A PROJECT WORK REPORT

on

Invoice Processing Using Machine Learning

Submitted in partial fulfillment of the requirement for the project work

MASTER OF TECHNOLOGY

in

DATA SCIENCE

by

Meesala Lakshmanna Hannah 2467415

Under the Guidance of

Babu Kumar S
Assistant Professor

and

Dr. Xavier C
Professor

Department of Computer Science and Engineering
School of Engineering and Technology,
CHRIST (Deemed to be University),
Kumbalgodu, Bangalore - 560074
March-2025

School of Engineering and Technology
Department of Computer Science and Engineering

CERTIFICATE

Certified that the project work entitled **Invoice Processing Using Machine Learning** is the bonafide work carried out by **Meesala Lakshmanna Hannah 2467415** in partial fulfillment of the project work(MTDS281) of **M. Tech in Data Science** of **Department of Computer Science and Engineering at CHRIST(Deemed to be University), Bangalore** during academic year 2024-2025. It is certified that all the corrections/suggestions indicated for the internal assessment have been incorporated in the report to the department. The project work has been approved as it satisfies the academic requirements.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements

Signature of the Guide
Babu Kumar S
Assistant Professor
Department of CSE

Signature of the Co-guide
Dr. Xavier C
Professor
Department of CSE

Signature of HoD/Program Coordinator
Department of CSE.

Signature of Associate Dean
Dr. Mary Anita E A
School of Engineering and Technology

School of Engineering and Technology

Department of Computer Science and Engineering

BONAFIDE CERTIFICATE

It is to certify that this Project work titled" Invoice Processing Using Machine Learning " is the bonafide work of

Name	**Register Number**
Meesala Lakshmanna Hannah	2467415

Examiners [Name and Signature]

1.

2.

Name of the Candidate:

Register Number:

Date of Examination:

ACKNOWLEDGEMENT

I would like to thank CHRIST (Deemed to be University) Vice Chancellor, **Dr Fr Joseph C C**, Pro Vice Chancellor, **Dr Fr Viju P D**, Director of School of Engineering and Technology, **Dr Fr Sony J Chundattu** and the Associate Dean **Dr. Mary Anita E A** for their kind patronage

I would like to express my sincere gratitude and appreciation to the Head of the Department of Computer Science and Engineering, School of Engineering and Technology, **Dr Balamurugan M**, for giving me this opportunity to take up this project.

I also take this opportunity to express a deep sense of gratitude to **Dr Manohar M**, M. Tech Program Coordinator and Project Coordinator, Department of Computer Science and Engineering, School of Engineering and Technology for their timely support and valuable guidance.

I am also extremely grateful to my Supervisor , **Professor Babu Kumar S,** who has supported and helped to carry out the project. His constant monitoring and encouragement helped me keep up to the project schedule.

I am also extremely grateful to my Mentor, **Professor Dr Xavier C**, who has supported and helped to carry out the project. His constant monitoring and encouragement helped me keep up with the project schedule.

Meesala Lakshmanna Hannah

ABSTRACT

The study aimed to automate invoice processing using LayoutLM and LayoutLMv2 to enhance accuracy, efficiency, and scalability in structured data extraction. Traditional invoice processing methods relied on manual effort, leading to inefficiencies and errors in handling large volumes of invoices with diverse formats. To address these challenges, the research utilized deep learning-based document understanding models trained on the FUNSD dataset to extract key invoice fields such as Invoice Number, Date, Buyer and Seller Details, Total Amount, and Tax Information. The models processed invoice data using OCR techniques to convert scanned documents into machine-readable text while preserving spatial relationships. The extracted entities were structured using the BIEO (Beginning, Inside, End, Outside) tagging format, which facilitated precise key-value pair recognition.

The developed pipeline automated invoice extraction by first performing OCR to extract textual content, followed by LayoutLM-based entity classification and tagging. The structured information was then validated against business rules to ensure accuracy. The integration of multimodal embeddings in LayoutLMv2 improved entity recognition by leveraging spatial, textual, and visual features of invoices. The system successfully reduced processing time and minimized errors in invoice data extraction. The use of transformer-based models enabled better generalization across varying invoice formats, making the approach suitable for real-world applications.

Despite the improvements achieved, challenges remained, particularly in handling unstructured invoices, OCR misinterpretations in low-quality scans, and variations in invoice layouts. The study identified the need for further enhancements in OCR accuracy, dataset diversity, and model adaptability to improve generalization across industries. Future work was suggested to explore more advanced transformer-based architectures and fine-tuning techniques to address these limitations.

The research demonstrated that LayoutLM-based models significantly improved invoice processing automation by reducing manual intervention, ensuring higher accuracy, and making the system scalable for large datasets. The findings contributed to the field of document intelligence and business process automation, providing a structured and efficient solution for invoice management.

LIST OF FIGURES

NOMENCLATURE

Term	**Description**	**Formula**
CER (Character Error Rate)	Measures the percentage of incorrectly recognized characters in the extracted text compared to the ground truth.	CER = (S + D + I)/N Where: **S** = Substitutions **D** = Deletions **I** = Insertions **N** = Total characters in the ground truth
WER(Word Error Rate)	Measures the percentage of incorrectly recognized words in the extracted text compared to the ground truth.	WER = (S + D + I)/W Where: **S** = Substitutions **D** = Deletions **I** = Insertions **W** = Total words in the ground truth

LIST OF TABLES

CONTENTS		
		Page No
Certificate		2
Acknowledgement		4
Abstract		5
List of Figures		6
List of tables		8
Nomenclature		7
Chapter 1: INTRODUCTION		
	1.1 Overview of the project domain	10
	1.2 Motivation and Problem Statement	11
	1.3 Objectives	12
	1.4 Scope of the Project	12
Chapter 2: LITERATURE SURVEY		
	2.1 Literature Review	13
	2.2 Research gaps	20
Chapter 3: PROBLEM FORMULATION AND PROPOSED WORK		
	3.1 Introduction	21
	3.2 Problem Statement	22
	3.3 Objectives	22
Chapter 4: METHODOLOGY		
	4.1 Introduction	23
	4.2 Implementation Strategy	25
	4.3 Tools/Hardware/Software used	31
	4.4 Expected Outcome	31
Chapter 5: Result and Discussion		
	5.1 Implementation Details	33
	5.2 Results	40
	5.3 Discussions	45
Chapter 6: Conclusions and future scope		53
References		54

CHAPTER 1

INTRODUCTION

1.1 Overview of the project domain

This project is based on Invoice processing using LayoutLM and LayoutLMv2. Invoices are a pivotal part of business transactions and contain structured data such as Order number, Invoice Identification Number, product descriptions, Tax, and total amount. Processing an Invoice manually is time-consuming, may contain many errors and is difficult to do in large-scale operations.

The advanced algorithms in Natural Language processing(NLP), Computer vision (CV), and Machine learning models have effective solutions for Invoice Document Processing. The Deep Learning model LayoutLM and its improved version LayoutLMv2 integrate textual, spatial and visual features, enabling better understanding of document structure. In Optical Character Recognition (OCR) it extracts text but doesn't contain any relationship between extracted elements. But in the case of LayoutLMv2 the model understands the relationship between text elements and their layouts within Invoice Documents, which leads to the improvement of accuracy in key value extraction and table recognition. The natural language models are based on words and their patterns. The Natural Language Processing just deals with the textual part of the document. , but the deep learning models LayoutLMv2 learn the layout of the models and the relationships between words.

This Project focuses on Invoice processing using LayoutLM and LayoutLMv2, which are deep learning models specifically used for Visually rich Document Understanding(VrDU). These models effectively address the challenges of layout variations, text misalignment and OCR errors by combining layout, text and image embeddings to extract structured data effectively from invoice documents.

1.2 Motivation and Problem Statement

Motivation

Businesses are increasingly transitioning to digital invoicing to streamline financial operations. However, manual data entry remains a persistent challenge, leading to inefficiencies, errors, and increased processing time. As invoice volumes grow, automation becomes crucial for improving accuracy and reducing operational costs. Traditional Optical Character Recognition (OCR) technology offers a partial solution by extracting text from invoices. However, OCR struggles with diverse invoice formats, unstructured layouts, varying fonts, and poor image quality. These challenges often result in incorrect data extraction, requiring extensive manual corrections that negate the benefits of automation.

To overcome these limitations, advanced AI-driven approaches such as LayoutLM and LayoutLMv2 provide a more robust solution for invoice data extraction. Unlike traditional OCR, these models integrate textual, spatial, and visual information, allowing them to understand the structure and relationships within an invoice. By recognizing patterns and adapting to different layouts, LayoutLM models significantly enhance data extraction accuracy. By leveraging these AI techniques, businesses can minimize manual intervention, reduce processing errors, and achieve greater efficiency in invoice automation. This project focuses on utilizing LayoutLM and LayoutLMv2 to extract structured data from invoices, ensuring higher accuracy and improved adaptability across various invoice templates. The integration of these AI models offers a scalable and reliable solution that surpasses OCR's limitations, paving the way for more efficient financial workflows and automation-driven business processes.

Problem Statement

The effectiveness of OCR diminishes when processing invoices with diverse layouts, font styles, and alignments. These variations make it challenging for OCR engines to accurately extract relevant information. Additionally, poor-quality scans further reduce accuracy, often necessitating manual corrections. This contradicts the goal of automation, as OCR fails to consistently extract structured data. In practical scenarios, OCR-based methods frequently misclassify critical invoice elements such as invoice numbers, dates, and total amounts. The reliance on manual intervention to rectify errors slows financial processes and increases the likelihood of inaccuracies, ultimately reducing operational efficiency.

1.3 Objectives

1. Extract structured data from invoices using LayoutLM and LayoutLMv2.
2. Compare the effectiveness of LayoutLM and LayoutLMv2 in handling structured invoice data.
3. Enhance invoice data extraction accuracy by leveraging AI-driven approaches.
4. Evaluate the performance of both models based on accuracy, efficiency, and adaptability across various invoice templates.
5. Reduce manual intervention in invoice processing by improving automation capabilities.
6. Demonstrate how AI models surpass traditional OCR methods in structured data extraction.
7. Provide insights into the best-suited model for invoice automation based on real-world invoice formats.

1.4 Scope of the Project

To improve accuracy and efficiency over conventional OCR techniques, this project focuses on automating invoice data extraction using LayoutLM and LayoutLMv2. To determine how well these models handle different structures, layouts, and font variations, they must be trained and evaluated on a variety of invoice templates. Comparing their performance in terms of precision, flexibility, and effectiveness in structured data extraction is the main goal.

By leveraging AI-driven techniques, this study aims to address the limitations of OCR, which often struggles with unstructured layouts and inconsistent formatting. The comparison between LayoutLM and LayoutLMv2 will provide insights into their effectiveness in real-world invoice processing. The project is strictly limited to invoices and does not cover other document types, such as receipts or contracts.

The findings will contribute to the optimization of AI-based invoice automation, reducing the need for manual intervention and improving operational efficiency. Businesses in finance, accounting, and procurement can benefit from the improved accuracy and scalability offered by these advanced models, leading to streamlined workflows and cost savings.

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Review

Xu et al.[1] present LayoutLM, a model that combines textual and layout information to improve document image understanding, in their work "LayoutLM: Pre-training of Text and Layout for Document Image Understanding." LayoutLM achieves state-of-the-art outcomes in applications, including form comprehension, receipt understanding, and document image classification, by modeling text and layout together. There are still certain research gaps despite these developments. The model's applicability in situations without such resources may be limited by its reliance on extensive annotated data for pre-training. Furthermore, LayoutLM may perform less well on tasks where visual cues are essential since it only integrates text and layout; it does not completely integrate visual elements from document photos. Creating strategies to lessen reliance on data and improving the model's utilization of visual data could be two ways to close these gaps.

The study by Baviskar et al. (2021)[2] introduced a multi-layout unstructured invoice dataset aimed at enabling template-free invoice processing. The dataset comprises 630 invoice PDFs with four distinct layouts from different suppliers, making it a valuable resource for training AI models in key field extraction. To ensure its reliability, the dataset underwent statistical validation, enhancing its credibility for AI-based invoice automation. Additionally, the study evaluated various feature extraction techniques, including GloVe, Word2Vec, and FastText, alongside AI models such as BiLSTM and BiLSTM-CRF. Among these, the BiLSTM-CRF model demonstrated superior performance in extracting key invoice fields, proving effective in handling diverse layouts.

Krieger et al. (2023)[3] investigated machine learning-based information extraction for long-tail suppliers—suppliers that provide invoices infrequently and in diverse formats. The study highlighted the impact of training data distribution on model performance, as models trained primarily on frequent suppliers' invoices exhibited lower accuracy when processing long-tail invoices. Their findings demonstrated that machine learning models can generalize well across varying invoice structures, but key limitations remain. The study identified the need for more diverse training datasets encompassing invoices from different suppliers to improve model robustness. Additionally, while several machine learning models were assessed, the research suggested that exploring transformer-based models, such as LayoutLM, could further enhance extraction capabilities. Another gap identified was the limited integration of visual features, such as logos and formatting cues, which could improve accuracy in invoice data extraction.

Saout et al. (2024)[4] provide a comprehensive overview of automated invoice data extraction, emphasizing the integration of multiple techniques to enhance accuracy. The study highlights the role of digitalization and Natural Language Processing (NLP) in extracting meaningful information from invoices, demonstrating how these methods improve text interpretation and processing. Additionally, it reviews machine learning and deep learning approaches that handle diverse invoice layouts, reducing manual intervention and improving adaptability. However, the research identifies key gaps in the integration of various techniques for seamless automation, suggesting the need for cohesive frameworks that effectively combine different methods. Furthermore, the study acknowledges the challenge of invoice layout variability and the necessity for models that can generalize across different formats without requiring extensive retraining. Addressing these gaps would enhance the reliability and scalability of automated invoice processing systems, making them more adaptable to real-world applications.

To overcome the drawbacks of human data extraction, such as the absence of standardized datasets and the variety of invoice forms, Chazhoor et al. (2022)[5] looked into automating invoice parsing with computer vision techniques. They produced a dataset of 315 invoices that were annotated for eight different entities: product name, price, quantity, total amount, billing address, shipping address, invoice date, and invoice number. The study employed Optical Character Recognition (OCR) for text extraction following an evaluation of advanced object detection algorithms, including YOLO, SSD, and R-CNN, to identify key entities. To enhance model performance, hyperparameter tuning was conducted, and the models were assessed using metrics such as mean Average Precision (mAP) and COCO evaluation metrics. However, the limited dataset size posed a challenge, potentially restricting the generalizability of the findings. While the models exhibited promising results, further research is necessary to validate their effectiveness on larger and more diverse datasets to ensure their reliability in real-world scenarios.

Arslan et al. [6] proposed an automatic invoice processing system designed to handle various invoice file types, allowing companies to submit invoices via a web interface or email. The system distinguishes between text files and images: for text files, it employs template matching to extract information, while for images, it detects and extracts text and table areas using both image processing techniques and a YOLOv5-based deep learning method. Subsequently, Optical Character Recognition (OCR) is performed using Tesseract. Recognizing the limitations of existing OCR models for Turkish invoices, the study fine-tuned a new model trained specifically on Turkish invoices, resulting in improved accuracy over the default English and Turkish models provided by Tesseract. The experimental results demonstrated that the YOLOv5-based table detection model outperformed traditional image-processing methods. However, the study's reliance on a specific dataset may limit the generalizability of its findings. Future research should focus on

validating the system's effectiveness across diverse invoice formats and languages to ensure robustness in various real-world applications.

Baviskar et al. (2021) [7] conducted a systematic literature review focusing on the application of artificial intelligence (AI) techniques for processing unstructured documents. Their findings indicate that AI-based approaches hold significant potential in automatically extracting valuable information from such documents. However, the study also identifies challenges, particularly in handling documents with multiple layouts, which can hinder the efficiency of AI applications. The authors emphasize the necessity for further research to develop more robust AI models capable of effectively managing the variability inherent in unstructured documents, thereby enhancing the automation process.

Perin et al. (2024)[8] introduced DynGraph-BERT, a novel semi-supervised text classification approach that synergistically combines Bidirectional Encoder Representations from Transformers (BERT) with Graph Neural Networks (GNNs) through dynamic graph structures. This method integrates text augmentation, dynamic graph construction, and label propagation during inference to enhance classification performance. Notably, DynGraph-BERT constructs homogeneous graphs using BERT embeddings, allowing for adaptability and improved accuracy across various datasets. Experimental evaluations demonstrated that DynGraph-BERT outperforms existing semi-supervised text classification models, achieving consistent accuracy improvements as more labeled data becomes available. However, the study primarily focuses on the integration of BERT and GNNs using dynamic graphs, suggesting future research could explore the application of this approach to other domains and the incorporation of additional linguistic features to further enhance model performance.

Devika et al. (2021)[9] proposed the Semkey-BERT model, a deep learning approach that integrates Bidirectional Encoder Representations from Transformers (BERT) with sentence transformers to enhance semantic keyphrase extraction from large-scale social media data. This model leverages BERT's capability to maintain semantic and syntactic relationships between tweets, facilitating the automatic extraction of representative phrases from Twitter content. The Semkey-BERT model demonstrated an accuracy of 86%, surpassing existing models in performance. However, the study did not extensively address the challenges associated with processing noisy and informal language prevalent in social media platforms, which could impact the model's robustness. Future research should focus on refining the model to better handle such linguistic variability, thereby enhancing its applicability across diverse social media contexts.

Giarelis et al. (2024)[10] conducted a comprehensive literature review on deep learning and embedding-based methods for keyphrase extraction, a critical task in natural language processing that involves automatically identifying terms encapsulating a document's main themes. Their analysis highlighted the efficacy of deep learning models, particularly those

leveraging embeddings, in capturing semantic nuances and contextual information, thereby enhancing keyphrase extraction accuracy. However, they also identified challenges, such as the need for extensive labeled datasets and the complexity of models, which can impede practical implementation. The study suggests that future research should focus on developing more efficient models that require less annotated data and can generalize across diverse domains.

Gon et al. (2024)[11] explored the application of Bidirectional Encoder Representations from Transformers (BERT) for text classification tasks within Natural Language Processing (NLP). Their study delineated a comprehensive methodology encompassing pre-processing strategies, tokenization using BERT's tokenizer, and the integration of special tokens like [CLS] and [SEP] to enhance model understanding. They utilized the pre-trained BERT model to obtain contextual embeddings for each token in the text, followed by pooling techniques, such as mean or max pooling, to derive fixed-size representations of input sequences. A classification layer was then employed to perform specific tasks, including sentiment analysis and spam detection. The model underwent fine-tuning on labeled datasets, involving the adjustment of weights in both the classification layer and selected BERT layers to better adapt to specific tasks. The study highlighted the efficacy of BERT in capturing contextual nuances, thereby improving classification accuracy. However, the authors acknowledged challenges related to the need for substantial amounts of labeled data for effective transfer learning and domain adaptation. Future research directions include optimizing BERT's performance in resource-constrained environments and exploring methods to reduce the dependency on large labeled datasets.

Qiu et al. (2024)[12] proposed a novel approach for document image layout detection in scientific literature by combining ConvNext and Cascade Mask R-CNN networks. Their study aimed to enhance the accuracy of detecting and segmenting different layout components, such as text blocks, figures, tables, and equations in complex document images. By leveraging the strengths of ConvNext in feature extraction and the multi-stage refinement of Cascade Mask R-CNN, the model demonstrated superior performance in handling diverse document layouts with varying structures and noise levels. The evaluation, conducted on publicly available scientific literature datasets, showed that the proposed approach outperformed existing state-of-the-art models in terms of precision and recall. However, the study acknowledged certain limitations, including the computational complexity of the combined architecture, which may hinder real-time processing for large-scale document repositories. Additionally, the generalizability of the model to non-scientific documents remains an open research question. Future work could focus on optimizing the model for efficiency and exploring domain adaptation techniques to extend its applicability to other document types beyond scientific literature.

Kiatphaisansophon et al. (2024)[13] introduced an efficient text bounding box identification method for Thai documents using Mask R-CNN. Their approach aimed to enhance text

localization accuracy in complex layouts, addressing challenges such as overlapping characters, diverse font styles, and noisy document backgrounds. The study demonstrated that Mask R-CNN effectively segmented text regions with high precision, outperforming traditional OCR-based methods in handling Thai script, which often contains intricate character connections and varying spacing. The evaluation on a Thai document dataset showed promising results, improving both recall and F1 scores compared to existing techniques. However, the research highlighted certain limitations, including the need for a large annotated dataset for robust model training and the high computational cost of Mask R-CNN, which may not be optimal for real-time applications. Additionally, while the model performed well on structured documents, its effectiveness on handwritten or highly degraded documents remains uncertain. Future research directions could focus on optimizing the model for faster inference, integrating lightweight architectures, and expanding its applicability to multilingual text detection in diverse document types.

Xu et al. (2020)[14] introduced LayoutLMv2, an advanced multi-modal pre-trained model designed for visually rich document understanding by integrating text, spatial, and image information. The study demonstrated that LayoutLMv2 outperformed its predecessor, LayoutLM, by incorporating a more effective cross-modal interaction mechanism, enabling better comprehension of complex document layouts such as invoices, forms, and receipts. The model was evaluated on benchmark datasets like FUNSD, CORD, and SROIE, where it achieved state-of-the-art performance in key information extraction tasks. Key findings highlighted its superior ability to process diverse layouts, improving structured data extraction accuracy compared to traditional OCR and rule-based methods. However, despite these advancements, the study identified gaps, such as the need for extensive labeled datasets for fine-tuning and the computational cost associated with multi-modal training, which may limit deployment in real-time applications. Additionally, while LayoutLMv2 performed well on structured documents, its adaptability to highly unstructured or handwritten documents remains an open challenge. Future research could focus on optimizing the model for low-resource environments, reducing dependency on large-scale annotated data, and extending its applicability to multilingual document processing.

Appalaraju et al. (2021)[15] introduced DocFormer, an end-to-end Transformer-based model for document understanding that integrates text, layout, and visual features. Unlike traditional OCR-based systems that process text independently, DocFormer utilizes a unified architecture that effectively captures the relationships between textual and spatial information, improving key information extraction from complex documents such as invoices and forms. The study demonstrated that DocFormer outperformed prior models, including LayoutLM, on benchmark datasets like FUNSD, CORD, and SROIE, achieving higher accuracy in structured data extraction tasks. One key finding was its ability to generalize across different document layouts without requiring extensive template-specific fine-tuning. However, the study also identified several research gaps. The model's reliance on large-scale annotated datasets for

pre-training remains a challenge, as data scarcity can hinder real-world deployment. Additionally, DocFormer exhibits high computational complexity, making it resource-intensive for real-time applications. While the model performed well on printed documents, its effectiveness on handwritten or low-quality scanned documents remains an area for further exploration. Future research could focus on optimizing computational efficiency, enhancing adaptability to diverse document types, and developing semi-supervised or unsupervised learning approaches to mitigate the dependency on labeled data.

Wang et al. (2022)[16] introduced a benchmark for structured data extraction from complex documents, addressing challenges posed by diverse layouts, multi-column structures, and varying font styles. The study presented a large-scale dataset designed to evaluate AI models' ability to extract key information from unstructured and semi-structured documents. The benchmark tested state-of-the-art methods, including LayoutLMv2, Tesseract OCR, and Transformer-based architectures, revealing that existing models still struggle with irregular layouts, noisy document images, and overlapping text regions. The findings demonstrated that while models like LayoutLMv2 performed well on structured documents, they faced difficulties in accurately identifying and categorizing data in complex layouts. A key limitation highlighted in the study was the insufficient generalizability of models trained on a limited set of document formats, leading to performance degradation when applied to unseen templates. Additionally, the research emphasized the need for improved multimodal architectures that integrate text, visual, and spatial features more effectively. Future directions suggested exploring semi-supervised learning and graph-based approaches to enhance structured data extraction while reducing reliance on extensively labeled datasets.

Oussaid et al. (2021)[17] explored information extraction from visually rich documents by incorporating font style embeddings to enhance document understanding. The study highlighted that traditional OCR-based techniques struggle with documents containing varied font styles, sizes, and formatting, leading to inaccuracies in extracting structured information. To address this, the authors proposed integrating font style embeddings alongside text and layout features to improve the model's ability to differentiate between key textual elements. Experiments on visually diverse datasets demonstrated that incorporating font attributes significantly improved extraction accuracy, particularly for hierarchical document structures. The study compared performance with models like LayoutLM and Transformer-based architectures, revealing that font-aware embeddings enhanced text classification and entity recognition. However, a notable limitation was the reliance on high-quality scanned documents, as noise and distortions in lower-quality images reduced model performance. Additionally, while the approach improved information extraction, generalisation to unseen document layouts remained a challenge. The authors suggested future research on multi-modal learning techniques, integrating font embeddings with vision-language models to achieve more robust document parsing across diverse formats.

Graliński et al. (2020)[18] introduced Kleister, a novel benchmark for information extraction from long documents with complex layouts, addressing challenges in structured data extraction from multi-page documents. The study emphasized that existing datasets primarily focus on short documents with simple structures, limiting the generalization of AI models to real-world financial, legal, and administrative documents. The Kleister dataset included long documents with intricate layouts, requiring models to process text continuity across pages, multi-column structures, and nested information. The authors evaluated state-of-the-art Transformer-based models, such as BERT and LayoutLM, and found that while these models performed well on structured forms, they struggled with contextual dependencies across multiple pages. The study demonstrated that pre-training on layout-aware features improved model performance but highlighted a gap in effectively handling discontinuous text flows and document-length constraints. The research also pointed out limitations in current evaluation metrics, which fail to fully capture model performance on long and complex document structures. The authors suggested future work on multi-modal learning approaches that integrate visual, textual, and hierarchical document context to enhance long-document understanding in AI-driven information extraction systems.

Stanisławek et al. (2021)[19] expanded on the Kleister benchmark by introducing new datasets specifically designed for key information extraction (KIE) from long, complex-layout documents. The study highlighted that most existing datasets focus on short, structured forms, making them insufficient for real-world scenarios involving multi-page financial, legal, and administrative documents. The authors developed datasets that included varied document layouts, multiple columns, and cross-page dependencies, posing significant challenges for traditional OCR-based and Transformer-based models. They evaluated state-of-the-art models such as BERT, RoBERTa, and LayoutLM, finding that while layout-aware models like LayoutLM performed better than text-only approaches, they still struggled with text continuity across pages, hierarchical structures, and low-resource document types. The study identified key research gaps, including the need for improved multi-modal learning techniques that integrate visual, textual, and hierarchical document context to enhance long-document understanding. Additionally, the authors emphasized the limitations of existing evaluation metrics, which fail to capture model performance on complex multi-page document structures, suggesting the development of more comprehensive benchmarks for structured document processing.

Salgado et al.(2023)[20] explored information extraction from electricity invoices using Named Entity Recognition (NER) with Transformer models. The study aimed to improve structured data extraction by identifying key entities such as billing amounts, invoice dates, customer details, and consumption data. The authors experimented with pre-trained Transformer models, including BERT and RoBERTa, fine-tuning them on a dataset of electricity invoices. Their findings showed that Transformer-based NER models outperformed traditional OCR-based approaches, particularly in handling unstructured invoice layouts and

variations in text formatting. However, the study also highlighted challenges such as the misclassification of similar entity types, difficulty in handling out-of-vocabulary terms, and performance degradation on invoices with low-quality scans. A key research gap identified was the limited availability of labeled training data for domain-specific invoice processing, which restricted the model's generalizability. Additionally, the authors emphasized the need for hybrid approaches that combine rule-based methods with deep learning models to enhance extraction accuracy in real-world invoice automation scenarios.

2.2 Research gaps

1. Limited Generalizability of Models Across Diverse Invoice Layouts. Despite advancements in Transformer-based models (e.g., LayoutLM, DocFormer), many approaches still struggle with template-free invoice processing across multiple formats and languages. Studies like Baviskar et al. (2021) [2] and Saout et al. (2024)[4] highlight the need for models that can generalize across diverse invoice layouts without requiring extensive fine-tuning.
2. Inadequate Handling of Low-Quality and Noisy Documents. Several studies, including Qiu et al. (2024) [12] and Kiatphaisansophon et al. (2024)[13] , focus on document layout detection and text bounding box identification; however, challenges remain in addressing low-resolution scans, handwritten annotations, and distortions. To enhance extraction accuracy from degraded invoices, more robust preprocessing techniques and noise-resistant deep learning architectures are required.
3. Lack of Effective Multi-Modal Fusion for Visually-Rich Documents. While LayoutLMv2 (Xu et al., 2020)[14] and DocFormer (Appalaraju et al., 2021) [15] attempt to integrate text, vision, and layout information, they still lack efficient fusion mechanisms that fully leverage spatial and semantic relationships in complex invoices. Future research should focus on enhancing multi-modal attention mechanisms to improve structured data extraction.
4. Limited Adaptability to Long-Tail Suppliers and Unseen Invoice Structures. Krieger et al. (2023)[3] and Wang et al. (2022)[16] emphasize that current models struggle with invoices from small or long-tail suppliers, where structures and terminologies vary significantly. Developing few-shot or zero-shot learning techniques using self-supervised pretraining could help improve adaptability to unseen invoice templates.
5. Insufficient Benchmark Datasets for Domain-Specific Invoice Processing. Although datasets like Kleister (Stanisławek et al., 2021) [19] provide benchmarks for document extraction, there is a lack of diverse, annotated datasets tailored for financial documents, especially in multilingual settings. Future work should focus on creating comprehensive datasets that include global invoice variations and multi-language support to improve model robustness.

CHAPTER 3

PROBLEM FORMULATION AND PROPOSED WORK

3.1 Introduction

Automation now depends extensively on document processing, especially in sectors that deal with financial accounts, forms, bills, and receipts. To improve decision-making, expedite workflows, and increase operational efficiency, organizations that handle enormous volumes of documents need effective techniques for extracting structured data. Text extraction has long made use of traditional optical character recognition (OCR) techniques, but these methods frequently produce less-than-ideal results when dealing with intricate document layouts, tabular structures, and key-value connections. Research into deep learning-based models, especially transformer topologies like LayoutLMV2, has been prompted by the need for a sophisticated, layout-aware document processing method.

Invoices, in particular, present unique challenges in document processing. They come in various formats, contain multiple structured fields such as invoice numbers, buyer and seller details, itemized tables, and total amounts, and often include variations in fonts, logos, and alignments. Rule-based and template-driven approaches have traditionally been used to extract structured data, but these methods lack the flexibility to handle diverse invoice formats effectively. Moreover, simple OCR systems, while capable of extracting raw text, fail to recognize the spatial relationships between entities, which are essential for accurate data extraction. This limitation results in errors such as misaligned text, missing values, and incorrect associations between fields.

LayoutLMV2, a multi-modal transformer-based model, offers a significant improvement in document understanding by integrating textual, spatial, and visual information. Unlike conventional OCR-based approaches, LayoutLMV2 processes both the textual content and the layout structure, enabling it to understand documents as humans do. The model is pre-trained on large datasets of visually rich documents, allowing it to learn the relationships between different elements in a document. It leverages spatial embeddings to encode positional information, which enhances its ability to identify structured fields, table structures, and key-value pair relationships with high accuracy.

The primary objective of this project is to build and optimize LayoutLMV2 to extract structured data from invoices. The primary goal is to increase the accuracy of key-value pair extraction by utilizing visual and spatial clues. The implications of feature extraction optimization, training on IOB-annotated datasets, and comparing the model's performance to traditional OCR-based methods are also examined in this study.

3.2 Problem Statement

Extracting structured data from invoices presents significant challenges due to their highly variable layouts, unstructured text, and complex table structures. Unlike standardized documents, invoices come in diverse formats depending on the vendor, industry, and geographical location. This variability makes it difficult to design a one-size-fits-all approach for data extraction. Traditional Optical Character Recognition (OCR) techniques, while effective for extracting raw text, often struggle with misaligned text, overlapping elements, and multi-column layouts. These issues lead to inaccurate extractions, particularly when invoices contain handwritten notes, stamps, or non-standard fonts. Rule-based extraction methods further complicate the process as they require extensive customization for each invoice template, making them inefficient and difficult to scale.

A major challenge in invoice processing is the identification of key-value pairs, such as "Invoice Number" and its corresponding value. Since these pairs do not always follow a fixed structure or predefined position, simple text recognition methods fail to capture their spatial relationships. Many invoices also contain tables with multiple rows and columns, where extracting the correct values requires an understanding of both content and layout. Conventional OCR-based approaches lack this spatial awareness, leading to misinterpretation of data, particularly in scenarios where table headers, numerical values, and text fields are closely packed. Beyond accuracy, scalability is another critical issue. Businesses and financial institutions process large volumes of invoices daily, making manual validation infeasible. An automated system must be robust enough to handle thousands of invoices with minimal human intervention while ensuring high accuracy. Any inefficiencies in extraction can lead to financial discrepancies, compliance issues, and delays in payment processing. To address these challenges, this project explores the use of LayoutLMV2, a transformer-based model designed for document processing. By leveraging both textual and visual information, LayoutLMV2 enhances key-value pair extraction, improving accuracy and minimizing manual effort.

3.3 Objectives

The objectives of this study are:

1. To implement and fine-tune LayoutLMV2 for structured data extraction from invoices.
2. To preprocess and annotate invoice datasets in an BIEO format suitable for model training.
3. To evaluate the performance of LayoutLMV2 on extracting key invoice details such as seller, buyer, invoice ID, and total amount.
4. To compare results with traditional OCR-based extraction to determine improvements in accuracy.

CHAPTER 4

METHODOLOGY

4.1 Introduction

The rapid growth of digital transactions has increased the demand for automated document processing, particularly for invoices, receipts, and other business documents. Extracting structured information from invoices is a challenging task due to their complex layouts, variations in formatting, and multi-modal nature. Traditional Optical Character Recognition (OCR) techniques often fail to capture key-value relationships, tables, and spatial structures effectively, leading to data extraction errors and inefficiencies.

To address these challenges, deep learning-based models such as LayoutLMV2 have been developed. LayoutLMV2 is a transformer-based model that leverages textual, spatial, and visual features to enhance document understanding. Unlike traditional OCR-based approaches, which rely on predefined rules and template-matching techniques, LayoutLMV2 generalizes across different invoice templates without requiring extensive manual intervention. By integrating visual and positional information with textual data, the model can effectively extract key-value pairs, table structures, and other crucial elements from invoices.

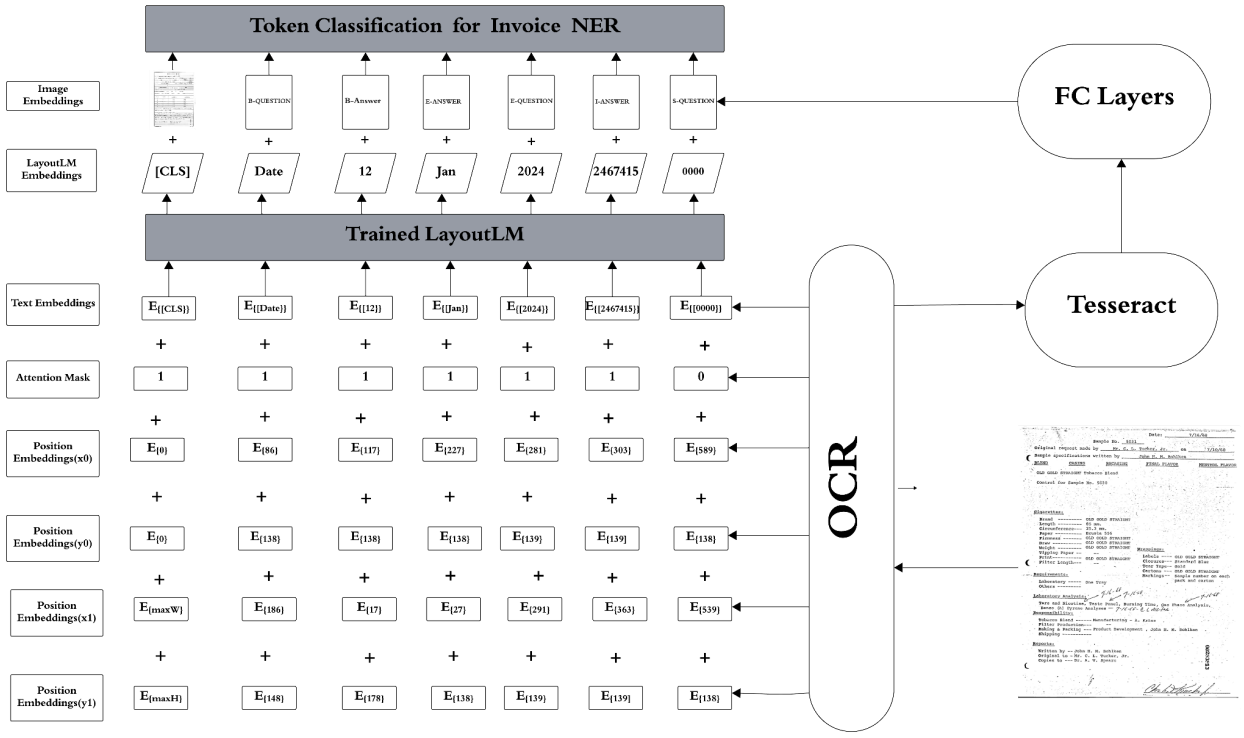


Figure 1: LayoutLM Architecture

The proposed methodology focuses on implementing LayoutLMV2 for automated invoice processing. The process begins with data preprocessing, where invoices are converted into image formats if necessary, and OCR is applied to extract raw text and bounding box coordinates. The extracted data is then structured into a format suitable for LayoutLMV2 input. Feature extraction is performed by leveraging token embeddings, bounding box information, and visual features. The training phase involves fine-tuning LayoutLMV2 on a labeled invoice dataset to optimize its performance for key-value pair extraction. Finally, the prediction and post-processing phase ensures that the extracted data is structured and formatted for integration into business workflows.

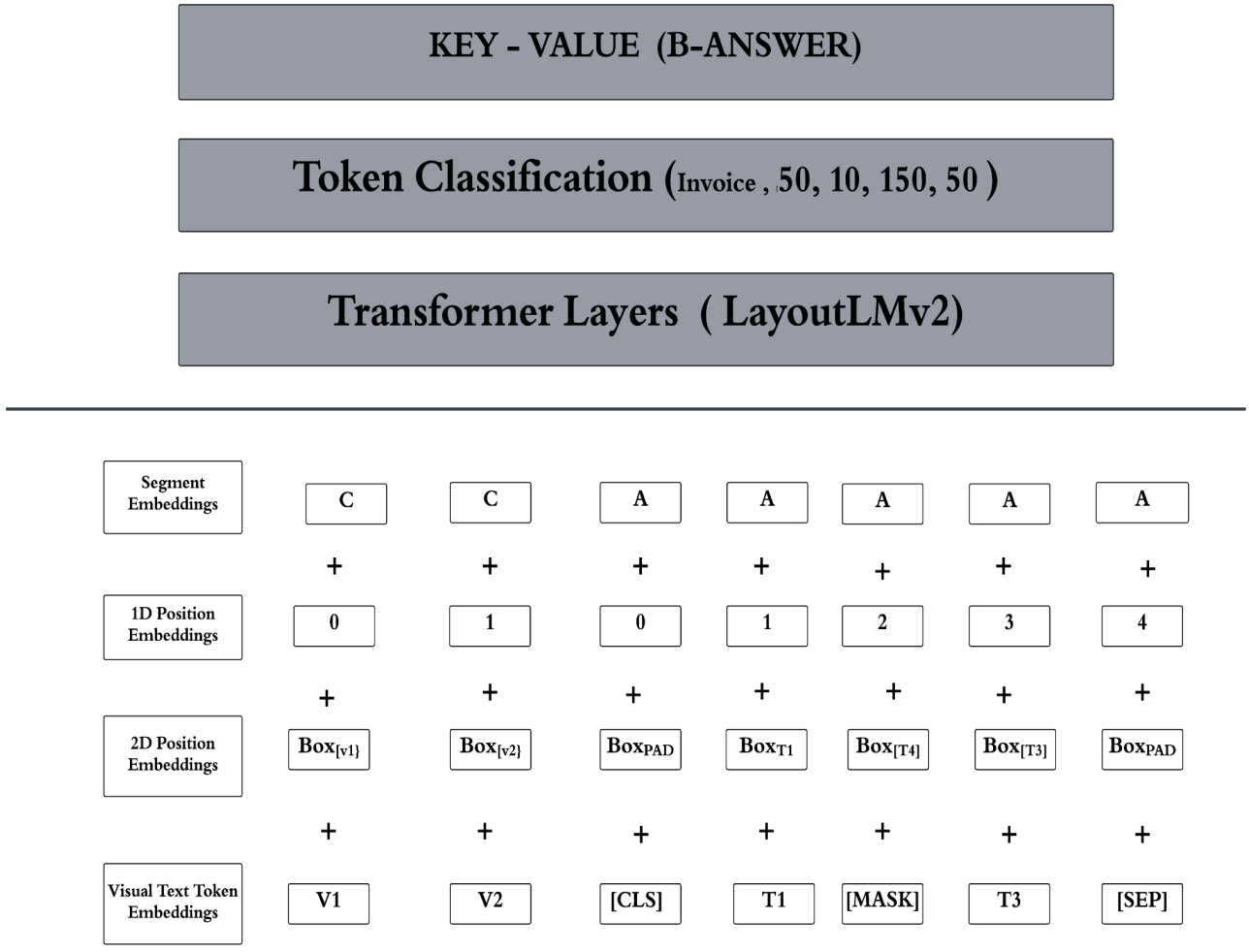


Figure 2: LayoutLMv2 Architecture

The primary objective of this methodology is to enhance the accuracy and efficiency of invoice data extraction by leveraging advanced deep learning techniques. The project aims to minimize manual data entry efforts, reduce errors, and streamline invoice processing workflows. LayoutLMV2 can significantly outperform traditional rule-based and OCR-only methods by

incorporating spatial and visual cues. The methodology also includes performance evaluation using standard metrics such as Precision, Recall, and F1-score to assess the approach's effectiveness.

Furthermore, the proposed implementation ensures scalability, allowing the system to process invoices in bulk with minimal human intervention. The use of a GPU-enabled infrastructure accelerates training and inference, making it feasible for real-time or batch processing applications. The methodology is adaptable and can be extended to process other document types, such as receipts, purchase orders, and financial statements.

In summary, this chapter presents a systematic approach to implementing LayoutLMV2 for automated invoice processing. It details the preprocessing steps, feature extraction methods, model training strategies, and evaluation metrics used to assess performance. The integration of deep learning and multi-modal document understanding techniques marks a significant advancement in the field of intelligent document processing. The proposed approach is expected to enhance the accuracy of key-value pair extraction, improve scalability, and provide a robust solution for businesses looking to automate invoice processing with minimal manual effort.

4.2 Implementation Strategy

The implementation of the automated invoice processing system follows a structured pipeline, integrating preprocessing, tokenization, feature extraction, model training, evaluation, and final structured output generation. The objective is to efficiently extract metadata from invoice images and fine-tune deep learning models, such as LayoutLM and LayoutLMv2, to recognize and label key invoice fields.

The process begins with input data acquisition, where invoice images are collected from different sources. Given the diverse formats and structures of invoices, this step ensures that the dataset is representative of real-world variations. The next step is preprocessing, where invoice images undergo cleaning and transformation. This involves resizing, grayscale conversion, noise reduction, and text region enhancement to improve OCR-based text extraction. Additionally, bounding box coordinates for extracted text are obtained to preserve spatial relationships in the invoice.

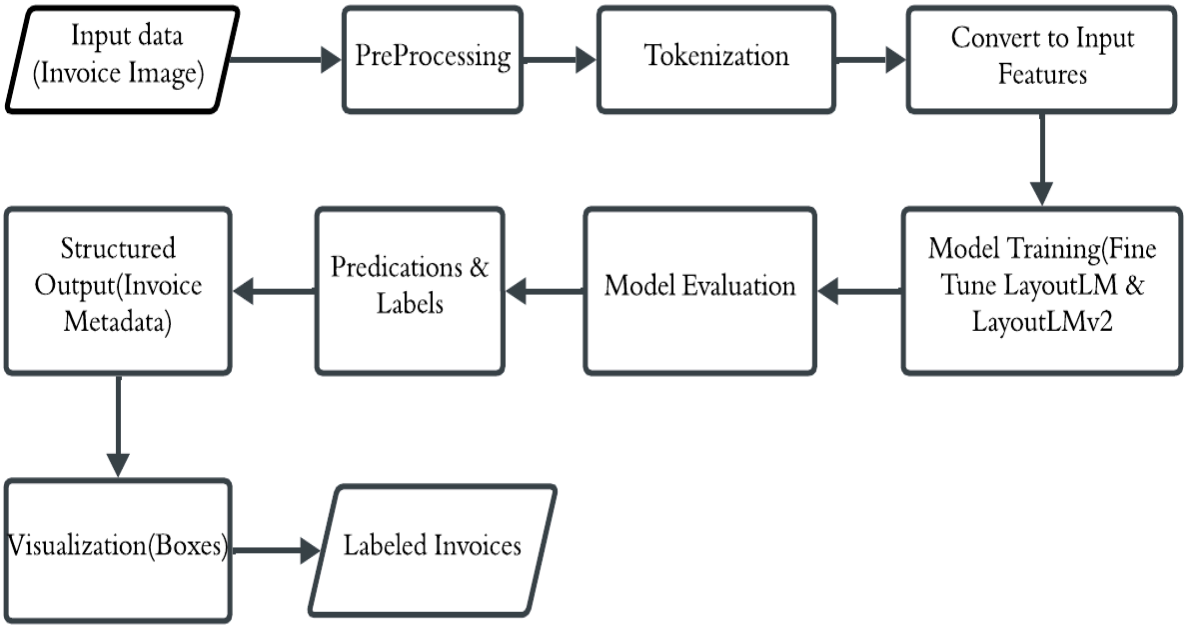


Figure 3: Invoice Processing Flow Chart

Following preprocessing, the tokenization phase segments the extracted text into meaningful units. Tokenization is crucial for maintaining context and enabling the deep learning model to understand invoice elements. The tokenized text is then converted into input features, incorporating textual, positional, and structural information. This transformation is essential for fine-tuning LayoutLM models, as these models require both textual and spatial features to accurately classify invoice fields.

The model training phase involves fine-tuning LayoutLM and LayoutLMv2 using annotated invoice datasets. These models leverage multimodal learning, combining text embeddings with layout-specific features to improve field recognition. Fine-tuning is performed using supervised learning, where the model learns to associate extracted tokens with predefined invoice labels, such as “Invoice Number,” “Seller Address,” and “Total Amount.”

Once training is complete, the model undergoes evaluation to assess its performance in extracting structured invoice data. Evaluation metrics, such as precision, recall, and F1-score, determine the accuracy of extracted fields. If necessary, hyperparameter tuning and additional training iterations are performed to enhance model performance.

After evaluation, the trained model generates predictions and labels, producing structured metadata for invoices. The extracted information is formatted for downstream processing, ensuring usability in financial and accounting systems. Additionally, structured output visualization is implemented by drawing bounding boxes around detected fields, facilitating manual validation and error analysis. The final step involves labeling invoices, where extracted

fields are validated and exported in structured formats, such as JSON or CSV, for integration into enterprise workflows.

This end-to-end pipeline ensures that invoice data extraction is accurate, scalable, and adaptable to various invoice formats. By leveraging LayoutLM-based deep learning models, the system achieves high precision in text extraction while preserving invoice structure, making it highly effective for document processing.

4.2.1 Dataset

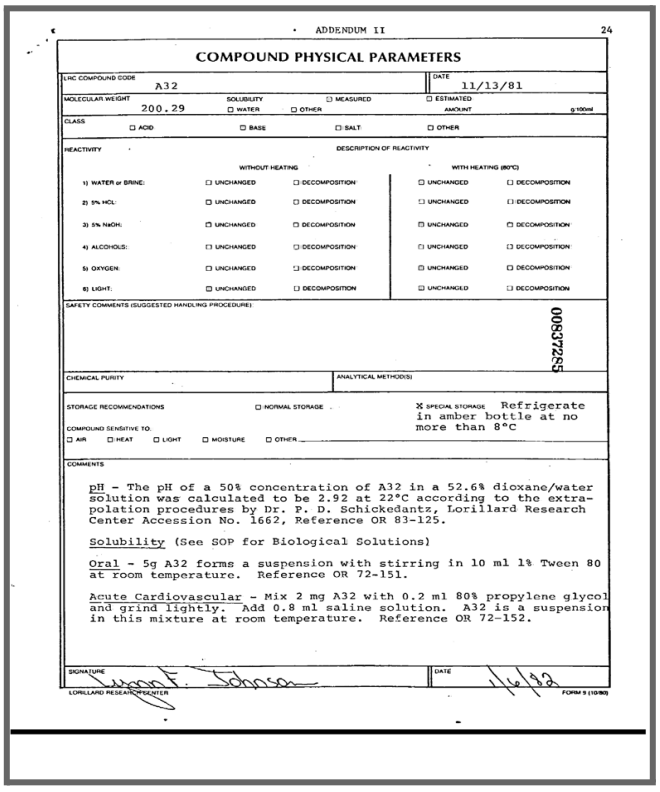
The FUNSD dataset is a well-known standard for identifying entities, extracting key-value combinations, and deciphering document structures in scanned texts. It is particularly useful for training and assessing machine learning models like LayoutLM, LayoutLMv2, and other transformer-based document processing architectures.

The FUNSD dataset contains 199 scanned forms, primarily in English, where each document has been manually annotated to capture the structure and content of text elements. These forms include text blocks, key-value pairs, and hierarchical relationships, making it an ideal dataset for invoice processing, receipt extraction, and general form recognition tasks. Each document is labeled with four categories:

1. Header: Titles and section names.
2. Question: Keys that define a field (e.g., “Invoice Number”).
3. Answer: Values corresponding to keys (e.g., “INV-2025-001”).
4. Other: Text that does not fit into a structured key-value format.

The dataset is provided in JSON format, where each text element is associated with:

1. Bounding box coordinates (to preserve spatial layout).
2. Text transcription (OCR-extracted text).
3. Entity category (header, question, answer, other).
4. Links between entities (defining key-value relationships).



ADDENDUM II 24

COMPOUND PHYSICAL PARAMETERS

LRC COMPOUND CODE: A32 DATE: 11/13/81

MOLECULAR WEIGHT: 200.29 SOLUBILITY: ☐ WATER ☐ MEASURED ☐ ESTIMATED ☐ OTHER AMOUNT: g/100ml

CLASS: ☐ ACID ☐ BASE ☐ SALT ☐ OTHER

REACTIVITY — DESCRIPTION OF REACTIVITY

	WITHOUT HEATING		WITH HEATING (80°C)	
1) WATER or BRINE:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION
2) 5% HCL:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION
3) 5% NaOH:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION
4) ALCOHOLS:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION
5) OXYGEN:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION
6) LIGHT:	☐ UNCHANGED	☐ DECOMPOSITION	☐ UNCHANGED	☐ DECOMPOSITION

SAFETY COMMENTS (SUGGESTED HANDLING PROCEDURE): 00637285

CHEMICAL PURITY: ANALYTICAL METHOD(S):

STORAGE RECOMMENDATIONS: ☐ NORMAL STORAGE ☒ SPECIAL STORAGE: Refrigerate in amber bottle at no more than 8°C

COMPOUND SENSITIVE TO: ☐ AIR ☐ HEAT ☐ LIGHT ☐ MOISTURE ☐ OTHER

COMMENTS:

pH - The pH of a 50% concentration of A32 in a 52.6% dioxane/water solution was calculated to be 2.92 at 22°C according to the extrapolation procedures by Dr. P. D. Schickedantz, Lorillard Research Center Accession No. 1662, Reference OR 83-125.

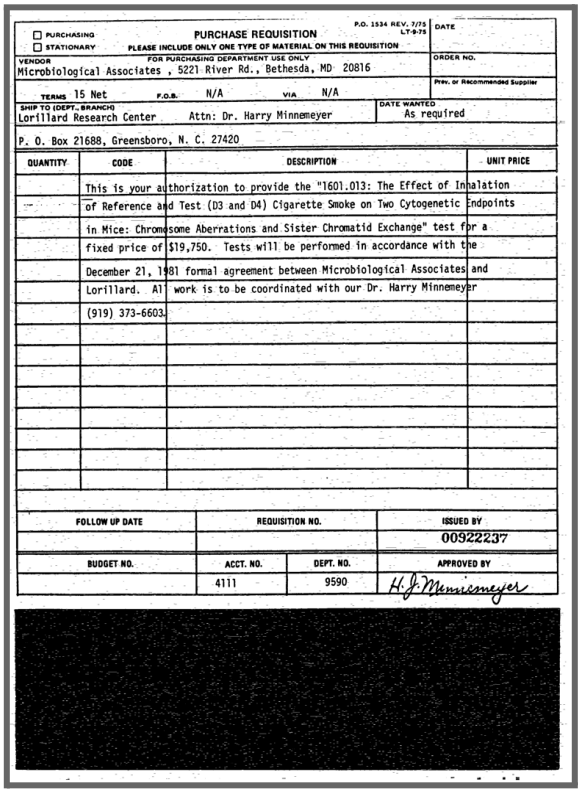
Solubility (See SOP for Biological Solutions)

Oral - 5g A32 forms a suspension with stirring in 10 ml 1% Tween 80 at room temperature. Reference OR 72-151.

Acute Cardiovascular - Mix 2 mg A32 with 0.2 ml 80% propylene glycol and grind lightly. Add 0.8 ml saline solution. A32 is a suspension in this mixture at room temperature. Reference OR 72-152.

SIGNATURE: William E. Johnson DATE: 11/16/82

LORILLARD RESEARCH CENTER FORM 5 (10-80)



☐ PURCHASING ☐ STATIONARY — PURCHASE REQUISITION — P.O. 1534 REV. 7/75 LT-9-75

PLEASE INCLUDE ONLY ONE TYPE OF MATERIAL ON THIS REQUISITION

VENDOR: Microbiological Associates , 5221 River Rd., Bethesda, MD 20816 — FOR PURCHASING DEPARTMENT USE ONLY — DATE — ORDER NO. — Prev. or Recommended Supplier

TERMS: 15 Net F.O.B.: N/A VIA: N/A

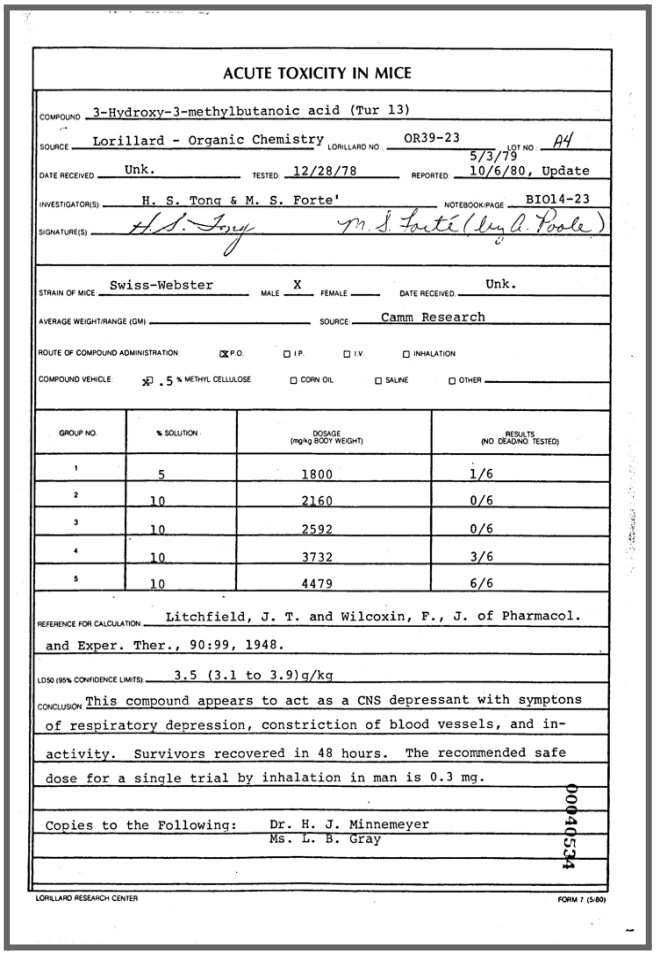
SHIP TO (DEPT., BRANCH): Lorillard Research Center Attn: Dr. Harry Minnemeyer — DATE WANTED: As required

P. O. Box 21688, Greensboro, N. C. 27420

QUANTITY	CODE	DESCRIPTION	UNIT PRICE
		This is your authorization to provide the "1601.013: The Effect of Inhalation	
		of Reference and Test (D3 and D4) Cigarette Smoke on Two Cytogenetic Endpoints	
		in Mice: Chromosome Aberrations and Sister Chromatid Exchange" test for a	
		fixed price of $19,750. Tests will be performed in accordance with the	
		December 21, 1981 formal agreement between Microbiological Associates and	
		Lorillard. All work is to be coordinated with our Dr. Harry Minnemeyer	
		(919) 373-6603.	

FOLLOW UP DATE	REQUISITION NO.		ISSUED BY
			00922237
BUDGET NO.	ACCT. NO.	DEPT. NO.	APPROVED BY
	4111	9590	H.J. Minnemeyer

[redacted]



ACUTE TOXICITY IN MICE

COMPOUND: 3-Hydroxy-3-methylbutanoic acid (Tur 13)

SOURCE: Lorillard - Organic Chemistry LORILLARD NO.: OR39-23 LOT NO.: A4

DATE RECEIVED: Unk. TESTED: 12/28/78 REPORTED: 5/3/79, 10/6/80, Update

INVESTIGATOR(S): H. S. Tong & M. S. Forte' NOTEBOOK/PAGE: BIO14-23

SIGNATURE(S): H.S. Tong M. S. Forte (by A. Poole)

STRAIN OF MICE: Swiss-Webster MALE: X FEMALE: DATE RECEIVED: Unk.

AVERAGE WEIGHT/RANGE (GM): SOURCE: Camm Research

ROUTE OF COMPOUND ADMINISTRATION: ☒ P.O. ☐ I.P. ☐ I.V. ☐ INHALATION

COMPOUND VEHICLE: ☒ .5 % METHYL CELLULOSE ☐ CORN OIL ☐ SALINE ☐ OTHER

GROUP NO.	% SOLUTION	DOSAGE (mg/kg BODY WEIGHT)	RESULTS (NO. DEAD/NO. TESTED)
1	5	1800	1/6
2	10	2160	0/6
3	10	2592	0/6
4	10	3732	3/6
5	10	4479	6/6

REFERENCE FOR CALCULATION: Litchfield, J. T. and Wilcoxin, F., J. of Pharmacol. and Exper. Ther., 90:99, 1948.

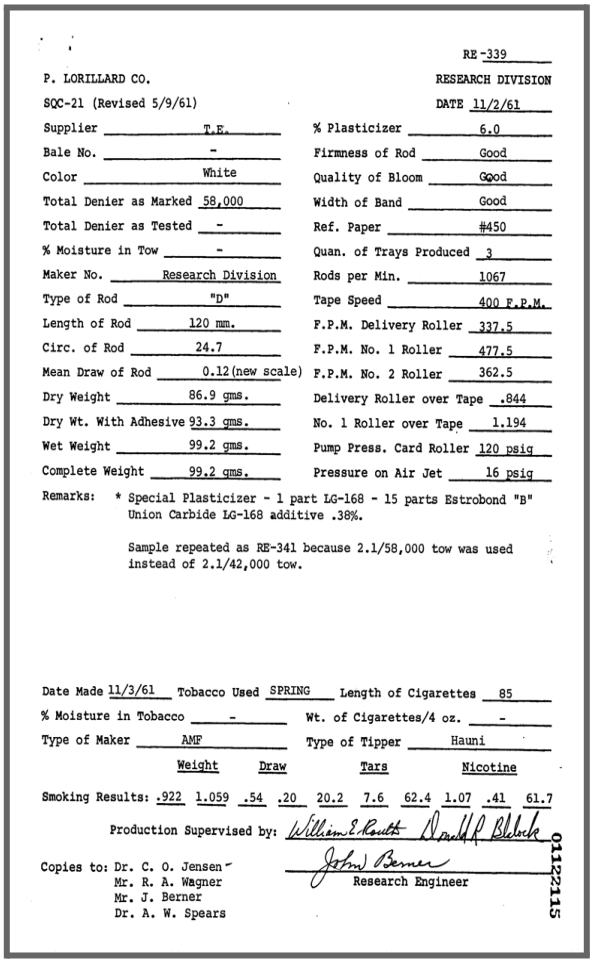
LD50 (95% CONFIDENCE LIMITS): 3.5 (3.1 to 3.9)g/kg

CONCLUSION: This compound appears to act as a CNS depressant with symptons of respiratory depression, constriction of blood vessels, and inactivity. Survivors recovered in 48 hours. The recommended safe dose for a single trial by inhalation in man is 0.3 mg.

Copies to the Following: Dr. H. J. Minnemeyer, Ms. L. B. Gray

00040534

LORILLARD RESEARCH CENTER FORM 7 (5-80)



RE-339

P. LORILLARD CO. RESEARCH DIVISION

SQC-21 (Revised 5/9/61) DATE 11/2/61

Supplier	T.E.	% Plasticizer	6.0
Bale No.	-	Firmness of Rod	Good
Color	White	Quality of Bloom	Good
Total Denier as Marked	58,000	Width of Band	Good
Total Denier as Tested	-	Ref. Paper	#450
% Moisture in Tow	-	Quan. of Trays Produced	3
Maker No.	Research Division	Rods per Min.	1067
Type of Rod	"D"	Tape Speed	400 F.P.M.
Length of Rod	120 mm.	F.P.M. Delivery Roller	337.5
Circ. of Rod	24.7	F.P.M. No. 1 Roller	477.5
Mean Draw of Rod	0.12 (new scale)	F.P.M. No. 2 Roller	362.5
Dry Weight	86.9 gms.	Delivery Roller over Tape	.844
Dry Wt. With Adhesive	93.3 gms.	No. 1 Roller over Tape	1.194
Wet Weight	99.2 gms.	Pump Press. Card Roller	120 psig
Complete Weight	99.2 gms.	Pressure on Air Jet	16 psig

Remarks: * Special Plasticizer - 1 part LG-168 - 15 parts Estrobond "B" Union Carbide LG-168 additive .38%.

Sample repeated as RE-341 because 2.1/58,000 tow was used instead of 2.1/42,000 tow.

Date Made 11/3/61 Tobacco Used SPRING Length of Cigarettes 85

% Moisture in Tobacco - Wt. of Cigarettes/4 oz. -

Type of Maker AMF Type of Tipper Hauni

	Weight	Draw		Tars		Nicotine	
Smoking Results:	.922	1.059	.54	.20	20.2	7.6	62.4	1.07	.41	61.7

Production Supervised by: William E. Ruth Donald P. Blalock

Copies to: Dr. C. O. Jensen, Mr. R. A. Wagner, Mr. J. Berner, Dr. A. W. Spears

John Berner
Research Engineer

01122115

Figure 4 : Sample Invoices of Different Layouts

Since invoices share structural similarities with forms, the FUNSD dataset serves as an excellent pre-training dataset for invoice automation. Fine-tuning models on FUNSD enables robust entity recognition, spatial-text understanding, and layout preservation, which are essential for extracting structured invoice metadata such as Invoice Number, Date, Seller Details, and Total Amount.

4.2.2 Data Preprocessing

In the process of training a deep learning model for invoice data extraction, effective preprocessing of annotated data is crucial. This workflow ensures that raw invoice data is structured appropriately before feeding it into a machine learning model. The process starts with JSON annotation files, which contain structured information about the text present in invoices, including labels and bounding boxes.

Next, the system loads the JSON file and extracts relevant details such as text content, classification labels (e.g., Invoice Number, Seller Name, Amount), and bounding boxes that define the spatial location of these elements in the invoice image. These extracted features are essential for training models that recognize structured invoice components.

After retrieving text and bounding box information, the corresponding invoice image dimensions (width and height) are obtained. These dimensions help in normalizing the bounding boxes and ensuring proper alignment of textual data with the invoice layout. At this stage, each word in the annotation file undergoes processing to prepare it for further segmentation and analysis. This preparation includes converting the text to a standardized format, removing any unnecessary characters, and organizing the data for efficient access during the subsequent stages of data extraction and validation. The extracted data is then saved into three temporary (.tmp) files, ensuring a structured format that can be easily manipulated in subsequent steps. This organization aids in handling large datasets efficiently, minimizing memory usage while processing multiple invoices simultaneously. Following data extraction, tokenization is performed using a pre-trained BERT model. Tokenization breaks down extracted text into smaller units (tokens), which allows the model to understand the contextual relationships between words in an invoice. This step is essential for improving the model's ability to differentiate between invoice fields and non-relevant text. By accurately identifying and categorizing these fields, the model can enhance its overall performance in tasks such as data validation and anomaly detection, ultimately leading to more reliable and automated invoice processing solutions. Once tokenized, the segmented data is stored in three structured text files, ensuring consistency before model training. These text files contain key-value pairs that will serve as input features for the deep learning model. Finally, after preprocessing, the data is ready for model training, where it will be used to develop an accurate and efficient

invoice processing system. This structured preprocessing workflow enhances the accuracy and efficiency of invoice data extraction, ensuring seamless integration into machine learning pipelines.

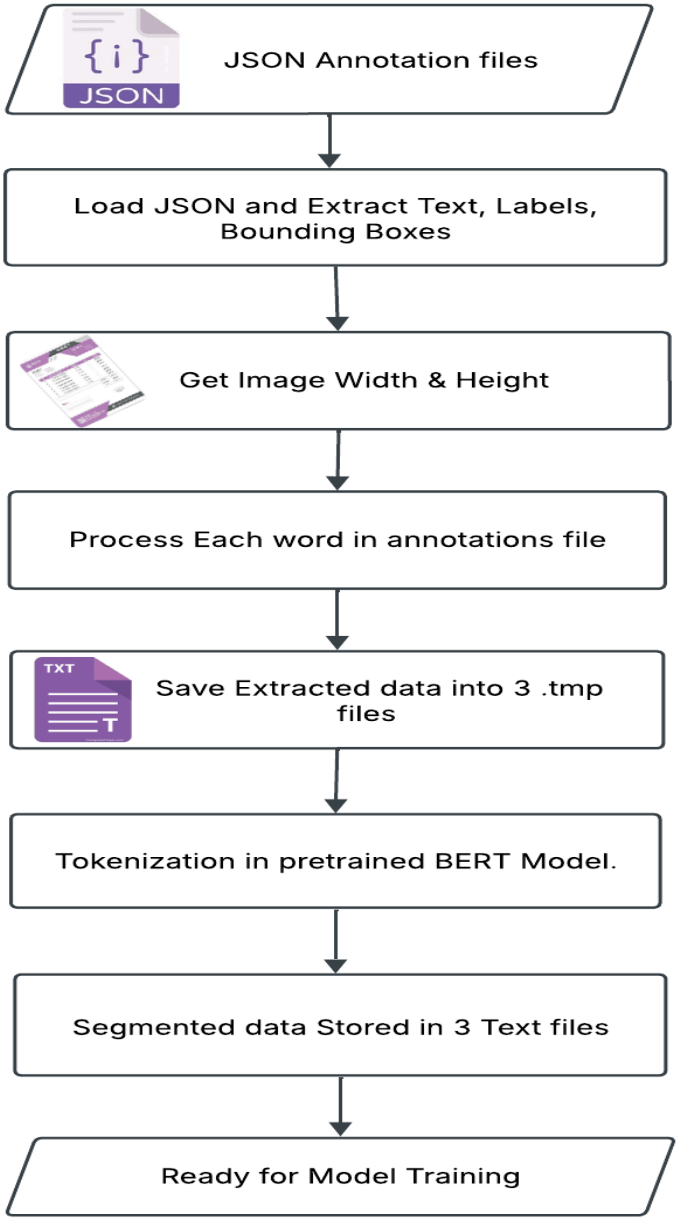


Figure 5: Data Preprocessing Pipeline

4.3 Tools/Hardware/Software used

4.3.1 Hardware Requirements

1. Neural Engine optimized for machine learning tasks, accelerating model inference.
2. Unified Memory Architecture (UMA) that enhances data processing speed.
3. Metal API Support, which can improve GPU-based computations for deep learning models.
4. Software Requirements
5. Operating System: macOS (optimized for development with Unix-based tools).
6. Python Environment: Anaconda Distribution (Python 3.12) is used for managing dependencies and virtual environments.
7. Jupyter Notebook: Used for interactive coding and model experimentation

4.3.2 Machine Learning Libraries

1. PyTorch – The primary deep learning framework used for fine-tuning LayoutLM and LayoutLMv2 models.
2. Transformers (Hugging Face) – Provides pre-trained LayoutLM models for document understanding.
3. Torchvision – Supports image transformations and preprocessing.
4. Tokenizers – Used for tokenizing text and extracting spatial relationships from invoices.
5. pytesseract (OCR) – Extracts text from scanned invoices before passing it to LayoutLM.

4.3.3 Data Processing & Visualization Tools

1. Pandas & NumPy – For handling invoice metadata and structured data processing.
2. OpenCV & PIL (Pillow) – Used for image processing and visualizing bounding boxes.
3. Matplotlib & Seaborn – Helps in visualizing extracted fields and model predictions.

4.4 Expected Outcome

The primary goal of this study is to evaluate the performance of LayoutLM and LayoutLMv2 for extracting structured data from invoices. These models leverage both textual and spatial information, making them well-suited for document understanding tasks. The assessment will be based on key performance metrics, including Recall, Precision, F1-score, Accuracy, Character Error Rate (CER), and Word Error Rate (WER). By comparing the results of LayoutLM and LayoutLMv2, this study aims to determine the most effective model for automating invoice processing.

The evaluation of LayoutLM and LayoutLMv2 is based on key performance metrics such as Accuracy, F1-score, Precision, Recall, Character Error Rate (CER), and Word Error Rate

(WER). It is expected that LayoutLMv2 will achieve higher accuracy and F1-score compared to LayoutLM due to its advanced architecture. With improved spatial embeddings and multimodal learning, LayoutLMv2 can effectively capture both textual and visual features, leading to more accurate data extraction. A higher F1-score reflects a better balance between precision and recall, ensuring that essential invoice details such as Invoice Number, Date, Total Amount, Buyer, and Seller Details are extracted correctly.

Precision and recall play a crucial role in evaluating the correctness and completeness of extracted information. Precision measures the proportion of correctly extracted values, whereas recall indicates the ability to retrieve all relevant fields. LayoutLMv2 is expected to have a higher recall rate, meaning it will capture more invoice fields with fewer omissions. Similarly, precision is expected to improve, reducing the number of false positives and enhancing the reliability of extracted values.

Character Error Rate (CER) and Word Error Rate (WER) are essential for assessing text extraction accuracy at the character and word levels, respectively. Lower values indicate better recognition performance, which is critical for extracting structured fields such as Invoice ID, Dates, and Total Amount without errors. LayoutLMv2 is expected to achieve lower CER and WER compared to LayoutLM, thanks to its enhanced OCR integration and improved handling of complex document layouts. By reducing extraction errors, LayoutLMv2 can provide more precise and structured data, making it a more reliable model for automated invoice processing.

By analyzing key performance metrics, this study aims to determine the extent to which LayoutLMv2 outperforms LayoutLM. The expected improvements stem from enhanced visual-text fusion, allowing LayoutLMv2 to better understand document structure and reduce errors in recognizing table-based invoice data. Additionally, better spatial embeddings improve how the model captures invoice layouts, ensuring a more accurate alignment of extracted data with its original document position. Furthermore, improved OCR capabilities enable LayoutLMv2 to handle variations in font, style, and text positioning more effectively, leading to a more precise extraction of structured data. If LayoutLMv2 demonstrates significant improvements in accuracy, F1-score, and error reduction, it will be the preferred model for real-world invoice automation. A more accurate extraction system reduces manual verification efforts, enabling faster processing times and improving operational efficiency. It also enhances financial data reliability, ensuring compliance and accuracy in business transactions. Moreover, LayoutLMv2's superior performance allows for scalability, making it a viable solution for organizations dealing with large invoice datasets. This study will provide a detailed comparative evaluation of LayoutLM and LayoutLMv2, assessing their effectiveness in structured invoice data extraction. The results will highlight whether LayoutLMv2 offers a significant performance advantage, guiding future implementations in automated invoice processing.

CHAPTER 5

RESULT AND DISCUSSION

5.1 Implementation Details

5.1.1 Data Preprocessing for LayoutLM and LayoutLMV2

The preprocessing script is designed to handle structured invoice data by extracting text, labels, and bounding box information from JSON annotation files. It reads the annotations, identifies key-value pairs, and applies the BIOES (Begin, Inside, Outside, End, Single) tagging scheme to classify words based on their entity type. The script also normalizes bounding box coordinates by converting them into a relative format, ensuring they align with the image dimensions. This process enables the extracted data to be structured appropriately for training deep learning models like LayoutLM and LayoutLMv2. Additionally, the script implements robust error handling, skipping non-JSON files and logging any errors related to JSON parsing or missing images. This ensures a smooth and automated preprocessing workflow without manual intervention.

Bounding Box Normalization:

Bounding box normalization is crucial for maintaining spatial consistency across varying invoice dimensions. The bbox_string() function converts bounding box coordinates into a fixed 1000x1000 scale. This ensures that all invoice layouts are processed uniformly, allowing the model to learn spatial dependencies effectively.

```
def bbox_string(box, width, length):
    return (
        str(int(1000 * (box[0] / width)))
        + " "
        + str(int(1000 * (box[1] / length)))
        + " "
        + str(int(1000 * (box[2] / width)))
        + " "
        + str(int(1000 * (box[3] / length)))
    )
```

Figure 6: Bounding Box Normalization

The function takes the original bounding box coordinates (x1, y1, x2, y2) and normalizes them based on the image width and height. This normalization ensures consistency when processing images of different resolutions.

Extracting and Structuring OCR Data:

The convert() function processes JSON-formatted invoice annotations, extracts text and bounding boxes, and writes the results into separate files for text, bounding boxes, and image references. This ensures that text-based information is aligned with spatial features for LayoutLM processing.

```
for file in os.listdir(args.data_dir):
    file_path = os.path.join(args.data_dir, file)
    if not file_path.endswith(".json"):
        print(f"Skipping non-JSON file: {file_path}")
        continue

    with open(file_path, "r", encoding="utf-8") as f:
        try:
            data = json.load(f)
        except json.JSONDecodeError as e:
            print(f"Error reading JSON file {file_path}: {e}")
            continue
```

Figure 7: Extracting and Structuring OCR Data

The script iterates through JSON files in the dataset directory. It reads each file and extracts structured annotations. Non-JSON files are skipped, ensuring that only relevant data is processed.

Image Loading and Error Handling:

The script utilizes the PIL library to load and process images, extracting their height and width to compute bounding boxes accurately. It also incorporates error-handling mechanisms to prevent crashes due to missing or corrupted images.

```
try:
    image = Image.open(image_path)
    width, length = image.size
except Exception as e:
    print(f"Error loading image {image_path}: {e}")
    continue
```

Figure 8: Image Loading and Error Handling

When an image is opened with PIL, its dimensions are retrieved, ensuring precise spatial alignment of text elements. To enhance robustness, the script displays an error message if an image fails to load, allowing seamless processing of large invoice datasets.

Labeling Tokens for Named Entity Recognition (NER):

The extracted words are labeled using a BIO (Begin, Inside, End) tagging scheme, which is essential for training LayoutLM on key-value pair extraction tasks.

```
if label == "other":
    for w in words:
        fw.write(w["text"] + "\tO\n")
        fbw.write(w["text"]+ "\t"+ bbox_string(w["box"], width, length) + "\n")
        fiw.write( w["text"]+ "\t" + actual_bbox_string(w["box"], width, length)+ "\t" + file_name + "\n")
else:
    if len(words) == 1:
        fw.write(words[0]["text"] + "\tS-" + label.upper() + "\n")
        fbw.write( words[0]["text"]+ "\t" + bbox_string(words[0]["box"], width, length) + "\n")
        fiw.write( words[0]["text"]+ "\t"+ actual_bbox_string(words[0]["box"], width, length) + "\t" + file_name+ "\n")
```

Figure 9: Labeling Tokens for Named Entity Recognition (NER)

Each word is assigned a label based on its role in a key-value pair. Single-word entities are marked as S-<LABEL>, while multi-word entities follow the B-, I-, and E- (Beginning, Inside, End) notation. This structure helps the model recognize entity boundaries.

Tokenization for Model Processing:

Tokenization is performed using the AutoTokenizer, ensuring that words are converted into subwords that fit within the model's input constraints.

```
def seg_file(file_path, tokenizer, max_len):
    subword_len_counter = 0
    output_path = file_path[:-4]

    with open(file_path, "r", encoding="utf8") as f_p, open(output_path, "w", encoding="utf8") as fw_p:
        for line in f_p:
            line = line.strip()

            if not line:
                fw_p.write("\n")
                subword_len_counter = 0
                continue
```

Figure 10: Tokenization for Model Processing

The function processes each tokenized word, ensuring that the total number of subwords remains within max_len. If a tokenized sequence exceeds the limit, a newline is inserted to maintain the document structure.

5.1.2 LayoutLM Model

The implementation employs the LayoutLMForTokenClassification model, which is fine-tuned for Named Entity Recognition (NER) on invoice documents. The preprocessing step converts invoices into tokenized sequences while preserving spatial relationships.

Model Training:

The training process involves fine-tuning LayoutLM on structured invoice data. The model takes text and bounding box coordinates as inputs. The training loop follows these steps:

1. Tokenization and bounding box extraction.
2. Model training using a cross-entropy loss function.
3. Gradient updates through backpropagation.
4. Performance evaluation using precision, recall, and F1-score.

The model is trained for token classification using the FUNSD dataset. We define a label set for named entity recognition (NER) tasks and use cross-entropy loss for optimization. The training involves feeding input sequences, bounding box coordinates, and token types into the LayoutLM model. The AdamW optimizer is employed to fine-tune the model over multiple epochs. A critical aspect of training is handling the padding label ID to ignore unnecessary tokens in the loss calculation.

The training loop iterates over batches of invoice data, computing loss and updating model weights. The following snippet highlights the training process:

```
model.train()
for epoch in range(num_train_epochs):
  for batch in tqdm(train_dataloader, desc="Training"):
      input_ids = batch[0].to(device)
      bbox = batch[4].to(device)
      attention_mask = batch[1].to(device)
      token_type_ids = batch[2].to(device)
      labels = batch[3].to(device)

      # forward pass
      outputs = model(input_ids=input_ids, bbox=bbox, attention_mask=attention_mask, token_type_ids=token_type_ids,
                      labels=labels)
      loss = outputs.loss
      logits = outputs.logits
      true_labels.extend(labels.cpu().numpy().flatten())
      pred_labels.extend(logits.argmax(dim=2).cpu().numpy().flatten())
```

Figure 11: LayoutLM Model Training

Model Evaluation:

After training, the model is evaluated on a test dataset. Evaluation involves computing precision, recall, and F1-score. The evaluation loop iterates through test samples, predicting entity labels and comparing them with ground truth labels.

```
model.eval()
for batch in tqdm(eval_dataloader, desc="Evaluating"):
    with torch.no_grad():
        input_ids = batch[0].to(device)
        bbox = batch[4].to(device)
        attention_mask = batch[1].to(device)
        token_type_ids = batch[2].to(device)
        labels = batch[3].to(device)
```

Figure 12(a) LayoutLM Model Evaluation

Performance metrics are computed using the seqeval library, with results presented as:

```
results = {
    "loss": eval_loss,
    "precision": precision_score(out_label_list, preds_list),
    "recall": recall_score(out_label_list, preds_list),
    "f1": f1_score(out_label_list, preds_list),
}
print("Evaluation Results:", results)
```

Figure 12(b) LayoutLM Performance Metrics

Inference and Visualization:

The trained model is applied to invoice images for entity extraction. Preprocessing involves OCR-based text extraction, bounding box detection, and feature conversion. The predictions are mapped to entity labels and overlaid on the invoice image for visualization.

```
model_path='layoutlm.pt'
model=model_load(model_path,num_labels)
image, words, boxes, actual_boxes = preprocess("/Users/hannahml/Documents/MTECH/Project/Invoice/FUNSD_DATASE

word_level_predictions, final_boxes=convert_to_features(image, words, boxes, actual_boxes, model)
```

```
draw = ImageDraw.Draw(image)
font = ImageFont.load_default()
def iob_to_label(label):
  if label != 'O':
    return label[2:]
  else:
    return ""
label2color = {'question':'blue', 'answer':'green', 'header':'orange', '':'violet'}
for prediction, box in zip(word_level_predictions, final_boxes):
    predicted_label = iob_to_label(label_map[prediction]).lower()
    draw.rectangle(box, outline=label2color[predicted_label])
    draw.text((box[0] + 10, box[1] - 10), text=predicted_label, fill=label2color[predicted_label], font=font)
image
```

Figure 13: LayoutLM Inference and Visualization

5.1.3 LayoutLMv2 Model

LayoutLMv2 is a deep learning model designed for document image understanding, leveraging both spatial and textual information for tasks such as Optical Character Recognition (OCR), key-value extraction, and document classification. Unlike traditional OCR methods that only extract text, LayoutLMv2 understands the spatial layout of documents, making it particularly effective for processing invoices, receipts, forms, and other structured documents. This report provides a detailed overview of the LayoutLMv2 implementation, focusing on training, evaluation, inference, and visualization.

Model Training:

Training LayoutLMv2 involves initializing the model, defining the training parameters, and fine-tuning it on a structured dataset. The model is designed to learn token classification tasks, meaning it can extract specific entities such as names, dates, and amounts from a document. The training process typically involves:

1. Dataset Preparation: The dataset consists of tokenized text extracted from the document, bounding box coordinates of each token, and labels for named entity recognition (NER) or classification tasks.
2. Model Initialization: A pre-trained LayoutLMv2 model is loaded and fine-tuned with task-specific labels.
3. Training Execution: The model is trained using a transformer-based Trainer API, specifying hyperparameters such as batch size, number of epochs, and learning rate.

```
model.train()
for epoch in range(num_train_epochs):
    for batch in tqdm(train_dataloader, desc="Training"):
        input_ids = batch["input_ids"].to(device)
        bbox = batch["bbox"].to(device)
        attention_mask = batch["attention_mask"].to(device)
        token_type_ids = batch["token_type_ids"].to(device)  # If required
        labels = batch["labels"].to(device)  # Ensure correct shape
        images = batch["image"].to(device)  # For LayoutLMv2

        labels_padded = torch.full((labels.shape[0], 512), fill_value=-100, dtype=torch.long).to(device)
        labels_padded[:, :labels.shape[1]] = labels
        # Forward pass
        outputs = model(
            input_ids=input_ids,
            bbox=bbox,
            attention_mask=attention_mask,
            image=images,  # LayoutLMv2 requires images
            labels = labels_padded  # Ensure shape is (batch_size, sequence_length)
        )

        loss = outputs.loss
```

Figure 14: LayoutLMv2 Model Training

Once the training process is complete, the model is saved for evaluation and inference. The training results include loss values and accuracy metrics, which help analyze how well the model is learning. Fine-tuning can be improved by adjusting hyperparameters, adding more labeled training data, or applying data augmentation techniques.

Model Evaluation:

Evaluating the trained model is crucial to assess its performance. The model's predictions are compared against ground truth labels, and various performance metrics, such as accuracy, precision, recall, and F1-score, are calculated. Evaluation helps identify weaknesses in entity recognition and areas where the model can be improved.

Accuracy: Measures how many predictions match the true labels.
F1-score: Balances precision and recall to give a better overall performance metric.
Confusion Matrix: Helps visualize classification errors across different entities.

```
eval_loss = eval_loss / nb_eval_steps
preds = np.argmax(preds, axis=2)

out_label_list = [[] for _ in range(out_label_ids.shape[0])]
preds_list = [[] for _ in range(out_label_ids.shape[0])]

for i in range(out_label_ids.shape[0]):
    for j in range(out_label_ids.shape[1]):
        if out_label_ids[i, j] != pad_token_label_id:
            out_label_list[i].append(label_map[out_label_ids[i][j]])
            preds_list[i].append(label_map[preds[i][j]])

results = {
    "loss": eval_loss,
    "precision": precision_score(out_label_list, preds_list),
    "recall": recall_score(out_label_list, preds_list),
    "f1": f1_score(out_label_list, preds_list),
}
print(results)
```

Figure 15: LayoutLMv2 Model Evaluation

The evaluation phase ensures that the model generalizes well to unseen data. If the accuracy is low, adjustments such as increasing training data, using different pre-processing techniques, or refining the label annotations may be necessary. Additionally, analyzing misclassified entities can help improve model predictions.

Inference and Visualization:

After training and evaluating the model, it can be used for inference on new document images. Inference involves feeding an unseen document into the trained model, extracting text and spatial features, and obtaining predictions for named entities. The extracted information is then visualized by overlaying predictions on the document image.

1. Image Processing: The document image is converted into a format suitable for LayoutLMv2.
2. Token Prediction: The model processes the image and assigns labels to detected tokens.
3. Visualization: Bounding boxes are drawn around detected entities to provide a clear representation of extracted information.

```
# Get predicted labels and bounding boxes
word_level_predictions = torch.argmax(outputs.logits, dim=-1)[0].tolist()
final_boxes = inputs["bbox"][0].tolist()  # Extract bounding boxes

# Draw on the image
draw = ImageDraw.Draw(image)
font = ImageFont.load_default()

# Convert LayoutLMv2 bounding boxes to absolute pixel coordinates
for prediction, box in zip(word_level_predictions, final_boxes):
    predicted_label = iob_to_label(prediction)  # Convert to readable format

    if predicted_label:  # Ignore 'O' labels
        abs_box = (
            (box[0] / 1000) * width, (box[1] / 1000) * height,  # Top-left
            (box[2] / 1000) * width, (box[3] / 1000) * height   # Bottom-right
        )
```

Figure 16: LayoutLMv2 Inference and Visualization

This step enables real-world applications such as automated invoice processing, form understanding, and document classification. Visualization aids in verifying the correctness of extracted entities and debugging issues related to misclassification or incorrect bounding boxes.

5.2 Results

Form understanding and key-value extraction from scanned documents are frequently assessed using the FUNSD dataset. 31,485 words and 9,707 identified semantic items are included in 199 scanned forms. The performance of LayoutLM and LayoutLMv2 in processing invoices may be assessed using FUNSD, which is a perfect benchmark because invoices frequently include structured elements like tables, key-value pairs, and free-text fields.

The models were assessed using the following key metrics:

1. Accuracy: Measures the correctness of extracted key-value pairs.
2. Precision: The proportion of correctly extracted key-value pairs among all extracted values.
3. Recall: The proportion of correctly extracted key-value pairs among all actual values.
4. F1-Score: The harmonic mean of precision and recall.
5. Character Error Rate (CER): Percentage of incorrect characters relative to total characters.
6. Word Error Rate (WER): Percentage of incorrect words relative to total words.

5.2.1 Analysis of LayoutLM Results

Performance Comparison Across Different Hyperparameters:

The performance comparison of LayoutLM across different hyperparameter settings highlights the impact of learning rates and epochs on the model's accuracy, precision, recall, and text extraction quality. The evaluation considered two learning rates (5e-5 and 3e-5) and two epoch settings (5 and 10), with key insights drawn from Table 1.

A lower learning rate of 3e-5 resulted in better accuracy when trained for 5 epochs, achieving 77.37%, compared to 74.42% with a 5e-5 learning rate. This suggests that a slower learning rate allows for finer weight adjustments, improving the model's ability to generalize. However, when trained for 10 epochs, the accuracy for 3e-5 (75.31%) was nearly identical to that of 5e-5 (75.30%), indicating that additional training did not lead to significant performance gains. Increasing the number of epochs from 5 to 10 resulted in slightly improved accuracy for a 5e-5 learning rate (from 74.42% to 75.30%). However, for 3e-5, the accuracy slightly dropped from 77.37% to 75.31%, suggesting that prolonged training might lead to minor overfitting. While longer training can enhance performance, its benefits diminish beyond a certain point.

Table 1: Performance Comparison of LayoutLM

	Epoch	Learning Rate	Accuracy	Precision	Recall	F1-Score	CER	WER
LAYOUTLM	5	3e-5	**77.37%**	**76.51%**	**80.59 %**	**76.51%**	0.1880	0.2258
	10	3e-5	75.31%	69.66%	80.29%	74.59%	0.2009	0.2458
	5	5e-5	74.42%	70.712%	77.43%	73.92%	0.2231	0.255
	10	5e-5	75.30%	72.61%	79.73%	75.70%	0.2050	0.2462

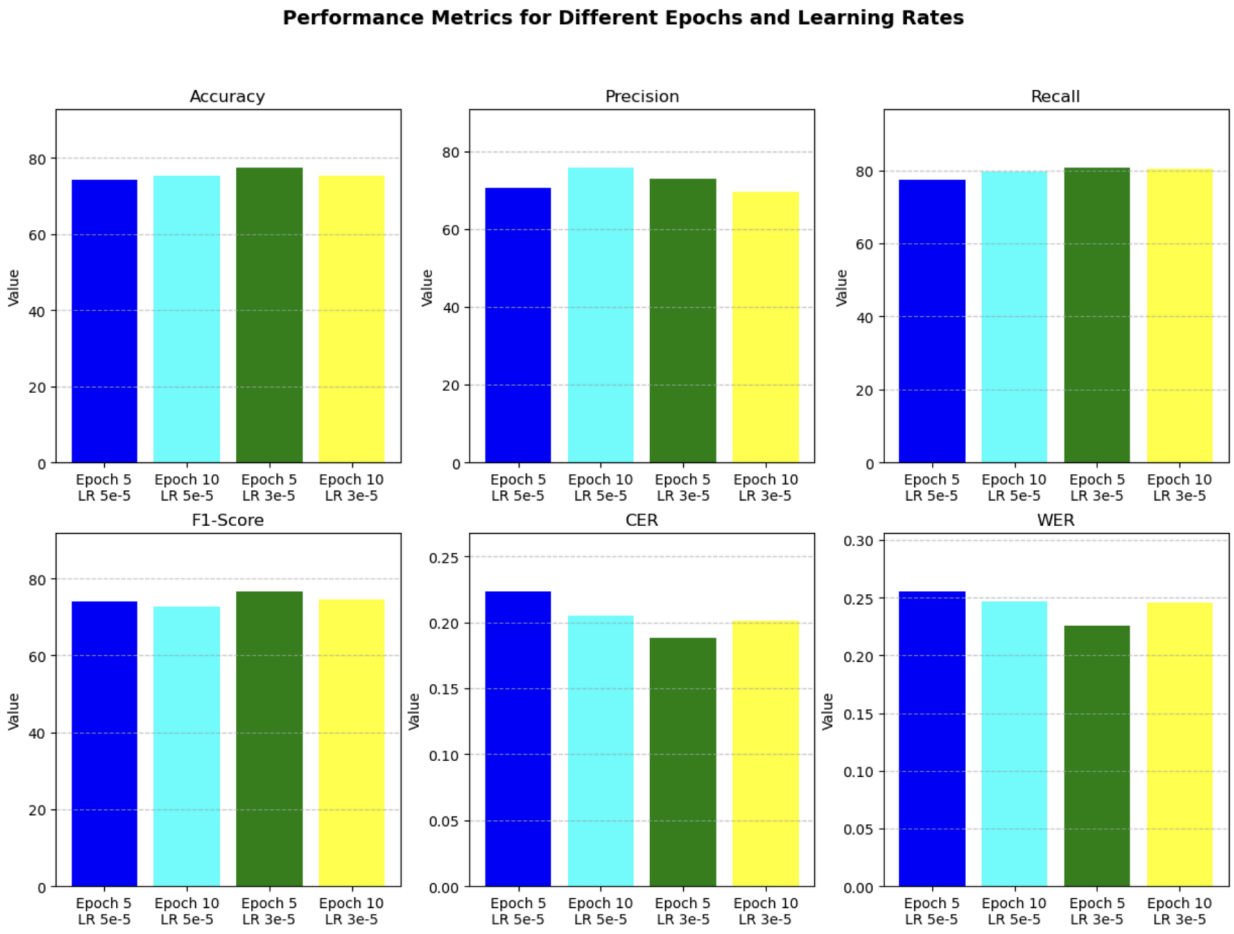


Figure 17 : Performance Metrics for Different Epochs and Learning Rates of LayoutLM

The highest precision (75.70%) was achieved with 5e-5 and 10 epochs, indicating that the model was highly selective in identifying key entities. Meanwhile, the highest recall (80.59%) was recorded at 3e-5 with 5 epochs, suggesting that this setting allowed the model to capture more relevant instances. The best F1-score (76.51%), which balances precision and recall, was also observed for 3e-5 and 5 epochs, making it the most optimal configuration.The lowest Character Error Rate (CER) of 0.1880 and Word Error Rate (WER) of 0.2258 were observed with 3e-5 and 5 epochs, further confirming its superior text extraction accuracy.

Overall, a learning rate of 3e-5 with 5 epochs emerged as the most effective setting, providing a balance between accuracy, recall, and text recognition quality, while avoiding the risks of overfitting that come with prolonged training.

Field-Wise Performance Analysis of LayoutLM:

The field-wise performance analysis of LayoutLM highlights variations in precision, recall, F1-score, and accuracy across different entity types, reflecting the model's effectiveness in extracting structured and unstructured data from invoices. Among the three field categories—Answer Fields, Header Fields, and Question Fields—the Header Fields consistently achieved the highest accuracy, with a peak of 71% when trained with a 5e-5 learning rate and 5 epochs. This indicates that the model effectively identifies structured headings, likely due to their consistent format across different invoice templates. However, when increasing the number of epochs to 10 with the same learning rate, the accuracy for Header Fields dropped to 65%, suggesting a potential overfitting issue where the model learned patterns too rigidly.

Table 2: LayoutLM model for Learning rate 5e-5 and 5 Epochs

	Precision	Recall	F1-Score	Accuracy
Answer Fields	0.41	0.48	0.44	0.61
Header Fields	0.52	0.56	0.54	0.71
Question Fields	0.51	0.52	0.52	0.53

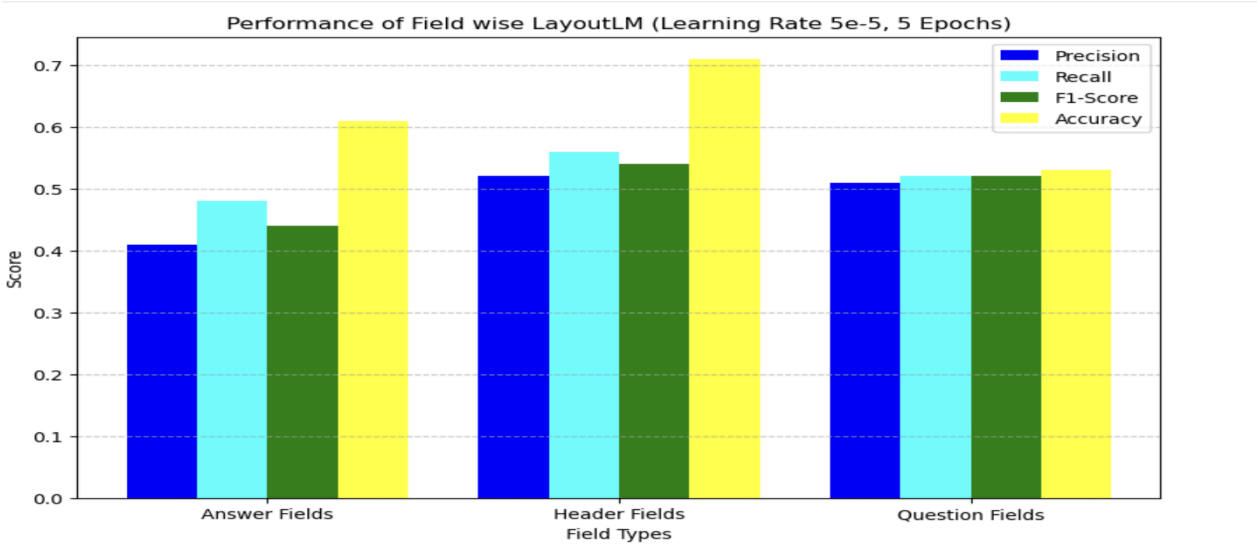


Figure 18 : Performance of Field wise LayoutLM (Learning Rate 5e-5, 5 Epochs)

Question Fields exhibited relatively stable performance across different configurations, with accuracy values ranging between 52% and 55%, showing that the model can recognize these entities with moderate reliability. Despite minor variations in F1-score, Question Fields maintained a balance between precision and recall across all hyperparameter settings.

Table 3: LayoutLM model for Learning rate 5e-5 and 10 Epochs

	Precision	Recall	F1-Score	Accuracy
Answer Fields	0.41	0.44	0.43	0.62
Header Fields	0.51	0.52	0.51	0.65
Question Fields	0.53	0.55	0.54	0.54

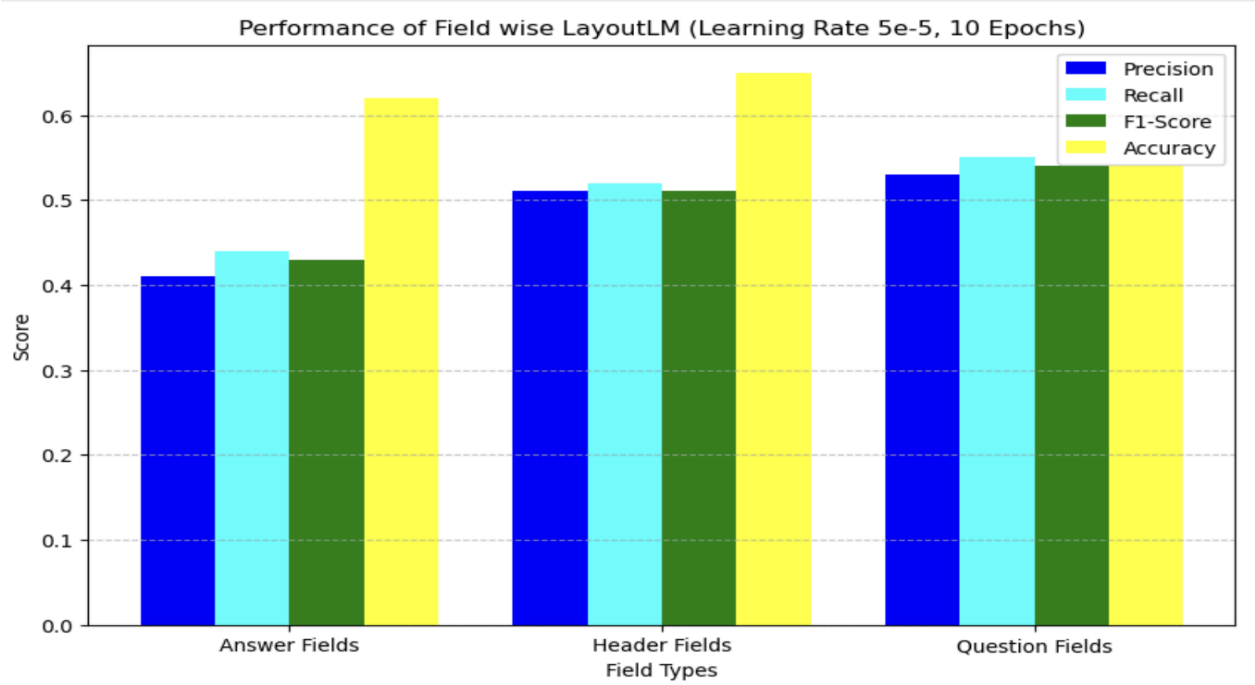


Figure 19 : Performance of Field wise LayoutLM (Learning Rate 5e-5, 10 Epochs)

Conversely, Answer Fields posed the greatest challenge for LayoutLM, as seen in the lower precision and recall scores. The best performance for Answer Fields occurred at a 3e-5 learning rate with 5 epochs, where the F1-score reached 0.49 and accuracy improved to 65%. This suggests that a lower learning rate allows the model to generalize better for extracting numerical and textual values. However, in most other cases, Answer Fields had low recall, indicating that the model frequently missed extracting key-value pairs, which may be due to variations in invoice structures.

Table 4: LayoutLM model for Learning rate 3e-5 and 5 Epochs

	Precision	Recall	F1-Score	Accuracy
Answer Fields	0.46	0.52	0.49	0.65
Header Fields	0.46	0.50	0.48	0.63
Question Fields	0.51	0.53	0.52	0.52

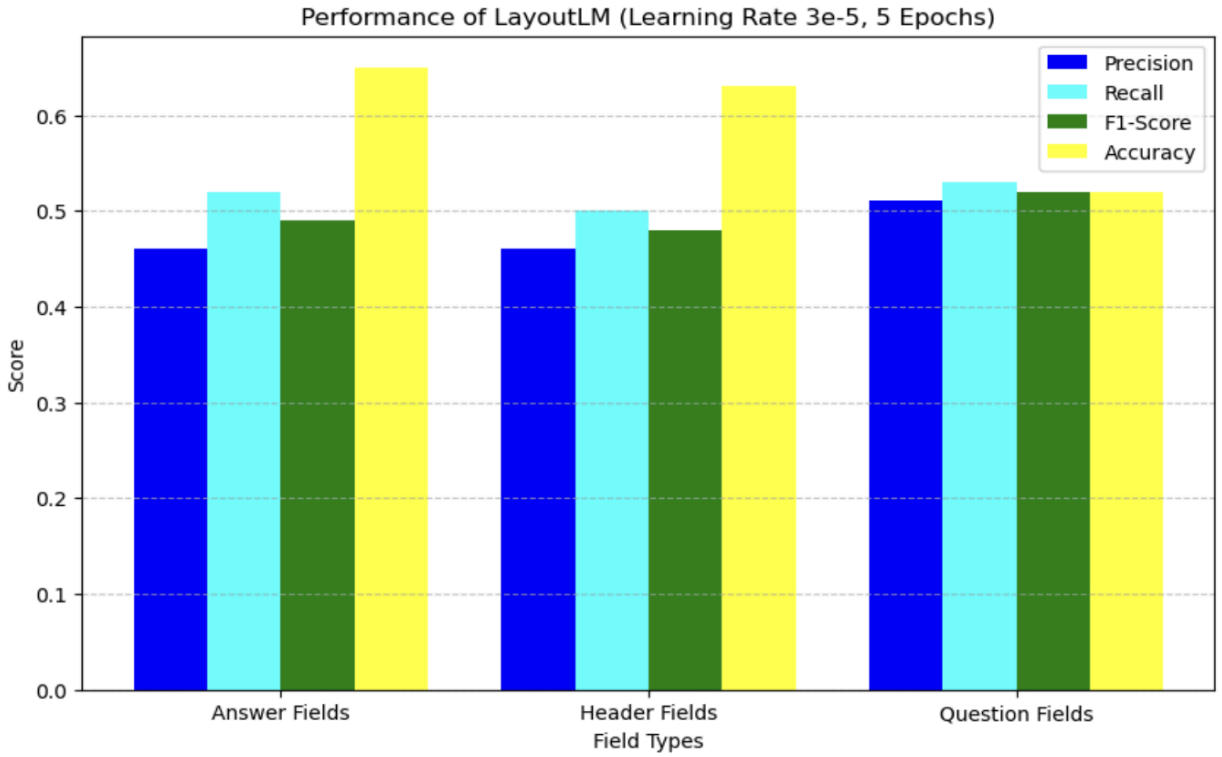


Figure 20 : Performance of Field wise LayoutLM (Learning Rate 3e-5, 5 Epochs)

Overall, while increasing epochs slightly improved recall in some cases, it negatively impacted precision, particularly for Header Fields, reinforcing the risk of overfitting. The results suggest that LayoutLM performs best for structured fields like headers but struggles with unstructured text, especially Answer Fields. Future improvements could involve fine-tuning the model on entity-specific datasets, applying post-processing techniques, or integrating rule-based validation to improve the accuracy of extracted key-value pairs, particularly in challenging cases.

Table 5: LayoutLM model for Learning rate 3e-5 and 10 Epochs

	Precision	Recall	F1-Score	Accuracy
Answer Fields	0.42	0.47	0.44	0.63
Header Fields	0.39	0.42	0.40	0.59
Question Fields	0.51	0.55	0.53	0.54

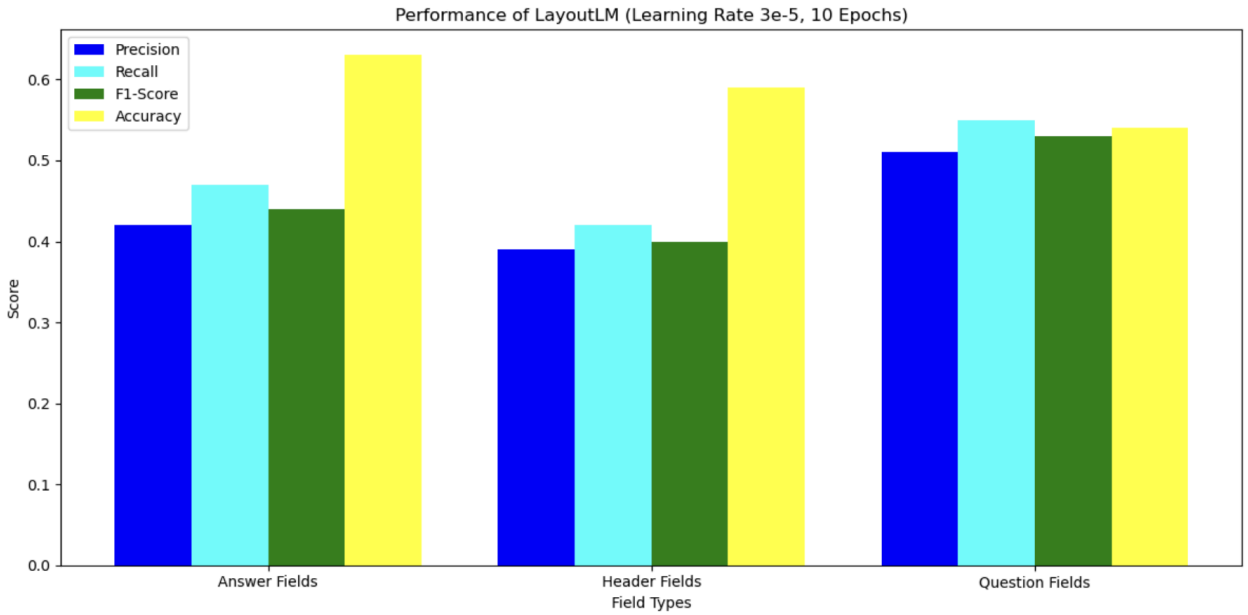


Figure 21 : Performance of LayoutLM (Learning Rate 3e-5, 10 Epochs)

5.2.2 Analysis of LayoutLMv2 Performance

The performance of the LayoutLMv2 model was evaluated under different hyperparameter settings, including learning rates (2e-5 and 3e-5) and epochs (5 and 10). The key metrics analyzed include accuracy, precision, recall, F1-score, Character Error Rate (CER), and Word Error Rate (WER), as presented in Table 6.The highest accuracy (95.19%) was achieved with a 2e-5 learning rate and 10 epochs, demonstrating that extended training improves model performance when using a lower learning rate. However, for 3e-5, the accuracy was slightly lower (93.59% for 5 epochs and 93.39% for 10 epochs), suggesting that increasing epochs for a higher learning rate does not significantly impact accuracy.The best precision (94.19%) was observed with a 3e-5 learning rate and 5 epochs, indicating that this setting yielded highly selective and accurate extractions. However, the highest recall (94.12%) was recorded with a 2e-5 learning rate and 5 epochs, meaning that the model was able to capture more relevant entities in this configuration. The best balance between precision and recall, as measured by the F1-score (88.80%), was achieved at 2e-5 with 10 epochs, making it the most optimal setting for both accurate and complete entity recognition.

Table 6: Performance Comparison of LayoutLMv2

	Epoch	Learning Rate	Accuracy	Precision	Recall	F1-Score	CER	WER
LAYOUTLMv2	5	3e-5	93.59%	**94.19%**	75.75%	83.97%	0.14	0.064
	10	3e-5	93.39%	82.81%	88.55%	85.58%	0.1495	0.0660
	5	2e-5	88.85%	67.93%	**94.12%**	78.91%	0.252	0.111
	10	2e-5%	**95.19%**	91.80%	85.99%	**88.80%**	0.108	0.0480

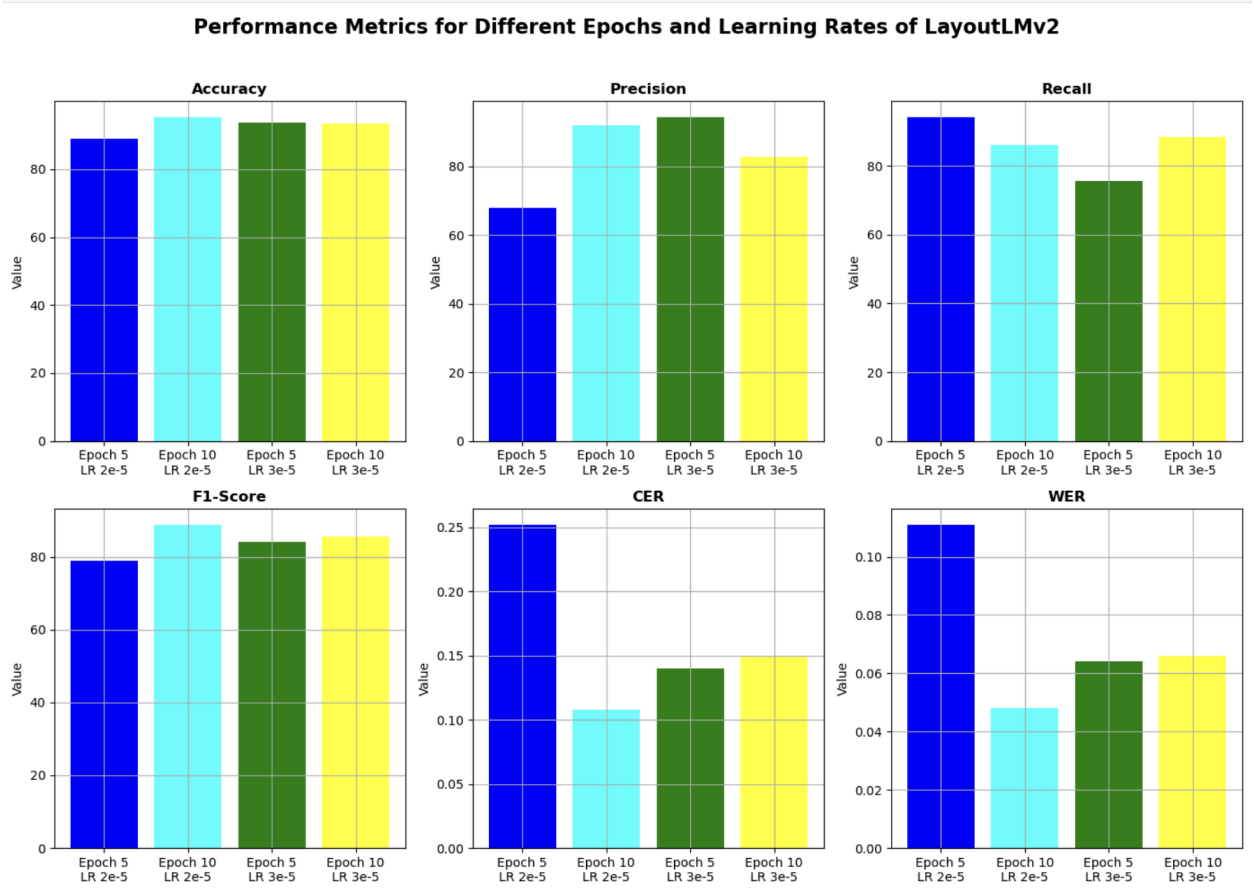
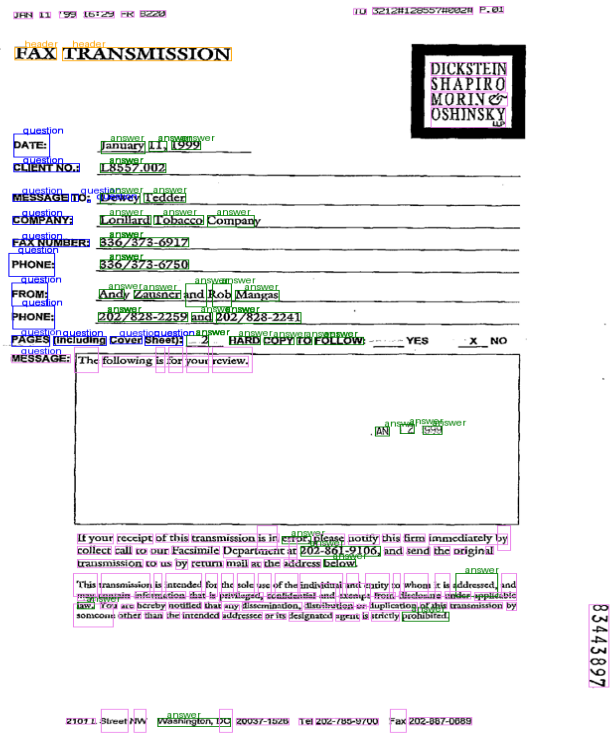
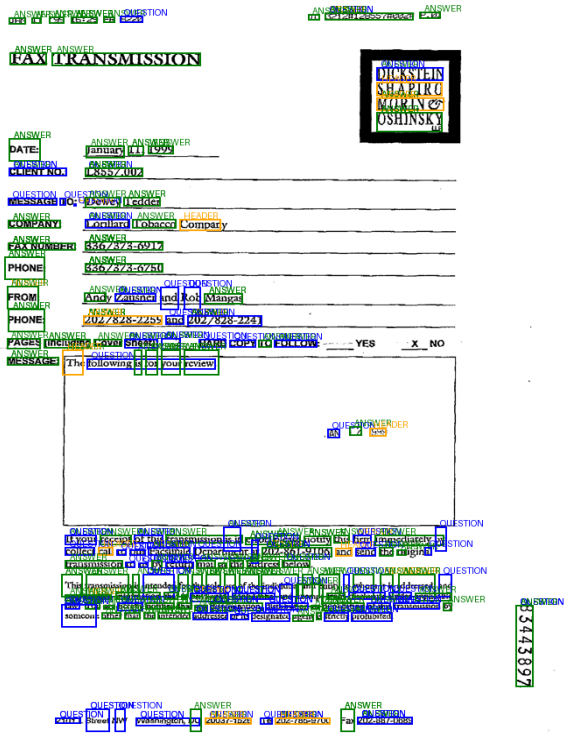


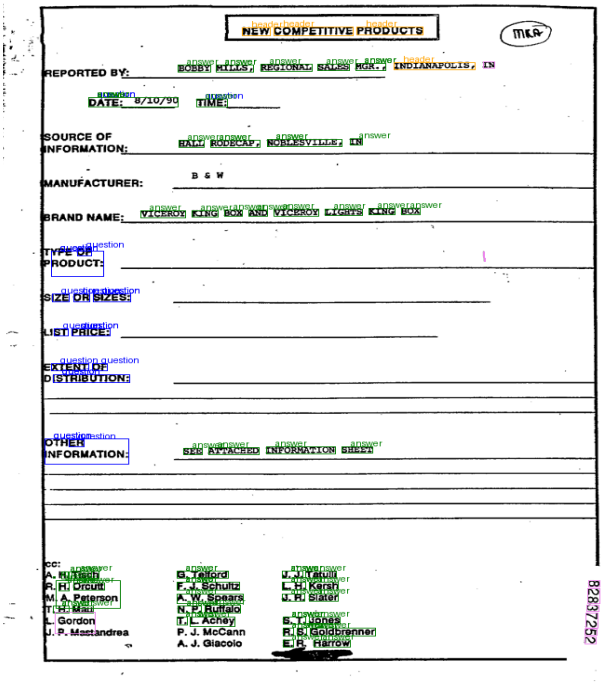
Figure 22 : Performance Metrics for Different Epochs and Learning Rates of LayoutLMv2



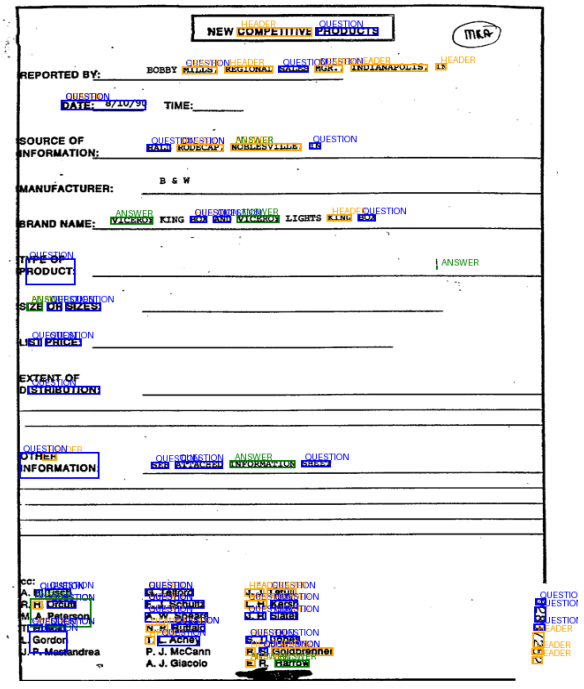
23(a) Sample 1 LayoutLM Output



23(b) Sample 1 LayoutLMv2 Output



23(c) Sample 2 LayoutLM Output



23(d) Sample 2 LayoutLMv2 Output

Figure 23: Output

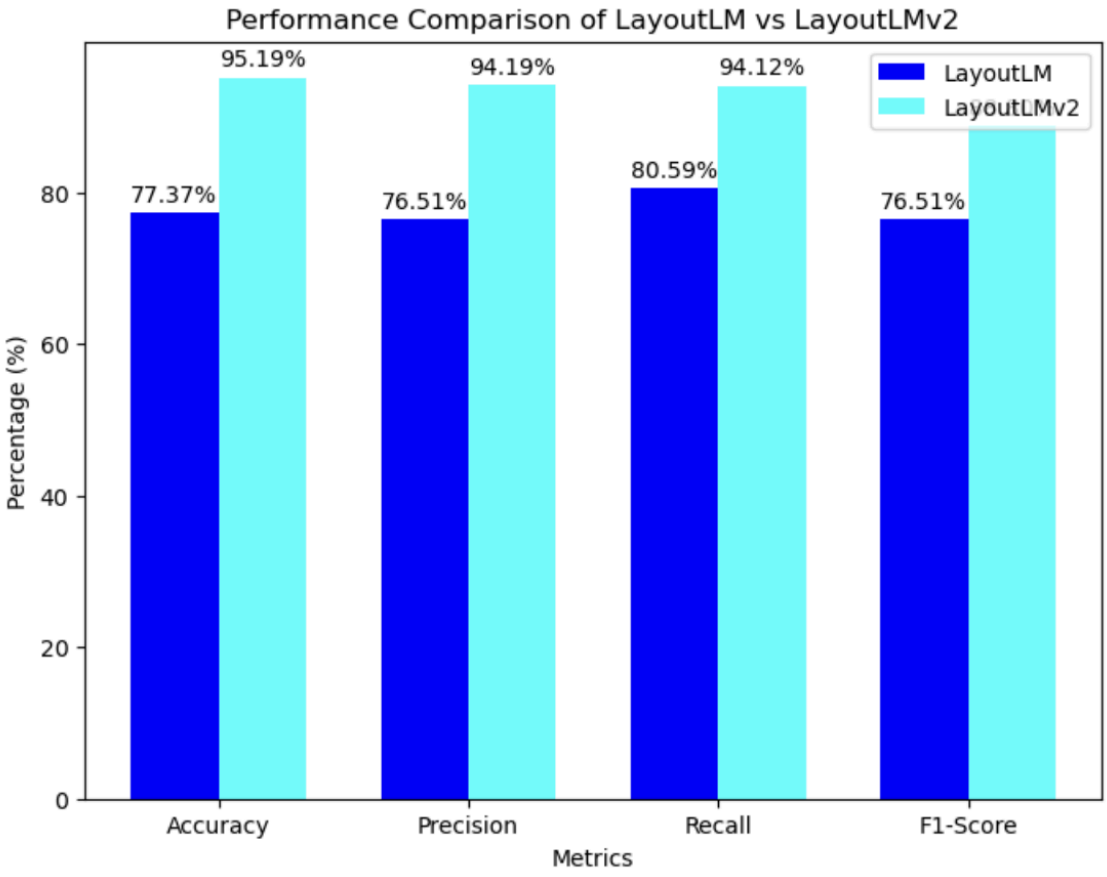


Figure 24: Comparison between LayoutLM & LayoutLMv2

Lower CER and WER indicate better text recognition performance. The lowest CER (0.108) and WER (0.0480) were achieved with a 2e-5 learning rate and 10 epochs, confirming that this setting produces the most accurate text extractions with minimal errors.Overall, LayoutLMv2 with a 2e-5 learning rate and 10 epochs emerged as the best-performing configuration, achieving the highest accuracy, F1-score, and lowest error rates. However, for applications requiring higher precision, a 3e-5 learning rate with 5 epochs may be preferable, while higher recall is obtained with 2e-5 and 5 epochs.

5.3 Discussion

This study evaluates and compares the performance of LayoutLM and LayoutLMv2 models under different hyperparameter settings, including variations in learning rate (2e-5, 3e-5) and epochs (5, 10). The key performance metrics analyzed include accuracy, precision, recall, F1-score, Character Error Rate (CER), and Word Error Rate (WER).

From the results, it is evident that LayoutLMv2 consistently outperforms LayoutLM across most evaluation metrics. The highest accuracy observed for LayoutLMv2 (95.19% with a 2e-5 learning rate and 10 epochs) is significantly higher than that of LayoutLM (77.37% with a 3e-5 learning rate and 5 epochs). This suggests that LayoutLMv2 is more effective in extracting structured data from invoices, likely due to its enhanced architecture and improved spatial-text alignment capabilities.In terms of precision, LayoutLM achieved its best score (75.70% with 5e-5 learning rate and 10 epochs), whereas LayoutLMv2 significantly outperformed it, reaching 94.19% with 3e-5 learning rate and 5 epochs. This indicates that LayoutLMv2 is much more precise in extracting relevant entities while reducing false positives.

For recall, LayoutLM reached its highest value (80.59% with 3e-5 learning rate and 5 epochs), while LayoutLMv2 achieved 94.12% with 2e-5 learning rate and 5 epochs. The superior recall of LayoutLMv2 suggests that it is better at capturing all relevant information without missing important details.The F1-score, which balances precision and recall, was highest for LayoutLM (76.51% with 3e-5 learning rate and 5 epochs) and for LayoutLMv2 (88.80% with 2e-5 learning rate and 10 epochs), reinforcing the superior overall performance of LayoutLMv2.

Lower Character Error Rate (CER) and Word Error Rate (WER) indicate better text extraction accuracy. LayoutLMv2 achieved the lowest error rates (CER = 0.108, WER = 0.0480) with a 2e-5 learning rate and 10 epochs, whereas LayoutLM had a CER of 0.1880 and WER of 0.2258 at its best-performing setting. This highlights the significant reduction in extraction errors with LayoutLMv2.

CHAPTER 6

Conclusions and future scope

Using deep learning models, the Invoice Processing using Machine Learning project effectively automates the extraction of structured data from invoices. A comparison of LayoutLM with LayoutLMv2 showed that LayoutLMv2 performs better than LayoutLM, reaching a peak accuracy of 95.19% with lower Character Error Rate (CER) and Word Error Rate (WER) and higher precision, recall, and F1-score. This demonstrates how well the model comprehends invoice patterns and extracts important elements with increased precision. Furthermore, by decreasing manual labor, enhancing data consistency, and boosting processing performance, the combination of OCR methods, deep learning-based entity extraction, and rule-based validation guarantees dependable invoice processing. But there are still issues that need to be resolved, like managing various invoice formats, enhancing OCR precision, and honing model generalization.

For future enhancements, improving the model's adaptability by incorporating a more diverse dataset and domain-specific pretraining can enhance accuracy across varying invoice structures. Additionally, integrating advanced OCR models like TrOCR or Donut could improve text recognition and reduce extraction errors. Real-time invoice processing can be achieved by deploying the model as an API, making it compatible with enterprise applications such as ERP and accounting software. Enhancing the rule-based validation system with adaptive learning mechanisms will allow the model to refine its data validation process based on past errors. Extending support for multi-language invoices and handwritten text recognition will make the system more versatile for global use. Moreover, incorporating Explainable AI (XAI) techniques can improve transparency, helping businesses ensure compliance with financial regulations and auditability of extracted data. By implementing these improvements, the system can evolve into a highly scalable, intelligent, and efficient end-to-end invoice processing solution, further reducing manual intervention and enhancing automation in financial workflows.

References

[1]. Xu, Yiheng & Li, Minghao & Cui, Lei & Huang, Shaohan & Wei, Furu & Zhou, Ming. (2019). LayoutLM: Pre-training of Text and Layout for Document Image Understanding. 10.48550/arXiv.1912.13318.

[2]. Baviskar, D., Ahirrao, S. and Kotecha, K., 2021. Multi-layout unstructured invoice documents dataset: A dataset for template-free invoice processing and its evaluation using AI approaches. *IEEE Access*, *9*, pp.101494-101512.

[3]. Krieger, F., Drews, P. and Funk, B., 2023. Automated invoice processing: Machine learning-based information extraction for long tail suppliers. *Intelligent Systems with Applications*, *20*, p.200285.

[4] Saout, T., Lardeux, F. and Saubion, F., 2024. An Overview of Data Extraction From Invoices. *IEEE Access.*

[5] Chazhoor, Anisha & Sarobin, Vermin. (2022). Intelligent automation of invoice parsing using computer vision techniques. Multimedia Tools and Applications. 81. 10.1007/s11042-022-12916-x.

[6] Arslan, H., 2022. End to end invoice processing application based on key fields extraction. *IEEE access*, *10*, pp.78398-78413.

[7] Baviskar, D., Ahirrao, S., Potdar, V. and Kotecha, K., 2021. Efficient automated processing of the unstructured documents using artificial intelligence: A systematic literature review and future directions. IEEE Access, 9, pp.72894-72936.

[8] Perin, E.L.S., de Souza, M.C., de Andrade Silva, J. and Matsubara, E.T., 2024. DynGraph-BERT: Combining BERT and GNN Using Dynamic Graphs for Inductive Semi-Supervised Text Classification.

[9] Devika, R., Vairavasundaram, S., Mahenthar, C.S.J., Varadarajan, V. and Kotecha, K., 2021. A deep learning model based on BERT and sentence transformer for semantic keyphrase extraction on big social data. *IEEE Access*, *9*, pp.165252-165261.

[10] Giarelis, Nikolaos & Karacapilidis, Nikos. (2024). Deep learning and embeddings-based approaches for keyphrase extraction: A literature review. Knowledge and Information Systems. 66. 6493-6526. 10.1007/s10115-024-02164-w.

[11]. Gon, Anudeepa & Mukherjee, Gunjan & Chanda, Kaushik & Nandi, Subhadip & Ganguly, Aryabhatta. (2024). BERT Model: A Text Classification Technique in NLP.

[12]. Qiu, Qinjun & Wang, Yang & Hao, Mengqi & Liu, Jiandong & Li, Weijie & Tao, Liufeng & Xie, Zhong. (2024). Document image layout detection from scientific literature using combined ConvNext and cascade mask RCNN networks. International Journal on Document Analysis and Recognition (IJDAR). 1-20. 10.1007/s10032-024-00508-4.

[13]. Kiatphaisansophon, Phanthakan & Wanvarie, Dittaya & Cooharojananone, Nagul. (2024). Efficient Text Bounding Box Identification Using Mask R-CNN: Case of Thai Documents. IEEE Access. PP. 1-1. 10.1109/ACCESS.2024.3383911.

[14] Xu, Yang & Xu, Yiheng & Lv, Tengchao & Cui, Lei & Wei, Furu & Wang, Guoxin & Lu, Yijuan & Florencio, Dinei & Zhang, Cha & Che, Wanxiang & Zhang, Min & Zhou, Lidong. (2020). LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding. 10.48550/arXiv.2012.14740.

[15] Appalaraju, Srikar & Jasani, Bhavan & Urala Kota, Bhargava & Xie, Yusheng & Manmatha, R.. (2021). DocFormer: End-to-End Transformer for Document Understanding. 10.48550/arXiv.2106.11539.

[16].Wang, Zilong & Zhou, Yichao & Wei, Wei & Lee, Chen-Yu & Tata, Sandeep. (2022). A Benchmark for Structured Extractions from Complex Documents. 10.48550/arXiv.2211.15421.

[17]. Oussaid, Ismail & Vanhuffel, William & Ratnamogan, Pirashanth & Hajaiej, Mhamed & Mathey, Alexis & Gilles, Thomas. (2021). Information Extraction from Visually Rich Documents with Font Style Embeddings.

[18]. Graliński, F., Stanisławek, T., Wróblewska, A., Lipiński, D., Kaliska, A., Rosalska, P., Topolski, B. and Biecek, P., 2020. Kleister: A novel task for information extraction involving long documents with complex layout. *arXiv preprint arXiv:2003.02356*.

[19]. Stanisławek, T., Graliński, F., Wróblewska, A., Lipiński, D., Kaliska, A., Rosalska, P., Topolski, B. and Biecek, P., 2021, September. Kleister: key information extraction datasets involving long documents with complex layouts. In *International Conference on Document Analysis and Recognition* (pp. 564-579). Cham: Springer International Publishing.

[20] Salgado, A. and Sánchez, J., 2023, June. Information extraction from electricity invoices through named entity recognition with Transformers. In *Proceedings of the 5th International Conference on Advances in Signal Processing and Artificial Intelligence, Tenerife, Spain* (pp. 7-9).