University of Brasília

Institute of Exact Sciences
Department of Computer Science

Visual and Textual Feature Fusion for Document Analysis

Fusão de Informações Visuais e Textuais para Análise de Documentos

Patricia Medyna Lauritzen de Lucena Drumond

Thesis presented for conclusion of the Ph.D. Program in Computer Science

Supervisor
Prof. Dr. Teófilo Emidio de Campos

Brasília
2024

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

LAURITZEN DE LUCENA DRUMOND, PATRICIA MEDYNA
LD795v Visual and Textual Feature Fusion for Information Extraction of Document Image / PATRICIA MEDYNA LAURITZEN DE LUCENA DRUMOND; orientador TEÓFILO EMIDIO DE CAMPOS. -- Brasília, 2024.
75 p.

Tese(Doutorado em Informática) -- Universidade de Brasília, 2024.

1. Inteligência Artificial. 2. Linguagem de Processamento Natural. 3. Machine Learning. I. DE CAMPOS, TEÓFILO EMIDIO, orient. II. Título.

University of Brasília

Institute of Exact Sciences
Department of Computer Science

Visual and Textual Feature Fusion for Document Analysis

Fusão de Informações Visuais e Textuais para Análise de Documentos

Patricia Medyna Lauritzen de Lucena Drumond

Thesis presented for conclusion of the Ph.D. Program in Computer Science

Prof. Dr. Teófilo Emidio de Campos (Supervisor)
CIC/UnB

Prof. Dr. Fabricio Ataides Braz
FGA/UnB

Prof. Dr. Li Weigang
CIC/UnB

Prof. Dr. Carolina Scarton, PhD
University of Sheffield

Prof. Dr. Ricardo M. Marcacini
ICMC/USP

Prof. Dr. Ricardo Pezzuol Jacobi
Computer Science Graduate Program Coordinator

Brasília, March 12, 2024

Dedication

To my dearest family,

This thesis is a testament to the boundless love, unwavering support, and belief in my abilities that you, my family, have given me throughout this academic journey. I dedicate this work to you.

My eternal gratitude to my beloved father, Francisco Assis de Lucena (in memorial), who believed in my studies more than I believed in myself, who was proud of every step I reached in academia and praised me to everyone. Father, you have always been affectionate with me, you have loved me and blessed me since I was born, you have taught me the value of the simplest things, to smile and cry with emotion, live simply, enjoy every moment of life and always sing. Today you are not here to see this work completed, but you are present in my heart and in my memories. My mother, Eulineide Lauritzen de Lucena, her love, encouragement and guidance lit my way even when it was arduous. You are a wonderful mother, a great woman, and you taught me to fight for my ideals. Her example as a hard-working woman instilled in me the resilience to pursue this Ph.D. endeavor. Gratitude for everything. I love you.

To my loving husband, Raimundo Marques Campos Drumond Junior, his patience, understanding and constant support fueled my determination to reach this milestone. Your belief in me and your willingness to share the responsibilities of daily living so that I could focus on my studies were instrumental in my success. I dedicate my love, admiration and respect to you.

To my siblings, Veruscka, Cristiano, Rossana and Kalina, who filled my life with love, laughter and motivation. Your encouragement has been very important for the completion of this work. To my nephews and nieces, grand-nephews and godchildren, for the moments of love, and joy, which made my days lighter.

To the Drumond family, especially my parents in-laws, Raimundo and Carminha, for their welcome, love, affection and support. Finally, this thesis is dedicated to the entire family with a token of gratitude for their continued support and belief in my efforts. Thank you for being the pillars of strength in my life and for making this achievement possible. Love you all.

Acknowledgments

First of all, I thank GOD because without Him I am nothing. I can do all things through Christ who strengthens me.

I am deeply grateful to my supervisor, Teófilo de Campos, whose guidance, expertise and unwavering support have been fundamental throughout this doctoral journey. Your insightful feedback, encouragement, and collaborative approach enriched the quality of this work.

I extend my heartfelt appreciation to Fabricio Braz, who was my de-facto co-supervisor (even though this was never formalised), for his invaluable contributions, valuable insights, expertise, and the time he has dedicated to ensuring the success of this research. He strongly supported my whole trajectory, from the first to the last day of my PhD.

I also thank my committee members, Carolina Scarton, Ricardo Marcacini and Li Weigang, for investing time in reading and evaluating this thesis and for contributing to its improvement. I thank everyone from the KnEDLE project and AILab laboratory, especially professors Thiago Faleiros and Fabricio Braz for providing equipment to execute part of the work code.

I thank all the professors at UnB for their valuable teachings, stimulating discussions, and constructive feedback. In particular, I want to thank Li Weigang, Célia Ghedini, Claúdia Nalon and Alba Melo. I extend my thanks to the many colleagues who faced with me various challenges in the disciplines and supported me with constructive discussions and group study. In particular, Pedro Luz, Fred Guth, Dennis Sávio, George and Flávia. Many thanks to my colleague Lindeberg Leite who helped me with some experiments with BERT.

I am sincerely grateful to all the professors of the Information Systems course at CSHNB-UFPI, whose encouragement and friendship have been a constant source of strength and motivation throughout this doctoral journey. I extend my thanks to my friend Juscelino for his support and encouragement.

Special thanks to my friends Amélia, Anísia, Gilvana, Juliana, Kelly and Patricia Vieira who gave me strength in the most difficult times, to my family for giving me support in the midst of all difficult times, for not letting me give up and praying with me.

I cannot forget my dear nephew Felipe Lauritzen, an enthusiast who was always excited about my studies and who helped me with the English language.

I am grateful to the University of Brasília (UnB) for providing an environment conducive to research and learning. The resources, facilities and academic support were essential for the completion of this thesis. I am also grateful to the Federal University of Piauí (UFPI) for the support and encouragement of professional and personal qualification, with me taking time off from my teaching activities to dedicate myself exclusively to my doctorate. The present work was also supported by CAPES' Portal de Periódicos.

Abstract

The large volume of documents produced daily in all sectors, such as industry, commerce, and government agencies, has increased the amount of research aimed at automating the process of reading, understanding, and analyzing. Business documents can be born digital, as electronic files, or a digitized form that comes from writing or printed on paper. In addition, these documents often come in various layouts and formats. They can be organized differently, from plain text multi-column layouts and various tables/forms/figures. In many documents, the spatial relationship of text blocks usually contains important semantic information for downstream tasks. The relative position of text blocks plays a crucial role in document understanding. However, embedding layout information in the representation of a page instance is not trivial. In the last decade, Computer Vision (CV) and Natural Language Processing (NLP) pre-training techniques have been advancing in extracting content from document images considering visual, textual, and layout features. Deep learning methods, especially the pre-training technique, represented by the Transformer architecture, have become a new paradigm for solving various downstream tasks. However, a major drawback of such pre-trained models is that they require a high computational cost. Unlike these models, we propose LayoutQT, a simple rule-based spatial layout encoding method, which combines textual and spatial information from text blocks. Given that our focus is on developing a low computational cost solution, we performed the experiments with AWD-LSTM neural network. We show that this enables a standard NLP pipeline to be significantly enhanced without requiring expensive mid or high-level multimodal fusion. We evaluated our method on three datasets (Tobacco800, RVL-CDIP, and VICTOR) for page stream segmentation tasks and document image classification and identified an improvement in the results obtained about the baseline. For document page stream segmentation, the LayoutQT method combining text and layout features was evaluated with the following backbones: LSTM, AWD-LSTM, and BERT, leading to the F1 scores of 86.1%, 99.6% and 93.0%, respectively on the Tobacco-800 dataset. In contrast, the baseline results were F1 82.9%, 97.9% and 92.0%. For classifying documents on the RVL-CDIP dataset, our proposed approach also demonstrated superior performance, resulting in an advantage of 5.5% and 4.4% in the F1 score metric

compared to the baseline using AWD-LSTM and BERT models, respectively. Furthermore, the result of our approach obtained with the AWD-LSTM model was 1.4% better than that with BERT. Finally, the performance of our LayoutQT surpasses the state-of-the-art proposed by Luz et al. (2022) on the VICTOR dataset for document image classification, proving the effectiveness of our model.

Keywords: Document Intelligence, Natural Language Processing, Computer Vision, Document Image Classification

Fusão de Informações Visuais e Textuais para Análise de Documentos

Resumo Expandido

Diariamente é produzido um grande volume de documentos nas organizações industriais, comerciais, governamentais, entre outras. Além disso, com o mercado competitivo na internet, as transações de negócios têm crescido numa velocidade imensa. Esses fatos aumentam cada vez mais a necessidade da automação e extração de informações de documentos. Os documentos podem ter sido originados digitalmente como um arquivo eletrônico ou podem ser uma cópia digitalizada de documento impresso em papel. Esses documentos, geralmente, são ricos de informações visuais e podem estar organizados de diferentes maneiras, desde páginas simples contendo apenas texto, até páginas com layouts de várias colunas de texto e uma ampla variedade de elementos não textuais como figuras e tabelas. Para análise e classificação desses documentos a extração de informações baseadas somente em blocos de texto ou em características visuais nem sempre é eficaz. Em geral, a relação espacial desses elementos e blocos de texto contém informações semânticas cruciais para compreensão de documentos.

O processo de automação da análise e extração de informações de documentos é desafiador devido aos vários formatos e layouts dos documentos de negócios, e tem atraído a atenção em áreas de pesquisa como Visão Computacional (CV) e Processamento de Linguagem Natural (NLP). *Document Intelligence* é um termo recente utilizado para aplicações da Inteligência Artificial que envolve a automatização de leitura, compreensão e análise de documentos visualmente ricos de informação. O primeiro workshop de *Document Intelligence* (DI'2019) foi realizado no dia 14 de dezembro de 2019 na Conferência sobre Sistemas de Processamento de Informações Neurais (NeurIPS) em Vancouver, Canadá. Essas aplicações, também conhecidas como *Document AI*, são geralmente desenvolvidas para resolver tarefas como análise de layout de documentos, extração de

informações visuais, resposta-pergunta visuais de documento e classificação de imagem de documentos, etc.

Na última década, várias abordagens multimodais unindo técnicas de CV e NLP vêm avançando em tarefas de compreensão de documentos, como por exemplo, análise de layout, segmentação de páginas e classificação de imagens de documentos considerando a junção de pelo menos duas das modalidades de recursos: visuais, textuais e de layout. Existem algumas abordagens que foram propostas para lidar com layouts nas imagens do documento. As abordagens tradicionais baseadas em regras (top-down, bottom-up e híbridas) e as abordagens baseadas em Machine Learning e Deep Learning. No entanto, o surgimento da abordagem Deep Learning, principalmente com as técnicas de pré-treinamento, utilizando Redes Neurais Convolucionais e Arquitetura Transformer tem avançado em pesquisa reduzindo o número de pesquisas com abordagens tradicionais.

A tecnologia de Deep Learning usada em *Document Intelligence* envolve a extração de informações de diferentes tipos de documentos através de ferramentas de extração, como OCR, extração de HTML/XML e PDF. As informações de texto, layout e visuais depois de extraídas são pre-treinadas em redes neurais para realizar as tarefas downstream. O modelo de linguagem BERT (Bidirectional Encoder Representations from Transformers) tem sido usado como backbone para outros modelos de pre-treinamento combinando recursos visuais e textuais para tarefas downstream. Apesar do excelente desempenho dos modelos Transformer existem vários desafios associados à sua aplicabilidade para configurações prática. Os gargalos mais importantes incluem requisitos para grandes quantidades de dados de treinamento e altos custos computacionais associados.

Ao contrário desses modelos, nós propomos um método de codificação de layout espacial simples e tradicional baseado em regras, LayoutQT, que combina informações textuais e espaciais de blocos de texto. Nós mostramos que isso permite que um pipeline de NLP padrão seja significativamente aprimorado sem exigir custos de fusão multimodal de médio ou alto nível. O LayoutQT divide a imagem de documento em quadrantes e associa a cada quadrante um token. Na extração de blocos de texto, são inseridos os tokens relativo às posições de início e fim dos blocos de texto. Além disso, foram inseridos tokens relativos às posições centrais de texto. Para avaliar nosso método, nós realizamos experimentos utilizando as redes neurais LSTM e AWD-LSTM em três bases de dados (Tobacco800, RVL-CDIP e VICTOR) disponíveis publicamente, sendo uma para tarefas de segmentação de fluxo de páginas e as outras duas para classificação de imagens de documentos. A base de dados Tobacco800, possui 1.290 imagens de documentos dividida em duas classes (FirstPage e NextPage), utilizada para classificar se a imagem é a primeira página de um documento ou se é uma página de continuidade. RVL-CDIP contém 400.000 imagens de documentos divididos em 16 classes e é utilizada para classificação de documentos.

VICTOR é uma base de dados mais robusta contendo 692.966 documentos de processos judiciais do Supremo Tribunal Federal (STF) do Brasil compreendendo 4.603.784 páginas dividida em 6 classes. Essa base de dados faz parte de um projeto com mesmo nome, resultado da parceria entre a UnB, STF e a Finatec Como *baseline* realizamos os mesmos experimentos sem os tokens de posição.

Inicialmente nós escolhemos empiricamente dividir os documentos em 24 quadrantes, sendo 6 linhas por 4 colunas. Em seguida nós alteramos os parâmetros como valores de quadrantes, inserção/exclusão de tokens posicionais e realizamos vários experimentos com números de quadrantes diferentes, menos e mais do que 24. No entanto, os melhores resultados foram obtidos com os 24 quadrantes. Para segmentação de fluxo de páginas de documentos, o método LayoutQT combinando recursos de texto e layout obteve os melhores resultados, obtendo pontuação F1 usando LSTM, AWD-LSTM e BERT modelo, respectivamente de 86,1%, 99,6% e 93,0%. Em contraste, o resultado da *baseline* obteve F1 de 82,9%, 97,9% e 92,0% no conjunto de dados Tobacco-800. Para classificar documentos no conjunto de dados RVL-CDIP, nossa abordagem proposta também demonstrou desempenho superior, resultando em uma vantagem de 5,5% e 4,4% na métrica de pontuação F1 em comparação com a *baseline* usando os modelos AWD-LSTM e BERT, respectivamente. Além disso, o resultado da nossa abordagem obtido com o modelo AWD-LSTM foi 1,4% melhor do que com BERT. Por fim, o desempenho do nosso LayoutQT supera o estado da arte proposto por Luz et al. (2022) no conjunto de dados VICTOR para classificação de imagens de documentos, comprovando a eficácia do nosso modelo.

Palavras-chave: Inteligência de Documento, Processamento de Linguagem Natural, Visão Computacional, Classificação de Imagem de Documento

Contents

List of Acronyms and Abbreviations . . . xiv
1 Introduction . . . 1
1.1 Contextualization . . . 1
1.2 Problem Statement . . . 5
1.2.1 Document Image Classification . . . 7
1.2.2 Page Stream Segmentation . . . 7
1.3 Research Question and Objectives . . . 8
1.4 Contributions . . . 9
1.5 Document Outline . . . 9
2 Background and Related Concepts . . . 11
2.1 Processes of document digitalization . . . 11
2.1.1 Binarization . . . 12
2.1.2 Noise Removal . . . 13
2.1.3 Rectification . . . 13
2.1.4 Page Segmentation . . . 14
2.1.5 Zone Classification . . . 17
2.1.6 Reading Order Determination . . . 18
2.2 Feature Extraction . . . 19
2.3 Document Classification with Machine Learning Approaches . . . 20
2.3.1 Convolutional Neural Networks . . . 23
2.3.2 Recurrent Neural Networks . . . 23
2.3.3 Long Short Term Memory networks . . . 24
2.4 ULMFiT . . . 25
2.5 Transformers . . . 26
2.5.1 Pretraining Objectives Downstream Tasks . . . 30
2.6 Related Works . . . 32
2.6.1 Page Stream Segmentation . . . 32
2.6.2 Document Image Classification . . . 34
2.7 Summary . . . 37

3 Methodology . . . 39
3.1 Layout Quadrant Tags (LayoutQT) . . . 39
3.2 Baseline . . . 43
3.3 Metrics . . . 43
3.4 Evaluation . . . 45
3.5 Summary . . . 46
4 Experiments . . . 47
4.1 Experiment Setting . . . 47
4.2 Datasets . . . 48
4.2.1 Tobacco800 . . . 49
4.2.2 RVL-CDIP . . . 50
4.2.3 Tobacco-3482 . . . 52
4.2.4 VICTOR . . . 53
4.3 Results and Discussions . . . 55
4.3.1 Page Stream Segmentation on Tobacco800 dataset . . . 55
4.3.2 Document Classification on RVL-CDIP dataset . . . 58
4.3.3 Document Classification in Portuguese on the VICTOR dataset . . . 60
4.4 Summary . . . 65
5 Concluding Remarks . . . 66
5.1 Conclusion . . . 66
5.2 Limitations . . . 67
5.3 Future Works . . . 67
References . . . 69

List of Acronyms and Abbreviations

AI Artificial Intelligence

ANN Artificial Neural Network

ARE *Agravo de Recurso Extraordinário*

ASGD Asynchronous Stochastic Gradient Descent

AWD-LSTM ASGD Weight-Dropped Long Short-Term Memory

BERT Bidirectional Encoder Representations from Transformers

BoVW Bag of Visual Words

BPTT Backpropagation Through Time

BVic Big VICTOR

CCs Connected Components

CDIP Complex Document Information Processing

CNN Convolutional Neural Network

CPC Cell Position Classification

CV Computer Vision

DI Document Intelligence

DIC Document Image Classification

DL Deep Learning

DLA Document Layout Analysis

DNN Deep Neural Network

FFN FeedForward Network

Finatec Fundação de Empreendimentos Científicos e Tecnológicos

GNN Graph Neural Network

GPT Generative Pre-Training

IIT Illinois Institute of Technology

KNN K-Nearest Neighbor

LayoutQT Layout Quadrant Tags

LLMs Large Language Models

LSTM Long Short-Term Memory

LTDL Legacy Tobacco Documents Library

LTR Learning-To-Reconstruct

MDC Multi-label Document Classification

ML Machine Learning

MLM Masked Language Modeling

MLP Multilayer Perceptron

MM-MLM Multi-Modal Masked Language Modeling

MMT Multi-Modal Machine Translation

MVic Medium VICTOR Dataset

MVLM Masked Visual-Language Model

NeurIPS Conference on Neural Information Processing Systems

NLP Natural Language Processing

NSP Next Sentence Prediction

OCR Optical Caracter Recognition

PSS Page Stream Segmentation

R-CNN Regions with CNN features

RDF Random Decision Forest

RE *Recurso Extraordinário*

RNN Recurrent Neural Network

RVL-CDIP Ryerson Vision Lab Complex Document Information Processing

SLP Single-Layer Perceptron

STF Supremo Tribunal Federal

SVic Small VICTOR Dataset

SVic+ Extension of Small VICTOR Dataset

SVM Support Vector Machines

TDI Text Describes Image

TIA Text-Image Alignment

TIM Text-Image Matching

UCSF University of California San Francisco

ULMFiT Universal Language Model Fine-Tuning

UnB Universidade de Brasília

VGG16 Visual Geometry Group 16

VICTOR Legal Documents Dataset

VQA Visual Question Answering

Chapter 1

Introduction

This Chapter briefly contextualizes our field of study, the motivation, and the statement of the problem we intend to face. It also includes our objectives, the contributions we have achieved, and the expected contributions. To conclude the Chapter, an outline of the entire document is presented.

1.1 Contextualization

Business documents are essential for the operations carried out in their organizations. Automated processing has helped to organize and extract information from these documents. However, the massive amount of digitized documents produced in the last decades requires a significant effort in developing document image processing methods for information extraction. Document Artificial Intelligence or Document Intelligence (DI) [48], is an application of Artificial Intelligence (AI) that involves automatic reading, comprehension, and analysis of business documents. The first workshop on Document Intelligence was held on December 14, 2019, at the Conference on Neural Information Processing Systems (NeurIPS) in Vancouver, Canada [48]. Document AI is very challenging due to the diversity of layouts and formats from webpages, digital-born or scanned documents, low-quality scanned document images, and the template structure's complexity. With the various structures of business document images, extracting semantic information from its textual content favors downstream tasks such as document retrieval, information extraction, and text classification [14].

Furthermore, each document can also be classified as belonging to a certain class of documents. Given the image of a document, the layout can help to recognize and classify this document. Document image classification (DIC) is often an important step of the document image processing system. This classification aims to assign to document image to one or several pre-defined categories. The document image classification task often

facilitates the downstream process since images from different categories may undergo different processes. It can also help automate document image workflows by routing a document when classes of interest are detected [40].

Documents follow some layout, including vital structural and visual information (e.g., font sizes and geographic position of the text). It is important to locate the region of the structural elements, like text, figures, and tables; it contains most document layout information. Figure 1.1 presents four documents with different layouts: form, scientific publication, invoice, and memo. In addition, the information in business documents is presented in various ways, from plain text to multi-column formats and a wide variety of tables. These documents often reflect complex legal agreements and refer explicitly or implicitly to regulations, legislation, case law, and standard business practices. Consequently, the information is not easily accessible for extraction and recognition [72].

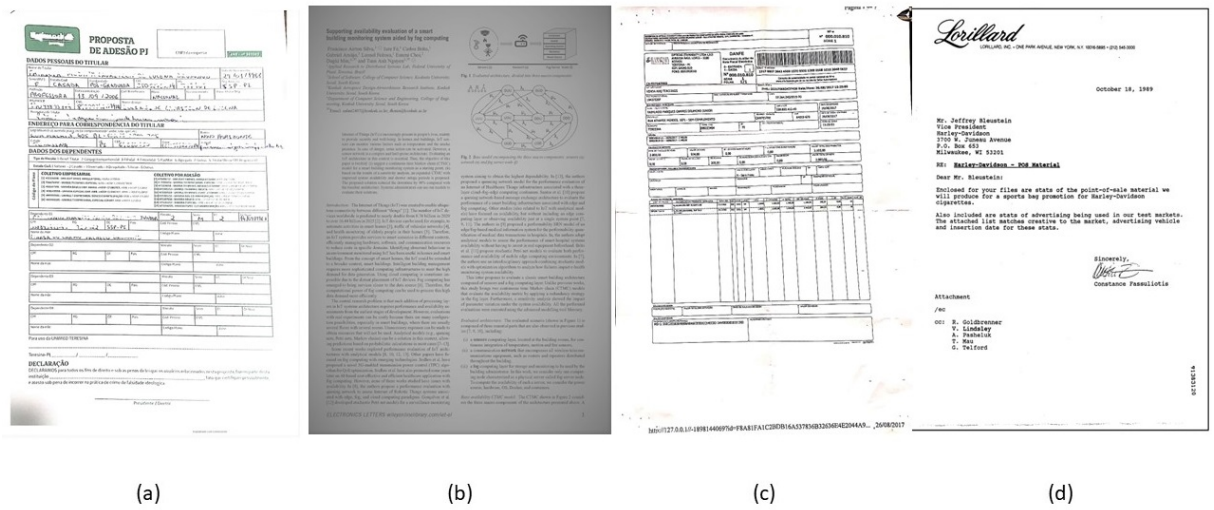


Figure 1.1: Examples of document images with different visual styles (a) a form, (b) a scientific publication page, (c) an invoice, and (d) a memo.

Due to the fundamental importance of document image classification, it has been explored extensively over the years with the growth of methods from computer vision (CV) and natural language processing (NLP). Computer vision methods have been used for optical character recognition (OCR) systems to extract text from image documents based on their visual appearance [5]. To some extent, OCR could be a solution that can extract the text from an image of a document and convert it into computer-readable form, which may further be used for editing. Nonetheless, OCR is prone to errors and is not always applicable to all documents, e.g., handwriting text is still difficult to read, and those document images must have high resolution [50]. The main issue with traditional OCR is that it does not extract and attach the positional values of the text with extracted text [22].

On the other hand, much of the relevant information is in the text, so extracting text-based information from documents has been the subject of NLP studies for some time. However, a system cannot rely on text alone but requires incorporating structure and image information. Although the text allows retrieving information about the document's content, the visual layout plays an equally important role [50]. The document layout comprises both the structure and visual information (e.g., font sizes, text centering, location of parts of the text) that are vital to the understanding of the document by readers but often ignored by models that consider only the textual content. Thus, combining visual, textual, and document image layout resources in extracting information is of great importance [31]. Recent approaches in literature have explored frameworks that utilize information from text, layout, and document images to serve specific downstream tasks. However, they are limited by the inability to learn cross-modal representations in text, layout, and image dimensions for documents and process multi-page documents.

With the acceleration of digitization, the structured analysis and content extraction of documents, images, and others has become a key part of digital success. Key information extraction from business document images requires understanding texts in various layouts. Many AI technologies have advanced to improve the use and handling of industrial documents, such as machine [44] and deep learning [78]. Deep learning methods have become a new paradigm for solving many machine learning problems.

In addition, most recent approaches try to solve the task by developing pre-training language models [27, 37, 72, 73] focusing on combining visual features from document images with texts and their layout using a unified Transformer architecture [65]. The development of Document AI also reflects a similar trend with other applications in deep learning, especially in the pre-training technique represented by Convolutional Neural Networks (CNN), Graph Neural Networks (GNN), and Transformer architecture. Among all these approaches, a typical pipeline for pre-training Document AI models usually starts with the vision-based understanding, such as Optical Character Recognition (OCR) or document layout analysis.

Furthermore, Large Language Models (LLMs) are gaining increasing popularity in academia and industry owing to their unprecedented performance in various applications. The core module behind many LLMs is the self-attention module in Transformers [65] and GPT [53], which serves as the fundamental building block for language modeling tasks. However, LLMs have high training and updating costs due to the high computational cost and the spatial complexity of the self-attention mechanism concerning the length of the input sequence. In real-world application scenarios, a typical Document Intelligence System mainly includes five types of tasks, namely: Document Layout Analysis, Visual Information Extraction, Document Visual Question Answering, Document Image

Classification, and Page Stream Segmentation [14].

Document Layout Analysis (DLA) is a means to identify different functional/logical content elements (e.g. sentences, titles, captions, author names, and addresses) on a given page. It is realized by segmenting physical contents (e.g. pixels, characters, words, lines, figures, tables, and background) on the page and classifying them into predefined functional/logical categories, in other words, by assigning these classified entity labels. Document layout analysis plays a crucial role within the document digitization procedure because the correctness of layout analysis determines whether a subsequent text recognition procedure is operated on the correct text object. When implementing layout analysis, there are generally two approaches to carry out this procedure, the top-down approach and bottom-up approach [39], discussed in subsection 2.1.4.

Visual Information Extraction refers to the technology of extracting semantic entities and their relationships from many unstructured visually-rich documents. Visual information extraction differs in different document categories, and the extracted entities are also different. Unlike traditional pure text information extraction, the construction of the document turns the text from a one-dimensional sequential arrangement into a two-dimensional spatial arrangement. This makes text, visual, and layout information extremely important influencing factors in visual information extraction [72].

Document Visual Question Answering (VQA) is a high-level understanding task for document images. Specifically, given a document image and a related question, the model needs to correctly answer the question based on the given image [14]. A set of VQA tasks is defined based on various application scenarios, including statistical charts, daily-life photos, and digital-born documents. Document VQA task aims to extract information from documents and answer natural language questions.

Document Image Classification is the process of analyzing and identifying document images while classifying them into different categories, such as scientific papers, resumes, invoices, receipts, and many others. Document image classification is a special subtask of image classification. Thus, classification models for natural images can also address the problem of document image classification [72]. Document Image Classification task tries to predict the class to which a document belongs by means of analyzing its image representation.

Page Stream Segmentation is the process of recovering document boundaries from aggregated streams of pages [64]. Page Stream Segmentation refers to the combined prob-

lem of both finding document separation points in an ordered collection of page images and assigning the correct semantic labels to the output documents [23]. One of the key steps in the batch scanning process is the segmentation of the resulting page stream into continuous sets of pages corresponding to the physical documents, a procedure also referred to as document separation.

For these five main Document AI tasks, there have been many open-sourced benchmark datasets in academia and industry, which has greatly promoted the development of new algorithms and models by researchers in related research areas. However, this work mainly focuses on approaches to document image classification and page stream segmentation tasks combining visual and textual features. The next section presents the document classification problem statement and divides it into two downstream tasks: Document Image Classification and Page Stream Segmentation.

1.2 Problem Statement

Automatic information extraction from documents is a challenging task. The physical documents are generally scanned or photographed before the information extraction process begins. Document classification has been widely adopted for various document image processing applications as a fundamental step of document-related tasks. Developing an automated system to classify arbitrary document images into their respective true categories is computationally complex. The complexity of this task is increased due to the similarity between document classes. Two documents from different classes may look similar, while two from the same class may look very different. For example, an advertisement may look like a news item, and scientific publications may appear very different depending on the publisher's layout (two vs. single-column).

Another challenge for the document image classifier is receiving many pages as input and needing to separate the documents when one document ends and another begins. This huge document flow must be processed accurately and quickly. Each scanned document must be segmented on subsequent pages and then forwarded to the right recipient to be processed and saved in a database. One of the key steps in the batch scanning process is segmenting the resulting page stream into continuous sets of pages corresponding to the physical documents, a procedure also referred to as document separation [23]. In this context, page stream segmentation refers to the combined problem of both finding document separation points in an ordered collection of page images and assigning the correct semantic labels to the output documents. The page stream does not contain any separator pages or other marks.

In contrast, the documents in the page stream comprise sets of pages that do not necessarily bear any similarity between each other. Only document-level labels are available, while there is no prior information about the number of documents in the stream. Furthermore, This separation can be done by identifying the first page of each document in the ordered flow of documents. Figure 1.2 illustrates this segmentation of document page images into different documents, recognizing the first page of each document. In this case, the pages of the documents are ordered from the first to the last page, and the sequence continues with the next document.

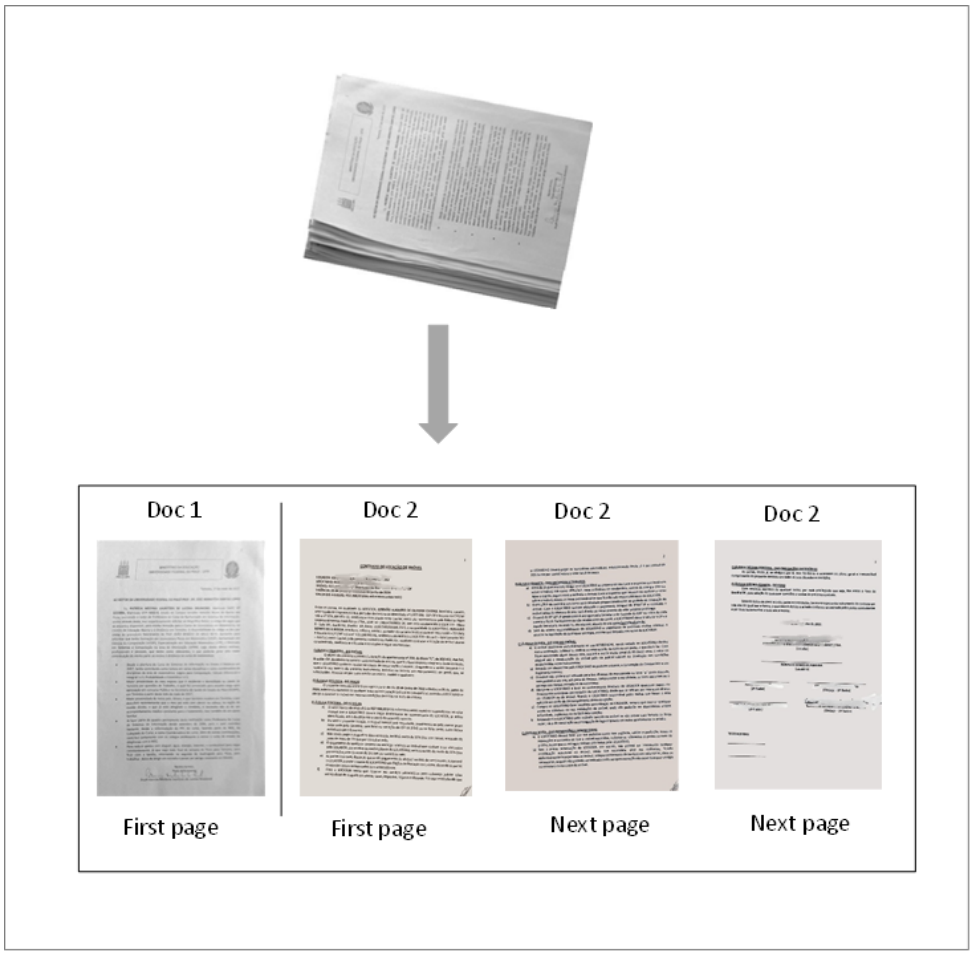


Figure 1.2: Illustration of page stream segmentation.

Some classifiers use only image, structural, or textual features; others use a combination of resources from several groups. Global image features are extracted directly from the entire document image, and local features are extracted from a segmented image region. Structural characteristics are obtained from physical or logical layout analysis. Textual features can be extracted from OCR output or directly from document images. Recent work [72, 73, 71] has used 2D coordinates to represent visual features of regions of interest (ROI) and join to text embedding.

1.2.1 Document Image Classification

Document image classification consists of assigning a document image to one of a set of predefined document classes. In most research papers and their respective datasets, the methods treat every page as a sample with a single class. Classification can be based on various features, such as visual, layout, or textual features. Classifiers solve various document classification problems, differ in how they use training data to construct models of document classes, and differ in their choice of document features and recognition algorithms. Choice of document features is an important step in classifier design [12].

Classification may be performed at different stages of document processing, with a diverse choice of document features, feature representations, class models, and classification algorithms. These aspects are interrelated: design decisions regarding one aspect influence the design of other aspects. For example, if document features are represented in fixed-length feature vectors, then statistical models and classification algorithms are usually considered [12].

Formally, let a set $D = \{d_1, d_2, ..., d_M\}$ of M documents and $C = \{c_1, c_2, ..., c_K\}$ represent the set of possible document classes, then for each d_i exists a class c_K such that $d_i \ \epsilon \ c_K$. The function $f : D \rightarrow C$ represents Document Image Classification DIC. In general, different documents d_K will contain a different number of pages, even if they belong to the same class [21].

The publicly available datasets for evaluating the performance of document image classifiers are Tobacco-3482 [32] and RVL-CDIP [26]. These datasets are subsets of annotated documents from the Truth Tobacco Industry dataset found in the literature. The VICTOR [17] dataset was built from Brazil's Supreme Court (Supremo Tribunal Federal or STF) digitalized legal documents. Three datasets are described in detail in 4.2.

1.2.2 Page Stream Segmentation

Page Stream Segmentation (PSS) is the task of automatically separating a stream of scanned document page images into a set of documents. In document digitization pipelines, it is common that multi-page digital documents arrive at the Document Management System as an ordered set of digital images without indicating the document boundaries. (PSS) consists in breaking pages into either continuity of the same document (SD) or the beginning of a new document (ND) [69].

Page Stream Segmentation PSS is defined as a function $g : P \rightarrow D$, where $P = \{p_1, p_2, ..., p_N\}$ is a set of N pages transformed to $D = \{d_1, d_2, ..., d_M\}$, set of M multi-page documents of sequential pages, using a binary classification function $g : \mathbb{N} \rightarrow \{0, 1\}$,

where $d_k = [p_i, p_i + 1, ..., p_j]$ for $i < j \leq N$. Here, 0 denotes the first page of any document, and 1 denotes any page other than the first page of any document [24].

Page stream segmentation is not a straightforward task because the limits of the documents are not always obvious, and it is not always easy to find common features between the pages of the same document. Some studies in the related domain have been conducted recently, depending on textual features [16] and some on image features [10] or combining both resources. An approach [69] that uses image features using convolutional neural networks (CNN) was built for PSS. By Braz et al. (2021) [10] improved the network architecture using EfficientNet pre-trained CNN architecture, replacing the earlier proposed VGG16 Network and focusing only on the image features.

A key challenge in PSS is finding real-world datasets containing publicly available multi-page documents. For this work, we identified two datasets. Tobacco800 [35] is a small annotated subset of the Truth Tobacco Industry Documents used in various PSS research, which is described in detail in 4.2.1.

This task is not trivial because the document categories can be numerous, and increasing the number of classes increases the complexity of the problem. This work proposes a preprocessing method to improve DIC and PSS tasks with a low computational cost. In the next section, we present research question and the objectives of our proposal.

1.3 Research Question and Objectives

Our main research question is "how can we take advantage of layout information to improve the representation of documents for document classification?". This question motivated the elaboration of the following secondary questions:

- "which approaches support researchers in the semantic representation of textual and visual features for information extraction?"
- "which natural language processing and computer vision techniques were used for document classification task?"
- "what metrics and datasets were used to evaluate the performance of these approaches?"

To answer the main question, we set out the following main objective: to propose, implement, and evaluate document processing methods that combine textual information and layout by performing experiments on document image classification and page stream segmentation tasks with low computational cost. More specifically, we aim to:

1) propose a joint feature learning approach that combines positional information of text block and text embedding for extracting information.

2) evaluate this approach for document classification and page stream segmentation tasks.

3) compare the models with baselines and state-of-art.

1.4 Contributions

Our LayoutQT method enriches the textual representation beyond the reading order and word context. The document is divided into quadrants that serve to mark the text box location within a page. These quadrants are then injected into the representation in the form of tags (spatial tokens), a bit like special words that do not belong to any language but carry layout information.

Instead of representing spatial tokens using fine Cartesian coordinates on the page, the spatial representation is highly quantised, which reduces the cardinality of the representations of coordinates, enabling their relevance to be learned by neural representations. This novel spatial representation of text blocks is encoded along with the text and passed to the language model.

Therefore, our main contribution is a novel approach to fuse textual and layout information which exploits a by-product of the text digitalization process, incurring insignificant additional computational cost in feature extraction. In addition, our model had real application in the classification of STF legal documents on the VICTOR dataset and showed an improvement over the baseline of at least 1.4% over the state-of-the-art [17].

The work has generated the following publication:

- De Lucena Drumond, P. M. L. *et al.* LayoutQT—Layout Quadrant Tags to embed visual features for document analysis [18].

1.5 Document Outline

This manuscript is structured into 6 chapters. Chapter 1 consists of this introduction, containing a brief contextualization that served as motivation for defining the problem, the research question, the objectives, and the contribution of the work.

In Chapter 2, we present concepts necessary to understand how document features are extracted from image processing. Initially, we show the physical layout analysis processes for and their strategies.

Chapter 3 describes the methodology of a new approach for inserting spatial tokens of text blocks into text embedding to improve document classification. LayoutQT divides

a document page into quadrants to indicate the location of blocks of text without using cartesian coordinates.

Chapter 4 presents the experiment scenario, four publicly available Document AI benchmarks for document image classification and page flow segmentation tasks. Furthermore, the results of the LayoutQT evaluation on the two downstream tasks are presented in tabular and graphical form.

Chapter 5 discusses the proposed model's final considerations and limitations and presents some proposals for future work.

Chapter 2

Background and Related Concepts

Paper documents have traditionally been the main source for acquiring, disseminating, and preserving knowledge. Automating the analysis and extraction of information from documents is challenging due to the various document formats and layouts. A scanning device can obtain a digital image of a paper document. Scanning a paper document into electronic format needs a way to transform the document into data structures that computers can understand. A document image may contain text, graphics, and tables. It can be organized differently, from simple pages containing only text to pages with layouts of several text columns.

Layout analysis is often a performance-limiting step of optical character recognition (OCR) systems since the errors made at this stage propagate to all further stages. Divining page images into text blocks and lines and determining their reading order is a major performance-limiting step in large-scale document digitization projects. Prior to text extraction using character recognition and word detection methods in OCR, a series of physical layout analysis processes are applied for document analysis. The rest of the chapter is organized as follows. Different processes involved in layout analysis are outlined in Section 2.1.

2.1 Processes of document digitalization

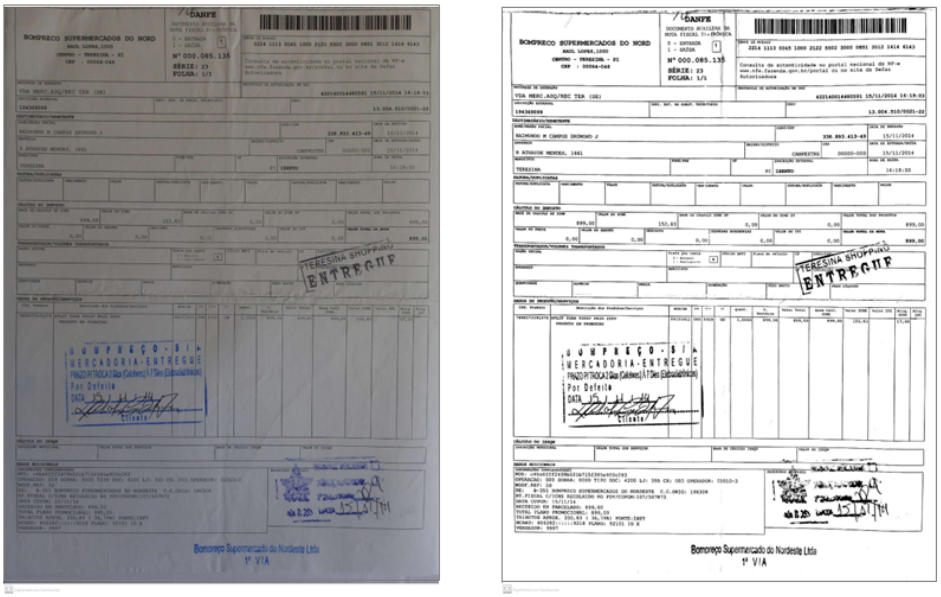
Physical layout analysis is the step that locates lines of text in the image and identifies its reading order, and involves different processes. In document layout analysis step, an input document image is segmented into different regions. These regions are then classified as text or non-text. The non-text regions are further classified into different sub-classes like table, image, separator, graphic, chart, etc., whereas text regions are classified as title, paragraph, header, footer, caption, drop-capital, etc [8]. Most of the layout analysis

systems use processes of binarization, noise removal, skew correction, page segmentation, zone classification and reading order determination in some form.

2.1.1 Binarization

Binarization is an important first step in most document analysis systems. To minimize the impact of physical document degradation on document image analysis tasks (e.g. page segmentation, OCR), it is often desirable to first the digital images. Such binarize degradations include non-uniform background, stains, faded ink, ink bleeding through the page, and uneven illumination. Binarization separates the raw content of the document from these noise factors by labeling each pixel as either foreground ink or background.

Document binarization aims to convert a given greyscale or color document image into a bi-level representation. When a document with black text on a white background is scanned with a flatbed scanner to convert it to digital form, noise from several sources is added to its digital counterpart. This noise comes from imaging mechanisms like finite spatial sampling rate, noise in electronic components, pixel sensor sensitivity variations, scanning processes like de-focusing non-uniform or poor illumination, and print-through from the other side of the page. Even if the original paper document was bi-level, the image obtained after scanning is usually greyscale. There are different binarization techniques like Otsu, Adaptive, Sauvola, Global threshold-based, etc. The result of running a binarization algorithm on a scanned document is shown in Figure 2.1.



(a) Input Image (b) Binarized Image

Figure 2.1: The result of applying binarization algorithm: a) the input image is the scanned image of a document. (b) image of the document after the binarization process.

Thresholding is one of the popular techniques for image binarization, and many methods based on this technique are proposed in the literature. Thresholding is usually performed in two ways: either globally or locally. In global thresholding, one threshold value is computed to segment the image, whereas, in local thresholding, many different threshold values are computed to divide the image into objects and backgrounds. The global thresholding methods are widely used in many document image analysis applications for their simplicity and efficiency. However, these methods are usually unsuitable for degraded document images because they need a clear bimodal pattern separating foreground text and background. So, the local thresholding methods are better for degraded document images with nonuniform background and foreground distribution. There are different binarization techniques like Otsu, Adaptive, Sauvola, Global threshold-based, etc.

2.1.2 Noise Removal

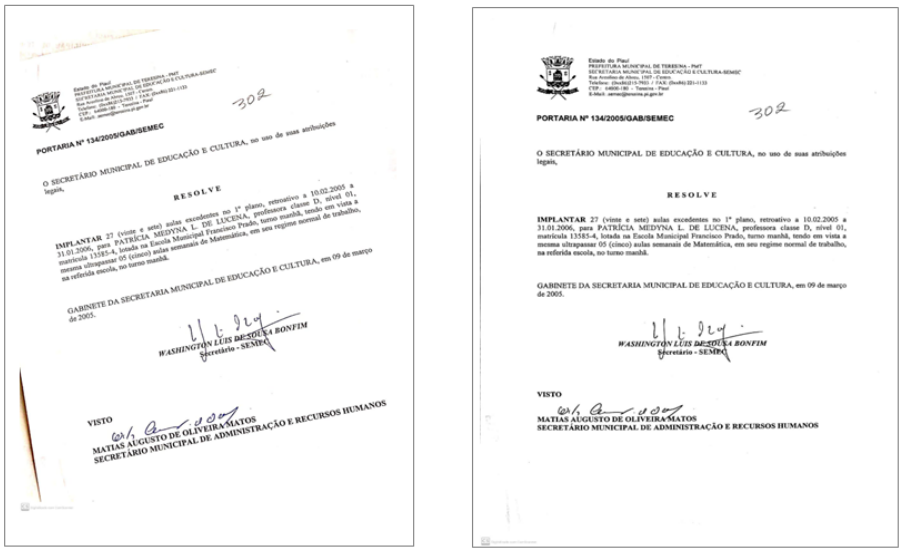
The documents may contain noise during scanning, transmission, or digital transformation. Usually, most of these contain defects and degradation such as complex backgrounds, non-uniform intensity, shadows, bleed-through ink, aging, ink fading, and holes. Noise is a common problem in most of the image understanding problems. Noise and blur can be caused by some situations, such as environmental conditions and image sensor problems due to miss-focus, motion, and transmission channels. Noise can also be induced in the document from sources like printing on low-quality paper, printing by old printers and old photocopying machines, images taken by portable cameras such as mobile phone cameras, or webcam of laptops [58].

The noise may be categorized after finding the features, and the same patterns in the document image may be found to obtain the accurate method for their elimination. Noise removal is a process that tries to detect and remove noisy pixels in a document introduced by scanning or binarization. Removing noise from the document image is important for developing a high-quality Optical Character Recognition (OCR) system.

2.1.3 Rectification

Rectification is a process that detects and corrects the deviation of a document's orientation angle from the horizontal direction (see Fig. 2.2). Rotation is introduced in a document image when a document is scanned or imaged at an angle concerning the reference axes. Paper positioning variations are a class of document degradation that results in skew and translation of the page contents in the scanned image. The problem of rectification plays an important role in the effectiveness of many document analysis

algorithms, such as text line estimation, region boundary detection, etc. For example, algorithms based on projection profiles assume an axis-aligned scan.



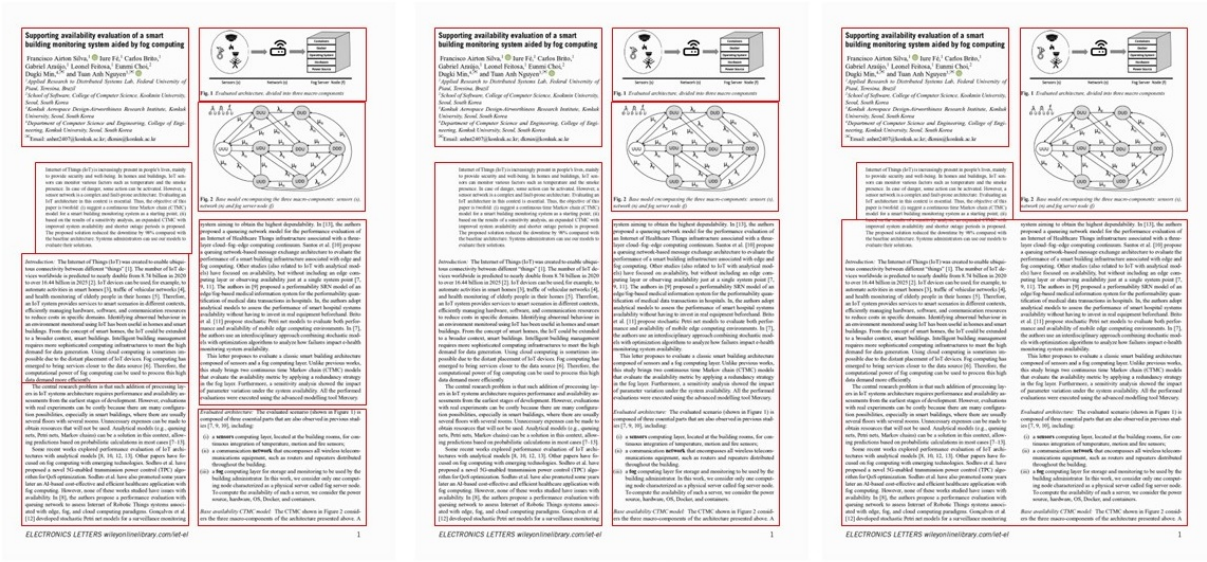
(a) Input Image (b) Output Image

Figure 2.2: Example of rectification of a document image with a rotation of 15 degrees.

The primary challenge in rectification is estimating the exact rotation angle of a document image. A variety of techniques are used for the detection of skew. Most of them assume the presence of some text component in the document and estimate the orientation of text lines using different methods. A commonly used technique is projection profiles, in which a given image is rotated at different angles for a range. The maximum difference between the peaks of the pixel histogram of that image at each angle is calculated. The angle of rotation for rectification will be the angle for which the maximum difference is obtained.

2.1.4 Page Segmentation

Page Segmentation is a process that divides a document image into homogeneous zones, each consisting of only one physical layout structure (text, graphics, pictures, etc.) while respecting the columnar structure of the document. Note that there are several possible ways to segment a document image correctly; this is the case with segments A and B in Fig. 2.3(a) and Fig. 2.3(b), respectively. At the same time, segmentation C in Figure Fig. 2.3(c) shows some common segmentation errors that negatively affect OCR accuracy. In that case, the page segmentation algorithm should segment all text columns separately so that the text-lines in different text-columns are not merged together.



(a) Segmentation A (b) Segmentation B (c) Segmentation C

Figure 2.3: An example image showing different segmentations of the same document. Segmentations A and B are both correct as they separate text in different columns as well as images from text. Segmentation C is considered to have two errors: 1) text in the first two columns is merged 2) Caption of the top figure is merged with the figure itself.

Page segmentation is a key component of geometric layout analysis. The performance of OCR systems depends heavily on the page segmentation algorithm. The segments thus obtained are classified as containing text or non-text elements. The text segments or zones are then fed to a character recognition module to convert them into electronic format. If a page segmentation algorithm fails to segment text from images correctly, the character recognition module outputs many garbage characters originating from the image parts. Page segmentation algorithms can be categorized into three analysis methods: top-down, bottom-up, and hybrid approaches. These methods rely heavily on heuristic rules and require many parameters to improve performance. When the layout of a document is relatively complex, these methods may fail to deliver optimal results.

Top-down separates the original document into different regions and then uses many heuristic filters to classify each region [38, 52]. Top-down algorithm starts from the whole document image and iteratively splits it into smaller ranges using the horizontal and vertical projection profiles. It segments a page as a whole into one or more content blocks and recursively segments the segmented blocks into paragraphs, lines, words, and characters. The splitting procedure stops when some criterion is met, and the obtained ranges constitute the final segmentation results. These methods were proposed quite early; some famous examples of these top-down algorithms are the recursive X-Y cut algorithm [49], the constrained run-length algorithm (CRLA) [66], and whitespace analysis

[6]. Traditional top-down methods are only effective when the document has a Manhattan layout[1] [62]. While these methods work well in some documents, they require much human effort to discover better rules. These methods have a low generalization capability since they depend on the layout structure of the document represented in the input image. Furthermore, they depend highly on the parameters chosen based on a priori knowledge of the layout structure, which can vary greatly. In recent decades, documents have become more varied and complex and do not necessarily follow those rules.

Bottom-up methods are more flexible as they do not require prior knowledge of the layout structure. Instead, they operate by processing an image from its lowest levels, such as its pixels or connected components, and increasingly group them into higher-level regions. The first group of connected components is produced by the black and white pixel in characters, then words, then lines, then text blocks [39]. The document segmentation process combines them in blocks or paragraphs according to the different structural characteristics. The Docstrum algorithm [51], the Voronoi-Diagram-based algorithm [30], and the Run-Length Smearing Algorithm (RLSA) [70] are typical bottom-up algorithms. Bottom-up methods are usually applicable to various layouts but are generally at least quadratic in time and space. These methods use a lot of memory space and are time-consuming. They need higher computational costs as an exchange.

Hybrid Methods are created from the combination of the two basic approaches. One of the most representative methods is Connected Components (CCs) analysis: CCs are detected from the entire images first, and then researchers analyze these CCs to acquire areas of interest [11, 60, 62, 63]. These algorithms mostly analyzed the connected components and the whitespaces between them. Hybrid methods can handle a variety of documents at a relatively fast speed. However, the results of these methods are still not convincing for problems such as non-text identification. Hybrid methods overcome the error accumulation from low-level components by foreground analysis and the under-segmentation of background analysis due to their ignorance of foreground components' homogeneity. However, it is not trivial to extract proper background regions and to group them into separators accurately. There is no feasible general approach to achieve this goal [11].

These rule-based methods are mostly developed to perform document layout analysis. A DLA system primarily segments an input document image into various regions and classifies these as text or non-text regions. The non-text regions are further classified into

[1]Manhattan layouts are defined as layouts that can be decomposed into individual segments by vertical and horizontal cuts.

sub-classes like table, image, separator, graphic, and chart. In contrast, text regions are classified as title, paragraph, header, footer, caption, etc. [8].

2.1.5 Zone Classification

Zone Classification aims at classifying the blocks detected by the page segmentation step (2.1.4) of a geometric layout analysis system into one of a set of predefined classes. A document is varied in content. It can contain text, math, figure zones, etc. Each of these zones has its characteristic features. Every page contains numerous zones. Each zone is specified by a rectangular box enclosing it and its type, as shown in Figure 2.4. The classifier uses properties of each zone for the process of classification. A zone classification is associated with a multi-class discrimination problem. Blocks identified as text can then be fed to a character recognition module. Similarly, other actions can be taken for zones of specific types; for instance graphics regions can be sent to a raster to a vector conversion program, whereas table zones can be fed to a table understanding system.

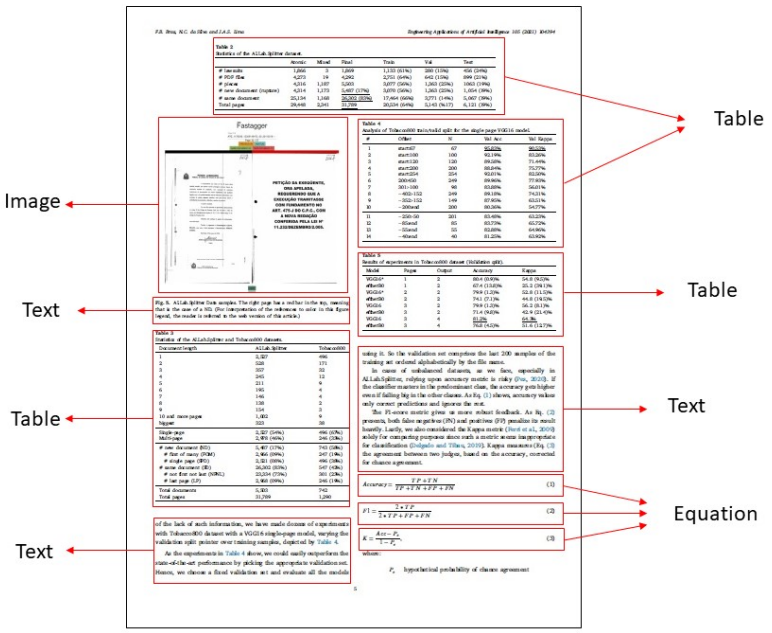


Figure 2.4: Example image showing a document with zone classification.

A complete system for extracting information from document images can transform paper documents into a hierarchical representation of their structure and content. The transformed representation of the document allows the exchange of documents, editing, navigation, indexing, archiving, and retrieval. The zone classification method plays a key role in the success of such a document understanding system. It is useful for successive applications like OCR and table understanding and can also assist and validate document

segmentation. In the design of a zone classifier, a set of measurements are made in the area. Each measurement is a feature. Some features are calculated along the zone's horizontal, vertical, right, and left diagonal directions [67].

2.1.6 Reading Order Determination

Reading Order Determination tries to recover the order in which a human will go through different parts (segments) of the document. Detecting the reading order among the layout components of a document's page is fundamental to ensure the effectiveness or even applicability of subsequent content extraction steps. While in single-column documents the reading flow can be straightforwardly determined, in more complex documents the task may become very hard. In some cases, the document layout is quite complex, requiring suitable strategies to determine the correct reading order of these components. Figure 2.5 exemplifies the detection of the reading order of two different documents. Current off-the-shelf methods usually directly borrow the results from the OCR engines [72], while most OCR engines arrange the recognized tokens or text lines in a top-to-bottom and left-to-right way [13].

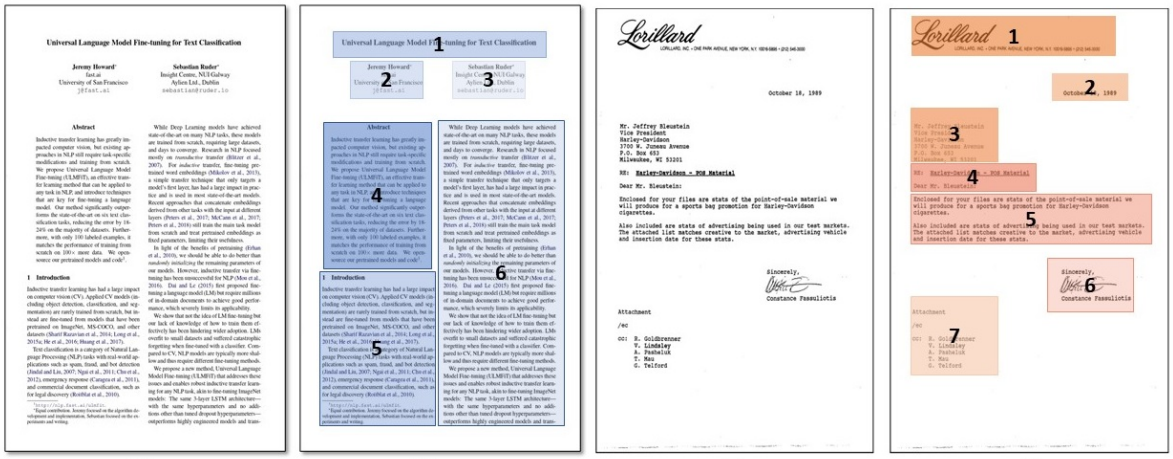


Figure 2.5: An example image showing two documents with reading order detection performed correctly for each: (a) and (b) a scientific article and its correct reading order detection, respectively. (c) and (d) an email and its detection of the correct reading order, respectively.

Many applications require sorting elements according to a partial or a total ordering relationship. The problem can be efficiently solved by applying sorting algorithms when the ordering relationship is known a priori. However, there are cases where no definition of an ordering relationship is available due to several difficulties in formalizing one. A typical example is represented by newspapers, where a page includes several articles that can be read independently and have a well-defined reading order among their constituent

components. Since the various articles are composed on the page in several unpredictable combinations and have different sizes and numbers of components, a simple topdown, left-to-right reading order of the pages would be ineffective, returning a text flow that interleaves components from different unrelated articles. So, Reading Order Detection in a document is a hot problem. In some classes of documents (e.g., single- or multi-column scientific articles) the reading order is quite stable and hence might, in principle, be learned if suitable examples for that class are provided. Reading order detection was first proposed by Aiello and Smeulders (2003) [2], using a propositional approach language of qualitative rectangular relations to detect reading order from document images. Rule-based approaches usually aim at one specific domain, while learning-based approaches are more general but have scalability issues. The work of [68] proposed ReadingBank, a benchmark dataset with 500,000 real-world document images for reading order detection.

Binarization, noise removal and rectification are typically considered as pre-processing steps in layout analysis. The core part of geometric layout analysis consists of page segmentation and zone classification modules. Reading order determination is generally considered a post-processing step in which a simple ordering criterion can be used to identify the reading order of the detected page segments. Specific layout approaches have been proposed in the literature where knowledge used to label zones in document images comes from geometric characteristics and the physical appearance of the layouts that the model has already seen during training. Existing approaches for document image classification and retrieval differ from each other based both on the type of extracted information (textual or visual) and/or the type of image analysis that is performed over the processed documents (global or local) [50].

2.2 Feature Extraction

Once the text blocks have been extracted and reading order has been determined, algorithms can focus on the extraction of the text from within each text box to assemble words, paragraphs, section titles, etc. This is the core process of OCR engines, which analyze the pixel-level information within each bounding box to recognize and transcribe the text. A wide range of OCR methods have been proposed over the years. Some methods start by detecting each individual character, which traditionally was done by locating connected components in binary images and more recently is done using methods that are similar to generic object detectors. Once characters are detected, they can be classified into the letters, numbers and other symbols of the vocabulary using machine learning techniques.

To group characters into words and then use the context to fix potential mistakes of character detectors and classifiers, sequence processing methods can be employed.

OCR systems as well as formatted document file formats (such as PDF) internally store information about the 2D coordinates of word bounding boxes and even individual characters. Images are 2-dimensional pixel values, while text is typically treated as a one-dimensional sequence, making it difficult to perform early fusion directly with their raw representations. In the case of document image classification, the text produced through OCR comes directly from the document. Hence, the spatial location of the text can be used to embed a feature representation corresponding to the 2-dimensional bounding box of where the text occurs in the document.

In order to use the extracted text for machine learning applications, it is necessary to convert the sequence of characters into a sequence of vectors that represents the text. These sequences are known as textual embedding, which are dense, low-dimensional vector representations that capture textual content's semantic meaning and contextual relationships. Popular techniques for generating textual embedding include word embedding such as Word2Vec, GloVe, and FastText and more advanced methods such as transformer-based models like BERT [19], GPT [53], and RoBERTa [41].

These textual embeddings enable the numerical representation of textual content, facilitating the application of machine learning and deep learning models for various text-related tasks. By encoding textual information into dense vector representations, textual embeddings preserve semantic similarities between words, sentences, and documents, enabling algorithms to understand and process textual content more effectively. Recent advancements in computer vision and natural language processing have led to significant text extraction and representation progress. Some works [68, 71, 73, 72] combine visual and textual features, using the 2-coordinates of the text block position in text embedding to improve document classification.

2.3 Document Classification with Machine Learning Approaches

All processes of document analysis that can be modeled as classification or regression problems can be dealt with using Machine Learning (ML) approaches. Some researchers define Machine Learning as a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns, and make decisions with minimal human intervention. Algorithms and statistical frameworks help the system learn by itself and make predictions about certain functions. Image classification and text extraction are some of the applications of machine learning.

Machine learning can be divided into supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. In supervised learning, the corresponding outputs of the training data have been labeled. In contrast, the corresponding outputs of the training data in unsupervised learning are unlabeled. For semi-supervised learning, some training data are labeled, and the remaining data are unlabeled; the amount of unlabeled data often exceeds the number of labeled data. In reinforcement learning, reinforcement signals provided by the environment are used to evaluate the quality of the generated actions and improve the strategies for adapting to the environment.

Machine learning techniques create a predictor, such as a classifier or a regressor, through an inductive learning process. A classifier is created based on relationships between documents and associated labels in the document parsing task. Then the algorithm classifies a document not yet known in one of the categories learned in the training phase, making decisions based on experiences gained through previous successful problem-solving. Several classic machine learning techniques, such as support vector machine (SVM) [20], K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP) [57], Adaptive impulse decision tree (Adaboost) [34] and Artificial Neural Networks (ANN) [45], have been applied to classification.

Artificial Neural Networks (ANNs) are inspired by brain studies and based on the operation of biological neural networks. They contain a series of mathematical equations that simulate biological systems processes such as learning and memory. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. ANNs learning process involves adjustments to the synaptic connections between the neurons. ANNs combine several artificial neurons to process information. Neural networks are trained to execute complex functions in various fields of application, including pattern recognition, identification, classification, clustering, speech, vision, and control systems. ANNs combine several artificial neurons to process information.

Artificial neurons essentially consist of 'inputs', which are multiplied by 'weights' and then computed by a mathematical function, which determines the 'activation' of the neuron, as depicted in Fig. 2.6 (a). Another function computes the 'output' of the artificial neuron, sometimes dependent on a certain 'threshold'. Weights can also be negative, so it can be said that the negative weight inhibits the signal. Depending on the weights, the computation of the neuron will be different. The weights are iteratively adjusted during the learning or training process until the output for specific inputs is close to the desired one.

Figure 2.6 (b) shows an ANN, consisting of a layer of input and output nodes (neurons) connected by one or more layers of hidden nodes. Input layer nodes pass information to

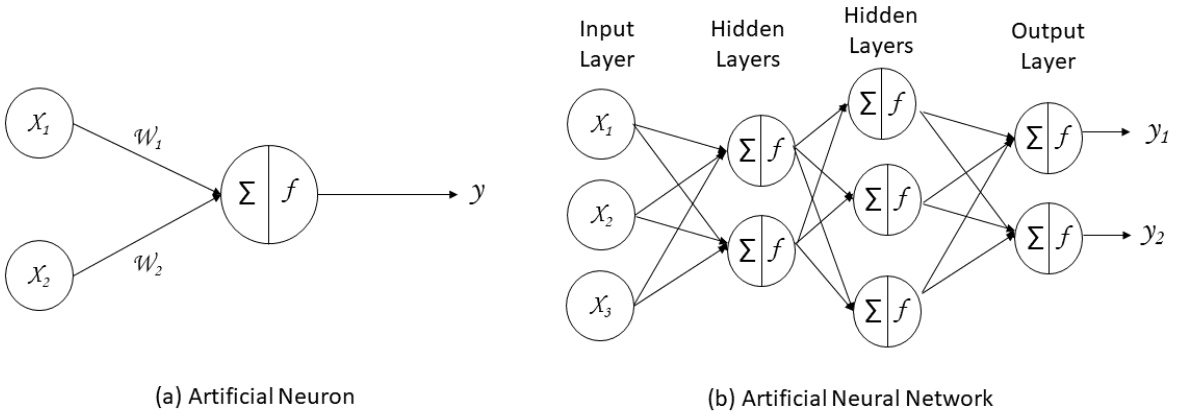


Figure 2.6: Illustration of an artificial neuron (a) and a single artificial neural network (b)

hidden layer nodes by firing activation functions, and hidden layer nodes fire or remain dormant depending on the evidence presented. The hidden layers apply weighting functions to the evidence, and when the value of a particular node or set of nodes in the hidden layer reaches some threshold, a value is passed to one or more nodes in the output layer.

Feedforward artificial neural networks ANNs have a unidirectional flow of information, while feedback ANNs return feedback. Single-layer perceptrons (SLPs) are simple feedforward ANNs often used for linear binary data classification. In contrast, multilayer perceptions (MLPs) feature not only an input layer and an output layer but also one or more hidden layers of fully connected neurons. Unlike SLPs, they incorporate nonlinear activation functions. Applying supervised machine learning with multilayer perceptrons falls under deep learning (DL) techniques. A multilayer perceptron is a class of feedforward artificial neural networks. A MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a non-linear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

Deep learning (DL) methods have recently become a new paradigm for solving many machine learning problems. Deep Learning is a branch of machine learning that deals with deep neural networks, where each layer is trained to extract higher-level representations of the previous ones. Deep learning methods have been confirmed to be effective in many research areas [78].

2.3.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a Deep Learning algorithm that can capture an input image, assign importance (learned weights and biases) to various aspects/objects of the image, and differentiate one from the other. These algorithms can identify faces, individuals, objects, characters, and many other aspects of visual data. Convolutional networks perform OCR to digitize text and make natural language processing possible in analogue and handwritten documents, where images are symbols to be transcribed.

A Convolutional Neural Network (CNN) is a regularized form of Multilayer Perceptron (MLP) with a layer (convolutional layer) that usually applies a rectified linear unit activation function. Unlike MLPs with fully connected neurons, in CNNs, the input data are convolved with individual neurons in the convolutional layer receiving data only for a specific receptive field. This reduces the probability of data overfitting, a disadvantage of MLPs.

Recently, deep learning has been widely explored in document layout classification. A fast CNN based document layout analysis was introduced, where two one-dimensional projections of images were considered to train the model. A CNN architecture that learns a hierarchy of features from a raw image was proposed for the document image classification to identify complex document layouts. A Deep CNN architecture was applied for classification, where CNNs were extensively used for feature extraction and model training.

2.3.2 Recurrent Neural Networks

A Recurrent Neural Network (RNN) is a special artificial neural network adapted to work for time series data or data that involves sequences of data such as text. These Neural Networks have been applied to several problems, such as NLP tasks, speech recognition, genomes, and numerical series. RNNs have the concept of 'memory' that helps them store the states or information of previous inputs to generate the next sequence output. The decision of a recurrent step reached in time step 1 affects the decision to reach a later time. Thus, recurrent networks have two input sources, the present and the recent past, which are combined to determine the result on new data, as shown in Fig. 2.7.

Recurrent Neural Networks leverage the backpropagation through time (BPTT) algorithm to determine the gradients. BPTT slightly differs from traditional backpropagation as it is specific to sequence data. It also differs from the traditional approach in that BPTT sums errors at each time step, whereas feedforward networks do not need to sum errors as they do not share parameters across each layer.

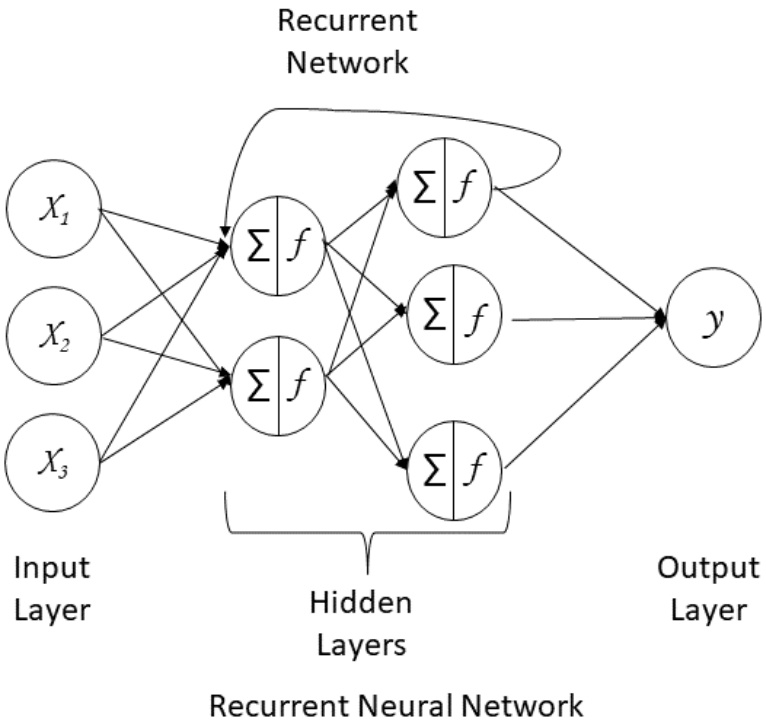


Figure 2.7: Illustration of a Recurrent Neural Network.

2.3.3 Long Short Term Memory networks

Long Short-Term Memory Networks (LSTMs) [61] are a special kind of RNN, capable of learning long-term dependencies. They work tremendously well on many problems and are widely used in NLP. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods is practically their default behavior. The basic difference between the architectures of RNNs and LSTMs is that the hidden layer of LSTM is a gated unit or gated cell. It consists of four layers that interact with one another to produce the output of that cell along with the cell state. These two things are then passed onto the next hidden layer.

The LSTM consists of three parts, as shown in Fig. 2.8, and each part performs an individual function. At a high level, LSTM works like an RNN cell. LSTM cells possess three gates, an input, a forget, and an output gate, that allow changes on a cell state vector propagated iteratively to capture long-term dependencies. This controlled information flow within the cell enables the network to memorize multiple time dependencies with different characteristics. LSTM provides a mechanism that limits the change gradient realized at each iteration. Hence, LSTM does not allow past information to be completely discarded.

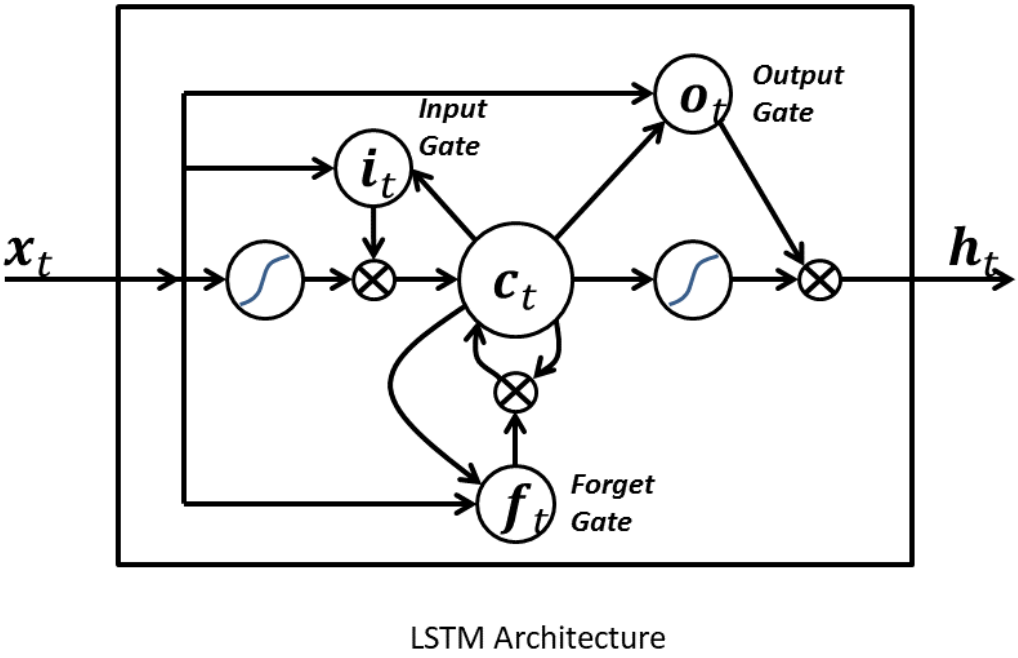


Figure 2.8: Illustration of the main elements of the architecture of the cell of a Long Short Term Memory network.

2.4 ULMFiT

Universal Language Model Fine-tuning (ULMFiT) [29] was a pioneering transfer learning method proposed for NLP tasks.

ULMFiT consists of the following steps, as shown in Fig. 2.9: In the first step, a Language Model is pre-trained on a large general-domain corpus to capture general features of the language in different layers. Then, the model can predict the next word in a sequence (with a certain degree of certainty). Following the transfer learning approach, the knowledge gained in the first step should be utilized for the target task. However, the target task dataset is likely from a different distribution than the source task dataset. The LM is consequently fine-tuned on the target task data in the second stage to address this issue. The full LM is fine-tuned on target task data using discriminative fine-tuning following a slanted triangular learning rate policy to learn task-specific features. Finally, the classifier is fine-tuned on the target task in the third stage using gradual unfreezing. This strategy preserves low-level representations and adapts high-level ones.

ULMFiT involves a 3-layer architecture for its representations, ASGD Weight-Dropped LSTM [47], a.k.a. AWD-LSTM. The AWD-LSTM architecture is a type of recurrent neural network that employs DropConnect for regularization, as well as NT-ASGD for optimization - non-monotonically triggered averaged Stochastic Gradient Descent - which returns an average of the last iterations of weights. Additional regularization tech-

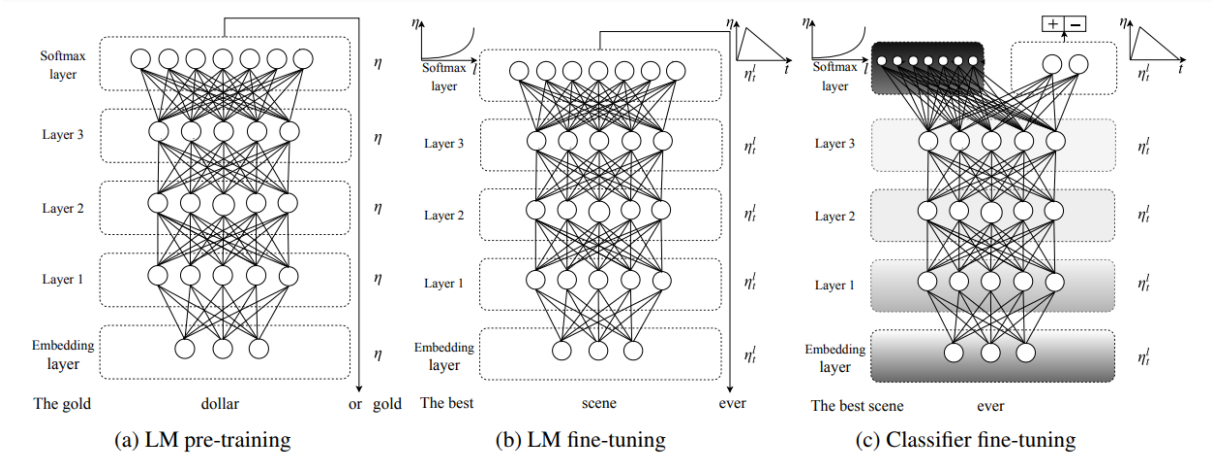


Figure 2.9: Illustration of ULMFiT architecture.
Source: reproduced from Howard and Ruder (2018) [29] (2018)

niques include variable-length backpropagation sequences, variational dropout, embedding dropout, weight tying, independent embedding/hidden size, activation regularization, and temporal activation regularization.

2.5 Transformers

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values and output are all vectors. The output is computed as a weighted sum of the values, where a compatibility function of the query with the corresponding key computes the weight assigned to each value. The vanilla transformer [65] is the first transduction model relying entirely on an attention mechanism without using sequence-aligned RNNs or convolution to draw global dependencies between input and output. The original Transformer model follows the architecture of Figure 2.10 using six stacked self-attention layers. The output of layer l is the input of layer l+1 until the final prediction is reached.

Each encoder layer has two sub-layers. The first is a multi-head self-attention [2] mechanism, and the second is a simple, position-wise, fully connected feed-forward network. A residual connection surrounds each main sub-layer in the Transformer model. These connections transport the unprocessed input of a sub-layer to a layer normalization function. This way, we are certain that key information such as positional encoding is not lost on the way [55].

[2]Self-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence

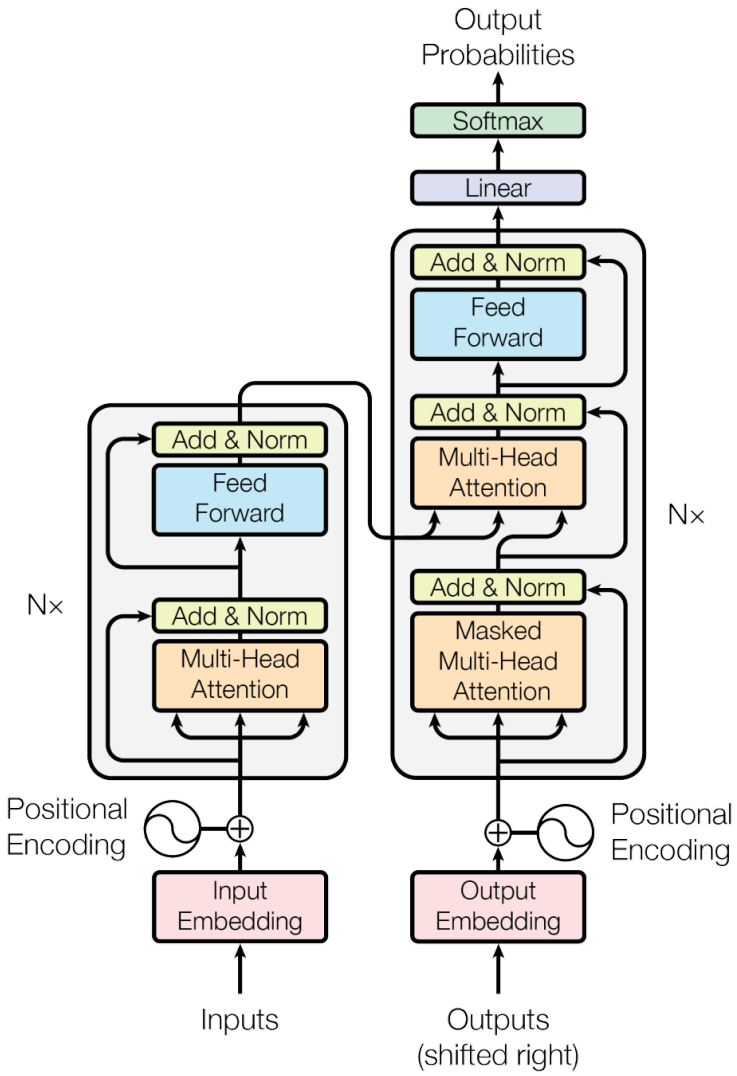


Figure 2.10: Vanilla Transformer Model Architecture [55, 65]. On the left, there is an N = 6 layers encoder stack. The inputs enter the encoder side of the Transformer through an attention sub-layer and FeedForward Network (FFN) sub-layer. On the right, there is an N = 6 layers decoder stack. The target outputs go into the decoder side of the Transformer through two attention sub-layers and an FFN sub-layer.

The decoder layer structure remains the same as the encoder layer for all N = 6 layers of the Transformer model. Each layer contains three sub-layers: a multi-headed masked attention mechanism, a multi-headed attention mechanism, and a fully connected position-wise feed-forward network. The decoder has a third main sub-layer, the masked multi-head attention mechanism. In this sublayer output, the following words are masked at a certain position, so Transformer bases its assumptions on its inferences without seeing the rest of the sequence. The Transformer only performs a small, constant number of steps (chosen empirically). Each step applies a self-attention mechanism that directly models relationships between all words in a sentence, regardless of their respective position.

In recent years, self-attention-based models like Transformers, Bidirectional Encoder Representations from Transformers (BERT) [19], and GPT models have achieved state-of-the-art performance on several Natural Language Processing tasks. The BERT model is designed to pre-train deep bidirectional representations from the unlabeled text by jointly conditioning on both left and right contexts in all layers. The overall framework of BERT is a multi-layer bidirectional Transformer encoder as shown in Fig. 2.11. It accepts a sequence of tokens and stacks multiple layers to produce final representations.

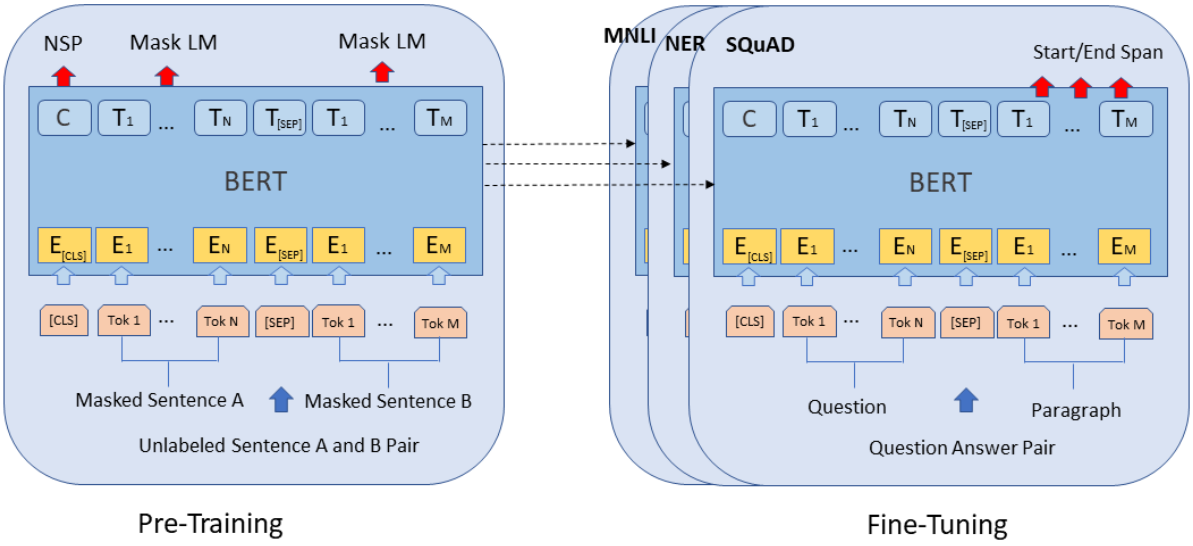


Figure 2.11: The overall framework of BERT adapted from Devlin et al. (2019) [19]. Apart from output layers, the same architectures are used in pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different downstream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added before every input example, and [SEP] is a special separator token.

There are two steps in the framework of the BERT [19]: pre-training and fine-tuning. During the pre-training, the model uses two objectives to learn the language representation: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP), where MLM randomly masks some input tokens, and the objective is to recover these masked

tokens, and NSP is a binary classification task taking a pair of sentences as inputs and classifying whether they are two consecutive sentences, further discussed in section 2.5.1. In fine-tuning, task-specific datasets are used to update all parameters end-to-end.

LayoutLM [72] model is proposed as the pioneer pre-training method of text and layout for document image understanding tasks, which expands 1D positional encoding of BERT to 2D to avoid the loss of layout information. It is trained over a large corpus of business documents to understand spatial dependencies between text blocks. Image embeddings are combined in the fine-tuning stage, and the image information is integrated into the pre-training stage. The overall framework of LayoutLM is shown in Fig. 2.12.

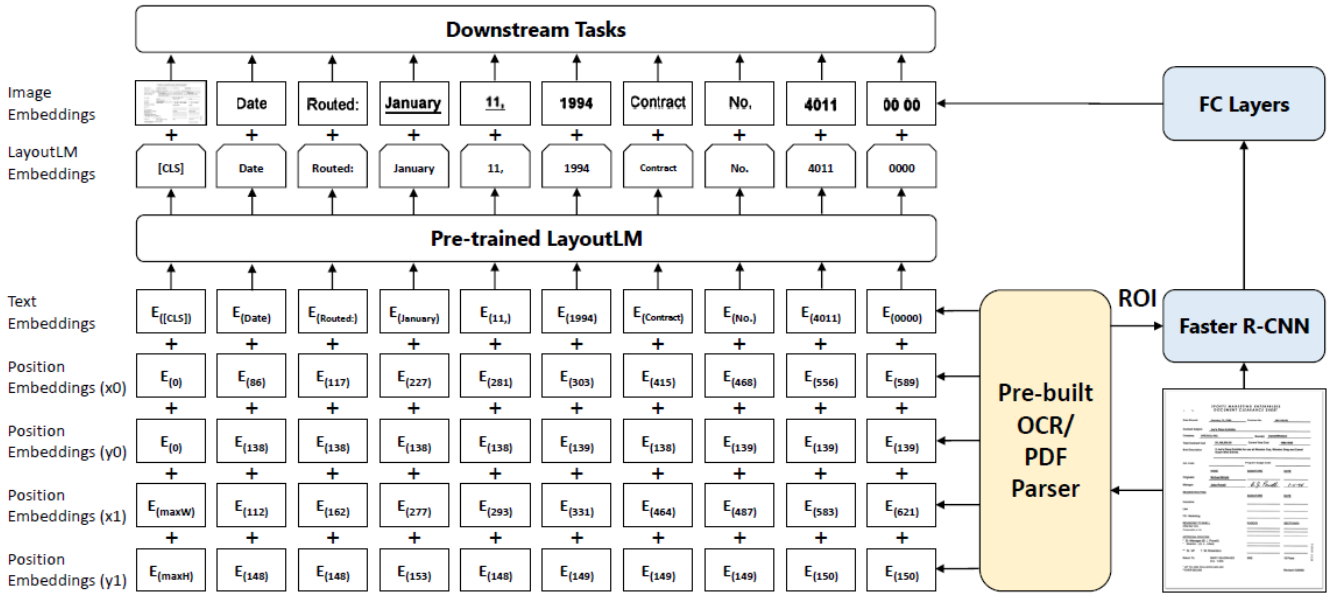


Figure 2.12: The overall framework of LayoutLM [72], where 2-D layout and image embeddings are integrated into the original BERT architecture. The LayoutLM embeddings and image embeddings from Faster R-CNN work together for downstream tasks.
Source: reproduced from Xu et al. (2020) [72] (2020)

Xu et al. (2020) [72] add the 2-D position embedding layers with four embedding representations (x0, y0, x1, y1), where (x0, y0) corresponds to the position of the upper left in the bounding box, and (x1, y1) represents the position of the lower right. On the other hand, to align the image feature of a document with the text, they add an image embedding layer to represent image features in language representation. With the bounding box of each word from OCR results, they split the image into several pieces and one-to-one correspondence with the words. Embeddings of the image region are generated using the Faster R-CNN model to be added to the text embedding. However, the image processing step with the Fast R-CNN model has a high computational cost and typically require an expensive GPU to be executed. This motivated our design, which incorporates

tokens from text regions coming directly from the formatted document or the OCR and avoids further image processing.

In addition, LayoutLM adopts a multi-task learning objective, including a Masked Visual-Language Model (MVLM) loss and a Multi-label Document Classification (MDC) loss, which are discussed in subsection 2.5.1.

2.5.1 Pretraining Objectives Downstream Tasks

Inspired by BERT, many pre-trained language models have emerged to understand visually rich documents. These models use pre-training jointly with different modalities such as text, layout and visual information in a single framework. Pre-training objectives have been used in pre-training and fine-tuning language models.

Masked Language Model (MLM) was proposed firstly in BERT [19] architecture to learn bidirectional representations by predicting the original vocabulary id of a randomly masked word token based on its context. The MLM objective allows the representation to fuse the left and the right context, which allows pre-training for a deep bidirectional Transformer. Some percentage of the input tokens at random are masked to train a deep bidirectional representation. In this case, the final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary, as in a standard Language Model. BERT randomly masks 15% of all WordPiece tokens with a special token [MASK] in each sequence and only predicts the masked words rather than reconstructing the entire input. The training data generator randomly chooses 15% of the token positions for prediction. Masked tokens are replaced with a special [MASK] token 80% of the time, a random word 10%, and an unaltered 10%. Figure 2.13 (a) shows that MLM is a fill-in-the-blank task; words are masked from the input, and the transformer network must predict the missing words. The BERT model is then trained to reconstruct these masked tokens given the observed set.

Next Sentence Prediction (NSP) enables the model to capture sentence-to-sentence relationships, which are crucial in many language modelling tasks such as Question Answering and Natural Language Inference. Given a pair of sentences, the model predicts a binary label, i.e., whether the pair is valid from the original document or not, see Fig. 2.13 (b). Specifically, when choosing the sentences A and B for each pre-training example, 50% of the time, B is the actual next sentence that follows A, and 50% of the time, it is a random sentence from the corpus.

Figure 2.13: BERT [19] (a) Masked Language Model and (b) Next Sentence Prediction objectives. BERT operates over sequences of discrete tokens comprised of vocabulary words and a small set of special tokens: [CLS], [MASK] and [SEP]. The first token of every sequence is always a special classification token [CLS]. The special token [MASK] masks a word that will be predicted. [SEP] is a special separator token.

Masked Visual-Language Model (MVLM) was proposed to learn language representation with the clues of 2-D position embeddings and text embeddings. The model randomly masks some input tokens during pre-training but keeps the 2-D position embeddings and other text embeddings. The model is then trained to predict the masked tokens given the context. In this way, the LayoutLM [72] model not only understands the language contexts but also utilizes the corresponding 2-D position information, thereby bridging the gap between the visual and language modalities.

Multi-label Document Classification (MDC) refers to assigning multiple relevant labels to each input document, while the entire label set might be extremely large. LayoutLM [72] uses MDC loss during the pretraining phase. Given a set of scanned documents, the model uses the document tags to supervise the pretraining process. The model can cluster the knowledge from different domains and generate better document-level representation [14].

Text-Image Alignment (TIA) was proposed in LayoutLMv2 [73] as a fine-grained cross-modality alignment task to help the model learn the spatial location correspondence between the image and coordinates of the bounding boxes. The covering operation ran-

domly selects some tokens lines and their image regions and covers them in the document image. During pretraining, a classification layer is built above the encoder outputs. This layer predicts a label for each text token depending on whether it is covered, i.e., [Covered] or [Not Covered], and computes the binary cross-entropy loss.

Text-Image Matching (TIM) task is applied to help the model learn image-text alignment, i.e., to the model learn the correspondence between document image and textual content. LayoutLMv2 [73] feeds the output representation at tag [CLS] into a classifier to predict whether the image and text are from the same document page. Regular inputs are positive samples. Moreover, in negative samples, an image is either replaced by a page image from another document or dropped. The TIM target labels are set to tag [Covered] in negative samples.

2.6 Related Works

Various document image classification and page stream segmentation approaches have been proposed over the past few years. Gordo et al. (2013) [23] focused on segmenting a continuous page stream into multi-page documents and classifying the resulting documents. This section provides an overview of some important works that have been reported about document classification methods that take textual and visual information as input.

2.6.1 Page Stream Segmentation

Agin et al. (2015) [1] presented a method for segmentation of document page flow applied to heterogeneous real bank documents. The approach is based on the content of images, and it also incorporates font-based features inside the documents. The authors involved a bag of visual words (BoVW) model on the designed image-based feature descriptors using three different classifiers: Support Vector Machines (SVM), Random Decision Forest (RDF) and Multilayer Perceptron (MLP). In addition, they combined the consecutive pages of a document into a single feature vector representing the transition between these pages. One of the two classes represented the transitions: the continuity of the same document or the beginning of a new document. They evaluated the method by comparing the performance of the three classifiers.

In Gallo et al. (2016) [21], page stream segmentation PSS is performed on top of the results from a document image classification DIC process. They proposed a supervised approach for page stream segmentation and document image classification using features learned by Convolutional Neural Networks (CNN). In the final step of the approach, the

CNN predictions are corrected using an additional deep model that analyzes the stream of classified image documents. The experiments were performed on two datasets that they built using real documents and evaluated using Accuracy and Kappa metrics.

Wiedemann and Heyer (2021) [69] developed an approach based on convolutional neural networks (CNN) combining image and text features to perform (PSS) as a binary classification task on single pages from a data stream. They first create two separate convolutional neural networks for the binary classification of pages classified into either continuity of the same document (SD) or the beginning of a new document (ND), one based on text data and another based on image scans. In a third step, they combine the learned parameters from the two final hidden layers of both CNN to an input vector of features for a multi-layer perceptron. This MLP delivers a third and final classification result based on both feature types. The authors used the VGG16 architecture for images and a pre-trained FastText model for word embeddings. They evaluated the proposed model on the Tobacco800 datasets and a sample of the data from the German archive of their project context using Accuracy and Kappa metrics.

The work of Braz et al. (2021) [10] was built upon the proposal of Wiedemann and Heyer (2021) [69] by improving the network architecture using EfficientNet pre-trained CNN architecture, replacing the earlier proposed VGG16 Network. However, they used techniques focused only on the images on the pages. They proposed a novel approach to the PSS problem, using four training classes, which can be reduced to the usual two classes of the PSS problem in the literature. They used two datasets to validate the proposed model for the PSS problem: Tobacco800 and AI.Lab.Splitter [10], a novel dataset composed of Brazilian court documents. Performance was evaluated using Accuracy statistics, F1 score, and Kappa and compared with the model by Wiedemann and Heyer (2021) [69] obtained better results.

A multimodal binary classification approach based on transfer learning techniques using BERT [19] to solve the PSS problem was proposed by Guha et al. (2022) [24]. The authors considered the model proposed by Wiedemann and Heyer (2021) [69] as the baseline. They simultaneously used the VGG16 architecture as an image feature extractor and the $BERT_{BASE}$ pre-trained model for text features. Both features are finally fused and passed through a fully connected layer of Multi-Layer Perceptron (MLP) to obtain the binary classification of the pages as the First Page (FP) and the Other Page (OP). The model was evaluated using real-time document image streams from the archive of production business processes obtained from a reputed Title Insurance (TI) company, and the metric used was the F1 score. They compared the results with the work of Wiedemann and Heyer (2021) [69] and Braz et al. (2021) [10] but used a different dataset.

Table 2.1 summarizes the work proposed for Page Stream Segmentation. Most models

used a Convolutional Neural Network as a backbone, and the evaluation metrics were Accuracy and F1 score. Each model combines at least two modalities (textual, visual, and layout) for downstream tasks, except for the models proposed by Gallo et al. (2016) [21] and Braz et al. (2021) [10] that used only visual features for PSS. Only two models [69, 10] used the same Tobacco800 dataset for PSS, and both results were compared and presented in the work by Braz et al. (2021) [10].

Table 2.1: Comparison between proposed models for the page stream segmentation task w.r.t. modality, backbone, datasets, accuracy and F1-score evaluation metrics. T, L, and I denote textual, layout, and image features.

Model	**Modality**	**Backbone**	**Dataset**	**Accuracy**	**F1**
Agin et al. (2015) [1]	T + I	BoVW + SVM RDF and MLP	Banking Dataset Private DS	87.24%	88.88%
Gallo et al. (2016) [21]	I only	CNN + DNN	Public Dataset	97.45%	-
Wiedemann and Heyer (2021) [69]	T + I	VGG16-CNN MLP	**Tobacco800** German dataset	91.10% 93.00%	90.40% -
Braz et al. (2021) [10]	I only	CNN EfficientNet	**Tobacco800** AI.Lab.Splitter	92.00% 95.20%	91.90% 95.30%
Guha et al. (2022) [24]	T + I	VGG16 + BERT	real-time document Title Insurance	98.56%	97.37%

2.6.2 Document Image Classification

Asim et al. (2019) [4] present a Naïve Deep Learning approach for the task of text document image classification, which utilizes both structural similarity and content of text document images. A filter-based feature-ranking algorithm was utilized to alleviate the dependency of the textual stream on the performance of underlying OCR. This algorithm ranks the features of each class based on their ability to discriminate document images and selects a set of top 'K' features retained for further processing. Simultaneously, the visual stream uses deep CNN models to extract structural features of document images, and the average ensembling method concatenates textual and visual streams. To assess the performance of two streams (text and visual) document classification approaches, they used publicly available Tobacco-3482 and RVL-CDIP datasets. Finally, they used the accuracy metric to compare results with the state-of-the-art and obtained better results.

A multimodal neural network is designed by Audebert et al. (2020) [5], which can learn from word embeddings and images. FastText word embedding and MobileNetv2 image embedding were introduced to perform joint visual and textual feature extraction. First, Tesseract OCR was used to extract the text from the image to perform a fine-

grained classification using visual and textual features. Then, they computed character-based word embeddings using FastText on the noisy Tesseract output and generated a document embedding representing our text features. The visual features are learned using MobileNetv2, a standard CNN from state of the art. Finally, they introduced an end-to-end learnable multimodal deep network that jointly learns text and image features and performs the final classification based on a fused heterogeneous representation of the document. The approach was evaluated using the accuracy metric on the Tobacco3482 and RVL-CDIP datasets for the document image classification problem. However, they do not compare the results with the state-of-the-art.

Bakkali et al. (2020) [7] presented a hybrid cross-modal feature learning approach that combines image features and text embedding to classify document images. They adopt a late fusion scheme methodology. The built-in network is based on the performance of lightweight, heavyweight architectures used in experiments for image stream and static, dynamic word embeddings used to perform text classification. NASNet-Large model and BERT model pre-trained were used on ImageNet to extract the image and textual features, respectively, for document classification on the Tobacco-3482 dataset. Every single modality was trained independently from one another, but merging both streams boosted the performance of the two fusion modalities and improved classification accuracy. They compared results with the state-of-the-art and obtained better results.

StructuralLM [36] is a self-supervised pretraining method designed to better model the interactions of cells and layout information in scanned document images. Unlike LayoutLM [72], StructuralLM is a structural pretraining approach that jointly exploits cell and layout information from scanned documents. It uses cell-level 2D-position embeddings to model the layout information of cells rather than word-level 2D-position embeddings. To represent the spatial position of cells in scanned document images, we consider a document page as a coordinate system with the top-left origin. In this setting, the cell (bounding box) can be precisely defined by (x0, y0, x1, y1), where (x0, y0) corresponds to the top-left position, and (x1, y1) represents the bottom-right position. It adopts two self-supervised tasks during the pretraining stage: MVLM [72] and Cell Position Classification (CPC) task. The authors conduct experiments on publicly available benchmark datasets for three downstream tasks. These three tasks are the form comprehension task, the document visual question answer task, and the document image classification task. They used the RVL-CDIP [26] dataset for the document image classification task. Furthermore, they compared the results with the LayoutLM model and achieved 1.65% better.

The approach proposed by Zingaro et al. (2021) [77] exploits the side-tuning framework for multimodal document classification. They combined incremental learning and multimodal features training to learn from both representations, visual and textual, jointly.

The base model consists of a CNN for image classification, pre-trained on the ImageNet dataset. The side component presents two different networks: the first one is identical to the base model but with unlocked weights to allow updates during training. In contrast, the second network is a CNN for text classification. To assess the proposed model's validity, they evaluated the approach on Tobacco-3482 and RVL-CDIP datasets and two deep-learning architectures, MobileNetV2 and ResNet50, with parameters 12M and 57M, respectively. The metric used to evaluate the performance of the model on the test set was the Accuracy metric. Furthermore, they compared the results with the work of Audebert et al. (2020) [5] and obtained better results.

DocFormer [3] adopts a discrete multi-modal structure self-attention with shared spatial embeddings in an encoder-only transformer architecture. It also has a CNN backbone for visual feature extraction and encoding image information to obtain higher resolution image features and simultaneously encodes text information into text embeddings. All components are trained end-to-end. DocFormer enforces deep multi-modal interaction in transformer layers using novel multi-modal self-attention. They describe three modality features (visual, language, and spatial) prepared before feeding them into transformer layers. The position information is added to the image and text information separately and passed to the Transformer layer separately. In addition, DocFormer proposes three pretraining tasks: multi-modal masked language modeling (MM-MLM), a modification of the original MLM pre-text task introduced in BERT; learning-to-reconstruct (LTR), is an image reconstruction task, and the text describes image (TDI) to teach the network if a given piece of text describes a document image. Each word k in the text also gets bounding box coordinates $b_k = (x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$. 2D spatial coordinates bk provide additional context to the model about the location of a word in the entire document. They reported performance on the test sample using the overall classification accuracy metric.

The LayoutLMv2 [73] improved over LayoutLM [72] by changing how visual features are input to the model - treating them as separate tokens instead of adding visual features to the corresponding text tokens. Further, additional pre-training tasks were explored to use unlabeled document data. Xu et al. (2021) [73] proposed the spatial-aware self-attention mechanism for the LayoutLMv2, which involves a 2-D relative position representation for token pairs. Different from the absolute 2-D position embeddings, the relative position embeddings explicitly provide a broader view of contextual spatial modeling. The multi-modal Transformer accepts inputs of three modalities: text, image, and layout. The input of each modality is converted to an embedding sequence and fused by the encoder. The model establishes deep interactions within and between modalities by leveraging the powerful Transformer layers. They adopted three self-supervised tasks

simultaneously during the pre-training stage: Masked Visual-Language Model (MVLM), Text-Image Alignment (TIA) and Text-Image Matching (TIM). They evaluated the classification task on RVL-CDIP dataset and compared it with LayoutLM [73].

Table 2.2 summarizes the works proposed for Document Image Classification in this section. The most recent works presented are pre-training multimodal models and used transformer architecture based on BERT as the backbone. The most recent works presented merge visual and textual features using 2-coordinates to represent the location of the text in the document. Instead, our approach uses special semantic tokens to represent the position of the text block and inserts it into the text embedding to be processed by a language model.

Table 2.2: Comparison between proposed models for the document image classification task w.r.t. modality, backbone, datasets, accuracy and F1-score evaluation metrics. T, L, and I denote textual, layout, and image features.

Model	**Modality**	**Backbone**	**Dataset**	**Accuracy**	**F1**
Asim et al. (2019) [4]	T + I	InceptionV3	Tobacco-3482	95.80%	-
		Multi-channel CNN	RVL-CDIP	96.40%	-
Audebert et al. (2020) [5]	T + L	Multimodal Neural Network	Tobacco-3482	92.10%	91.00%
			RVL-CDIP	90.60%	-
Bakkali et al. (2020) [7]	T + I	Cross-modal BERT	Tobacco-3482	99.71%	
			RVL-CDIP	97.05%	97.00%
LayoutLM (2020) [72]	T + L	Transformer BERT	RVL-CDIP	94.42%	-
StructuralLM (2021) [36]	T + L	BERT	RVL-CDIP	96.08%	-
Zingaro et al. (2021) [77]	T + I	DCNN	Tobacco-3284	90.50%	-
			RVL-CDIP	93.60%	-
DocFormer (2021) [3]	T + L + I	Multimodal Transformer	RVL-CDIP	96.17%	-
LayoutLMv2 (2021)[73]	T + L + I	Transformer	RVL-CDIP	95.25%	96.01%

2.7 Summary

This chapter presented concepts about layout analysis from scanning a document and its processes for extracting information. Layout analysis is often a performance-limiting step for optical character recognition (OCR) since errors in this phase propagate to all subsequent phases of the system. Geometric layout analysis of a document image typically involves different processes: binarization, noise removal, rectification, page segmentation, zone classification, and reading order. Some algorithms skip one or more of these processes

or apply their hybrid. However, most layout analysis systems use these processes in some form.

Paper documents, such as books, handwritten, magazines, and newspapers, have traditionally been used as the primary source of acquisition, dissemination, and preservation of knowledge. An electronic document represents a document using data structures that computers can understand. An electronic document can be converted into a paper document by printing device. Converting a paper document to an electronic format, on the other hand, requires a way to transform the document into data structures that computers can understand. Removing text bounding boxes and generating textual embedding is crucial in text extraction and representation tasks. By leveraging advanced image processing and natural language processing techniques, researchers and practitioners can unlock the potential of textual data for various applications in fields such as image understanding, document analysis, and intelligent information retrieval.

Several studies have addressed document analysis using visual and textual resource extraction for downstream tasks. Approaches have evolved from early-stage heuristic rules to statistical machine learning. Then, deep learning methods with greater attention to the pre-trained language models based on BERT [19] have become a trend in Document AI development. Moreover, some models have designed richer pretraining objective tasks for different modalities, such as the MLM objective task introduced by LayoutLM [72]. A major drawback of such pre-trained models based on the Transformer architecture [65] is that they require a high computational cost. Unlike these previous methods, our approach aims to improve the performance of language models by combining texts and their spatial information with a low computational cost. Specifically, we propose a spatial layout encoding method combining textual and spatial information from text blocks.

Chapter 3

Methodology

In this Chapter, we present LayoutQT - Layout Quadrant Tags, a lightweight preprocessing method focusing on combinations of texts and their spatial information without relying on visual features or activations from the visual modalities. Specifically, we propose a new set of tokens that encode spatial region language models and show that they improve results in downstream tasks with low computational cost.

3.1 Layout Quadrant Tags (LayoutQT)

Our algorithm is based on a bottom-up approach, which defines primitive components to start the clustering process. It starts with the bounding box of words as a primitive component of the page. The word grouping process identifies a group of nearest neighbours of each bounding box to form lines and blocks of text until the page ends. Furthermore, each document page is divided into rectangular regions with the same *height* and *width* dimensions. Each quadrant has layout location information that is represented by spatial tokens.

Spatial tokens are added at the beginning and end of each line when indicating the quantized coordinates of the bounding box that the line belongs to. The text group beginning tag considers the distances from the top left corner of the bounding box to the image's left edge and top edge. Likewise, the end tag considers the distance between the bottom right corner of the bounding box and the image's bottom edge and right edge. Table 3.1 presents spatial tokens and their descriptions used in our LayoutQT model. For example, the beginning of a text block is marked with $xxQr_i\_c_j$ *xxbob* to indicate the position (quadrant) of the beginning of the text block. The centered parts of the text are also marked with spatial tokens *xxbcet* and *xxecet*.

LayoutQT's Algorithm 1 takes single-page or multi-page documents as input and generates tokenized text t with layout information. The algorithm scans the page from top to

Table 3.1: Proposed spatial tokens

Special Token	Descriptions
$xxPn_k$	document page numbering tag, where n_k is the page index
$xxbob$	markup tag from the beginning of the text block
$xxeob$	markup tag from the end of the text block
$xxbcet$	tag that marks the beginning of the center of the block
$xxecet$	tag that marks the end of the center of the block
$xxQr_i\_c_j$	quadrant numbering tag, where r_i and c_j are the indexes of the quadrant row and column, respectively

bottom and left to right to find the boundaries of text groups and identify the group's top left corner. Initially, it adds a spatial token to the text to indicate page. It then uses an OCR engine [59] to generate word bounding boxes. For that, we used the combination of heuristics included in the Tesseract package [59]. However, more modern techniques can be applied using an object detection neural network trained to detect the bounding boxes of textual elements. An example of such networks is the series of YOLO networks, which was originally proposed for object detection benchmarks [54] then it has been adapted for all sorts of objects, including human body parts [46] and even tomatoes [33]. After getting textual bounding boxes, our algorithm exploits their coordinates by injecting that information through the spatial tokens. It sorts the groups in the same column on the page to check which groups are centralized and adds the tokens. Moreover, it ends by adding the end-of-group spatial token. The text extraction with spatial tags is saved to a text file.

Figure 3.1 illustrates LayoutQT tag computation on a single document page. The document input image is divided into quadrants and text groups. Each row is numbered from left to right, and each column is numbered from top to bottom, so the tags of the first and last quadrants are, respectively, $xxQ00\_00$ and $xxQn-1\_m-1$, where n and m are the total of vertical and horizontal quadrants. Inspired by the tokenization of Fastai [28], which adds spatial tokens at the beginning and end of the sentence, LayoutQT adds tokens with information about the bounding box position. All spatial tokens start with the character xx, which is not a common English word prefix. They are added using rules for the model to recognize the important parts of a text. The image of the text file tokenized by our model is on the right side of Fig. 3.1.

Following the flow of Figure 3.2, we start by providing document images as input to our preprocessing step, which virtually maps page space into equally spaced quadrants. Next, we map each text block's start and end position into the related quadrant and inject spatial tokens to mark each text box's start and end position. Then the text of each bounding box is extracted along with the spatial tokens considering their position on

Algorithm 1 LayoutQT Algorithm
Input: multi page document
Output: tokenized text t

1: $t =$ " " (empty string)
2: **for** $page = 0, \ldots, N-1$ **do**
3: $t+ = xxPn_k$ (add page token where $+ =$ means insert symbol in string t)
4: group each word by bounding boxes into lines and blocks
5: group the blocks into coherent page columns
6: **for each** group **do**
7: $t+ = xxQr_i\_c_j$ $xxbob$ (quadrant coordinate of group top left corner)
8: **for each** text line in this group **do**
9: check line centralization w.r.t. its page column center position
10: **if** the line is centralized **then**
11: $t+ = xxbcet$ (centre tag)
12: **end if**
13: $t+ =$ textual contents of the line
14: **if** the line is centralized **then**
15: $t+ = xxecet$ (centre tag)
16: **end if**
17: **end for**
18: $t+ = xxeob$ $xxQr_i\_c_j$ (quadrant coordinate of group bottom right corner)
19: **end for**
20: **end for**
21: return t

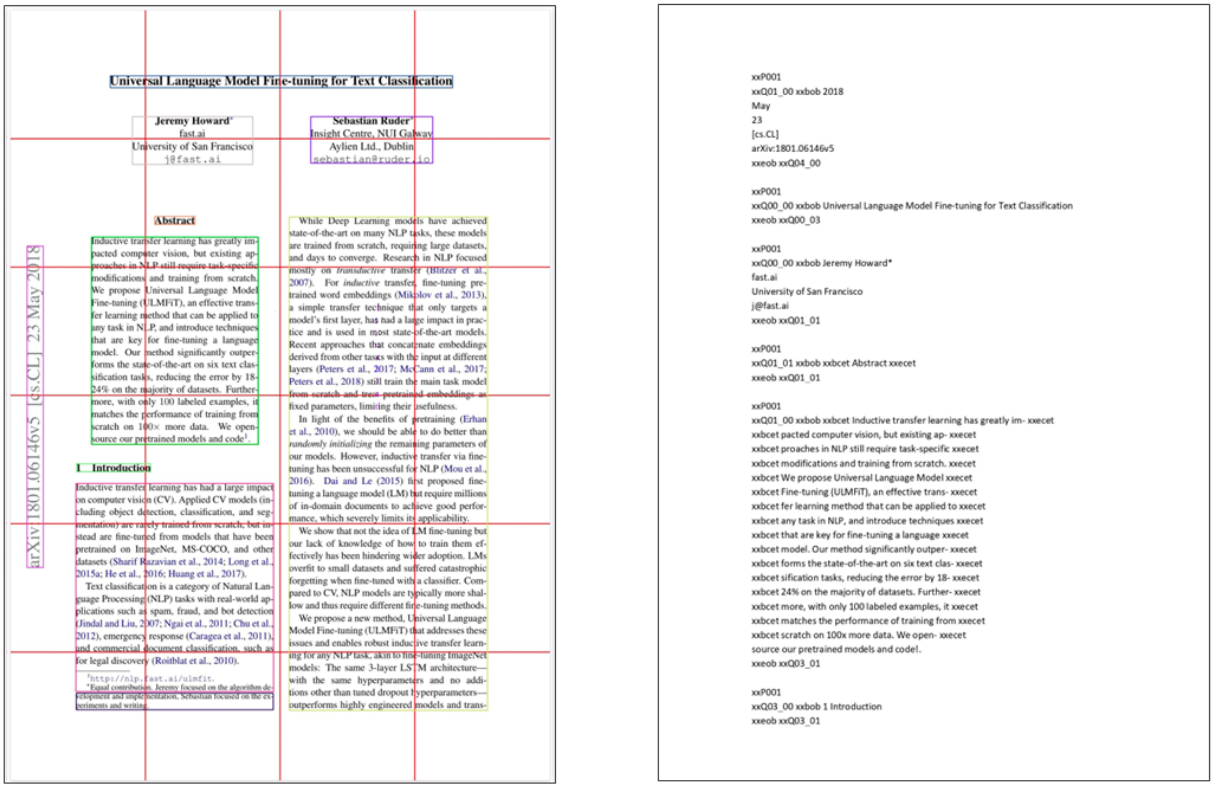


Figure 3.1: Illustration of Layout Quadrant Tags, LayoutQT. The rectangles represent the bounding boxes of text. On the left side, an input document is divided into quadrants and receives spatial tokens $xxQr_i\_c_j$ according to row i and column j positions. On the right side is the text extracted by the OCR system, with the tags indicating the position (quadrant) of each text block's beginning and end.

the document page. The resulting data then goes through a language modelling pipeline downstream tasks.

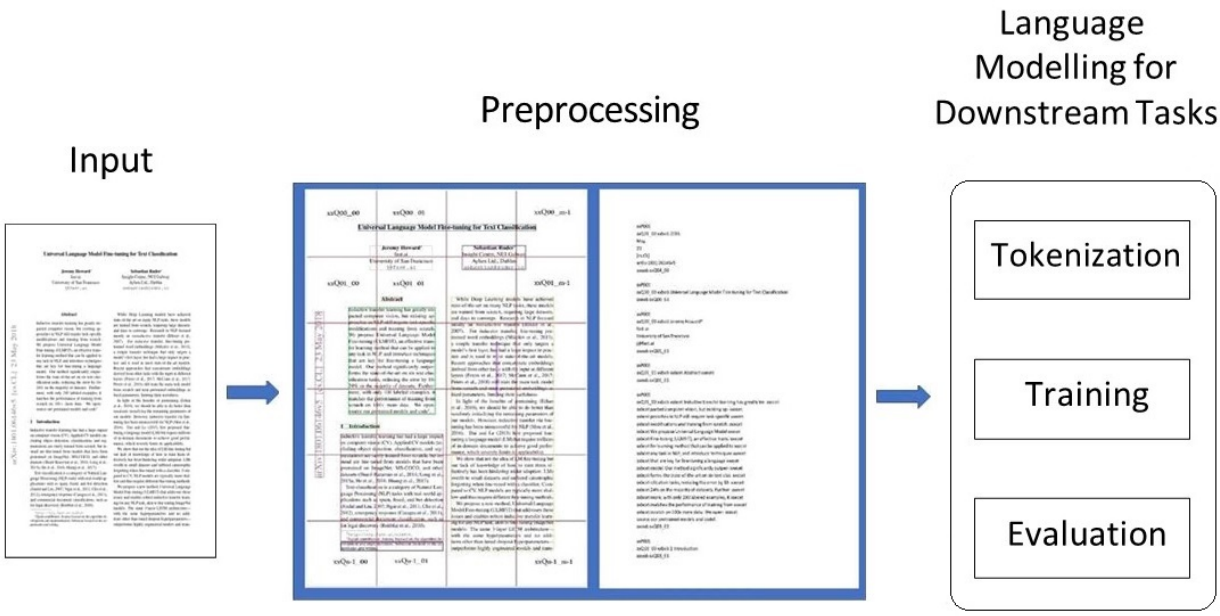


Figure 3.2: Illustration of the LayoutQT pipeline, going from input textual images to an NLP system. Our method computes layout tags as part of an OCR pipeline which is injected as special tokens in the text.

3.2 Baseline

As a baseline, we use an architecture similar to our approach. However, without our pre-processing, the document images fed an OCR engine to extract the text without the spatial tokens. Subsequently, the extracted texts were tokenized, trained, tested, and evaluated using the same language modelling for the downstream tasks, as shown in Fig. 3.3.

3.3 Metrics

The performance evaluation metrics used are Accuracy, Precision (P), Recall (R), and the average F-Score, which measures both. The Accuracy is defined as:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FN+TN+FP}}, \tag{3.1}$$

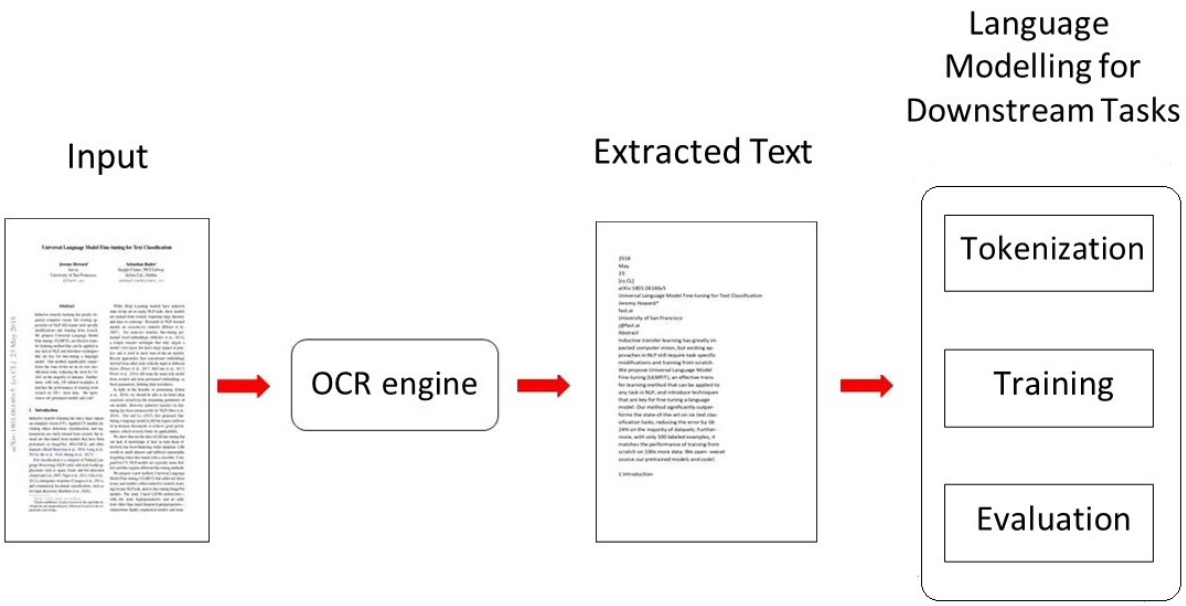


Figure 3.3: Experiment flow diagram showing the baseline without using the proposed method, to be compared with our pipeline, shown in Figure 3.2

where TP (true positive) is the number of documents correctly assigned to a category C they belong to, FP (false positive) is the number of documents incorrectly assigned to the same category C they do not belong to, TN (true negative) is the number of documents correctly classified to the other categories to which they belong other than category C and finally, FN (false negative) is the number of documents originally in category C but misclassified into other categories.

The confusion matrix is a table with two rows and two columns that reports the number of TP, FN, FP, and TN. This allows more detailed analysis than simply observing the proportion of correct classifications (accuracy). Accuracy will yield misleading results if the data set is unbalanced; that is when the numbers of observations in different classes vary greatly.

The F1-score takes into account the precision and recall rate. So, in this thesis, the F1-score is chosen to measure the algorithm's performance in classification tasks.

$$\mathrm{F1} = 2 * \frac{\mathrm{Precision*Recall}}{\mathrm{Precision+Recall}}, \tag{3.2}$$

whereas precision and recall are defined as follows:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}, \tag{3.3}$$

and

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}, \tag{3.4}$$

We evaluated the models using the parameters with the best average F1 score calculated on the validation set. After each epoch, we only save the model parameters if the validation performance is the highest up to that point.

3.4 Evaluation

To train and evaluate the document page stream segmentation, we used the Tobacco800 dataset in three network architectures: a Long Short-Term Memory (LSTM) [61], Universal Language Model Fine-Tuning (ULMFiT) [29] with ASGD Weight-Dropped LSTM (AWD-LSTM) [47] and BERT [19] for ranking the pages as *first_page* or *next_page* class on the same dataset.

For document classification with the RVL-CDIP dataset, inspired by Howard and Ruder (2018) [29], we used ULMFiT with AWD-LSTM for training, testing and evaluation. Each evaluation dataset was split into training, validation and test subsets. We minimized the loss function using the training set and assessed the model from each epoch on the validation set. We saved the model's weights of the lowest loss in the validation set iteration and evaluated the model with these weights in the test set after the whole training. We tried the same strategy with the BERT [19] model to classify the RVL-CDIP dataset.

We also performed experiments with the VICTOR dataset [17] to classify documents from the legal domain and in Portuguese, using ULMFiT with AWD-LSTM for training, testing, and evaluation. We performed preliminary experiments with the complete dataset using AWD-LSTM. Next, we split the VICTOR dataset into two sampling strategies, one containing only the first page of the documents and the other with the rest of the dataset inspired by [17]. Finally, we performed some experiments with the first-page sample of the VICTOR dataset using BERT, with the baseline and LayoutQT.

To evaluate, we compared the execution of the classifier using LayoutQT method generating the quadrant tags and without the preprocessing with Tobacco800, RVL-CDIP and VICTOR datasets. To compare the results of our approach with the baseline, we used accuracy and F1-score metrics. The loss function used by default is the cross-entropy loss, as we have a classification problem (the different categories are the words in our vocabulary).

3.5 Summary

In this chapter, we described the entire methodology of our LayoutQT approach - Layout Quadrant Tags, a lightweight pre-processing method that combines textual and layout information. Specifically, we presented a new set of tokens that encode language models of spatial regions. LayoutQT divides a document into quadrants. Each quadrant is identified by a positional token that is later inserted into the embedding of text blocks. Next, we define the baseline architecture. Finally, we present the statistical metrics for evaluating the model and the methodology for evaluating our approach.

Chapter 4

Experiments

This chapter presents our experiments and deals with the datasets chosen to evaluate our proposal. Publicly accessible document image collection with realistic scope and complexity is important to the document image analysis and search community.

We apply our model to two subsequent tasks, one for page flow segmentation and another for document type classification. We use the Tobacco800 dataset for the page stream segmentation task and the RVL-CDIP and VICTOR datasets for document type classification. For Tobacco800, we follow the training, validation, and testing split defined by [10], while we use the standard split for RVL-CDIP, and for the VICTOR dataset, we follow the split defined by [17]. We performed classification experiments with and without using our model to compare the results.

4.1 Experiment Setting

This section describes the implementation details used for the proposed approach. We used our preprocessing method, which starts with an OCR engine to generate blocks of text (bounding boxes) and delimit textual elements for each image in the document. Then, a parameterised set of quadrants is used to define quantised coordinates to compute our tags. For this feature extraction processing and insertion of positional tags into text blocks,We used an Intel Core i5-10210U CPU laptop with 20 GB of memory and 256 GB of PCIe SSD storage, which proves a low computational cost.

Initially, we performed two experiments with the Tobacco800 dataset for binary classification of document pages, one with LayoutQT using 24 quadrants (6 horizontal blocks x 4 vertical blocks) and the other experiment with the baseline. Later, we vary the number of quadrants by modifying the number of rows and columns. Our first model has an LSTM backbone (composed of 256 nodes fully connected with activation “ReLU” and a dropout of 0.3). Furthermore, we use binary cross-entropy as a loss function with softmax

activation and Adam as an optimizer. The model was trained for 100 epochs with a batch size of 128.

We also performed the experiments with an AWD-LSTM language model [47] trained with backpropagation through time with a batch size of 128, an embedding size of 400, 3 layers, 1150 hidden activations per layer, giving a total of 24 million parameters for the PSS job using the baseline and LayoutQT in the Tobacco800 dataset. We then repeated this experiment using BERT, which contains 12 layers in the encoder stack, 768 hidden units, 12 attention heads, totalling 110 million parameters. The model was trained using one cycle learning rate policy [56] for 100 epochs with a batch size of 128 documents and a sequence length 72 using NVIDIA Tesla V100 32GB GPU.

Finally, for the document image classification task, we performed similar experiments using as backbone AWD-LSTM and BERT with the same configurations of the previous experiments on the datasets RVL-CDIP and VICTOR. However, the experiments with the dataset VICTOR with AWD-LSTM were divided into three stages. First, we perform classification experiments with the baseline and LayoutQT on the full dataset. Next, we divided the dataset into two sets of samples, one containing only the first page of the documents and the other the not-first page, to see what is the relevance of the first page of the documents vs other pages. Then, we classified both samples to compare with the work by Luz et al. (2022) [17]. In the next section we present the publicly available benchmarks used to evaluate our method.

4.2 Datasets

The Truth Tobacco Industry Documents, formerly known as Legacy Tobacco Documents Library (LTDL), was created and hosted by the University of California San Francisco (UCSF). It was built to provide permanent access to the tobacco industry's internal corporate documents produced during litigation between the US States, the seven major tobacco industry organizations, and other sources. Complex document image processing (CDIP) test collection was constructed by the Illinois Institute of Technology (IIT), assembled from 42 million documents (in 7 million multi-page TIFF images) released by tobacco companies under the Master Settlement Agreement from the LTDL in 2006 [35]. The documents in LTDL range from the late 19th century to the present. The bulk of the collections dated 1950 through 2003.

At first, we used three publicly available datasets containing business documents in English, namely Tobacco800 [76, 75], RVL-CDIP [26], and Tobacco-3482 [32] datasets. These datasets are subsets of the CDIP dataset found in the literature for various downstream tasks, such as document image classification, PSS, and offline signature verification, among

others. Next, we briefly describe VICTOR [43, 42], a dataset of court documents in Portuguese proposed for document classification. The properties of all datasets are described below.

4.2.1 Tobacco800

Tobacco800 [35] is a public subset of the CDIP used for several tasks: offline signature verification, detection, extraction of document images, etc. Recently, it has been used for page stream segmentation. The Tobacco800 dataset has only 1,290 document images of many types, such as letters, fax, memos, etc., that were collected and scanned using various equipment over time. Since the Tobacco800 dataset sample file name comes with the page, like the ones shown in Figure 4.1 when merged, it mimics a stream of pages from multiple documents ideal for splitting by the PSS model. In addition, Tobacco800 [35] was manually annotated, targeting document signature and logos segmentation.

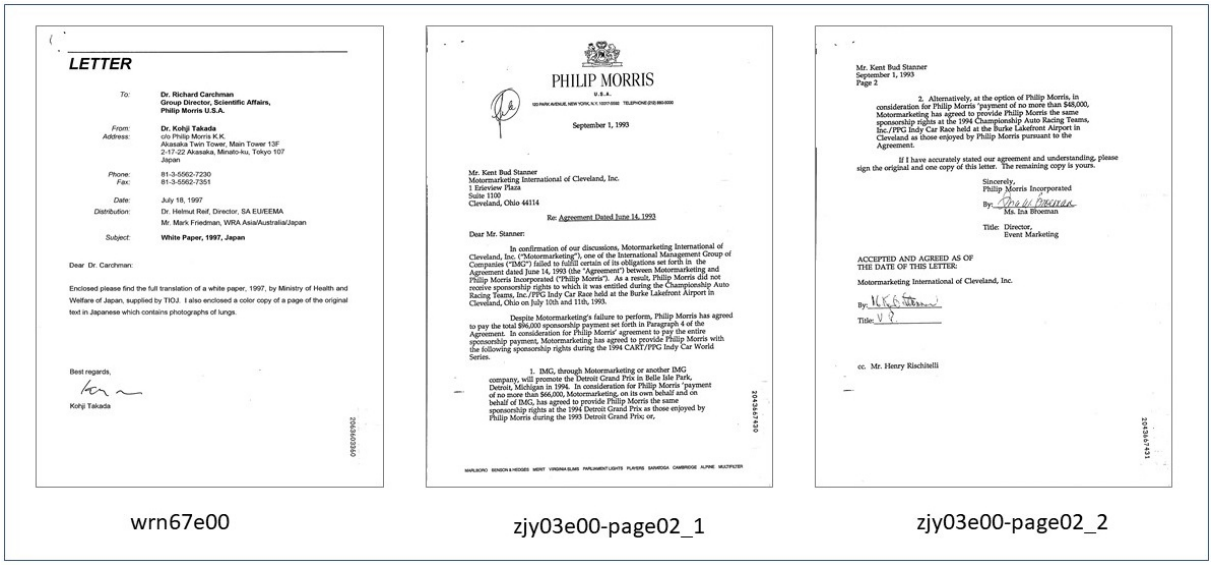


Figure 4.1: Image documents sample of Tobacco800 dataset. In left-to-right order, the first image is a single-page document, and the next two images are pages of the same document and are in ascending page order.

A significant percentage of Tobacco800 are consecutively numbered multi-page business documents, making it a valuable testbed for various content-based document image retrieval approaches. Resolutions of documents in Tobacco800 [35] vary significantly from 150 to 300 DPI, and the dimensions of images range from 1200 by 1600 to 2500 by 3200 pixels. Table 4.1 presents the division of the dataset into training, validation and test sets for each class.

The classification problem here involves two classes: whether the transition between consecutive pages indicates the continuity of the same document or the beginning of a new

Table 4.1: Class counts by division into training, testing and validation show the number of document pages belonging to the FirstPage and NextPage classes.

Label	Class	Train	Validation	Test
0	FirstPage	586	150	118
1	NextPage	445	109	88

document. Document images are classified in FirstPage or NextPage, in which FirstPage represents a document's first page, and NextPage class is formed by all document pages except the first page. The Tobacco800 Dataset was used by Wiedemann and Heyer (2021) [69] to evaluate a binary classification architecture proposed by them. This work developed a hybrid approach combining image and text for page stream segmentation (PSS). Braz et al. (2021) [10] also used this dataset to evaluate a series of models for the PSS problem. They defined a novel approach to the PSS problem using four training classes by dealing with pairs of pages, which can be reduced to the usual two classes of the PSS problem in the literature.

4.2.2 RVL-CDIP

RVL-CDIP, also known as BigTobacco, stands for Ryerson Vision Lab Complex Document Information Processing. The file structure of this dataset is the same as the IIT collection so that you can query this dataset for OCR and additional metadata. RVL-CDIP is a huge dataset with 400,000 grayscale images in 16 classes, with 25,000 images per class, which was introduced by Harley et al. (2015) [26]. There are 320,000 training images, 40,000 validation images, and 40,000 test images. The images are resized, so their largest dimension is not greater than 1,000 pixels.

The 16 classes include letter, form, email, handwritten, advertisement, scientific report, scientific publication, specification, file folder, news article, budget, invoice, presentation, questionnaire, resume, memo, see Figure 4.2. Table 4.2 presents the number of training, testing and validation data samples for each document type. The evaluation metric is the overall classification accuracy.

Recently, pre-training techniques have increased the development of Document AI, achieving notable progress on downstream tasks. RVL-CDIP is a representative dataset for evaluating document image classification tasks. It has been used in several state-of-the-art works for document AI [72, 73, 4, 5].

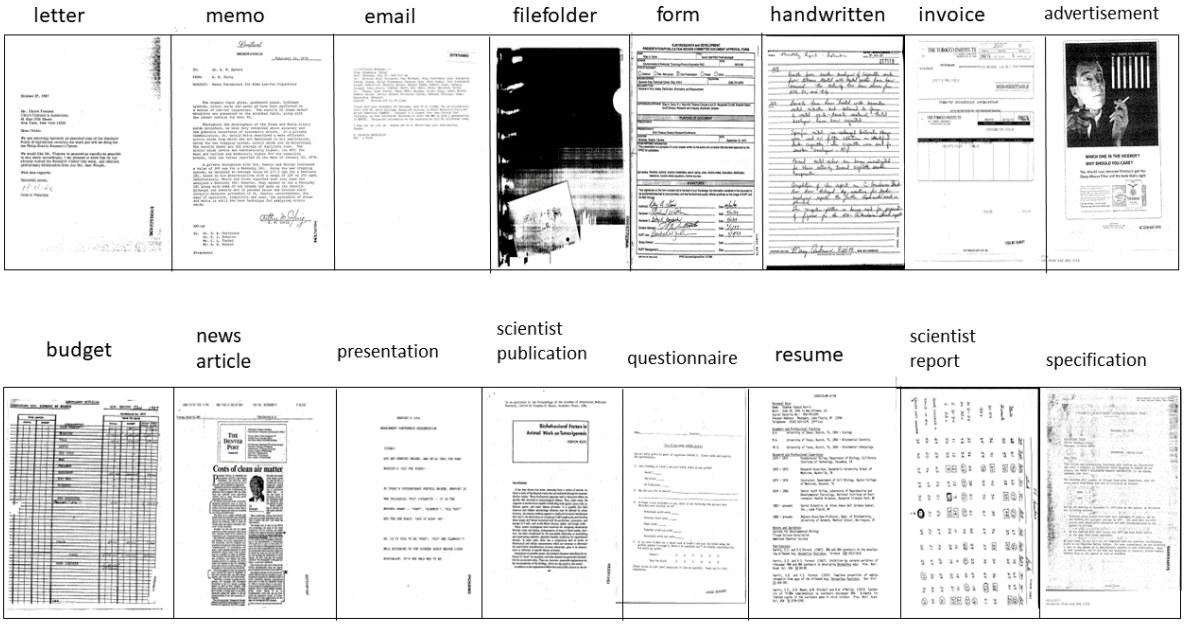


Figure 4.2: Samples of different document classes in the RVL-CDIP [26] dataset which illustrates the low inter-class discrimination and high intraclass variations of document images.

Table 4.2: Number of document pages in each class of the RVL-CDIP dataset, showing counts per split in training, testing and validation.

Class	Document Type	Train	Validation	Test
0	letter	20015	2424	2457
1	form	19871	2526	2495
2	email	19911	2529	2516
3	handwritten	19322	2341	2411
4	advertisement	19005	2383	2386
5	scientific report	19874	2502	2484
6	scientific publication	19834	2522	2568
7	specification	19917	2524	2468
8	file folder	17690	2153	2239
9	news article	19921	2523	2454
10	budget	19909	2474	2488
11	invoice	19868	2569	2468
12	presentation	19933	2457	2481
13	questionnaire	19909	2499	2421
14	resume	19993	2423	2535
15	memo	19877	2527	2478

4.2.3 Tobacco-3482

Tobacco-3482, also known as SmallTobacco, is another publicly available dataset comprising 3482 images of 10 different classes extracted. It was selected and labeled by Kumar et al. (2012) [32]. An example image from each of the ten classes (Advertisement, E-mail, Form, Letter, Memo, News, Note, Report, Resume, Scientific) in Tobacco-3482 is shown in Figure 4.3. Differently from RVL-CDIP, the Tobacco-3482 does not come with prebuilt subsets for train, validation, and test. Except for the Note and Report class, all others are already included in the RVL-CDIP dataset. Unlike the RVL-CDIP dataset, the distribution of the samples across the classes is not the same.

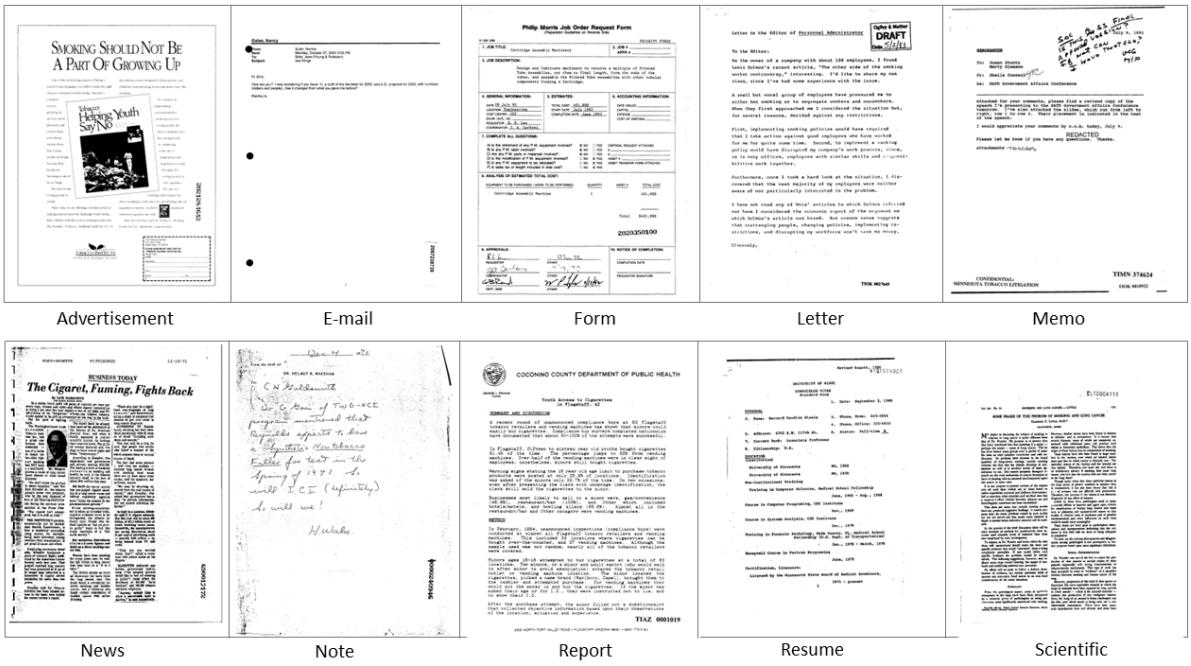


Figure 4.3: Samples of different document classes in the Tobacco-3482 [32] dataset which illustrates the low inter-class discrimination and high intraclass variations of document images.

SmallTobacco dataset was used in several related papers for document image classification. Tobacco-3482 was used by Noce et al. (2016) [50] to evaluate a document image classification method based on combined visual and textual information. Asim et al. (2019) [4] utilized the InceptionV3 model to classify text document images using transfer learning. They have trained InceptionV3 on the RVL-CDIP dataset using ImageNet weights and utilized transfer learning to classify Tobacco-3482 text document images. To evaluate the effectiveness of a cross-modal deep network that jointly learns text-image features to classify document images, Bakkalli et al. (2020) [7] utilized the benchmark Tobacco-3482 dataset.

4.2.4 VICTOR

VICTOR [17, 42] is a dataset of legal documents belonging to Brazil's Supreme Court (Supremo Tribunal Federal or STF) suits were labeled by a team of experts. This dataset was built as part of the VICTOR project, a partnership between the STF, UnB, and Finatec. The project aimed to develop an artificial intelligence tool to assist the STF in analyzing extraordinary appeals from all over the country, especially regarding their classification in the most recurrent themes of general repercussions. Some other works that resulted from this project using the VICTOR dataset are presented in [9, 42, 17, 15].

The VICTOR dataset comprises 45,532 Extraordinary Appeals (*Recursos Extraordinários*) from the STF. Each suit contains several documents, ranging from the appeal to certificates and rulings, totaling 692,966 documents comprising 4,603,784 pages. Most cases reach the court as PDF files, each representing a specific document or an unstructured volume containing multiple documents. A significant part of the data provided is in the form of images obtained by scanning printed documents that often contain handwritten notes, stamps, stains, and other sources of visual noise, like the ones shown in Figure 4.4. The dataset contains two types of annotations and supports two tasks: document type classification and theme assignment, a multilabel problem.

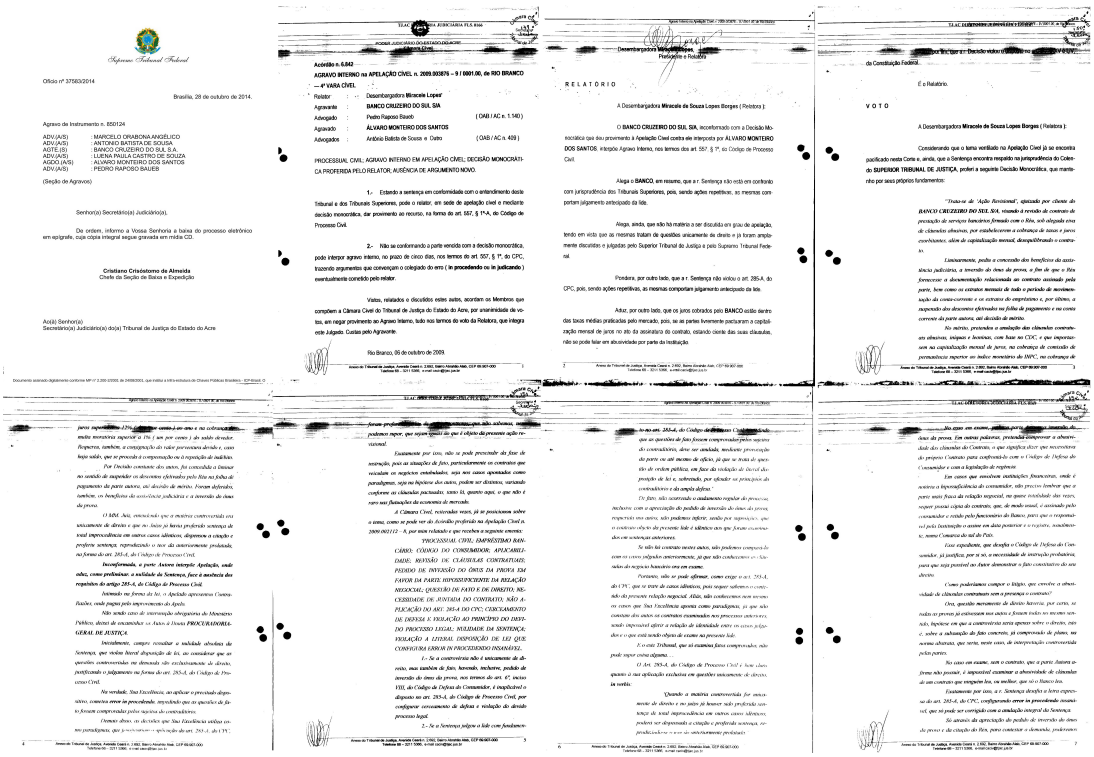


Figure 4.4: The first eight pages of a lawsuit of the VICTOR dataset [17]. While the first page is clean, the others come from an older document and contain ink stains, stamps, handwritten signatures, and other artifacts.

Table 4.3: Class counts per split in training, testing and validation show the number of the first page and the not-first page.

Class	**Train**		**Validation**		**Test**	
	First page	Not first page	First page	Not first page	First page	Not first page
Acórdão	301	282	198	116	197	88
ARE	266	3,954	227	2,423	203	2,334
Despacho	265	96	143	40	146	52
Others	37,114	103,672	24,292	67,110	24,193	63,709
RE	450	9,731	317	6,483	301	5,876
Sentença	420	1,757	277	1,336	262	1,216

There are six different labels for document type classification:*Acórdão*, for lower court decisions under review; *Recurso Extraordinário* (RE), for appeal petitions; *Agravo de Recurso Extraordinário* (ARE), for motions against the appeal petition; *Despacho*, for court orders; *Sentença* for judgments; and Others for documents not included in the previous classes.

First, Luz et al. (2020) [17] introduced three versions of this VICTOR dataset: Big, Medium, and Small. Big VICTOR (BVic) is used only for theme classifications since it contains all data, including the unlabeled documents. Medium VICTOR (MVic), with 44,855 suits, 628,820 documents, and 2,086,899 pages, is the result of filtering out those samples and can be employed for both theme and document type classification. The number of MVic processes was limited for each theme to 100 samples in each set to create the Small VICTOR (SVic) dataset, which contains 6,510 Extraordinary Features, 94,267 documents, and 339,478 pages.

Luz et al. (2022) [17] also introduced SVic+, a multimodal dataset of lawsuits composed of ordered document images and corresponding texts. This SVic+ dataset is an extension of Small VICTOR, which was expanded to include the document images and textual data. Every page in the expanded corpus is stored in at least one of two formats. First, as text extracted through optical character recognition, with the following additional preprocessing steps: lower-casing, removal of stop words and alphanumeric tokens, e-mail and URL tokenisation (e-mails and URLs are replaced by the tokens 'email' and 'link'), and special tokenisation of legislation references (e.g., Lei (law) 11.419 to LEI_11419). Second, JPEG images were converted from the original PDF files, with mean width and height of 1664 and 2322 pixels, respectively. Table 4.3 presents the number of training, testing and validation data samples for each class.

We present our results from the downstream tasks for each dataset in the form of tables and graphs in the next section.

4.3 Results and Discussions

We divided the results according to the downstream tasks performed and the dataset used. Initially, we present the results and discussions of the page stream segmentation task on the Tobacco800 dataset, followed by the results and discussions of the document classification task on the RVL-CDIP and VICTOR datasets.

4.3.1 Page Stream Segmentation on Tobacco800 dataset

We performed the page stream segmentation task based on Braz et al. (2021) [10], which aims to classify the first page of a document as FirstPage and the continuation pages as NextPage with the Tobacco800 dataset using our LayoutQT method by adding quadrant tags and as a baseline processing without placing tags using only text. Such experiments were processed using the LSTM, ULMFiT with AWD-LSTM and $BERT_{BASE}$ models.

The validation split results in Table 4.4 brought out that there was a large room for improvement in the baseline by only using text sequence architecture since we have surpassed Braz et al. (2021) [10] and Weidemann (2019) [69] baselines by at least 6 points of F1-score. After applying LayoutQT, we got 1.7 points more out of the 2.1 possible, which turns out to be 80.9% of the possible gain. Furthermore, comparing the results obtained from our model with tags and without tags (baseline) using the LSTM, AWD-LSTM and $BERT_{BASE}$ networks as the backbone, we obtained better results with AWD-LSTM.

Table 4.4: Accuracy and F1-score (in %) of the page stream segmentation on the Tobacco800 dataset obtained with the baseline and LayoutQT compared to the state-of-the-art.

Model	**Modality**	**Backbone**	**Accuraccy**	**F1-score**
Braz et al. (2021) [10]	image only	VGG16	92.0%	91.9%
Braz et al. (2021) [10]	image only	EfficientNet-B0	83.7%	81.9%
Wiedemann et al. (2019) [69]	text + image	VGG16	91.1%	90.4%
Baseline	text only	LSTM	84.1%	82.9%
LayoutQT baseline	text + layout	LSTM	85.9%	86.1%
BERT baseline	text only	$BERT_{BASE}$	92.2%	92.0%
BERT with LayoutQT	text + layout	$BERT_{BASE}$	93.0%	93.0%
ULMFiT baseline	text only	AWD-LSTM	97.5%	97.9%
ULMFiT with LayoutQT	text + layout	AWD-LSTM	**99.5**%	**99.6**%

Figure 4.5 shows the confusion matrix of binary classification to Tobacco800 dataset without tags (baseline) and with tags of quadrants (LayoutQT) using ULMFiT (AWD-LSTM) model. It is clear that for the detection of first page images, both the baseline

and our model missed only one image, but for detection of the follow-up pages, the model without our tags missed four images, while with our tags, there was only one error.

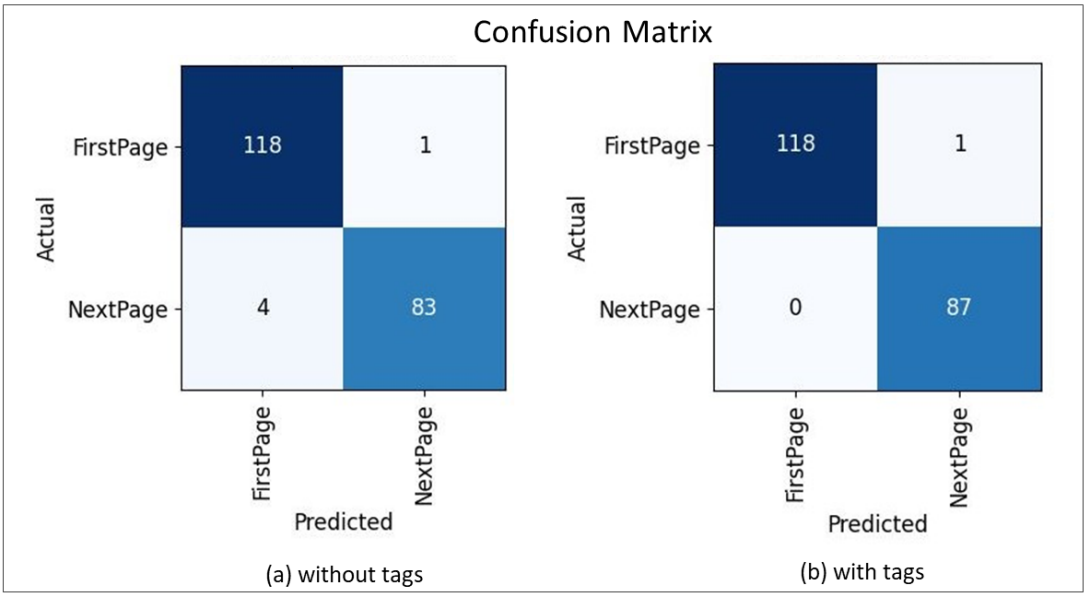
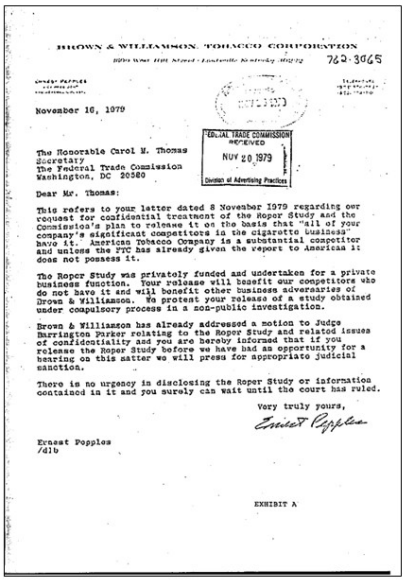


Figure 4.5: Confusion matrix of Tobacco800 binary classification using AWD-LSTM with 24 quadrants.(a) results found from the experiment without the tags, that is, with the baseline. (b) results obtained with the tags (LayoutQT).

This analysis provides insights into the classifier's performance for each class, highlighting areas where the classifier excels and areas where it may need improvement. After analyzing the two confusion matrices in Figure 4.5, we review the documents in which the binary classifier makes a wrong prediction. We verified a wrongly labeled document in the test sample, which appears in the confusion matrices as False Negative. The document in Figure 4.6 is the first page and is incorrectly labeled NextPage. The classifier's performance with LayoutQT achieved excellence in the PSS task on the Tobacco800 dataset.

To verify whether the number of quadrants influenced the results, we varied the baseline, without division of quadrants, with four quadrants, six quadrants, up to 35 quadrants, dividing the document into seven lines by five columns. The results obtained from varying the quadrants with AWD-LSTM on the Tobacco800 data set are shown in Figure 4.7. We can observe that the best result was obtained with 24 quadrants with 99.6% of the F1 and the worst with 35 quadrants with 96.2% of the F1. The results prove an ideal value for the number of quadrants. We show that the LayoutQT tags improve classification performance by using up to 4x6 quadrants per page. More than that may harm the performance. We hypothesize that the excessive location tasks are less informative and make the text noisy.

The LayoutQT method can be easily adapted to other architecture, including $BERT_{BASE}$. However, in the Tobacco800 dataset, the AWD-LSTM model outperforms the BERT



BROWN & WILLIAMSON TOBACCO CORPORATION

November 16, 1979

The Honorable Carol M. Thomas
Secretary
The Federal Trade Commission
Washington, DC 20580

FEDERAL TRADE COMMISSION RECEIVED NOV 20 1979 Division of Advertising Practices

Dear Ms. Thomas:

This refers to your letter dated 8 November 1979 regarding our request for confidential treatment of the Roper Study and the Commission's plan to release it on the basis that "all of your company's significant competitors in the cigarette business" have it. American Tobacco Company is a substantial competitor and unless the FTC has already given the report to American it does not possess it.

The Roper Study was privately funded and undertaken for a private business function. Your release will benefit our competitors who do not have it and will benefit other business adversaries of Brown & Williamson. We protest your release of a study obtained under compulsory process in a non-public investigation.

Brown & Williamson has already addressed a motion to Judge Harrington Parker relating to the Roper Study and related issues of confidentiality and you are hereby informed that if you release the Roper Study before we have had an opportunity for a hearing on this matter we will press for appropriate judicial sanction.

There is no urgency in disclosing the Roper Study or information contained in it and you surely can wait until the court has ruled.

Very truly yours,

Ernest Popples
/dlb

EXHIBIT A

Ukd41a00-ernest

Figure 4.6: Image of the first page of a document from the Tobacco800 dataset [35], which was mislabeled as a page within the document ("next page" class).

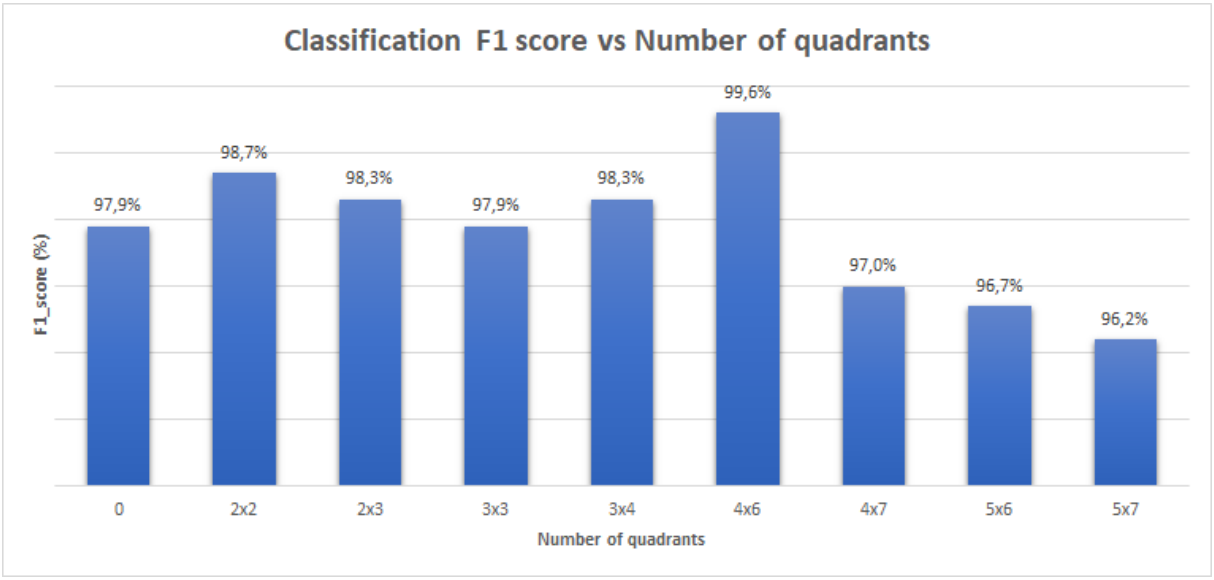


Figure 4.7: F1-score (in %) of the experiments carried out for the page stream segmentation task using LayoutQT with the variation of the number of quadrants on the Tobacco800 dataset in the AWD-LSTM architecture.

model in the classification F1 metric by a large margin (99.6% vs 93.0%). Considering the fewer parameters of the AWD-LSTM model - while the $BERT_{BASE}$ model has 110M parameters, the AWD-LSTM model has only 24M parameters. For this reason, we adopted the AWD-LSTM model as our default architecture.

4.3.2 Document Classification on RVL-CDIP dataset

Our proposed approach also demonstrated superior performance over the baseline for document classification on the RVL-CDIP dataset with AWD-LSTM backbone, as shown by the confusion matrices in Figure 4.8. When our location tokens are not used, the resulting F1 score is 80.7% (AWD-LSTM) and 80.1% ($BERT_{BASE}$). However, when we use our LayoutQT, the F1 score goes to 85.9% (AWD-LSTM) and 84.5% ($BERT_{BASE}$), as shown in Table 4.5.

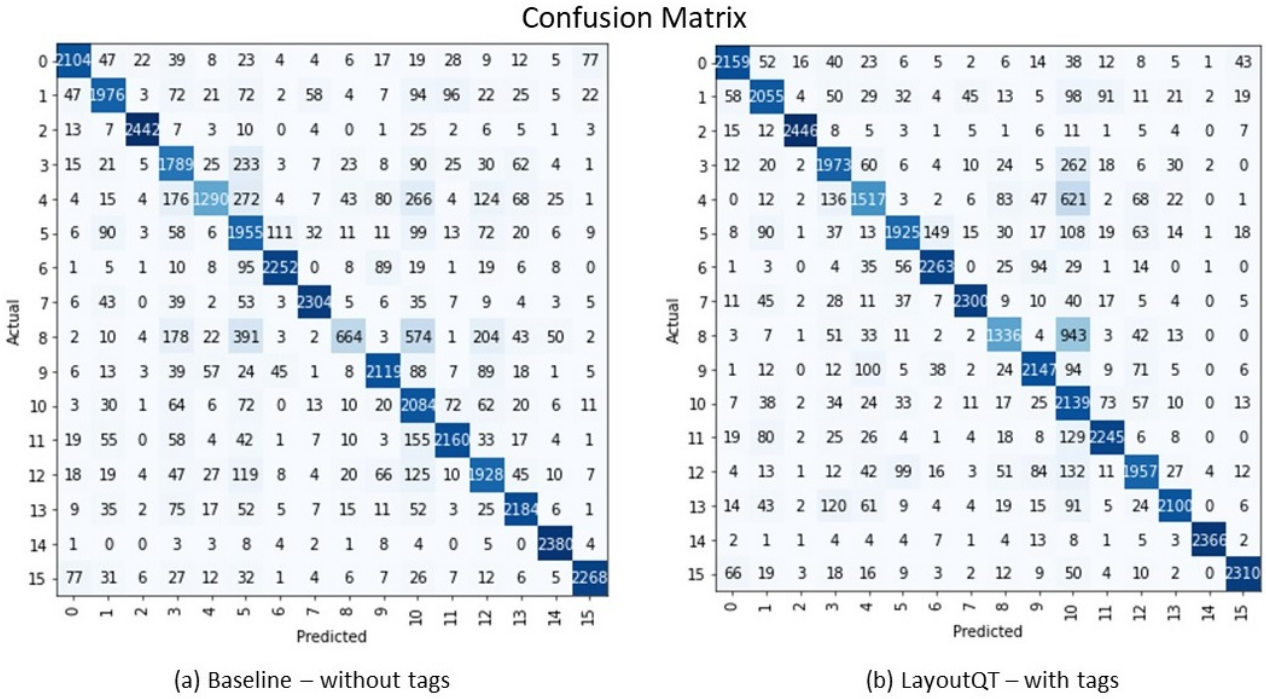


Figure 4.8: Confusion matrix of RVL-CDIP composed of 16 document classes: 0-letter, 1-form, 2-email, 3-handwritten, 4-advertisement, 5-scientific report, 6-scientific publication, 7-specification, 8-file folder, 9-news article, 10-budget, 11-invoice, 12-presentation, 13-questionnaire, 14-resume and 15-memo. Confusion matrix (a) shows the results of processing without tags, while confusion matrix (b) shows the results of our model using tags.

To classify documents using the RVL-CDIP dataset, we performed an error analysis of the confusion matrix in Figure 4.8 with tags, using the precision, recall, and F1-score metrics. Classes such as “Email”, “Resume”, and “Memo”, for example, demonstrate high precision, recall, and F1 score, indicating accurate classification. Some classes, such as

"File Folder" and "Advertisement", have relatively lower precision and recall, suggesting difficulty distinguishing instances belonging to these classes. We observed several samples of documents in the cited classes misclassified. The main characteristic of these documents is that they have little or no text, and the positional tags added by LayoutQT have little significance in classifying these documents. Figure 4.9 shows a document from the "File Folder" class, one from the "Advertisement" class, and the respective insertions of positional tags.

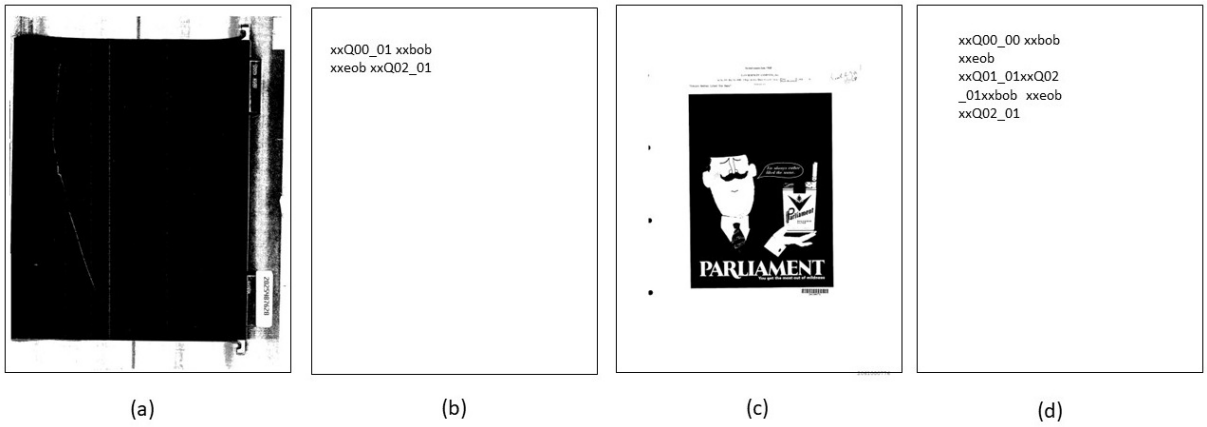


Figure 4.9: Examples of two documents from the RVL-CDIP [26] dataset where both baseline and the proposed method fail. In both pages, there is little or no textual content. (a) sample from the File Folder class and (b) tags added by LayoutQT to the document. (c) Sample of the Advertisement class and (d) tags added by LayoutQT to the document.

In addition, Table 4.5 compares the performance of the document classification proposals, baseline and LayoutQT, from the RVL-CDIP dataset for each document class using AWD-LSTM and $BERT_{BASE}$ model. The results show that our approach to adding positional tags performed better for all classes of documents w.r.t. the baseline with AWD-LSTM backbone. For detection of file folders, LayoutQT obtained a significantly better result than the baseline. That is, the percentage result of our approach more than doubled the baseline due to the relevance of the location text in such a document. The overall ranking result with LayoutQT showed an advantage of 5.2% in the F1 metric compared to the baseline. The highest F1 score result of 98.3% was obtained in the resume class with our approach using AWD-LSTM.

In contrast, in the case of $BERT_{BASE}$, LayoutQT results were lower than the baseline in only two classes: form and resume, both by a relatively small margin. In the case of forms, it is likely that those documents have too many text boxes, therefore there is an overload of layout tags, which are likely to make the representation too noisy for BERT. In addition, the comparison of LayoutQT results with AWD-LSTM and $BERT_{BASE}$ shows that AWD-LSTM performed better than $BERT_{BASE}$ in most classes, with only three

Table 4.5: F1-score (in %) of the document types classification on RVL-CDIP dataset obtained with the baseline and LayoutQT. The results in absolute numbers of hits and misses by classes are shown in Figure 4.8

Class	Document Type	Baseline AWD-LSTM	LayoutQT AWD-LSTM	Baseline $BERT_{BASE}$	LayoutQT $BERT_{BASE}$
0	letter	88.3%	**89.7%**	83.7%	**86.0%**
1	form	79.7%	**81.3%**	**77.8%**	77.3%
2	email	97.3%	**97.6%**	93.0%	**96.0%**
3	handwritten	70.9%	**83.3%**	63.6%	**80.0%**
4	advertisement	65.4%	**68.2%**	66.0%	**70.0%**
5	scientific report	65.5%	**80.8%**	74.8%	**80.3%**
6	scientific publication	90.5%	**92.1%**	87.4%	**89.0%**
7	specification	92.1%	**93.6%**	90.7%	**91.0%**
8	file folder	31.9%	**63.5%**	64.0%	**73.8%**
9	news article	86.4%	**86.8%**	78.8%	**82.6%**
10	budget	77.8%	**84.0%**	78.1%	**82.3%**
11	invoice	87.9%	**89.9%**	81.4%	**85.9%**
12	presentation	79.9%	**81.0%**	70.3%	**81.1%**
13	questionnaire	88.9%	**89.5%**	83.7%	**87.9%**
14	resume	98.1%	**98.3%**	**98.6%**	98.3%
15	memo	91.4%	**92.5%**	85.4%	**90.0%**
Average		80.7%	**85.9%**	80.1%	**84.5%**

exceptions (advertisement, file folder, and presentation). The overall average F1 score of our approach with AWD-LSTM was 1.4% points higher than that of $BERT_{BASE}$.

4.3.3 Document Classification in Portuguese on the VICTOR dataset

Figure 4.10 shows the document classification confusion matrix for the VICTOR dataset with only the first page of each document without quadrant tags (Baseline) and with quadrant tags (LayoutQT) using (AWD-LSTM) model. Analyzing the confusion matrix and the metrics used, we verified that the *Acórdão* class presents relatively high precision and recall, indicating that the classifier performs well in identifying instances of this class and that LayoutQT obtained better results than the baseline. Precision and recall are relatively low for the ARE and *Despacho* classes, suggesting that the classifier has more difficulty correctly identifying instances of these classes. The Others class exhibits high precision, recall, and F1 scores, indicating that the classifier performs exceptionally well in identifying instances of this class.

Table 4.6 exhibits the F1 scores of document image classification with the AWD-LSTM model on the VICTOR dataset. It also shows the difference in classification performance

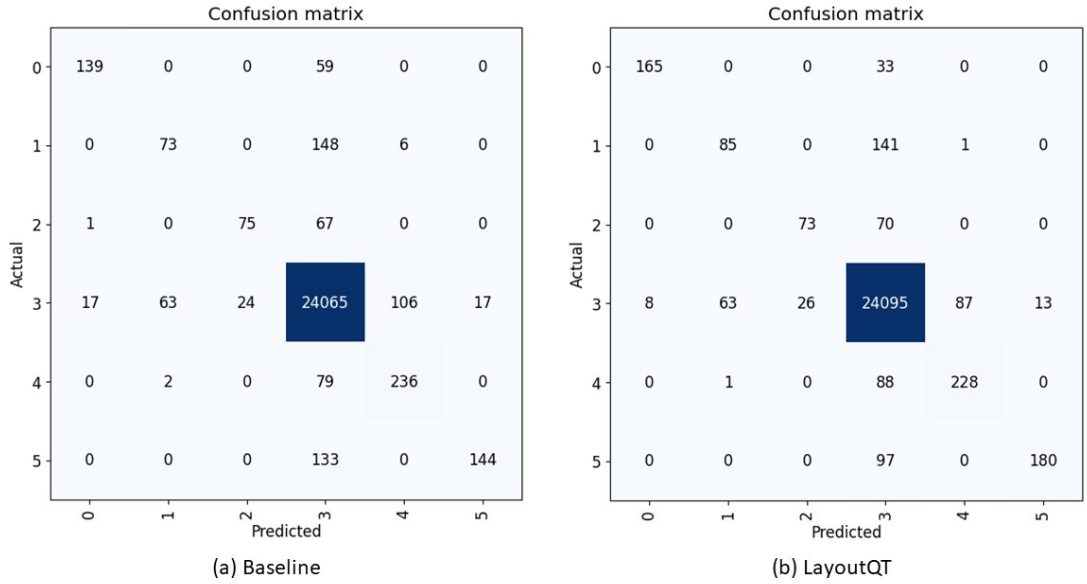


Figure 4.10: Confusion matrix of VICTOR composed of 6 document classes: 0-*Acórdão*, 1-*Agravo de Recurso Extraordinário* (ARE), 2-*Despacho*, 3-Others, 4-*Recurso Extraordinário* (RE), and 5-*Sentença*. Confusion matrix (a) shows the results of processing without tags, while confusion matrix (b) shows the results of our model using tags using ULMFiT

of samples on the first page of a document versus pages other, considering only text (baseline) versus fusion of text and layout (our method). We first compared the results obtained from the baseline with the LayoutQT across the entire dataset. LayoutQT's average F1 score result (73.6%) exceeds the baseline (72.3%) by 1.3%.

Also, all F1 score results obtained with LayoutQT were better than the baseline using the complete VICTOR dataset, except for the result of the *Acórdão* class, where the baseline outperformed our approach by 0.3%. The highest F1 score in document classification with AWD-LSTM model in the full VICTOR dataset was obtained in the Other class (96.7%) using LayoutQT, and the lowest was in the *Despacho* class (53.6%) using the baseline. This F1 score comparison can be best visualized through the bar chart in Figure 4.11.

In the case of experiments with the sample containing only the first page of documents, LayoutQT's F1 score performance was better than the baseline in almost all document classes except the RE class. Our approach to the Others class obtained the highest F1 score with 99.1%, and the ARE class obtained the worst F1 score with 57.5% using AWD-LSTM on the VICTOR dataset with the documents' First-page sample, as shown in Table 4.6. The last two columns present the F1 scores of the sample of the Not-first page or the First-page complement. LayoutQT was better than the baseline with the Not-first

Table 4.6: F1 score (in %) classification of document types in the VICTOR dataset obtained with baseline and LayoutQT using AWD-LSTM for the whole dataset. Then, the dataset is split into two samples: the first with only the first page of each document and the second with the rest of the pages.

Class	Baseline	LayoutQT	Baseline First page	LayoutQT First page	Baseline Not first page	LayoutQT Not first page
Acórdão	**80.4%**	80.1%	89.5%	**90.8%**	57.9%	**60.2%**
ARE	58.5%	**62.8%**	57.5%	**64.9%**	64.8%	**64.9%**
Despacho	53.6%	**55.1%**	68.9%	**77.4%**	33.5%	**40.8%**
Others	96.6%	**96.7%**	98.9%	**99.1%**	**96.3%**	96.0%
RE	70.4%	**71.9%**	**76.6%**	76.3%	**73.2%**	71.8%
Sentença	74.1%	**74.8%**	78.2%	**83.1%**	**77.2%**	76.1%
Average	72.3%	**73.6%**	78.3%	**81.9%**	67.2%	**68.3%**
Weighted	91.6%	**93.9%**	97.8%	**98.2%**	92.9%	**93.5%**

page sample in the first three classes (*Acórdão*, ARE, *Despacho*) and worse in the last three (Others, RE, *Sentença*). However, on average and weighted, LayoutQT's F1 score outperformed the baseline by 1.1% and 0.6% in the Not-first page sample with AWD-LSTM model, respectively. In all experiments, the average/weighted results of the F1 metric with LayoutQT are higher than the baseline.

The best F1 results for document classification were obtained with the first-page sample set of the documents with LayouQT. Figure 4.12 shows that the highest bar in each class is the LayoutQT First Page bar in the legend, the result of running the sample experiments using only the first page of each document with our approach. The first-page sample set achieved average/weighted F1 scores of 13.6%/4.7% higher than its complement. The first-page sample set using LayoutQT also had average/weighted F1 scores of 9.6%/4.3% higher than the full VICTOR dataset. These results show that the first pages are more informative from the point of view of both textual and layout features.

The comparison of the F1 scores of the performances of the AWD-LSTM, $BERT_{BASE}$ and BiLSTM-F [17] models on the first-page sample of the VICTOR dataset is presented in Table 4.7, categorized by the use of textual (Baseline), textual and layout (LayoutQT) or textual and visual (Luz et al. (2022) [17]) information. Combining positional tags with text embedding increases the performance of all classes except the RE class, where it dropped 0.2% from baseline using AWD-LSTM. Furthermore, our LayoutQT approach improved the performance of all classes using $BERT_{BASE}$, except for the Others class, where the same value of 99.0% of the F1 score remained. We can also observe that LayoutQT with $BERT_{BASE}$ performs better than LayoutQT with AWD-LSTM in almost all classes, except *Despacho* and Others classes, with 4.3% and 0.1% less, respectively. However, on average, $BERT_{BASE}$ outperforms AWD-LSTM by just 0.4% F1 score.

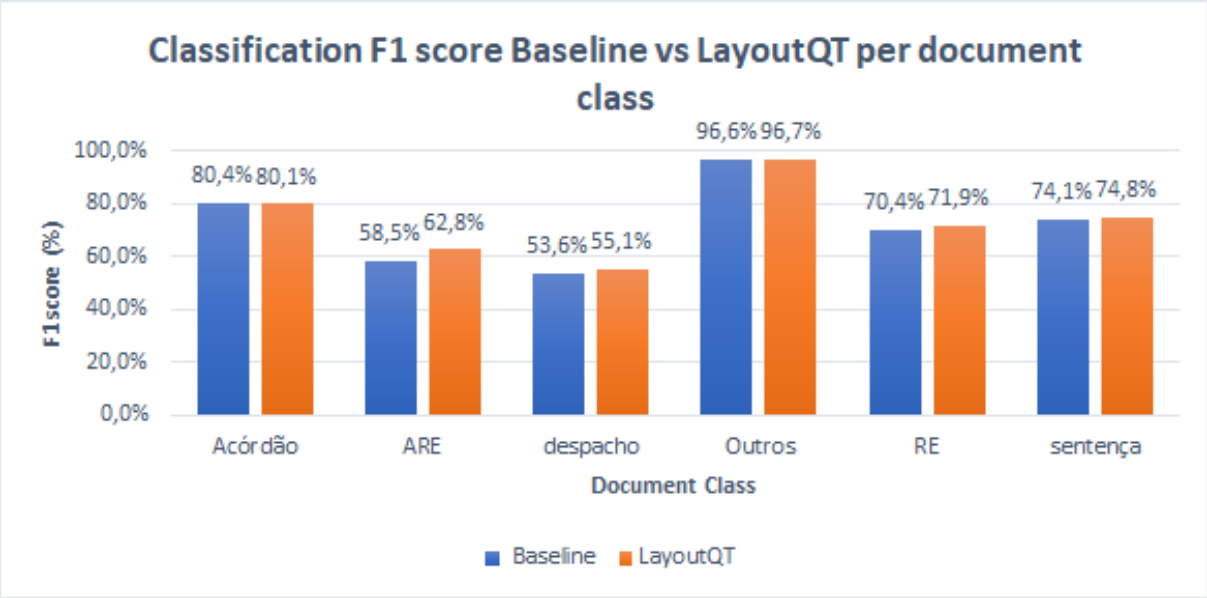


Figure 4.11: F1 score (in %) for document image classification using baseline and LayoutQT using AWD-LSTM architecture on the whole VICTOR dataset.

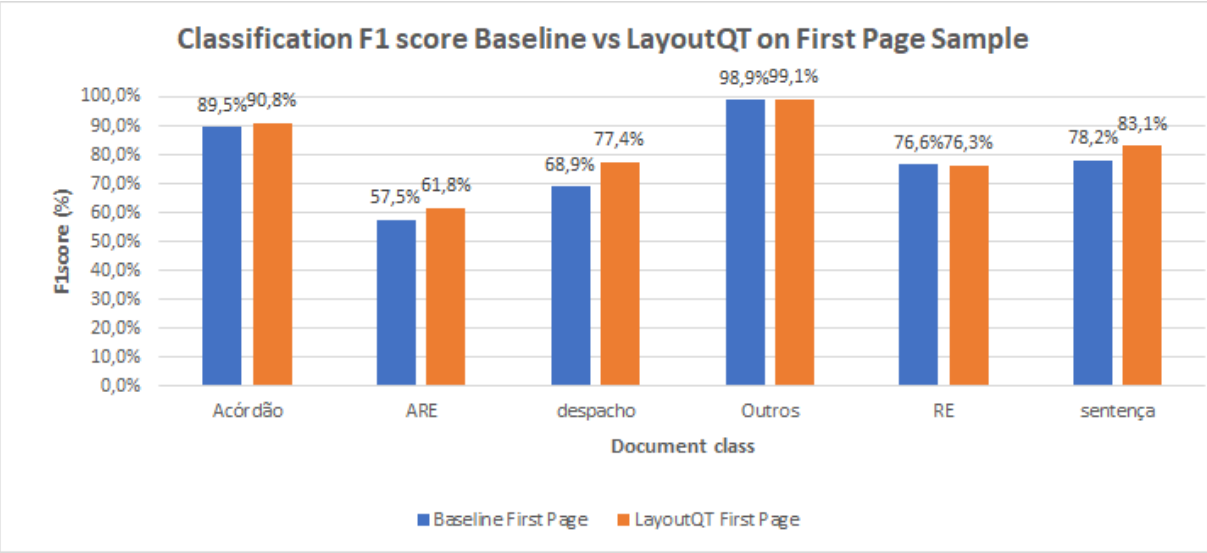


Figure 4.12: F1 score (in %) of document image classification using baseline and LayoutQT using AWD-LSTM architecture on the VICTOR First-page sample dataset.

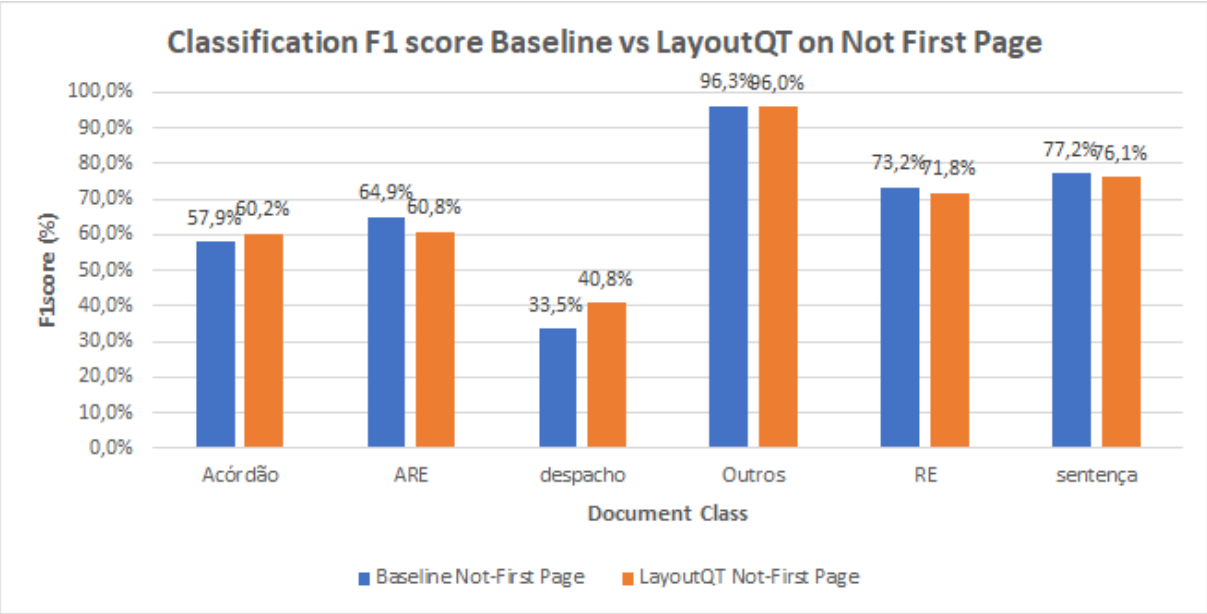


Figure 4.13: F1 score (in %) of document image classification using baseline and LayoutQT using AWD-LSTM architecture on the VICTOR Not First page sample dataset.

Table 4.7: F1-score (in %) of the document types classification in the sample of the VICTOR dataset obtained with the baseline and LayoutQT using $BERT_{BASE}$ and AWD-LSTM models compared to work by Luz et al. (2022) [17].

Class	**AWD-LSTM Baseline**	**AWD-LSTM LayoutQT**	$BERT_{BASE}$ **Baseline**	$BERT_{BASE}$ **LayoutQT**	**BiLSTM-F Luz et al. [42]**
Acórdão	89.5%	90.8%	92.1%	**93.4%**	**93.4%**
ARE	57.5%	61.8%	57.5%	**64.9%**	59.9%
Despacho	68.9%	**77.4%**	64.0%	73.1%	71.8%
Others	98.9%	**99.1%**	99.0%	99.0%	99.0%
RE	76.6%	76.3%	72.2%	**79.1%**	75.5%
Sentença	78.2%	83.1%	82.1%	**87.1%**	83.1%
Average	78.3%	81.9%	77.8%	**82.3%**	80.5%
Weighted	97.8%	98.2%	97.9%	**98.6%**	98.1%

Finally, our textual feature layout approach overcomes the visual and textual feature concatenation method proposed by Luz et al. (2022) [17]. In the LayoutQT experiments with AWD-LSTM, the performance was better than BiLSTM-F [17] in almost all classes, except for the *Acórdão* class with 2.6% smaller and the *Sentença* class whose value is the same for both models. In the LayoutQT with $BERT_{BASE}$, performance was better than BiLSTM-F in almost all classes, except for the *Acórdão* and the Others classes, which obtained the same value in both models. In addition, our LayoutQT approach using AWD-LSTM and $BERT_{BASE}$ performed, on average/weighted, higher than BiLSTM-F. Thus, the LayoutQT method on the VICTOR dataset showed an improvement over the baseline of at least three percentage points in average F1 score and an improvement of at least 1.4% over the state-of-the-art [17].

4.4 Summary

This chapter introduced the experiment settings that were performed using LSTM, AWD-LSTM and BERT architectures with our LayoutQT method generating the quadrant tags and baseline without the preprocessing on three datasets: Tobacco800, RVL-CDIP and VICTOR. In addition, four datasets of images of literature documents for page stream segmentation and document classification were described: Tobacco800, RVL-CDIP, Tobacco-3482, and VICTOR. The datasets contain visually rich documents containing both text and non-text. The first three (Tobacco800, RVL-CDIP and Tobacco-3482) contain images of publicly available English business documents extracted from the Legacy Tobacco Documents Library (LTDL) and are very similar to each other. The VICTOR dataset was obtained from STF court documents in Portuguese. After analyzing all the datasets, we verified that Tobacco-3482 is practically a subset of the RVL-CDIP. Given the above, only three datasets (Tobacco800, RVL-CDIP and VICTOR) were chosen to evaluate our proposed approach to the document classification task.

Our method was evaluated using the Page Stream Segmentation and Document Image Classification tasks. The results of the experiments are exhibited in the form of tables and graphs, along with the data evaluation metrics used, such as accuracy and F1 metrics. Furthermore, discussions relevant to the results obtained are presented. The LayoutQT method combines text and layout features, improving the baseline in experiments with all chosen datasets. The method can be easily used on various architectures and can improve the results of downstream tasks by combining text and layout features. AWD-LSTM gives the best results on Tobacco800 and RVL-CDIP. However, on VICTOR, which is a bigger dataset with less variation between different classes, BERT, which is a more complex model, gave better results.

Chapter 5

Concluding Remarks

In this Chapter, we present the main conclusions of this thesis based on the results obtained in this research and the limitations found from the error analysis. Finally, we discuss possible directions for future work.

5.1 Conclusion

We proposed a simple and effective method combining layout and textual features with a low computational cost for text processing. We use a rules-based and feature engineering approach. Specifically, it takes information from the bounding boxes issued by an OCR engine. It extracts coherent information from the text layout, like page and document position for each text block. Our method, introduced in Chapter 3, divides the document into quadrants and uses the quadrant location to add spatial tokens to mark each text box's start and end position. In addition, we also applied a greedy algorithm to organize the words in blocks, firstly processing lines and then processing the groups of words.

This method, dubbed LayoutQT, was tested and evaluated with artificial neural networks of LSTM, AWD-LSTM/ULMFiT and BERT architectures to perform page flow segmentation and document image classification. The datasets chosen for training/fine-tuning were Tobacco800, RVL-CDIP and VICTOR. The first two comprise document images in .tiff format, and the last was introduced in the Luz et al. (2020) [42] was generated from documents in .pdf format. Results were evaluated using Accuracy and F1 score, which are the most widely used metrics used for these problems.

We conducted experiments with an initially fixed number of 24 empirically chosen rectangular regions (quadrants) and compared them with the baseline, as presented in Chapter 4. Each quadrant is identified by a positional token that is later inserted into the embedding of text blocks. Next, we performed experiments with varying the number

of quadrants. We verified that our empirical choice of 24 quadrants gave better results than using other configurations.

Our model achieved the best result using ULMFiT and AWD-LSTM on the Tobacco800 dataset for the PSS task, achieving the following values in the evaluation metrics on the test set: accuracy of 99.5% and F1 score of 99.6%, surpassing the baseline model by at least two percentage points and the state-of-the-art by seven percentage points. In the document classification task on the RVL-CDIP dataset, LayoutQT achieved the best result using the ULMFiT model, with AWD-LSTM outperforming BERT by 1.4% of F1 score. On the VICTOR dataset sample set containing only the first page, LayoutQT achieved better F1 score results than the baseline and the work of Luz et al. (2020) [42]. Furthermore, this first-page sample set gave better than the full VICTOR and not-first-page sets. This is consistent with [42].

The general objective of producing a trained document processing model that combines textual information and layout was achieved. The specific objectives were also met in specific information fusion combining positional information from text blocks and text embeddings, using different learning models (LSTM, AWD-LSTM and BERT architecture). We emphasize that LayoutQT showed superior performance against baselines and state-of-the-art for document classification and PSS tasks on the Tobacco800 and VICTOR datasets, using a low additional cost in the positional tag insertion step.

5.2 Limitations

The main limitation of our approach is that it is designed to enrich the textual representation using layout information. The documents with little or no text may not benefit from layout representation.

5.3 Future Works

By analyzing the results per class in Table 4.5, we observed that classes with a small amount of text, such as file folder and form, are the most challenging. Therefore, our first recommendation for future work is to explore ways to automatically balance textual and visual features such that visual tokens can enrich document representations even when a very small quantity of textual boxes are present (or none at all).

A second possibility is to exploit the proposed method on other kinds of data where layout has an even higher level of importance, such as webpages, magazines, catalogues, etc. A webpage contains text content and images of various sizes, hyperlinks to navigate to other pages, domain and server information, HTML tags, and semantic web tags.

Therefore, automatic web page classification is challenging due to its complexity, diversity of content, images of different sizes, text, hyperlinks, and computational cost. On the other hand, they have a lot of layout features that can be exploited to enrich their representation.

A third suggestion is to evaluate the performance of LayoutQT with other downstream tasks, such as machine translation, next sentence predictions, etc. Research into multimodal machine translation (MMT) has surged, incorporating extra modalities like images to enhance the translation precision of text-based systems [25]. These multimodal approaches find specific utility in simultaneous machine translation tasks, where visual context augments the limited information from the source sentence, which is particularly beneficial in the initial translation stages [25]

We also recommend exploiting our layout coding method with recent Large Language Models (LLMs). Nowadays, LLMs have a significant impact on the AI community, and the advent of ChatGPT[1] and GPT-4[2] leads to rethinking the possibilities of artificial general intelligence (AGI). Despite the impressive progress and impact of those models, we believe it is worth exploring the possibility of using LayoutQT tags with them, potentially helping them in tasks where layout plays an important role for document analysis. For simpler tasks, such as page classification, we believe that simpler models like the ones used in this thesis are good enough and it may not be worth using LLMs. Such models are difficult to train and to perform experiments with (e.g. ablation studies) due to their huge computation cost [74].

[1] https://openai.com/chatgpt
[2] https://openai.com/gpt-4

References

[1] O. Agin, C. Ulas, M. Ahat, and C. Bekar. An approach to the segmentation of multi-page document flow using binary classification. In *Sixth International Conference on Graphic and Image Processing (ICGIP 2014)*, volume 9443, pages 216–222. SPIE, 2015. 32, 34

[2] M. Aiello and A. Smeulders. Bidimensional relations for reading order detection. In *EPRINTS-BOOK-TITLE*. University of Groningen, Johann Bernoulli Institute for Mathematics and Computer Science, 2003. Relation: http://www.rug.nl/informatica/organisatie/overorganisatie/iwi Rights: University of Groningen, Research Institute for Mathematics and Computing Science (IWI). 19

[3] S. Appalaraju, B. Jasani, B. U. Kota, Y. Xie, and R. Manmatha. Docformer: End-to-end transformer for document understanding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 973–983, 2021. 36, 37

[4] M. N. Asim, M. U. G. Khan, M. I. Malik, K. Razzaque, A. Dengel, and S. Ahmed. Two stream deep network for document image classification. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 1410–1416, 2019. 34, 37, 50, 52

[5] N. Audebert, C. Herold, K. Slimani, and C. Vidal. Multimodal deep networks for text and image-based document classification. In P. Cellier and K. Driessens, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 427–443, Cham, 2020. Springer International Publishing. 2, 34, 36, 37, 50

[6] H. S. Baird. Background structure in document images. In *International journal of pattern recognition and artificial intelligence*, 1994. 16

[7] S. Bakkali, Z. Ming, M. Coustaty, and M. Rusiñol. Visual and textual deep feature fusion for document image classification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2394–2403, 2020. 35, 37, 52

[8] S. Bhowmik and R. Sarkar. Classification of text regions in a document image by analyzing the properties of connected components. In *2020 IEEE Applied Signal Processing Conference (ASPCON)*, pages 36–40, 2020. 11, 17

[9] F. A. Braz, N. C. da Silva, T. E. de Campos, F. B. S. Chaves, M. H. S. Ferreira, P. H. Inazawa, V. H. D. Coelho, B. P. Sukiennik, A. P. G. S. de Almeida, F. de Barros Vidal, D. A. Bezerra, D. B. Gusmao, G. G. Ziegler, R. V. C. Fernandes, R. Zumblick, and F. Peixoto. Document classification using a bi-lstm to unclog brazil's supreme court. *ArXiv*, abs/1811.11569, 2018. 53

[10] F. A. Braz, N. C. da Silva, and J. A. S. Lima. Leveraging effectiveness and efficiency in page stream deep segmentation. *Engineering Applications of Artificial Intelligence*, 105:104394, 2021. 8, 33, 34, 47, 50, 55

[11] K. Chen, F. Yin, and C. Liu. Hybrid page segmentation with efficient whitespace rectangles extraction and grouping. In *12th International Conference on Document Analysis and Recognition*, pages 958–962, 2013. 16

[12] N. Chen and D. Blostein. A survey of document image classification: problem statement, classifier architecture and performance evaluation. *International Journal of Document Analysis and Recognition (IJDAR)*, 10(1):1–16, 2007. 7

[13] C. Clausner, S. Pletschacher, and A. Antonacopoulos. The significance of reading order in document recognition and its evaluation. In *2013 12th International Conference on Document Analysis and Recognition*, pages 688–692, 2013. 18

[14] L. Cui, Y. Xu, T. Lv, and F. Wei. Document ai: Benchmarks, models and applications. *ArXiv*, abs/2111.08609, 2021. 1, 4, 31

[15] N. C. da Silva, F. A. Braz, T. E. de Campos, A. L. P. Guedes, D. B. Mendes, D. A. Bezerra, D. B. Gusmao, F. B. S. Chaves, G. G. Ziegler, L. H. Horinouchi, M. U. Ferreira, P. H. Inazawa, V. H. D. Coelho, R. V. C. Fernandes, F. Peixoto, M. S. M. Filho, B. P. Sukiennik, L. Rosa, R. Silva, T. A. Junquilho, and G. H. T. Carvalho. Document type classification for brazil's supreme court using a convolutional neural network. *Proceedings of The Tenth International Conference on Forensic Computer Science and Cyber Law*, 2018. 53

[16] H. Daher, M.-R. Bouguelia, A. Belaid, and V. P. D'Andecy. Multipage administrative document stream segmentation. In *22nd International Conference on Pattern Recognition*, pages 966–971, 2014. 8

[17] P. H. L. de Araujo, A. P. G. S. de Almeida, F. A. Braz, N. C. da Silva, F. de Barros Vidal, and T. E. de Campos. Sequence-aware multimodal page classification of brazilian legal documents. *International Journal on Document Analysis and Recognition (IJDAR)*, jul 2022. 7, 9, 45, 47, 48, 53, 54, 62, 64, 65

[18] P. M. L. de Lucena Drumond, L. P. Leite, T. E. de Campos, and F. A. Braz. Layoutqt—layout quadrant tags to embed visual features for document analysis. *Engineering Applications of Artificial Intelligence*, 122:106091, 2023. 9

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, volume 1, page 4171–4186, 2019. 20, 28, 30, 31, 33, 38, 45

[20] M. Diem, F. Kleber, and R. Sablatnig. Text classification and document layout analysis of paper fragments. In *International Conference on Document Analysis and Recognition*, pages 854–858, 2011. 21

[21] I. Gallo, L. Noce, A. Zamberletti, and A. Calefati. Deep neural networks for page stream segmentation and classification. In *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–7, 2016. 7, 32, 34

[22] M. Gorai and M. J. Nene. Layout and text extraction from document images using neural networks. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, pages 1107–1112, 2020. 2

[23] A. Gordo, M. Rusiñol, D. Karatzas, and A. D. Bagdanov. Document classification and page stream segmentation for digital mailroom applications. In *2013 12th International Conference on Document Analysis and Recognition*, pages 621–625, 2013. 5, 32

[24] A. Guha, A. Alahmadi, D. Samanta, M. Z. Khan, and A. H. Alahmadi. A multimodal approach to digital document stream segmentation for title insurance domain. *IEEE Access*, 10:11341–11353, 2022. 8, 33, 34

[25] V. Haralampieva, O. Caglayan, and L. Specia. Supervised visual attention for simultaneous multimodal machine translation. *Journal of Artificial Intelligence Research*, 74:1059–1089, sep 2022. 68

[26] A. W. Harley, A. Ufkes, and K. G. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995, 08 2015. 7, 35, 48, 50, 51, 59

[27] T. Hong, D. Kim, M. Ji, W. Hwang, D. Nam, and S. Park. Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents. *arXiv preprint arXiv:2108.04539*, 2021. 3

[28] J. Howard and S. Gugger. *Deep Learning for Coders with fastai and PyTorch*. O'Reilly Media, 2020. 40

[29] J. Howard and S. Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. 25, 26, 45

[30] K. Kise. A computational geometric approach to text-line extraction from binary document images. In *Computer Science, Mathematics*, 1998. 16

[31] S. C. Kosaraju, M. Masum, N. Z. Tsaku, P. Patel, T. Bayramoglu, G. Modgil, and M. Kang. DoT-Net: Document layout classification using texture-based CNN. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 1029–1034, 2019. 3

[32] J. Kumar, P. Ye, and D. Doermann. Learning document structure for retrieval and classification. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 1558–1561, 2012. 7, 48, 52

[33] O. Lawal. Tomato detection based on modified YOLOv3 framework. *Scientific Reports*, 11, 01 2021. 40

[34] V. P. Le, N. Nayef, M. Visani, J. Ogier, and C. D. Tran. Text and non-text segmentation based on connected component features. In *13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1096–1100, 2015. 21

[35] D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, and J. Heard. Building a test collection for complex document information processing. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development*

in Information Retrieval, SIGIR '06, page 665–666, New York, NY, USA, 2006. Association for Computing Machinery. 8, 48, 49, 57

[36] C. Li, B. Bi, M. Yan, W. Wang, S. Huang, F. Huang, and L. Si. StructuralLM: Structural pre-training for form understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6309–6318, Online, Aug. 2021. Association for Computational Linguistics. 35, 37

[37] S. Li, X. Ma, S. Pan, J. Hu, L. Shi, and Q. Wang. Vtlayout: Fusion of visual and text features for document layout analysis. In *Pacific Rim International Conference on Artificial Intelligence*, pages 308–322. Springer, 2021. 3

[38] J. Liang, J. Ha, R. M. Haralick, and I. T. Phillips. Document layout structure extraction using bounding boxes of different entitles. In *Proceedings Third IEEE Workshop on Applications of Computer Vision. WACV'96*, pages 278–283, 1996. 15

[39] X. Liang, A. Cheddad, and J. Hall. Comparative study of layout analysis of tabulated historical documents. *Big Data Research*, 24, 05 2021. 4, 16

[40] L. Liu, Z. Wang, T. Qiu, Q. Chen, Y. Lu, and Y. Suen. Document image classification: Progress over two decades. *Neurocomputing*, 453, 05 2021. 2

[41] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. 20

[42] P. H. Luz de Araujo, T. E. de Campos, F. Ataides Braz, and N. Correia da Silva. VICTOR: a dataset for Brazilian legal documents classification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1449–1458, Marseille, France, May 2020. European Language Resources Association. 49, 53, 64, 66, 67

[43] P. H. Luz de Araujo, T. E. de Campos, and M. Magalhaes Silva de Sousa. Inferring the source official texts: can SVM beat ULMFiT? In *International Conference on the Computational Processing of Portuguese (PROPOR)*, Lecture Notes on Computer Science (LNCS), Evora, Portugal, March 2-4 2020. Springer. Code and data available from `https://cic.unb.br/~teodecampos/KnEDLe/`. 49

[44] A. L. L. M. Maia, F. D. Julca-Aguilar, and N. S. T. Hirata. A machine learning approach for graph-based page segmentation. In *31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 424–431, 2018. 3

[45] S. Marinai. Introduction to document analysis and recognition. In *Machine Learning in Document Analysis and Recognition*, 2008. 21

[46] W. McNally, K. Vats, A. Wong, and J. McPhee. Rethinking keypoint representations: Modeling keypoints and poses as objects for multi-person human pose estimation. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision – ECCV 2022*, pages 37–54, Cham, 2022. Springer Nature Switzerland. 40

[47] S. Merity, N. Keskar, and R. Socher. Regularizing and optimizing lstm language models. In *International Conference on Learning Representations*, pages 1–13, 2018. 25, 45, 48

[48] H. Motahari, N. Duffy, P. Bennett, and T. Bedrax-Weiss. A report on the first workshop on document intelligence (di) at neurips 2019. *SIGKDD Explor. Newsl.*, 22(2):8–11, jan 2021. 1

[49] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 25(7):10–22, 1992. 15

[50] L. Noce, I. Gallo, A. Zamberletti, and A. Calefati. Embedded textual content for document image classification with convolutional neural networks. In *Proceedings of the 2016 ACM Symposium on Document Engineering*, DocEng '16, page 165–173, New York, NY, USA, 2016. 2, 3, 19, 52

[51] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, 1993. 16

[52] Y. Pan, Q. Zhao, and S. Kamata. Document layout analysis and reading order determination for a reading robot. In *TENCON 2010 - 2010 IEEE Region 10 Conference*, pages 1607–1612, 2010. 15

[53] A. Radford and K. Narasimhan. Improving language understanding by generative pre-training. 2018. 3, 20

[54] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 40

[55] D. Rothman. *Transformers for Natural Language Processing: Build Innovative Deep Neural Network Architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and More*. Packt Publishing, 2021. 26, 27

[56] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv e-prints*, page arXiv:1609.04747, Sept. 2016. 48

[57] A. K. Sah, S. Bhowmik, S. Malakar, R. Sarkar, E. Kavallieratou, and N. Vasilopoulos. Text and non-text recognition using modified hog descriptor. In *IEEE Calcutta Conference (CALCON)*, pages 64–68, 2017. 21

[58] B. M. Singh, K. K. Verma, et al. Noise removal technique for document images. *Journal of Multimedia Information System*, 10(1):1–14, 2023. 13

[59] R. Smith et al. Tesseract ocr engine. *Lecture. Google Code. Google Inc*, 2007. 40

[60] R. W. Smith. Hybrid page layout analysis via tab-stop detection. In *10th International Conference on Document Analysis and Recognition*, pages 241–245, 2009. 16

[61] M. Sundermeyer, R. Schlüter, and H. Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, pages –, 2012. 24, 45

[62] T. A. Tran, I. S. Na, and S. H. Kim. Page segmentation using minimum homogeneity algorithm and adaptive mathematical morphology. *Int. J. Doc. Anal. Recognit.*, 19(3):191–209, Sept. 2016. 16

[63] T. A. Tran, K. Nguyen-An, and N. Quang Vo. Document layout analysis: A maximum homogeneous region approach. In *1st International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, pages 1–5, 2018. 16

[64] R. van Heusden, J. Kamps, and M. Marx. Wooir: A new open page stream segmentation dataset. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '22, page 24–33, New York, NY, USA, 2022. Association for Computing Machinery. 4

[65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages –. Curran Associates, Inc., 2017. 3, 26, 27, 38

[66] F. M. Wahl, K. Y. Wong, and R. G. Casey. Block segmentation and text extraction in mixed text/image documents. *Computer Graphics and Image Processing*, 20(4):375–390, 1982. 15

[67] Y. Wang, I. T. Phillips, and R. M. Haralick. Document zone content classification and its performance evaluation. *Pattern Recognition*, 39(1):57–73, Jan. 2006. 18

[68] Z. Wang, Y. Xu, L. Cui, J. Shang, and F. Wei. LayoutReader: Pre-training of text and layout for reading order detection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4735–4744, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. 19, 20

[69] G. Wiedemann and G. Heyer. Multi-modal page stream segmentation with convolutional neural networks. *Lang. Resour. Eval.*, 55(1):127–150, mar 2021. 7, 8, 33, 34, 50, 55

[70] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document analysis system. *IBM Journal of Research and Development*, 26(6):647–656, 1982. 16

[71] T.-L. Wu, C. Li, M. Zhang, T. Chen, S. A. Hombaiah, and M. Bendersky. Lampret: Layout-aware multimodal pretraining for document understanding. *ArXiv*, abs/2104.08405, 2021. 6, 20

[72] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou. Layoutlm: Pre-training of text and layout for document image understanding. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Aug 2020. 2, 3, 4, 6, 18, 20, 29, 31, 35, 36, 37, 38, 50

[73] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che, M. Zhang, and L. Zhou. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online, Aug. 2021. Association for Computational Linguistics. 3, 6, 20, 31, 32, 36, 37, 50

[74] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen. A Survey of Large Language Models. *arXiv e-prints*, page arXiv:2303.18223, Mar. 2023. 68

[75] G. Zhu and D. Doermann. Automatic document logo detection. In *In Proc. 9th International Conf. Document Analysis and Recognition (ICDAR 2007)*, pages 864–868, 2007. 48

[76] G. Zhu, Y. Zheng, D. Doermann, and S. Jaeger. Multi-scale structural saliency for signature detection. In *In Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR 2007)*, pages 1–8, 2007. 48

[77] S. P. Zingaro, G. Lisanti, and M. Gabbrielli. Multimodal side- tuning for document classification. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5206–5213, 2021. 35, 37

[78] A. Zulfiqar, A. Ul-Hasan, and F. Shafait. Logical layout analysis using deep learning. In *2019 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–5, 2019. 3, 22