que incorpora às informações geométricas e de escore, as informações de cores e graus dos vértices.

Experimentos	Média IOU
GEOM + CLASS	0.8208
GEOM + CLASS + COLOR	0.8246
GEOM + CLASS + COLOR + DIST	0.8323
GEOM + CLASS + COLOR + ANGLE	0.8393
GEOM + CLASS + COLOR + DEGREE	**0.8572**

Tabela 5.12: *Resumo dos Experimentos de Classificação de Arestas.*

A Tabela 5.13 estabelece uma comparação entre o caso base desta etapa e a configuração ótima obtida. Para uma avaliação mais detalhada, o gráfico da Figura 5.8 apresenta o resultado do IOU, por imagem, do conjunto de validação do caso base, em azul, e da configuração ótima, em vermelho.

Medida IOU	Caract. geométricas e escore	C. geométricas, escore, cor e grau
Média	0.8208	0.8572
Desvio Padrão	0.0832	0.0573
Menor valor	0.3231	0.4619
Maior valor	0.9605	0.9539

Tabela 5.13: *Resultado da medida IOU da Classificação de Arestas para os experimentos de Características geométricas e escore e Características geométricas, escore, informações cor e de granularidade.*

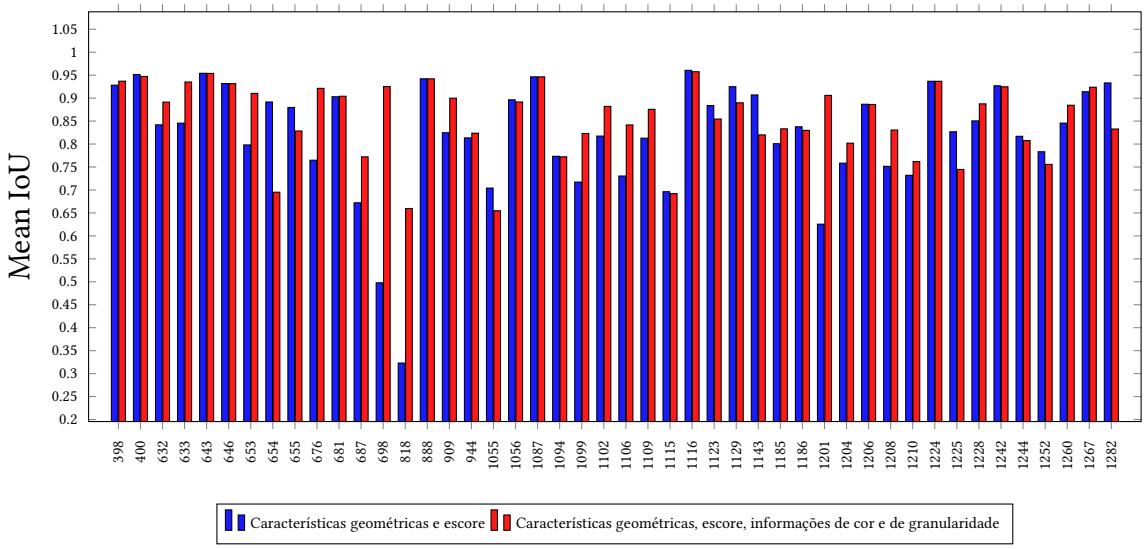


Figura 5.8: *Gráfico comparativo por imagem da medida IoU, por imagem, após experimentos de Classificação de Arestas entre o caso base desta etapa (azul) e a configuração ótima (vermelho). Os valores do eixo X representam os identificadores das imagens na base de dados.*

O diagrama da Figura 5.9 apresenta a configuração ótima após os experimentos efetuados para as etapas de Rotulação de Polígonos e Classificação de Arestas, juntamente com o valor do *MeanIOU* que será considerado o caso base para a próxima etapa de experimentos.

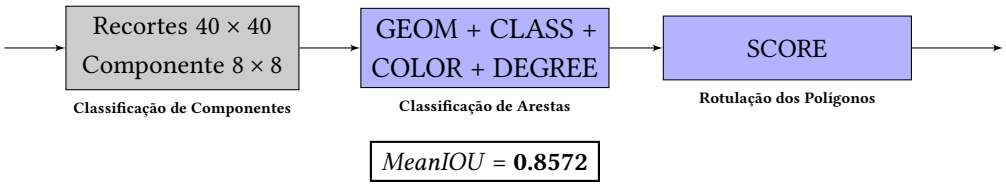


Figura 5.9: *Configuração do pipeline e valor do* MeanIOU *após os experimentos para a etapa de Classificação de Arestas. As configurações em azul correspondem às configurações ótimas para cada etapa.*

5.2.3 Classificação de Componentes Conexas

Para esta etapa escolhemos alterar o tamanho dos recortes (N) utilizados como entrada para a Rede Neural Convolucional. O processo de criação dos recortes foi apresentado na seção 3.2.3. A configuração testada foi:

- Aumento dos recortes (64×8): recortes aumentados para o tamanho 64×64 com a componente conexa centrada no recorte em um quadrado 8×8.

O diagrama que representa esta etapa de experimentos é apresentado na Figura 5.10. A configuração em verde refere-se ao caso base desta etapa, as configurações em vermelho é a configuração a ser testada e as configurações em azul são as configurações ótimas já escolhidas.

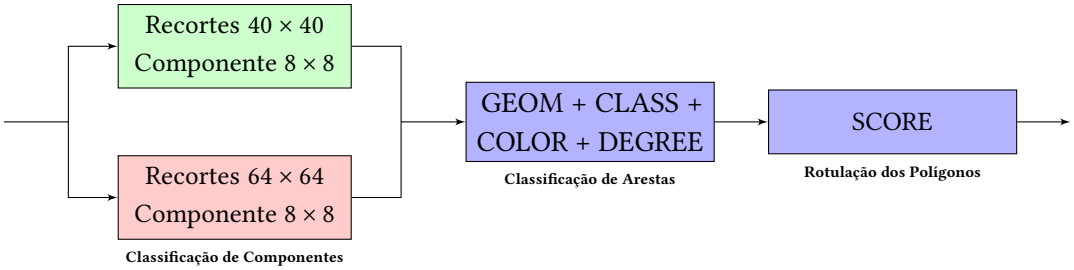


Figura 5.10: *Pipeline dos experimentos da etapa de Classificação de Componentes Conexas. A configuração em verde representa a configuração base e a configuração em vermelho representa a configuração a ser testada e as configurações em azul representam as configurações ótimas.*

A matriz de confusão dos resultados desse Classificador de Componentes é apresentada a seguir (Tabela 5.14).

	Other	Text	Chart	Graphic	Image	Maths	Noise	Separator	Table	Line Drawing
Other	4	19	0	1	234	0	0	37	0	0
Text	1	178498	0	14	871	0	0	12	537	0
Chart	0	82	0	45	21	0	0	150	29	0
Graphic	0	184	48	156	1375	0	0	2	149	0
Image	6	2495	261	47	55373	0	0	599	275	0
Maths	0	13	0	0	0	0	0	0	0	0
Noise	0	6	0	0	1	0	0	0	0	0
Separator0	17	63	42	68	0	0	0	987	0	
Line Drawing	0	296	0	9	112	0	0	0	21	0

Tabela 5.14: *Matriz de confusão dos resultados do Classificador de Componentes Conexas utilizando recortes de tamanho* 64 × 64.

Para esta classificação utilizamos as medidas de acurácia média e precisão e revocação por cada classe e os resultados estão apresentados na tabela 5.15.

	Precision	Recall
Other	0.307692	0.013559
Text	0.982545	0.992025
Chart	0.0000	0.0000
Graphic	0.435754	0.081505
Image	0.952277	0.937635
Maths	0.0000	0.0000
Noise	0.0000	0.0000
Separator	0.742848	0.921084
Table	0.493994	0.838573
Line Drawing	0.0000	0.0000

Tabela 5.15: *Resultados do Classificador de Componentes.*

A acurácia média deste experimento foi de 0.9660. A alteração do tamanho do recorte de 40 × 40 para 64 × 64 não aumentou significantemente as medidas de classificação das componentes conexas e ainda teve um valor do *MeanIOU* de 0.8448, inferior ao obtido com os recortes de tamanho 40.

5.3 Configuração Final do Pipeline

Após as etapas dos experimentos apresentados, obtivemos as configurações ótimas finais representadas no diagrama da Figura 5.11.

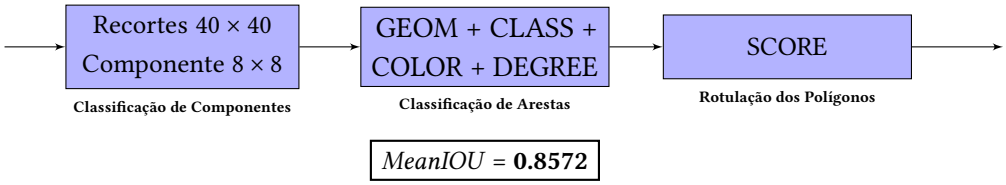


Figura 5.11: *Configuração ótima do pipeline e valor do* MeanIOU *após a finalização dos experimentos.*

Para encerrar o processo experimental, aplicamos os métodos da nossa configuração ótima a cada uma das 75 imagens do conjunto de testes e obtivemos os resultados apresentados na Tabela 5.16.

Medida IOU	Valor
Média	0.7831
Desvio Padrão	0.1692
Menor valor	0.1115
Maior valor	0.9626

Tabela 5.16: *Resultado da medida IOU para o conjunto de testes.*

A Figura 5.12 apresenta os resultados de IOU, por imagem, para as 75 imagens do conjunto de testes.

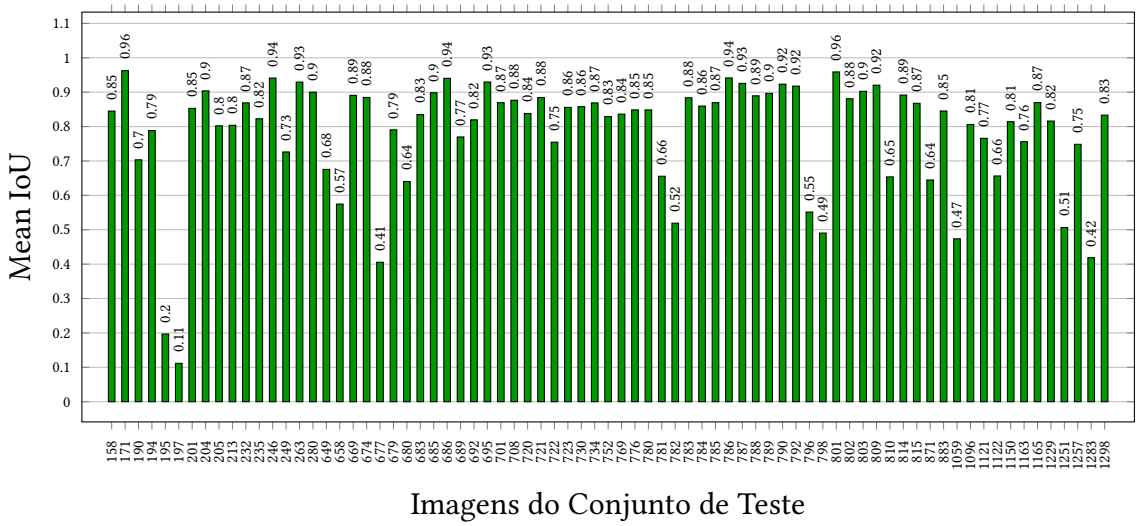


Figura 5.12: *Gráfico por imagem da medida IoU para o conjunto de Teste. Os valores do eixo X representam os identificadores das imagens na base de dados.*

5.4 Discussão dos Resultados

Após a finalização dos experimentos cabe aqui fazer uma avaliação mais detalhada dos resultados obtidos em nosso conjunto de testes. Algumas imagens do conjunto de testes podem ser destacados por apresentarem resultados com valores superiores de *MeanIOU*, como é o caso das imagens de documentos apresentados na Figura 5.13. Na Figura são apresentadas as imagens binarizadas, seguidas do resultado obtido pelo nosso modelo e em seguida o ground-truth para a imagem. Além disso, abaixo de cada um desses trios de imagens apresentamos o resultado do IOU.

Ao analisarmos essas imagens destacadas, podemos observar que estas possuem uma grande quantidade de texto, se não exclusivamente, em que se observam bons resultados com o uso de componentes conexas, e imagens escuras e densas que na binarização formam, praticamente, uma única componente conexa, porém ainda temos algumas falhas nessas classificações. Podemos observar que as porções de texto em nossos resultados não estão separados em parágrafos como no *ground-truth*, já que nosso modelo, na etapa de classificação de arestas, a partir das características utilizadas não é possível diferenciar explicitamente caracteres pertencentes ao mesmo parágrafo e caracteres pertencentes a parágrafos diferentes. Etapas de pós-processamento poderiam ser desenvolvidas neste caso para fazer um ajuste fino da segmentação. Uma outra falha evidente está na classificação dos separadores que tanto pode se dar por conta das reduções para construções dos patches que podem ocasionar perdas de informações importantes, quanto em falhas na binarização, que nos parece ser o principal motivo. Por possuírem características bem específicas (uma das dimensões com medidas bem pequenas e a outra dimensão com medidas bem maiores), os separadores poderiam ser classificados numa etapa prévia e não serem considerados na classificação geral.

De forma inversa, algumas imagens se destacaram negativamente por terem obtido valores muito baixos para a medida IOU, como é o caso das imagens apresentadas na Figura 5.14.

O que pudemos observar nestas imagens foram situações em que o classificador de arestas não foi capaz de distinguir componentes conexas pertencentes a componentes de página diferentes. Se olharmos mais detalhadamente estas arestas falso positivas, é possível perceber algumas situações interessantes. Neste caso para tentarmos entender as falhas no classificador, avaliamos as imagens dos documentos em suas versões binarizadas e as arestas que foram mantidas após a etapa de classificação de arestas. No caso da primeira imagem (Imagem 197) podemos observar na Figura 5.15 uma região de imagem binária e arestas mantidas (em azul). Percebemos que os vértices que permaneceram conectados após a classificação das arestas, são principalmente caracteres de texto e vértices da borda e internos da imagem que representam componentes conexas bem pequenas.

Na segunda imagem (Imagem 195) é possível perceber, na Figura 5.16 que apresentamos uma visão geral da imagem do documento e do resultado da classificação das arestas, que apenas quatro arestas foram as responsáveis por manter a imagem conectada às porções de texto, porções estas que em relação ao restante do documento tiveram suas arestas bem classificadas. Uma outra parte do documento que teve um resultado ruim é a porção referente aos gráficos, que conforme já era esperado, pela quantidade de amostras existentes

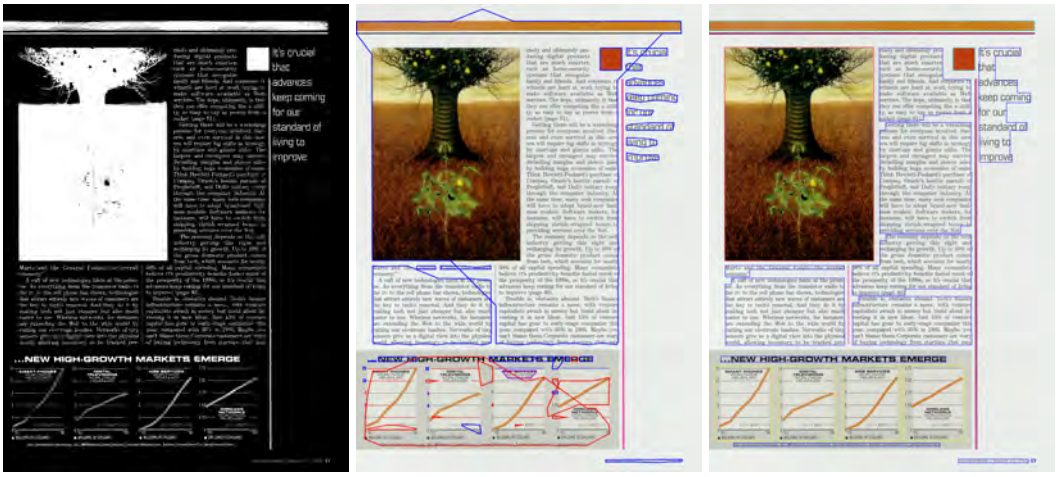
(a) *Imagem 171 –* ***IOU = 0.9626***

(b) *Imagem 801 –* ***IOU = 0.9589***

(c) *Imagem 786 –* ***IOU = 0.9414***

Figura 5.13: *Exemplos de documentos que obtiveram bons resultados na segmentação. Abaixo de cada documento está a sua identificação na base de dados e o valor do seu IOU.*

(a) *Imagem 197 –* ***IOU = 0.1115***



(b) *Imagem 195 –* ***IOU = 0.1967***

Figura 5.14: *Exemplos de imagens de documentos que obtiveram resultados ruins na segmentação. Abaixo de cada documento está a sua identificação na base de dados e o valor do seu IOU.*

já teve problema na classificação das componentes, além de podermos perceber o processo de binarização também teve impacto na classificação, uma vez que "apagou"as bordas dos gráficos. Na seção seguinte, falaremos um pouco mais sobre o impacto da binarização neste processo de segmentação de imagens de documentos.

5.4.1 Outros Experimentos

Com o intuito de minimizar as falhas detectadas nas imagens do conjunto de testes, mais dois experimentos foram efetuados:

- Limiar de classificação: alterar o limiar para a definição da classificação da aresta a partir do vetor de escores obtido;
- Remoção de componentes conexas pequenas: a partir da área da componente conexa, determinar se uma componente deve ou não ser considerada ou descartada.

Conforme já explicitado algumas vezes ao longo deste trabalho, as arestas falso po-

Figura 5.15: *Recortes da Imagem 197 binarizada com as arestas (em azul) que foram mantidas após a classificação.*

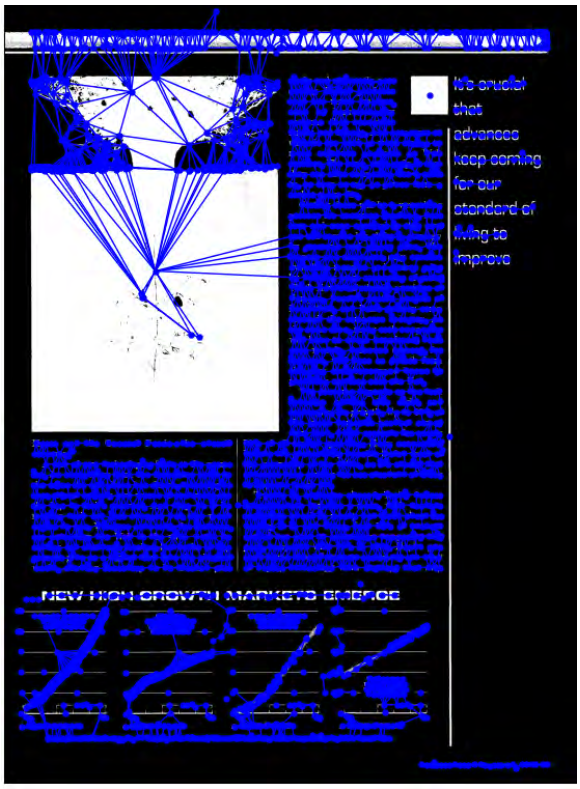


Figura 5.16: *Visão geral da Imagem 195 binarizada com as arestas (em azul) que foram mantidas após a classificação.*

sitivas têm uma influência muito maior do que as arestas falso negativas no resultado da segmentação. Dessa forma, o primeiro experimento visou testar uma flexibilização no limiar de classificação, aumentando esses valores com o intuito de diminuir as arestas falso positivas. Inicialmente o limiar considerado era o de 0.5, ou seja, se o escore da classificação fosse maior do que 0.5, a aresta era mantida e era "apagada"em caso contrário. Foram testados valores de 0.6 a 0.9, com variação de 0.1 e a melhor configuração foi considerando o limiar de 0.6 para a classificação.

O segundo experimento tratou de remover da imagem binária as componentes conexas consideradas muito pequenas (com área menor ou igual a um certo valor pré-estabelecido) e treinar os classificadores novamente com essas novas imagens. Três valores de áreas foram testados: 1, 5 e 10 e o melhor resultado obtido foi o experimento considerando a remoção de todas as componentes conexas com área menor ou igual a 1. Aplicando-se essa nova configuração ao conjunto de testes, tivemos um aumento na média *MeanIOU*, que obteve o valor de 0.8155. Os resultados desta classificação podem ser vistos na Tabela 5.17.

Medida IOU	**Valor**
Média	0.8155
Desvio Padrão	0.1191
Menor valor	0.4279
Maior valor	0.9594

Tabela 5.17: *Resultado da medida IOU para o conjunto de testes.*

A Figura 5.17 apresenta os resultados de IOU, por imagem, para as 75 imagens do conjunto de testes, com o limiar de 0.6 para a classificação de arestas e após a remoção de componentes com área menor ou igual a 1.

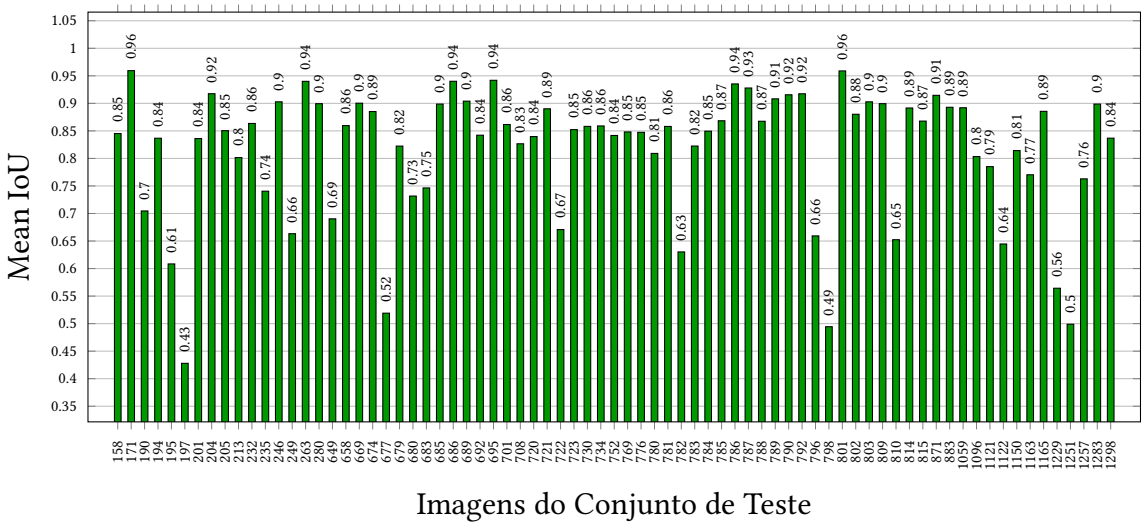


Figura 5.17: *Gráfico por imagem da medida IoU para o conjunto de Teste. Os valores do eixo X representam os identificadores das imagens na base de dados.*

5.4.2 Impacto da Binarização

Podemos observar que a binarização impacta diretamente a Classificação de Componentes Conexas, uma vez que as componentes conexas são extraídas a partir das imagens binárias. Dessa forma, se avaliarmos algumas imagens do conjunto de testes, podemos observar que em situações em que as imagens são de cor clara, o limiar da binarização pode fazer com que informações relevante sejam perdidas, como bordas de imagens ou até mesmo porções grandes de componentes de página sendo "apagados", conforme podemos observar no recorte da imagem do documento apresentado na Figura 5.18, isso compromete diretamente o resultado da segmentação de componentes de páginas, impondo uma limitação importante aos resultados possíveis de serem alcançados para este documento.

(a) **(b)**

Figura 5.18: *Recorte da imagem do documento original (a) e sua versão binarizada (b), em que a binarização "apaga"parte da figura.*

Devido à grande variedade de imagens outras situações também são afetadas pelo tipo de binarização escolhida, como por exemplo em situações em que um texto de cor clara é escrito em um fundo de cor escura, conforme o exemplo da Figura 5.19, que mostram recortes de um documento em que essa situação ocorre.

Conforme comentado anteriormente, optamos por fixar esta etapa de binarização utilizando o método de Otsu. Porém, decidimos fazer um experimento adicional utilizando o método de Sauvola (Sauvola e Pietikäinen, 2000), desenvolvido principalmente para a binarização de documentos, entretanto após alguns experimentos, verificamos que o desempenho do algoritmo de Otsu foi superior e, portanto, uma alternativa natural seria o uso de estratégias de aprendizagem de máquina para o processo de binarização, visando minimizar o impacto da binarização no restante do processo.

5.4.3 Classificação de Componentes Conexas

Analisando os resultados obtidos exclusivamente na classificação das componentes conexas considerando todas as classes, pudemos observar que os resultados do classificador de componentes para algumas classes não foram bons. Porém, dada a baixa quantidade de amostras disponíveis para algumas classes (Tabela 5.1), isso não é surpreendente, uma vez que, por escolha não utilizamos qualquer técnica para minimizar esse impacto. Vale ressaltar que essa escolha serviu para deixar bem claro o impacto desse desbalanceamento, que pode ser resolvido com técnicas de aumento de dados dentre outras.

Figura 5.19: *Ilustração de um recorte de um documento em que o texto é escrito em cor clara com um fundo de cor escura.*

Além disso, foi possível observar que algumas classes possuem algumas semelhanças, o que podem gerar certa ambiguidade, conforme podemos observar na figura 5.20 em que temos três imagens com diagramas e que, no *ground-truth*, foram classificados de forma diferente, como Elemento Gráfico (*Graphic*), Gráfico (*Chart*) e Desenho em Linhas (LineDrawing), respectivamente.

(a) **(b)** **(c)**

Figura 5.20: *Exemplo de imagens com componentes de página semelhantes, classificados de forma diferentes no ground-truth. Os componentes de página foram classificados como Elemento Gráfico, Gráfico e Desenho em Linhas, respectivamente.*

Dessa forma, um passo natural seria a junção de algumas classes como a junção das classes Other e Noise na classe Other, a junção das classes Chart, Graphic e Line Drawing na classe Graphic e a junção da classe Math na classe Text, além da utilização de técnicas de aumento de dados.

5.4.4 Comparação com resultados da competição

A base de dados que utilizamos neste trabalho, conforme apresentamos anteriormente é uma base de dados que foi utilizada na competição *Competition on Recognition of Documents with Complex Layouts - RDCL* da conferência *International Conference on Document Analysis and Recognition* e optamos por considerar como nosso conjunto de testes, o conjunto de avaliação da competição de 2017. Dessa forma, apesar de o propósito deste trabalho não tenha sido o de competir com os métodos estado da arte, e sim a proposição de um *pipeline* para a solução do problema em que as etapas possam ser avaliadas e alteradas de forma modular e, além disso, de não termos utilizado a mesma métrica utilizada da competição(Clausner, Antonacopoulos *et al.*, 2017), em que os parâmetros não são divulgados publicamente, o laboratório de pesquisa *Pattern Recognition & Image Analysis* da *University of Salford* responsável pela base de dados, disponibiliza uma ferramenta (Clausner, Pletschacher *et al.*, 2011a), que necessita de credenciais para o uso, que possibilita a avaliação dos resultados da competição de 2017, utilizando a métrica de avaliação desta.

A avaliação dos resultados da competição são divididos em três cenários descritos em detalhes em (Clausner, Pletschacher *et al.*, 2011b). A primeira avaliação (*Segmentation*) mede simplesmente o desempenho do método do ponto de vista da segmentação das regiões. A segunda avaliação (*Segmentation + Classification*)leva em consideração a segmentação e classificação no contexto de um típico sistema OCR, concentrado no texto mas sem ignorar as regiões de não-texto. O peso da classificação errada de texto é consideravelmente mais alta do que os erros em regiões de não-texto. A última avaliação (*Text regions only*) é concentrada apenas em regiões de texto, ignorando regiões de não-texto.

Dessa forma, apresentamos aqui um comparativo entre os resultados da competição e os nossos. Os nossos resultados foram próximos de alguns dos métodos, em uma situação até superando, mas inferiores a grande maioria, porém vale ressaltar que as métricas da competição priorizam as regiões de texto, diferente do nosso método que priorizou a generalidade. Ademais, como discutido anteriormente nesta seção, o método tem bastante potencial de melhorias.

A tabela 5.18 apresenta os nossos resultados juntamente com os resultados da competição, apresentados em Clausner, Antonacopoulos *et al.*, 2017.

Método	Segmentation	Segmentation + Classification	Text regions only
Método Proposto	75.12%	72.58%	77,90%
. Tesseract	75.83%	72.95%	78.09%
FineReader	83.87%	81.26%	86.32%
LIPADE	81.15%	78.70%	80.07%
MHS 2017	92.32%	90.62%	92.94 %
CVML	83.96%	83.11%	90.38%
AOSM	82.75%	81.23%	87.21%
JU_Aegean	76.31%	73.54%	77.15%

Tabela 5.18: *Comparativo entre o nosso método proposto e os métodos apresentados na competição RDCL2017.*

Capítulo 6

Conclusões

Nesta tese propomo-nos a desenvolver métodos para a segmentação de componentes de página que não fossem muito restritivos quanto ao leiaute da página nem aos tipos e formato dos componentes de página.

De forma geral, o método proposto é baseado em uma modelagem abstrata de abordagens *bottom-up* para segmentação de imagens. Nessa modelagem, uma imagem é inicialmente particionada em primitivas, as quais são em seguida agrupadas para formar regiões que correspondem aos componentes de página. Especificamente, utilizamos componentes conexas da versão binária da imagem original como primitivas, o método de triangulação de Delaunay para estabelecer a relação de adjacência entre as componentes conexas, e classificação de arestas do grafo de adjacências de forma a determinar quais arestas devem ser removidas para que os subgrafos conexos resultantes correspondam às componentes de página de interesse. Ao final, os componentes de página detectadas são representadas por polígonos envoltórios.

A utilização de algoritmos de aprendizado de máquina para a classificação das componentes conexas e para a classificação das arestas, assim como a possibilidade de associar diferentes características aos nós e arestas do grafo de adjacências, facilitou a aplicação do método sobre um conjunto de páginas com características heterogêneas. Um esquema de otimização sequencial reversa foi aplicado ao *pipeline* subjacente ao método para encontrar uma configuração com bom desempenho.

Resultados experimentais mostram um bom desempenho do método em páginas com leiaute Manhattan. Resultados interessantes são também alcançados em páginas com leiaute não-Manhattan, indicando o potencial do método.

Observamos que partes deste trabalho foram publicados nos seguintes artigos:

- F. D. Julca-Aguilar, A. L. L. M. Maia e N. S. T. Hirata. *Text/non-text classification of connected components in document images*. 30th Conference on Graphics, Patterns and Images (SIBGRAPI), 2017.
- A. L. L. M. Maia, F. D. Julca-Aguilar e N. S. T. Hirata. *A machine learning approach for graph-based page segmentation*. 31st Conference on Graphics, Patterns and Images (SIBGRAPI), 2018.

O método demonstra grande potencial, uma vez que permite a utilização de diferentes técnicas em diferentes etapas, o que pode contribuir para uma avaliação minuciosa de todo o processo e dos impactos de cada etapa, parâmetros. Além disso, em contrapartida aos métodos mais atuais, como os baseados em deep learning, o controle de cada etapa permite que possamos otimizar recursos que são críticos nos métodos mais modernos, como desempenho e eventualmente escassez de dados. Esse trabalho pode ser considerado um pontapé inicial em uma estratégia de flexibilizar e atender diferentes contextos na área de documentos associando diferentes técnicas nas diversas etapas do processo de Segmentação de Páginas utilizando esta abordagem.

6.1 Sugestões para trabalhos futuros

Por ser baseado em uma modelagem abstrata, o método proposto é bastante flexível quanto aos passos individuais. Por exemplo, o tipo de primitiva poderia ser superpixel. Isto afetaria a forma de se calcular o grafo de adjacências, mas os demais passos não mudariam em essência. Também, por construção, em princípio o método proposto é flexível quanto à família de documentos, ao tipo de leiaute (desde que sem sobreposição de componentes de página), e à variedade de tipos de componentes de página.

Neste sentido, uma das limitações desta tese é não ter explorado mais amplamente essas possibilidades. Destacamos, portanto, algumas linhas de investigação para trabalhos futuros.

O primeiro passo do método proposto é a binarização da imagem. Conforme apontado nas discussões sobre os resultados obtidos, notou-se que vários erros são decorrentes de uma binarização que não reflete adequadamente o conteúdo da página. Certamente outros algoritmos de binarização, inclusive baseados em aprendizado de máquina, podem ser empregados.

Na mesma linha, enquanto a binarização de regiões de alto contraste como aquelas ocupadas por texto geram uma representação binária quase fiel à imagem original, regiões da página contendo figuras podem resultar em componentes conexas muito distintas daquilo que se vê na imagem original. Esta observação sugere uma ideia para ser explorada: utilizar diferentes tipos de primitivas em diferentes partes da página, de acordo com as suas características, porém mantê-las integradas em um mesmo grafo de adjacências.

A avaliação do método proposto em outros conjuntos de dados, outra quantidade e tipos de componentes de páginas (por exemplo, detectar componentes de página específicos como tabela) será útil para demonstrar a efetividade e flexibilidade do método, assim como para identificar pontos a serem melhorados.

Uma linha de investigação futura promissora diz respeito à modelagem baseada em grafo. Em primeiro lugar, o emprego de grafos possibilita a exploração de informações de mais alto nível tais como características topológicas ou geométricas dos componentes de página e características globais da imagem. Tais informações, que não foram exploradas nesta tese, poderiam ser úteis para melhorar os resultados de segmentação. Grafos também são versáteis para modelar relações entre as componentes de página; relações são particularmente úteis para entender o leiaute da página. Assim, incorporar essas informações ao

grafo tem o potencial de permitir a análise lógica de leiaute.

Por último, apesar dos avanços mais recentes na área, algumas mencionadas no capítulo 2.6 (tais como maior quantidade de bases de dados, maior quantidade de tipos de componentes de página), observa-se que os trabalhos publicados não são em geral comparáveis por utilizarem conjuntos de dados distintos, ou considerarem diferentes tipos de componentes de páginas, diferirem quanto à aplicação de pós-processamento, entre outras diferenças. Por exemplo, a base de dados usada neste trabalho é uma das poucas no qual os componentes de página são especificados por polígonos. Na grande parte das bases de dados, os componentes são especificados por retângulos envoltórios (leiaute retangular). Além disso, apesar de a base de dados utilizada nesta tese ter sido utilizada em competições, a métrica de desempenho utilizada na competição não é o IoU, a métrica mais comumente utilizada. Desta forma, para fazer uma comparação do desempenho obtido neste trabalho e os obtidos na competição foi necessária a utilização de uma ferramenta específica com métricas cujos parâmetros não são divulgados publicamente. Este cenário indica que há necessidade de um esforço conjunto da comunidade atuante nesta área para a criação de *benchmarks*.

Referências

[Achanta *et al.* 2012] Radhakrishna Achanta *et al.* "SLIC superpixels compared to state-of-the-art superpixel methods". Em: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pgs. 2274–2282 (citado na pg. 12).

[Anitei *et al.* 2021] Dan Anitei, Joan Andreu Sánchez, José Manuel Fuentes, Roberto Paredes e José Miguel Benedí. "Icdar 2021 competition on mathematical formula detection". Em: *Document Analysis and Recognition – ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5-10, 2021, Proceedings, Part IV*. Lausanne, Switzerland: Springer-Verlag, 2021, pgs. 783–795. isbn: 978-3-030-86336-4. doi: 10.1007/978-3-030-86337-1_52. url: https://doi.org/10.1007/978-3-030-86337-1_52 (citado na pg. 11).

[Antonacopoulos, Bridson *et al.* 2009] Apostolos Antonacopoulos, David Bridson, Christos Papadopoulos e Stefan Pletschacher. "A realistic dataset for performance evaluation of document layout analysis". Em: *10th International Conference on Document Analysis and Recognition*. IEEE. 2009, pgs. 296–300 (citado nas pgs. 4, 39).

[Antonacopoulos, Pletschacher *et al.* 2009] Apostolos Antonacopoulos, Stefan Pletschacher, David Bridson e Christos Papadopoulos. "ICDAR 2009 page segmentation competition". Em: *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*. ICDAR '09. Washington, DC, USA: IEEE Computer Society, 2009, pgs. 1370–1374. isbn: 978-0-7695-3725-2 (citado nas pgs. ix, 8, 9).

[Arenas *et al.* 2018] Sebastian Wilde Alarcón Arenas, Yessenia Yari e Graciela Meza-Lovon. "A document layout analysis method based on morphological operators and connected components". Em: *2018 XLIV Latin American Computer Conference (CLEI)*. IEEE. 2018, pgs. 622–631 (citado na pg. 15).

[Binmakhashen e Mahmoud 2019] Galal M. Binmakhashen e Sabri A. Mahmoud. "Document layout analysis: a comprehensive survey". Em: *ACM Comput. Surv.* 52.6 (2019). issn: 0360-0300. doi: 10.1145/3355610. url: https://doi.org/10.1145/3355610 (citado nas pgs. 2, 9, 10, 19).

[Biswas *et al.* 2021] Sanket Biswas, Pau Riba, Josep Lladós e Umapada Pal. "Beyond document object detection: instance-level segmentation of complex layouts". Em: *Int. J. Doc. Anal. Recognit.* 24.3 (2021), pgs. 269–281. issn: 1433-2833. doi: 10.1007/s10032-021-00380-6. url: https://doi.org/10.1007/s10032-021-00380-6 (citado nas pgs. 22–24).

[Bukhari *et al.* 2010] Syed Saqib Bukhari, Mayce Ibrahim Ali Al Azawi, Faisal Shafait e Thomas M. Breuel. "Document image segmentation using discriminative learning over connected components". Em: *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. DAS'10. Boston, Massachusetts, USA: ACM, 2010, pgs. 183–190. isbn: 978-1-60558-773-8 (citado nas pgs. 15, 31, 32).

[Chen, Liu *et al.* 2016] Kai Chen, Cheng-Lin Liu *et al.* "Page segmentation for historical document images based on superpixel classification with unsupervised feature learning". Em: *12th IAPR Workshop on Document Analysis Systems (DAS)*. IEEE. 2016, pgs. 299–304 (citado na pg. 12).

[Chen, Yin *et al.* 2013] Kai Chen, Fei Yin e Cheng-Lin Liu. "Hybrid page segmentation with efficient whitespace rectangles extraction and grouping". Em: *12th International Conference on Document Analysis and Recognition (ICDAR),* IEEE. 2013, pgs. 958–962 (citado na pg. 15).

[Clausner, Antonacopoulos *et al.* 2017] Christian Clausner, Apostolos Antonacopoulos e Stefan Pletschacher. "Icdar2017 competition on recognition of documents with complex layouts - rdcl2017". Em: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. 2017, pgs. 1404–1410. doi: 10.1109/ICDAR.2017.229 (citado nas pgs. 39, 70).

[Clausner, Pletschacher *et al.* 2011a] Christian Clausner, Stefan Pletschacher e Apostolos Antonacopoulos. "Aletheia-an advanced document layout and text ground-truthing system for production environments". Em: *2011 International Conference on Document Analysis and Recognition*. IEEE. 2011, pgs. 48–52 (citado na pg. 70).

[Clausner, Pletschacher *et al.* 2011b] Christian Clausner, Stefan Pletschacher e Apostolos Antonacopoulos. "Scenario driven in-depth performance evaluation of document layout analysis methods". Em: *2011 International Conference on Document Analysis and Recognition*. IEEE. 2011, pgs. 1404–1408 (citado na pg. 70).

[Cohen *et al.* 2013] Rafi Cohen, Abedelkadir Asi, Klara Kedem, Jihad El-Sana e Itshak Dinstein. "Robust text and drawing segmentation algorithm for historical documents". Em: *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing*. ACM. 2013, pgs. 110–117 (citado na pg. 12).

[Dengel e Shafait 2014] Andreas Dengel e Faisal Shafait. "Analysis of the Logical Layout of Documents". Em: *Handbook of Document Image Processing and Recognition*. Ed. por David Doermann e Karl Tombre. London: Springer London, 2014, pgs. 177–222. isbn: 978-0-85729-859-1 (citado nas pgs. 2, 7, 8).

[Doermann e Tombre 2014] David S. Doermann e Karl Tombre, ed. *Handbook of Document Image Processing and Recognition*. Springer, 2014. isbn: 978-0-85729-858-4. doi: 10.1007/978-0-85729-859-1. url: https://doi.org/10.1007/978-0-85729-859-1 (citado na pg. 1).

[Edelsbrunner *et al.* 1983] Herbert Edelsbrunner, David Kirkpatrick e Raimund Seidel. "On the shape of a set of points in the plane". Em: *IEEE Transactions on information theory* 29.4 (1983), pgs. 551–559 (citado na pg. 36).

[Eskenazi *et al.* 2017] Sébastien Eskenazi, Petra Gomez-Krämer e Jean-Marc Ogier. "A comprehensive survey of mostly textual document segmentation algorithms since 2008". Em: *Pattern Recognition* 64 (2017), pgs. 1–14. issn: 0031-3203. doi: https://doi.org/10.1016/j.patcog.2016.10.023. url: https://www.sciencedirect.com/science/article/pii/S0031320316303399 (citado nas pgs. 9, 10, 19).

[Feofiloff *et al.* 2011] Paulo Feofiloff, Yoshiharu Kohayakawa e Yoshiko Wakabayashi. "Uma introdução sucinta à teoria dos grafos". Em: (2011) (citado na pg. 14).

[Fortune 1997] Steven Fortune. "Voronoi diagrams and delaunay triangulations". Em: *Handbook of Discrete and Computational Geometry*. USA: CRC Press, Inc., 1997, pgs. 377–388. isbn: 0849385245 (citado na pg. 17).

[Goodfellow *et al.* 2016] Ian Goodfellow, Yoshua Bengio e Aaron Courville. *Deep Learning*. MIT Press, 2016 (citado na pg. 2).

[Graham 1972] Ronald L. Graham. "An efficient algorithm for determining the convex hull of a finite planar set". Em: *Inf. Process. Lett.* 1.4 (1972), pgs. 132–133 (citado na pg. 36).

[Ha *et al.* 1995a] Jaekyu Ha, Robert M Haralick e Ihsin T Phillips. "Document page decomposition by the bounding-box project". Em: *3rd International Conference on Document Analysis and Recognition*. Vol. 2. IEEE. 1995, pgs. 1119–1122 (citado na pg. 10).

[Ha *et al.* 1995b] Jaekyu Ha, Robert M Haralick e Ihsin T Phillips. "Recursive xy cut using bounding boxes of connected components". Em: *Third International Conference on Document Analysis and Recognition*. Vol. 2. IEEE. 1995, pgs. 952–955 (citado na pg. 10).

[Jarvis 1973] R. A. Jarvis. "On the identification of the convex hull of a finite set of points in the plane". Em: *Information Processing Letters* 2.1 (1973), pgs. 18–21. ISSN: 0020-0190. DOI: https://doi.org/10.1016/0020-0190(73)90020-3. URL: https://www.sciencedirect.com/science/article/pii/0020019073900203 (citado na pg. 36).

[Julca-Aguilar *et al.* 2017] Frank D. Julca-Aguilar, Ana L. L. M. Maia e Nina S. T. Hirata. "Text/non-text classification of connected components in document images". Em: *30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE. 2017, pgs. 450–455 (citado nas pgs. 33, 47).

[Kasar *et al.* 2013] T. Kasar, P. Barlas, S. Adam, C. Chatelain e T. Paquet. "Learning to detect tables in scanned document images using line information". Em: *12th International Conference on Document Analysis and Recognition*. 2013, pgs. 1185–1189. DOI: 10.1109/ICDAR.2013.240 (citado na pg. 11).

[Kingma e Ba 2014] Diederik P. Kingma e Jimmy Ba. "Adam: a method for stochastic optimization." Em: *ICLR* (2014) (citado na pg. 48).

[Kise 2014] Koichi Kise. "Page segmentation techniques in document analysis". Em: *Handbook of Document Image Processing and Recognition*. Springer, 2014, pgs. 135–175 (citado nas pgs. 7–10, 16, 19).

[Krizhevsky *et al.* 2012] Alex Krizhevsky, Ilya Sutskever e Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". Em: *Advances in Neural Information Processing Systems*. Ed. por F. Pereira, C.J. Burges, L. Bottou e K.Q. Weinberger. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (citado na pg. 2).

[Le *et al.* 2015] V. P. Le, N. Nayef, M. Visani, J. M. Ogier e C. D. Tran. "Text and Non-text Segmentation based on Connected Component Features". Em: *International Conference on Document Analysis and Recognition (ICDAR)*. 2015, pgs. 1096–1100 (citado na pg. 15).

[Lee *et al.* 2019] Joonho Lee, Hideaki Hayashi, Wataru Ohyama e Seiichi Uchida. "Page segmentation using a convolutional neural network with trainable co-occurrence features". Em: *International Conference on Document Analysis and Recognition (ICDAR)*. 2019, pgs. 1023–1028. DOI: 10.1109/ICDAR.2019.00167 (citado nas pgs. 21, 23, 24).

[Li *et al.* 2021] Shoubin Li *et al.* "Vtlayout: fusion of visual and text features for document layout analysis". Em: *PRICAI 2021: Trends in Artificial Intelligence: 18th Pacific Rim International Conference on Artificial Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2021, pgs. 308–322. ISBN: 978-3-030-89187-9. DOI: 10.1007/978-3-030-89188-6_23. URL: https://doi.org/10.1007/978-3-030-89188-6_23 (citado nas pgs. 22–24).

[Liwicki *et al.* 2007] Marcus Liwicki, Alex Graves, Santiago Fernàndez, Horst Bunke e Jürgen Schmidhuber. "A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks". Em: *Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007*. 2007 (citado na pg. 2).

[Long *et al.* 2015] Jonathan Long, Evan Shelhamer e Trevor Darrell. "Fully convolutional networks for semantic segmentation". Em: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pgs. 3431–3440 (citado na pg. 20).

[Luo *et al.* 2022] Siwen Luo, Yihao Ding, Siqu Long, Josiah Poon e Soyeon Caren Han. "Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis". Em: *Proceedings of the 29th International Conference on Computational Linguistics, COLING*. Ed. por Nicoletta Calzolari *et al.* International Committee on Computational Linguistics, 2022, pgs. 2906–2916. URL: https://aclanthology.org/2022.coling-1.256 (citado nas pgs. 22–25).

[C. Ma *et al.* 2023] Chixiang Ma, Weihong Lin, Lei Sun e Qiang Huo. "Robust table detection and structure recognition from heterogeneous document images". Em: *Pattern Recognition* 133 (2023), pg. 109006 (citado na pg. 15).

[Maia *et al.* 2018] Ana Lucia Lima Marreiros Maia, Frank Dennis Julca-Aguilar e Nina Sumiko Tomita Hirata. "A machine learning approach for graph-based page segmentation". Em: *31st Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE. 2018, pgs. 424–431 (citado nas pgs. 47, 48).

[Marinai 2014] Simone Marinai. "Page similarity and classification". Em: *Handbook of Document Image Processing and Recognition*. Ed. por David Doermann e Karl Tombre. London: Springer London, 2014, pgs. 223–253. ISBN: 978-0-85729-859-1. DOI: 10.1007/978-0-85729-859-1_7. URL: https://doi.org/10.1007/978-0-85729-859-1_7 (citado nas pgs. 9, 10).

[Markewich *et al.* 2022] Logan Markewich *et al.* "Segmentation for document layout analysis: not dead yet". Em: *Int. J. Doc. Anal. Recognit.* 25.2 (2022), pgs. 67–77. ISSN: 1433-2833. DOI: 10.1007/s10032-021-00391-3. URL: https://doi.org/10.1007/s10032-021-00391-3 (citado nas pgs. 22–24).

[Mehri *et al.* 2015] Maroua Mehri, Nibal Nayef, Pierre Héroux, Petra Gomez-Krämer e Rémy Mullot. "Learning texture features for enhancement and segmentation of historical document images". Em: *Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing*. ACM. 2015, pgs. 47–54 (citado na pg. 12).

[Otsu 1979] Nobuyuki Otsu. "A threshold selection method from gray-level histograms". Em: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pgs. 62–66. DOI: 10.1109/TSMC.1979.4310076 (citado nas pgs. 14, 29).

[P. Soille 2003] P. Soille. *Morphological Image Analysis. Principles and Applications*. 2nd. Berlin: Springer-Verlag, 2003 (citado na pg. 15).

[Ren e Malik 2003] Xiaofeng Ren e Jitendra Malik. "Learning a classification model for segmentation". Em: *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE. 2003, pgs. 10–17. doi: 10.1109/ICCV.2003.1238308 (citado na pg. 12).

[Ronneberger *et al.* 2015] Olaf Ronneberger, Philipp Fischer e Thomas Brox. "U-net: convolutional networks for biomedical image segmentation". Em: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2015, pgs. 234–241 (citado na pg. 20).

[Rosenfeld 1974] Azriel Rosenfeld. "Adjacency in digital pictures". Em: *Information and Control* 26.1 (1974), pgs. 24–33 (citado na pg. 15).

[Sauvola e Pietikäinen 2000] Jaakko Sauvola e Matti Pietikäinen. "Adaptive document image binarization". Em: *Pattern recognition* 33.2 (2000), pgs. 225–236 (citado na pg. 68).

[Shafait e Smith 2010] Faisal Shafait e Ray Smith. "Table detection in heterogeneous documents". Em: *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. DAS'10. Boston, Massachusetts, USA: Association for Computing Machinery, 2010, pgs. 65–72. isbn: 9781605587738. doi: 10.1145/1815330.1815339. url: https://doi.org/10.1145/1815330.1815339 (citado na pg. 11).

[Tran, Na *et al.* 2016] Tuan Anh Tran, In Seop Na e Soo Hyung Kim. "Page segmentation using minimum homogeneity algorithm and adaptive mathematical morphology". Em: *International Journal on Document Analysis and Recognition (IJDAR)* 19.3 (2016), pgs. 191–209 (citado na pg. 15).

[Tran, Oh *et al.* 2017] Tuan Anh Tran, Kanghan Oh *et al.* "A robust system for document layout analysis using multilevel homogeneity structure". Em: *Expert Systems with Applications* 85 (2017), pgs. 99–113 (citado nas pgs. 11, 15).

[Wang *et al.* 2017] Murong Wang, Xiabi Liu, Yixuan Gao, Xiao Ma e Nouman Q. Soomro. "Superpixel segmentation: a benchmark". Em: *Signal Processing: Image Communication* 56 (2017), pgs. 28–39. issn: 0923-5965. doi: https://doi.org/10.1016/j.image.2017.04.007. url: https://www.sciencedirect.com/science/article/pii/S0923596517300735 (citado na pg. 12).

[Zou e J. Ma 2021] Yajun Zou e Jinwen Ma. "Deep learning based semantic page segmentation of document images in chinese and english". Em: *Intelligent Computing Theories and Application: 17th International Conference, ICIC*. Berlin, Heidelberg: Springer-Verlag, 2021, pgs. 484–498. isbn: 978-3-030-84521-6. doi: 10.1007/978-3-030-84522-3_40. url: https://doi.org/10.1007/978-3-030-84522-3_40 (citado nas pgs. 21, 23, 24).