



INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Optical Character Recognition, Text Analysis and Key Information Extraction

Samuel Filipe Agostinho dos Santos

Master in Computer Engineering

Supervisor:

Doctor Fernando Manuel Marques Batista, Associate Professor,
Iscte – Instituto Universitário de Lisboa

Co-Supervisor:

Doctor Ricardo Daniel Santos Faro Marques Ribeiro, Associate
Professor, Iscte – Instituto Universitário de Lisboa

September, 2024



TECHNOLOGY
AND ARCHITECTURE

Department of Information Science and Technology

Optical Character Recognition, Text Analysis and Key Information Extraction

Samuel Filipe Agostinho dos Santos

Master in Computer Engineering

Supervisor:

Doctor Fernando Manuel Marques Batista, Associate Professor,
Iscte – Instituto Universitário de Lisboa

Co-Supervisor:

Doctor Ricardo Daniel Santos Faro Marques Ribeiro, Associate
Professor, Iscte – Instituto Universitário de Lisboa

September, 2024

*I want to dedicate this work to the people I love, with a special reference to my parents,
to my brother, and to my life long partner for all the love and support they give me
everyday*

Acknowledgment

I want to thank Professor Fernando Batista and Professor Ricardo Ribeiro for all the patience to guide me throughout my thesis and for teaching me many important concepts. Also, I want to thank Professor Alexandre Ferraz for teaching me not only how to elaborate algorithms and write code, but also by sharing the basis that helped me pursue the academic and professional life that I have today. Lastly, I want to thank my parents, my brother, and my life long partner for always being there for me!

Resumo

Na atual era digital, a extração eficiente de dados de documentos digitais é essencial para a gestão da informação e para a automatização de processos. Esta tese investiga vários métodos de Reconhecimento Ótico de Caracteres (OCR), Análise de Texto (TA) e Extração de Informação Chave (KIE) a partir de documentos digitais, com um foco particular em recibos e facturas. Existem vários desafios associados à extração manual de dados, como o caso da fraca eficiência na gestão do tempo e o tratamento de erros no processamento de documentos.

Ao longo deste trabalho, examinamos a evolução dos métodos KIE, desde abordagens baseadas em regras e modelos até técnicas contemporâneas de Aprendizagem Automática e Modelos Generativos que superam os métodos tradicionais. Será utilizado um [dataset](#) de recibos digitalizados [SROIE](#) disponibilizado pelo [ICDAR](#), no qual iremos fazer extração de informação chave nesses mesmos recibos. Apesar de não serem tão eficientes, as abordagens tradicionais, como a utilização de regex, continuam a ser eficazes na extração de campos específicos, como números e datas. Identificamos desafios significativos em OCR, TA e KIE, incluindo má qualidade de imagem, tamanhos e estilos de letra variados e diferentes orientações de texto, e apresentamos potenciais soluções para estes desafios.

PALAVRAS CHAVE: Extração de Informação Chave, Análise de Texto, Reconhecimento Ótico de Caracteres, Aprendizagem Automática, Inteligência Artificial

Abstract

In today's digital era, efficiently extracting data from digital documents is essential for managing information and automation processes. This thesis researches various methods of [OCR](#), [TA](#), and [KIE](#) from digital documents, particularly focusing on scanned receipts and invoices. There are several challenges associated with manual data extraction and the potential for automation to save time and reduce errors regarding document management. There are several challenges associated with manual data extraction, such as poor time management efficiency and error handling in document processing.

Throughout this work, we examine the evolution of [KIE](#) methods from rule-based and template-based approaches to contemporary Machine Learning techniques and Generative Models that outperform traditional methods. We will be using a [SROIE dataset](#) provided by [ICDAR](#), from which we will extract key information. Despite not being as efficient, traditional approaches like the usage of regex remain effective at extracting specific fields, such as numbers and dates. We identify significant challenges in [OCR](#), [TA](#), and [KIE](#), including poor image quality, varied font sizes and styles, and different text orientations, and provide with potential solutions for these challenges.

KEYWORDS: Key Information Extraction, Text Analysis, Optical Character Recognition, Machine Learning, Artificial Intelligence

Contents

Acknowledgment	iii
Resumo	v
Abstract	vii
List of Figures	xi
List of Tables	xiii
List of Acronyms	xv
Chapter 1. Introduction	1
1.1. Motivation	1
1.2. Scope	2
1.3. Research Questions	2
1.4. Objectives	3
1.5. Research Method	3
1.6. Document Structure	4
Chapter 2. Relevant Concepts and Resources	7
2.1. Performance Metrics	7
2.2. Machine Learning	8
2.3. Neural Networks	11
2.4. Natural Language Processing	12
Chapter 3. Related Work	15
3.1. Key Information Extraction Approaches	15
3.2. Datasets	16
Chapter 4. Experiments and Results	23
4.1. ICDAR 2019 Challenge: Robust Reading Challenge on Scanned Receipts OCR and Information Extraction	23
4.2. Processing Pipeline	25
4.3. Approach: Custom Rules	25
4.3.1. Setup	26
4.3.2. Tesseract Setup	26
4.3.3. Defining Custom Rules	27
4.3.4. Results	28
	ix

4.4. Approach: Generative Models	29
4.4.1. Setup	29
4.4.2. Data Analysis	32
4.4.3. Results	32
4.4.4. Error Analysis	35
4.5. Summary	41
Chapter 5. Conclusions and Future Work	43
References	45
Glossary	49

List of Figures

Figure 1.1	DSRM Process Model	4
Figure 2.1	Confusion Matrix	7
Figure 3.1	Examples of CORD Scanned Receipts	17
Figure 3.2	Examples of SROIE Scanned Receipts	17
Figure 3.3	Examples of FUNSD Scanned Forms	18
Figure 3.4	Examples of Typical and Challenging Instances in EPHOIE	19
Figure 3.5	Example of Medical Invoice	20
Figure 3.6	Examples of Train Tickets	21
Figure 4.1	Example of a Scanned Receipt	24
Figure 4.2	Approach Workflow	26
Figure 4.3	SROIE Key for Receipt 625	29
Figure 4.4	SROIE Image for Receipt 625	30
Figure 4.5	SROIE Bounding Boxes for Receipt 625	31
Figure 4.6	Tesseract OCR and OpenAI GPT Outputs	33
Figure 4.7	Receipt 189	34
Figure 4.8	Receipt 189. Ground truth vs Tesseract+OpenAI keys	34
Figure 4.9	Receipt 510	35
Figure 4.10	Receipt 510. Ground truth vs Tesseract+OpenAI keys	36
Figure 4.11	Receipt 543, the border shows the amount of white space that exists on the original image	37
Figure 4.12	Receipt 543. Ground truth vs Tesseract+OpenAI keys	38
Figure 4.13	Tesseract and OpenAI Outputs	38
Figure 4.14	OpenAI GPT Outputs on Ground Truth Texts	39
Figure 4.15	OpenAI Outputs from Ground Truth Texts	40

List of Tables

Table 3.1	Overall evaluation results from the models of each paper	16
Table 4.1	Tesseract default - first set of tests	28
Table 4.2	Tesseract configured - first set of tests	28
Table 4.3	Tesseract default - second set of tests	28
Table 4.4	Tesseract configured - second set of tests	28
Table 4.5	Results using gpt on tesseract ocr outputs	32
Table 4.6	Results using gpt on ground truth texts	38

List of Acronyms

–oem: Engine Mode.
–psm: Page Segmentation Mode.
AI: Artificial Intelligence.
ANN: Artificial Neural Network.
API: Application Programming Interface.
BERT: Bidirectional Encoder Representations from Transformers.
CM: Confusion Matrix.
CNN: Convolutional Neural Network.
CORD: Consolidated Receipt Dataset.
CSV: Comma-Separated Values.
DL: Deep Learning.
FN: False Negative.
FNR: False Negative Rate.
FP: False Positive.
FPR: False Positive Rate.
FUNSD: Form Understanding in Noisy Scanned Documents.
ICDAR: International Conference on Document Analysis and Recognition.
IE: Information Extraction.
IIT-CDIP: Illinois Institute of Technology - Complex Document Information Processing.
JPG: Joint Photographic Experts Group.
JSON: JavaScript Object Notation.
KIE: Key Information Extraction.
KNN: K Nearest Neighbour.
LSTM: Long Short-Term Memory.
mAP: Mean Average Precision.
MIM: Masked Image Modeling.
ML: Machine Learning.
MLM: Masked Language Modeling.
MLP: Multilayer Perceptron.
NER: Named Entity Recognition.
NLP: Natural Language Processing.
NN: Neural Network.
OCR: Optical Character Recognition.
RegEx: Regular Expression.

RNN: Recurrent Neural Network.
RVL-CDIP: Ryerson Vision Lab - Complex Document Information Processing.
RvNN: Recursive Neural Network.
SciTSR: Complicated Table Structure Recognition.
Seq2Seq: Sequence-to-Sequence.
SNN: Shallow Neural Network.
SROIE: Scanned Receipts OCR and Information Extraction.
SVM: Support Vector Machines.
TA: Text Analysis.
TM: Text Mining.
TN: True Negative.
TNR: True Negative Rate.
TP: True Positive.
TPR: True Positive Rate.
VIE: Visual Information Extraction.
WPA: Word-Patch Alignment.

CHAPTER 1

Introduction

This chapter explains the motivation behind our work, its scope, the research questions we aim to answer, our objectives with it, as well as the research method used, and the way this document is structured. The motivation concerns three essential points: the information overload, the need for automation and the potential challenges regarding [Information Extraction \(IE\)](#). The scope focuses on giving insights about the work that is going to be developed. The research questions aim to address topics such as, practical applications, main challenges, and state-of-the-art techniques. The objectives that we want with this work, in this case the main goal is to perform [Optical Character Recognition \(OCR\)](#), [Text Analysis \(TA\)](#), [Text Mining \(TM\)](#) and [Key Information Extraction \(KIE\)](#), on digital documents. The research method explains each step taken during the work developed.

1.1. Motivation

In today's digital age, with the rapid inflow of information, the ability to efficiently extract data from digital documents, has become a subject of important matter. However, a significant amount of businesses and organizations still handle information in physical forms, such as receipts, invoices and contracts. Since they might be connected to transactions, record keeping and accountability, for many industries, including personal financial management, retail, accounting and auditing, these documents are often crucial. Even though some organizations have already shifted from paper to digital, by storing records in folders and repositories, the mere act of digitizing physical documents, without subsequent comprehensive processing does not add that much value. In fact, for an organization that needs to keep tracking thousands of records, and possibly having to sort or filter those same records, extracting the necessary information from these digital documents will be time consuming and also prone to errors, when done manually.

This is where [OCR](#) and [TA](#) come in. [OCR](#) is a technology that allows you to convert handwritten, printed, scanned, and image-based text into machine-understandable text [1]. Furthermore, [TA](#) is the practice of analyzing various collections of text, in order to identify key concepts and relationships between elements [2]. By combining these tools, organizations can save time collecting valuable data from documents.

Extracting data from scanned documents, stored as images, can be quite a challenge. Let us take the example of receipts. These documents contain essential financial information, such as details of transactions, vendor information, purchase amounts and timestamps, which provide with useful insights for budgeting, expense tracking and compliance. However, the process of extracting this crucial information from scanned images is not

that simple. Diverse layouts, different fonts or even the presence of image noise, these are all constraints that affect significantly, in a negative way, the accuracy and efficiency of [IE](#) methods.

1.2. Scope

After some research on [datasets](#) and challenges, it was concluded that the scope of this thesis will be centered around the [International Conference on Document Analysis and Recognition \(ICDAR\)](#) 2019 Robust Reading Challenge on [Scanned Receipts OCR and Information Extraction \(SROIE\)](#). This challenge, organized by the [ICDAR](#), presents a unique opportunity to address the complex processes of [OCR](#) and [KIE](#) from structured and semi-structured receipts and invoices.

The [ICDAR](#) 2019 Robust Reading Challenge focuses on two primary objectives [3]:

- **Scanned Receipts OCR:** This involves accurately recognizing text from scanned receipts, specially in scenarios where receipts exhibit various layouts and may suffer from low image quality.
- **Information Extraction (IE):** Apart from [OCR](#), the challenge emphasizes the extraction of key information from receipts and invoices, structuring this data into documents. This aspect holds immense potential for applications, such as [efficient archiving](#), [fast indexing](#) and [document analytics](#).

Although the main objective is to perform [KIE](#), the research will make use of a variety of technologies and tools. This includes [OCR libraries](#), [Natural Language Processing \(NLP\) frameworks](#), [Machine Learning \(ML\) models](#), and any custom software or [algorithm](#) developed to meet the challenge requirements.

It is necessary to clarify that the research will not cover the participation in the challenge itself. The intention is to analyze the work that was already made, by other participants, in order to gather insights and possibly improve their solutions.

1.3. Research Questions

Regarding the research questions, each of them is designed to address a specific problem or challenge in the context of performing [KIE](#) on scanned receipts:

- Receipt digitization is already a regular practice, however it is frequently not used to its full potential for obtaining useful data. Because of this, the first research question aims to understand the practical applications of [KIE](#) on scanned receipts, and to highlight the potential benefits and advancements in various areas, including financial management, auditing and compliance.
- Diverse layouts, various fonts and imagine noise, these are all factors that provide problems to the extraction of important information from scanned receipts. It is important to identify and describe these issues with more detail, in order to better understand the constraints regarding the extraction of important information, and have them into account when developing [KIE](#) techniques that address these same issues.

- **ML** and **NLP** play a crucial role on improving the accuracy of **KIE**. For that reason, the final question aims to identify the specific techniques and methods that are most suitable to perform **KIE** on scanned receipts, therefore improving its accuracy.

1.4. Objectives

This section outlines the main objectives of this thesis, and explains why each one of them is essential:

TM forms the foundation for **KIE**. Therefore, the first objective aims to explore various **ML** and **NLP** techniques for **TM**. Since, understanding and processing text data are critical components when performing **KIE** on scanned receipts.

Performing **KIE** on scanned receipts, this objective represents the practical implementation of the research itself. By applying the techniques and approaches developed and evaluated in the previous objective, the main goal is to demonstrate the feasibility and effectiveness of automated **KIE** on scanned receipts.

This research aims to explore how **KIE** from scanned receipts can benefit businesses and organizations by analyzing its real-world applications and implications. In order to connect **KIE** technology with its practical uses.

The final objective aims to identify the main challenges regarding **KIE** on scanned receipts, and present potential solutions or improvements.

1.5. Research Method

The research method for this thesis follows the Design Science Research Process model. This approach suits the research objectives, since it emphasizes the creation of innovative solutions to practical problems. This particular model is divided into six phases [4], as can be seen in the diagram in [Figure 1.1](#):

- (1) **Problem Identification**, which involves defining the specific research problem and justifying the value of the solution to be presented. In this case, the problem domain centers on the challenges of **OCR** and **KIE** from scanned receipts and invoices, as highlighted by the **ICDAR 2019 Robust Reading Challenge**.
- (2) **Objectives of the solution**, which focus on resolving the identified problem. In this situation, exploring advanced **OCR** techniques, developing efficient methods **KIE** and integrating these components to create a powerful **SROIE** system, which is consistent with the general objective of lowering manual intervention in procedures that include processing large quantities of documents.
- (3) **Design and Development**, presentation of proposals for architectures and technologies to be used that best suit the production of the system for this thesis. The research includes the design and development of **OCR models**, **KIE algorithms**, and the integration with the **framework** for **SROIE**, always focusing on innovation and practical applicability.

- (4) **Demonstration** of the effectiveness of the tool developed to solve the problem at hand. This may involve conducting experiments, simulations, case studies or other appropriate activities. In this case, by illustrate how the [OCR](#) and [KIE](#) systems work cohesively to extract valuable information from scanned receipts and invoices. This step serves to make the research tangible and accessible to a wider audience.
- (5) **Evaluation** based on observation and measurement, to understand how well the tool can handle a solution to the problem. This process involves comparing the intended goals with the results shown. For this thesis, the evaluation will be based on the evaluation scripts from the [ICDAR 2019](#) challenge.
- (6) **Communication**, in which it is intended to communicate the problem at hand, the proposed solution, as well as its usefulness, innovation and efficiency, even being able to compare it with other similar existing [frameworks](#).

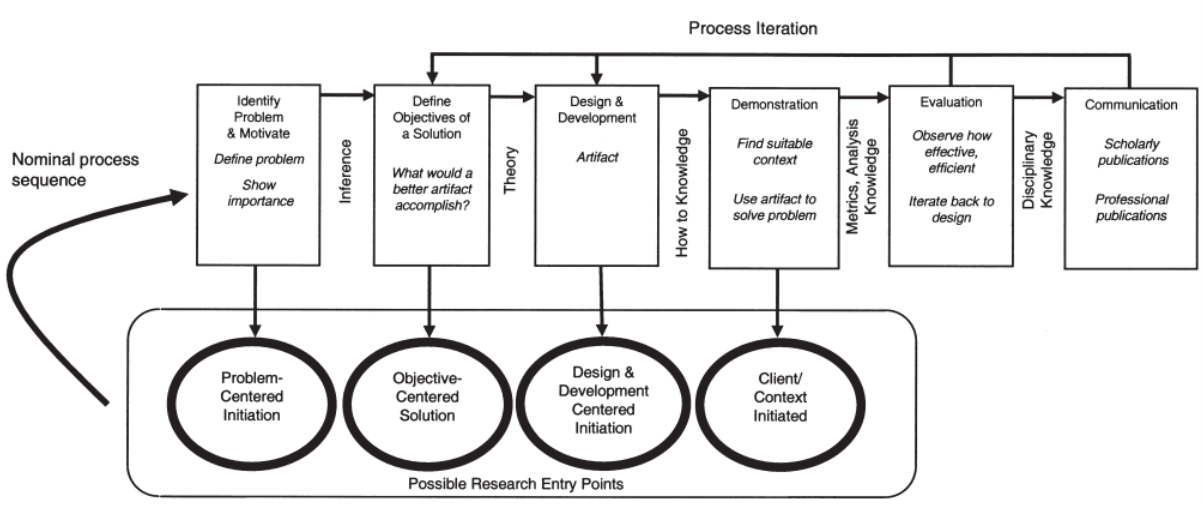


FIGURE 1.1. DSRM Process Model [4]

The model may not stick precisely to the initial plan [4] because research is usually a flexible and evolving process.

1.6. Document Structure

This section aims to describe the organization of this document, which follows:

- **Chapter 1 - Introduction**, which presents the motivation behind subject of this paper, the scope of our work, the research questions intend to answer, the objectives of this work, and the research method used, alongside the document structure.
- **Chapter 2 - Relevant Concepts and Resources**, this chapter explains some key concepts crucial to understand the work being developed, such as performance metrics, [ML](#), [Neural Network \(NN\)](#), [NLP](#), and also useful tools that are commonly used in this type of work.

- **Chapter 3 - Related Work**, aims to explore related work regarding [OCR](#), [TA](#), [TM](#) and [KIE](#), and the current state-of-the-art regarding [OCR](#) and [KIE](#).
- **Chapter 4 - Experiments and Results**, focuses on exploring the ICDAR 2019 Challenge and addressing it by using different approaches.
- **Chapter 5 - Conclusions and Future Work**, aims to summarize and reflect on the work done, as well as answering the research questions.

CHAPTER 2

Relevant Concepts and Resources

This chapter explains relevant concepts aligned with [OCR](#), [TA](#) and [KIE](#), as well as informing about useful resources that can be used for related work. We started by explaining the performance metrics normally that are normally used for test validations. Then we proceed with [ML](#) concepts, categories and algorithms. After that we give an introduction to [NN](#), [Deep Learning \(DL\)](#) and types of [NNs](#). We then talk about [NLP](#), and finalize this chapter by mentioning useful tools that can be used either for [ML](#), [NN](#) and [NLP](#).

2.1. Performance Metrics

This section explains performance metrics when dealing with imbalanced data. It introduces the Confusion Matrix and related terms like Precision, Recall and F1 Score. These metrics have several applications, such as in information retrieval, word segmentation and [Named Entity Recognition \(NER\)](#) [5].

Starting of with the Confusion Matrix, it is a 2x2 matrix used in [binary classification](#), as shown in [Figure 2.1](#). It helps visualize the performance of a classification [model](#) by comparing actual values to predicted values.

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

FIGURE 2.1. [Confusion Matrix](#) [6]

- **True Positive (TP)** — [model](#) predicts the positive class accurately (both prediction and actual result are positive).
- **True Negative (TN)** — [model](#) predicts the negative class accurately (both prediction and actual result are negative).
- **False Positive (FP)** — [model](#) wrongly predicts the negative class (predicted-positive, actual-negative). [FP](#) is also referred as a TYPE I error.

- **False Negative (FN)** — **model** wrongly predicts the positive class (predicted-negative, actual-positive). **FN** is also referred as a TYPE II error.

The above values enable the calculation of **True Positive Rate (TPR)**, **True Negative Rate (TNR)**, **False Positive Rate (FPR)** and **False Negative Rate (FNR)**.

$$TPR = \frac{TP}{ActualPositive} = \frac{TP}{TP+FN}$$

$$TNR = \frac{TN}{ActualNegative} = \frac{TN}{TN+FP}$$

$$FPR = \frac{FP}{ActualNegative} = \frac{FP}{TN+FP}$$

$$FNR = \frac{FN}{ActualPositive} = \frac{FN}{TP+FN}$$

Even with imbalanced data, it is possible to figure out if a **model** is predicting well or not. In order to achieve that, **TPR** and **TNR** should have high values, while **FPR** and **FNR** should be as low as possible.

Regarding Precision, this metric indicates the proportion of all positive predictions that are actually positive. Its value lies between 0 and 1. A higher value indicates that the **model** makes fewer **False Positive (FP)** errors.

$$Precision = \frac{TP}{TP+FP}$$

The Recall is a metric that indicates, based on the total of actual positives, what percentage were predicted positive by the **model**. It is the same as **TPR**. Its value lies between 0 and 1. A higher value indicates that the **model** is more accurate at identifying all positive occurrences, and has fewer **False Negative (FN)** errors.

$$Recall = \frac{TP}{TP+FN}$$

The F1-score is the **harmonic mean** of precision and recall. It takes into account both **FP** and **FN**. As a result, it works well with an unbalanced **dataset**, where one class significantly outnumbers the other. Its value lies between 0 and 1. A higher value indicates the **model** has a better performance.

$$F1\text{-score} = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} = \frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

2.2. Machine Learning

This section aims to describe with detail **Machine Learning** which is a branch of **Artificial Intelligence (AI)** that focuses on developing **models** and **algorithms** that allow computers to make predictions (outputs) by identifying patterns based on data (inputs), without having been explicitly programmed to do so. It requires statistical approaches, in order for machines to become better at predicting outputs over time [7] [8] [9]. **ML** comprises of three fundamental aspects:

- **Task**, the specific problem to be solved, such as making predictions, recommendations or estimations.
- **Experience**, the acquired learning from past data, which is then applied to tackle future tasks.
- **Performance**, measuring the ability of the model to solve a [ML](#) task effectively. Performance, can differ depending on the specific [ML](#) problem. More details about performance metrics can be found in [Section 2.1](#).

[ML](#) techniques can be categorized into four main types: Supervised Learning, Semi-Supervised Learning, Unsupervised Learning and Reinforcement Learning. The following list covers each of these categories:

- **Supervised Learning**, in this approach the machine works with a [dataset](#) that contains both inputs and their corresponding outputs, in order to correct its mistakes during the learning process.
- **Unsupervised Learning**, in this approach the machine works with data that lacks labeled outcomes. It explores the data to discover patterns and structures on its own.
- **Semi-Supervised Learning**, is a mix of supervised and unsupervised learning. It uses [datasets](#) that have some labeled data but mostly unlabeled data. This approach can reduce costs while still improving [model](#) performance.
- **Reinforcement Learning**, uses agents, such as computer programs, to interact with an environment. They take actions and receive rewards or penalties based on the outcome of the decisions they actions perform. The goal is to maximize their performance over time.

[ML algorithms](#) are developed to learn patterns from data, make predictions, and get better results with experience. Various [algorithms](#) serve different purposes, like linear regression for prediction tasks and [K Nearest Neighbour \(KNN\)](#) for classification tasks [10].

- **Linear Regression**, is a widely-used and straightforward [ML algorithm](#) that predicts real variables, such as experience, salary or cost. It establishes a statistical relationship between dependent and independent variables, demonstrating how the dependent variable changes based on the independent variable. This relationship is depicted by a line on a graph known as the Line of Regression.
- **Logistic Regression**, is a supervised learning method used to predict categorical outcomes based on independent variables. It provides probabilistic values between 0 and 1, making it suitable for classification tasks. It resembles Linear Regression but is tailored for classification problems. Logistic Regression employs [sigmoid functions](#) and predicts either 0 or 1 [8] [11]. There are three types of Logistic Regression:
 - **Binomial**, is used for problems where the [model](#) predicts one of two possible classes, such as "yes/no", "spam/ham" or "pass/fail".

- **Multinomial**, is used when there are more than two classes to predict, and these classes have no specific order, such as classifying fruits into categories like "apples", "oranges" and "bananas".
- **Ordinal**, is used when the dependent variable has multiple ordered classes or levels. For instance, classifying customer satisfaction into categories, such as "very dissatisfied", "dissatisfied", "neutral", "satisfied" and "very satisfied".
- **K Nearest Neighbour (KNN)**, is a simple supervised ML algorithm used for both regression and classification tasks. It categorizes new data based on its similarity to existing data. KNN is often called a "Lazy Learner" because it does not learn immediately from the training data but stores it instead. When classifying new data, it compares it to the stored data and assigns it to the category that is most similar. For example, KNN can be used to identify whether an image is of a dog or a cat by comparing it to known dog and cat images in its dataset.
- **K-Means Clustering**, is an unsupervised learning method that groups unlabeled data into clusters. This technique is used for clustering tasks, where the goal is to find natural groupings within the data. The "K" in K-Means specifies the number of clusters to create. For example, if K=2, it creates two clusters, and if K=3, it makes three clusters, and so on.
- **Decision Tree**, is a type of ML method used in supervised learning, mainly for solving classification problems. It is called a "tree" because it has a tree-like structure with decision nodes (internal nodes), branches representing decision rules, and leaf nodes indicating outcomes. The process starts at the root node and ends at a leaf node. Decision Trees are used to make decisions and determine the output of the model based on given conditions. They provide a graphical representation of possible outcomes for a problem or decision.
- **Random Forest**, is a powerful ML algorithm within supervised learning. Like KNN and Decision Trees, it handles both classification and regression problems, but it excels when dealing with complex tasks and enhancing model performance. Random Forest employs ensemble learning, which means it combines multiple classifiers. It is constructed from many decision trees and different subsets of the dataset. The algorithm takes input as an average prediction from all these trees, increasing model accuracy. More trees lead to higher accuracy and prevent overfitting. It is efficient in terms of training time compared to some other algorithms.
- **Support Vector Machines (SVM)**, is a widely-used ML algorithm in supervised learning. Its objective is to establish an optimal decision boundary, called a hyperplane, to separate data into distinct categories in a multi-dimensional

space. **Support Vector Machines (SVM)** addresses both classification and regression tasks, and finds applications in various fields such as face detection, image classification and text categorization.

- **Naïve Bayes**, is a simple and effective supervised **ML algorithm** based on Bayes' Theorem. It is suitable for classification tasks, especially in scenarios with high-dimensional data like text classification. This **algorithm** makes quick and accurate predictions by calculating probabilities. Common applications include spam filtering, sentiment analysis and article classification. The Bayes' Theorem formula can be expressed as follows [12]:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)} \quad (2.1)$$

Where:

- **P(A)**, is the probability of A
- **P(B)**, is the probability of B
- **P(A|B)**, is the probability of A when B happens
- **P(B|A)**, is the probability of B when A happens

2.3. Neural Networks

This section focuses on explaining **NNs**, also referred as **Artificial Neural Networks (ANNs)**, which are artificial systems inspired by biological **NNs**. They learn from data without pre-defined rules, and use computational **models** based on threshold logic, combining math and **algorithms** [13].

Typical components include:

- **Neurons**, receive inputs, have activation functions and produce outputs.
- **Connections or Synapses**, have **weights** and **biases** that control signal transfer.
- **Propagation**, computes inputs, sums neuron functions with **weights** and produces outputs.
- **Learning Rules**, involve adjusting **weights** and **biases** in three steps.

The learning process involves three key steps:

- (1) Simulating the **NN** in a new environment.
- (2) Modifying the **parameters** of the **NN** based on the simulation.
- (3) Observing the response changes to the environment of the **NN** due to the **parameter** adjustments.

DL is a branch of **ML** involving **NNs** with three or more **layers**. These networks aim to mimic the learning process of the human brain by analyzing vast amounts of data. While a single-layer **NN** can make rough predictions, adding hidden **layers** enhances accuracy and optimization [14].

There are several types of **NNs** with different specifications. The following list will cover each of these types of **NNs**.

- **Multilayer Perceptron (MLP)** has multiple [layers](#), including input, hidden and output [layers](#), and uses nonlinear activation functions.
- **Convolutional Neural Network (CNN)** are designed for grid-like data, such as images, using convolutional and pooling [layers](#) to extract features.
- **Recursive Neural Network (RvNN)** works on sequences of variable length, such as text, using weighted connections for structured predictions.
- **Recurrent Neural Network (RNN)** processes sequential data by forming directed cyclic connections between neurons.
- **Long Short-Term Memory (LSTM)** is a [Recurrent Neural Network \(RNN\)](#) variant that overcomes the vanishing gradient problem by using memory cells and gates for selective information handling.
- **Sequence-to-Sequence (Seq2Seq)** uses two [RNNs](#) to map input sequences to output sequences, such as language translation.
- **Shallow Neural Network (SNN)** has only one hidden [layer](#) and is often used for simpler tasks or as a component in larger networks.

2.4. Natural Language Processing

This section describes with detail [NLP](#), which is a branch of [AI](#) responsible for the interaction between computers and humans using natural language. It focuses on developing methods for machines to understand, interpret and generate human language and its semantics [15].

[NLP](#) includes a broad range of approaches and procedures for handling data from human language. The following list will cover each of the most commonly used [NLP](#) techniques and tasks [15].

- **Sentiment Analysis** is the analysis of data, such as text and speech, to establish its positivity, neutrality or negativity. It assigns a sentiment tag to each statement before summing up all of the statements in a particular [dataset](#). In this way, sentiment analysis can transform large repositories of user comments, reviews or social media reactions, into measurable outcomes. These outcomes can then be examined for further strategic results and customer intelligence [15].
- **Named Entity Recognition (NER)** involves identifying and categorizing important information in text into predefined groups, such as Person, Organization and Place. [NER](#) is a core part of [NLP](#), which has two main steps: finding entities in text and sorting them into categories. [NER](#) can also handle tasks like recognizing date/time, expressions, measurements and email addresses [16].
- **Text Summarization** is the process of transforming the text from a document, by reducing the number of sentences and words contained within it, without changing its original meaning [17].

- **Topic Modeling** is an unsupervised ML technique that identifies word and phrase patterns from a collection of scanned documents, from there it automatically clusters word groups and related phrases that share common topics [18] [15].
- **Text Classification** involves assigning predefined categories or labels to text from scanned documents. The goal is to automatically classify text documents into one or more predefined categories based on their content [15] [19].
- **Keyword Extraction** is the automated process of locating and extracting relevant words or phrases from a text document. The most important terms in the text are represented by these keywords [15] [20].
- **Lemmatization and Stemming** are both text normalization techniques used to break down words into their base or root forms. By eliminating nuances and variations, these methods contribute to the standardization of text data, and make it easier to compare and analyze words [15] [21].
 - **Stemming** is a simpler and faster text normalization technique, that consists on removing suffixes from words in order to obtain their root form, also called stem. The goal of stemming is to reduce words to their most basic form, even if the resulting stem may not be a real word. Stemming algorithms work by removing frequent suffixes using a set of rules or heuristics.
 - **Lemmatization algorithms** often employ linguistic rules and databases to identify the lemma of a word. They are more accurate but slower than stemming since they take into account the grammatical and semantic context.

CHAPTER 3

Related Work

This chapter aims to explore related work regarding [OCR](#), [TA](#) and [KIE](#). We want to understand its inherent challenges, study what has been done to solve them, and identify possible approaches to effectively develop a robust solution.

First, we start by reading and analyze several research papers that focus on these technologies, [OCR](#), [TA](#) and [KIE](#), in order to understand its state-of-the-art. We then compare each paper to identify points of convergence and divergence between the solutions presented by their authors.

3.1. Key Information Extraction Approaches

In recent years, deep learning approaches for [KIE](#) have emerged and outperformed traditional rule-based and template-based methods. Traditional approaches use hand-craft features, such as regex and template matching, they make use of textual and positional information to apply rules and therefore extract entities. Many methods consider [KIE](#) as a sequence tagging problem and implement solutions using [NER](#), these entity extraction approaches cause ambiguity, specially when dealing with complex documents.

We were able to identify three main types of document representation in [KIE](#), sequence-based, grid-based and graph-based. Sequence-based serializes a document into a 1D text sequence, grid-based generates a 2D grid of token embeddings, graph-based model a document into a graph and turn each text segment into a node. In general, graph-based approaches reach state-of-the-art results in benchmark datasets, however ViBERTgrid [22] which was inspired by BERT [23] and BERTgrid [24], uses a grid layout that has reached comparable results as well.

Besides document representation categorization, models can be pre-trained with different modalities, such as textual information, visual features and document layout. By using more modalities, specially visual features, pre-trained models require additional computational costs and demand more effective combinations of texts and their spatial information. PICK [25] was one the first attempts to efficiently make full use of all modalities by utilizing transformers. Some other approaches, such as BROS [26] try to simplify by removing visual features from the equation and only use text and layout information, by doing so they remove computational complexity. On the case of Fast-StrucTexT [27], the authors also intend to remove computational complexity, and they achieve it by using an hourglass transformer architecture and performing dynamic token merging. To remove ambiguity, different methods have been proposed such as in the case of FormNetV2 [28], where a graph contrastive learning objective is used to better identify neighbouring tokens

and getting rid of corrupted inputs. With LayoutLMv3 [29] we can also see a reduction in the number of input parameters compared to previous versions by implementing a unified model architecture and pre-trained objectives, such as [Masked Language Modeling \(MLM\)](#), [Masked Image Modeling \(MIM\)](#) and [Word-Patch Alignment \(WPA\)](#), that can be used for both text-centric and image-centric tasks.

[Table 3.1](#) represents the F1 Scores of each model regarding the three most commonly used benchmark datasets, they were taken from their respective papers.

TABLE 3.1. Overall evaluation results from the models of each paper

Model	Datasets		
	FUNSD	CORD	SROIE
PICK	96.1	—	—
ViBERTgrid (BERT _{BASE})	—	—	96.25
ViBERTgrid (RoBERTa _{BASE})	—	—	96.40
LayoutLMv3 _{BASE}	90.29	96.56	—
LayoutLMv3 _{LARGE}	92.08	97.46	—
Fast-StrucTexT _{Linear}	89.50	96.65	97.12
Fast-StrucTexT _{ResNet-18}	90.35	97.15	97.55
FormNetV2	86.35	97.37	98.31
BROS _{BASE}	83.05	96.50	96.28
BROS _{LARGE}	84.52	97.28	96.62

3.2. Datasets

This section discusses various important [datasets](#) that have significantly contributed to the advancement of [OCR](#) and [KIE](#) tasks in the field of document analysis and understanding. Each of these [datasets](#) plays a crucial role in addressing different challenges related to [OCR](#) and [KIE](#), offering a wide range of document types, layouts and complexities. This section will provide an overview of these [datasets](#), their characteristics, and their significance in the context of [OCR](#) and [KIE](#) research.

- [Complicated Table Structure Recognition \(SciTSR\) dataset](#) contains 15,000 tables in PDF format, along with their structure labels from LaTeX source files. It is split into 12,000 training images and 3,000 test images, serving as a [dataset](#) for table structure recognition [30].
- [Consolidated Receipt Dataset \(CORD\)](#) contains over 11,000 Indonesian receipts from shops and restaurants. It has a detailed classification system, including five main categories and 42 subcategories. Additionally, it includes group annotations for multi-level hierarchy analysis [31]. [Figure 3.1](#) contains three examples of three different scanned receipts from the [CORD dataset](#).
- [SROIE dataset](#) contains 1,000 scanned receipt images for use in three competition tasks, as mentioned in [Section 4.1](#). These images have various annotations. Each receipt image typically has four key text fields like product names and prices. Most of the text in the annotations consists of numbers and English characters [3].



FIGURE 3.1. Examples of CORD Scanned Receipts [32]

It is split into 600 training images and 400 test images. For receipt detection and OCR tasks, each image is annotated with text bounding boxes and the text inside them. These annotations are stored in text files.

For the IE task, the dataset also has annotations in a specific text file format. These annotations guide how to extract important data from the receipt images.

Figure 3.2 shows five different scanned receipts from the SROIE dataset, presenting variations between them regarding dimensions, orientation, image quality, condition (wear and tear) and font (style, color, family and size).



FIGURE 3.2. Examples of SROIE Scanned Receipts [32]

- **Form Understanding in Noisy Scanned Documents (FUNSD)** dataset is designed for document understanding and contains 199 fully annotated forms with a total of 31,485 words. It includes 9,707 semantic entities and 5,304 relations. The **dataset** focuses on semantic entity labeling, where entities are categorized as question, answer, header or other. It is split into 149 training samples and 50 testing samples. **FUNSD** aims to address the challenges of noisy scanned documents, and is used for tasks like semantic entity labeling and linking [33].

Figure 3.3 shows three examples of forms with different layouts from the FUNSD dataset.

The figure displays three distinct scanned forms from the FUNSD dataset. The first form, 'ACUTE TOXICITY IN MICE', is a laboratory report with handwritten entries for sample details and a table of results. The second form, 'Manuscript Review Form', is for 'TOBACCO SCIENCE' and includes fields for authors, title, and reviewer comments. The third form, 'Cigarette Request Form', contains checkboxes for various cigarette specifications and handwritten notes.

FIGURE 3.3. Examples of FUNSD Scanned Forms [32]

- **EPHOIE dataset** contains 1,494 images taken from actual Chinese school exam papers. These images focus on the important header sections of the papers, containing both handwritten and printed Chinese text in various shapes. The **dataset** features diverse layouts and noisy backgrounds. It provides detailed annotations for 15,771 text instances, categorized into ten predefined labels. The **dataset** is divided into 1,183 training images and 311 testing images, making it a valuable resource for tasks, such as **OCR** and **Visual Information Extraction (VIE)** from exam papers [34].

Figure 3.4 shows examples of complex layouts and also noisy background in papers from the EPHOIE dataset.

- **Payment dataset** contains approximately 10,000 documents that have been labeled with 7 semantic entities by human annotators [35].
- **Illinois Institute of Technology - Complex Document Information Processing (IIT-CDIP) dataset** contains around 11 million scanned document pages, primarily used for **pre-training** purposes [36].
- **PubLayNet dataset** contains 358,353 research paper images annotated with **bounding boxes** and polygonal segmentation for five document layout categories:

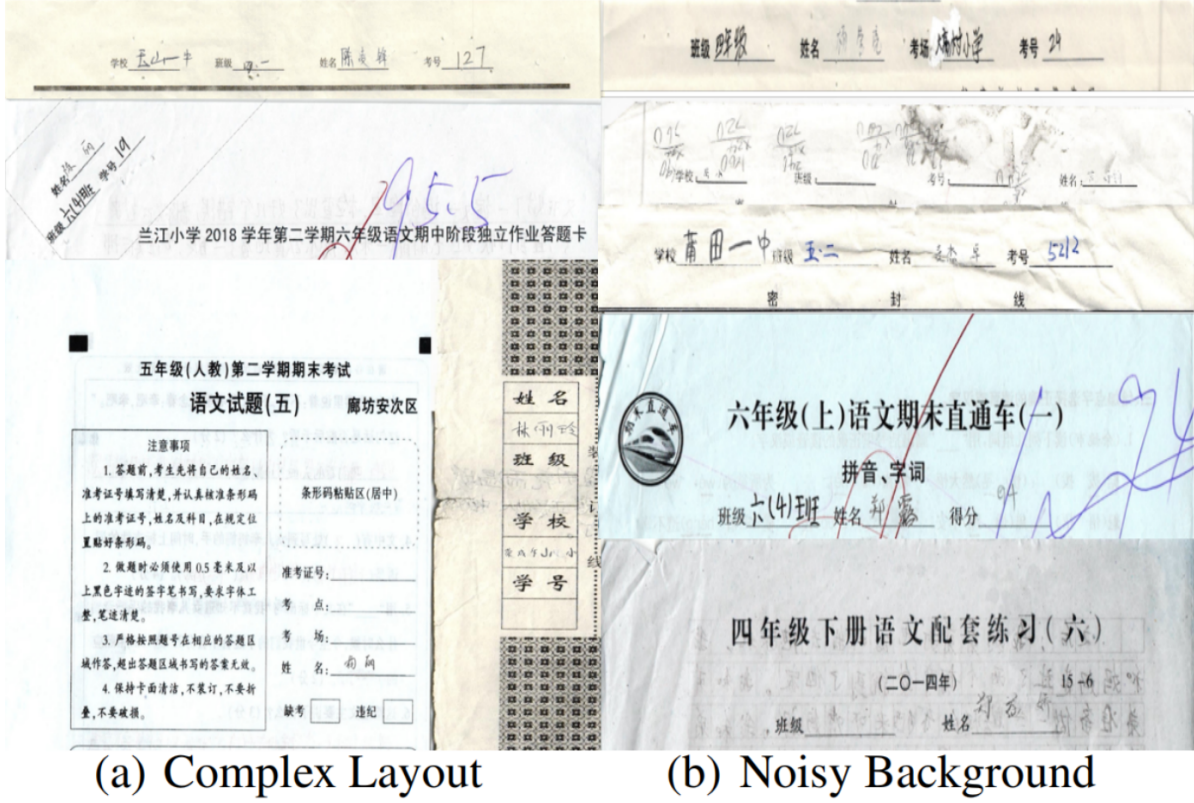


FIGURE 3.4. Examples of Typical and Challenging Instances in EPHOIE [32]

text, title, list, figure and table. It is split into 335,703 training images, 11,245 validation images and 11,405 test images [37].

- **DocVQA dataset** contains 12,767 document images and 50,000 questions designed for visual question answering on these document images. The **dataset** is split into 10,194 training images, 1,286 validation images, and 1,287 test images, along with 39,463 training questions, 5,349 validation questions and 5,188 test questions [38].
- **Ryerson Vision Lab - Complex Document Information Processing (RVL-CDIP) dataset** is a subset of the **IIT-CDIP** collection. It contains 400,000 document images categorized into 16 groups. It is split into 320,000 training images, 40,000 validation images and 40,000 testing images [36].
- **Medical Invoice dataset** contains 2,630 images with six important text fields, including medical insurance type, Chinese capital total amount, invoice number, social security number, name and hospital name. These images consist of a mix of digits, English characters and Chinese characters. Some images have variable layouts, illegible text and misaligned fonts. It is split into 2,104 training images and 526 testing images.

Figure 3.5 shows an example of a medical invoice from the Medical Invoice dataset.

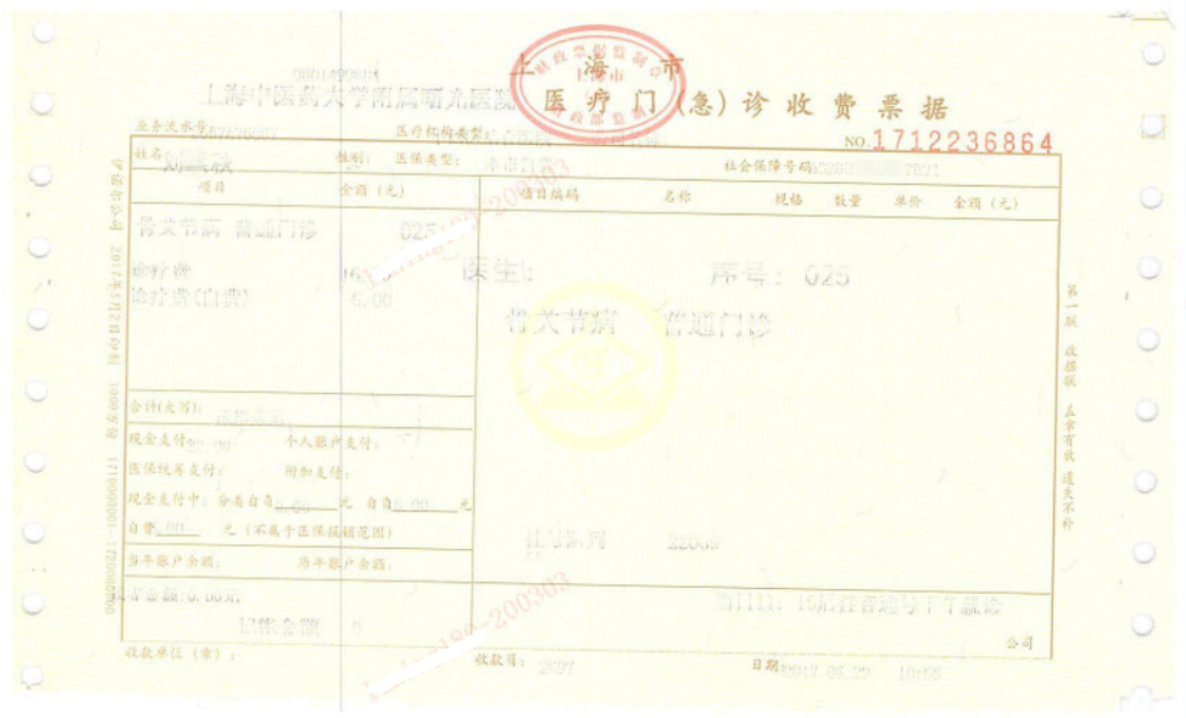


FIGURE 3.5. Example of Medical Invoice [32]

- **Train Ticket dataset** contains 2,000 real images and 300,000 synthetic images. Each image shows a train ticket with eight important text fields like ticket number, starting station, train number, destination station, date, ticket rates, seat category and name. These fields contain digits, English characters and Chinese characters. The layout of the tickets is consistent, but there might be some background noise and image distortions [39].

The **dataset** lacks text **bounding boxes** and their corresponding text content. To address this issue, 400 real images and 1,530 synthetic images were randomly selected and annotated by humans to create **bounding boxes** around the text. An **OCR** system was then used to extract the text within these boxes.

For **training** purposes, all the selected synthetic images (300,000) and 320 real images were used, while the remaining real images (1,680) were kept for testing.

Figure 3.6 shows a total of eight different examples from the Train Ticket dataset, where the four tickets on the left are real images and the other four on the right are synthetic images.

- **INVOICE dataset** contains 24,818 real-world invoice pages. It is split into 24,175 training images and 643 testing images. These invoices are from different templates and have 14 key fields, such as Customer, Address, Total and Amount, which were labeled by humans. Transcripts and word-level **bounding boxes** are extracted from these invoice images using the Microsoft Azure Read **API** [40].



FIGURE 3.6. Examples of Train Tickets: 4 Real Images (left), 4 Synthetic Images (right) [32]

Experiments and Results

This chapter focuses on studying with more detail the ICDAR 2019 Challenge, different approaches are explored to address this challenge, first by using more conventional methods, such as custom rules, and later by using generative models.

4.1. ICDAR 2019 Challenge: Robust Reading Challenge on Scanned Receipts OCR and Information Extraction

This section discusses the process of integrating OCR, TA, and KIE into a single solution. It outlines a workflow of decision-making based on specific requirements. The software for this integration was developed in Python¹ and executed in the Google Colab² environment. Scanned Receipts OCR is the process of extracting text from scanned receipts and invoices, which can have many applications, such as efficient archiving, fast indexing and document analytics. While recent studies have improved OCR accuracy and speed, there is still a need for further improvement, in order to fully automate certain tasks, as a result SROIE has gained more attention recently. However, there is still a lack of published research on SROIE, and existing studies and competitions do not address its specific challenges, such as robust reading, document layout analysis and NER [41].

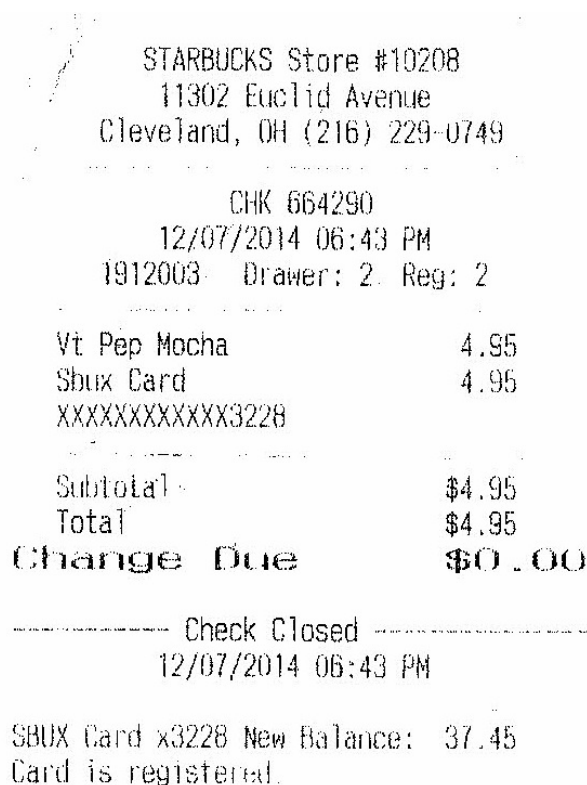
To address these issues the authors organized the ICDAR 2019 competition which involved a robust reading challenge on SROIE. The goal of the competition was to raise awareness and give several contributions. The authors provided a receipt dataset that contains several constraining factors, such as poor image quality, low resolution, folded invoices and interference from irrelevant texts. Some sensitive information is blurred for privacy reasons. The competition divides into two main tasks, performing OCR and KIE on scanned receipts. Additionally, a new evaluation method was developed for the two competition tasks, which can be used for future receipt datasets, in order to attract interest on SROIE [41].

The dataset for the challenge consists of 1,000 scanned receipt images, and each image contains several key text fields, such as product name, unit price and total cost. The receipts are mainly in English, and their text primarily consists of numbers and letters. The dataset is split into 600 training images ("trainval") and 400 testing images ("test"). The first set of images ("trainval") were given at the beginning of the challenge with the corresponding annotations, while the last set of images ("test") were only given a few weeks before the submission deadline (May 5th, 2019) [41].

¹<https://www.python.org/>

²<https://colab.research.google.com/>

Figure 4.1 shows an example of a scanned receipt from the [SROIE dataset](#). We can see the existence of the following fields: company, date, address and total, as well as others.



STARBUCKS Store #10208
11302 Euclid Avenue
Cleveland, OH (216) 229-0749

CHK 664290
12/07/2014 06:43 PM
1912003 Drawer: 2 Reg: 2

Vt Pep Mocha	4.95
Sbux Card	4.95
XXXXXXXXXXXX3228	
Subtotal	\$4.95
Total	\$4.95
Change Due	\$0.00

----- Check Closed -----
12/07/2014 06:43 PM

SBUX Card x3228 New Balance: 37.45
Card is registered.

FIGURE 4.1. Example of a Scanned Receipt [3]

For the receipt [OCR](#) task, each image in the [dataset](#) is annotated with text [bounding boxes](#) (bbox) and the actual text (transcript) contained within them. These annotations are saved in text files with the same names as the images [41]. The format used for these annotations is shown below:

```
x1_1, y1_1, x2_1, y2_1, x3_1, y3_1, x4_1, y4_1, transcript_1
x1_2, y1_2, x2_2, y2_2, x3_2, y3_2, x4_2, y4_2, transcript_2
x1_3, y1_3, x2_3, y2_3, x3_3, y3_3, x4_3, y4_3, transcript_3
...
```

For the [IE](#) task, each image in the [dataset](#) is annotated in a text file with a [JavaScript Object Notation \(JSON\)](#)³ format, as shown below:

```
{
  "company": "STARBUCKS STORE \#10208",
  "date": "14/03/2015",
  "address": "11302 EUCLID AVENUE, CLEVELAND, OH (216) 229-0749",
  "total": "4.95",
}
```

³<https://www.json.org/json-en.html>

The challenge consists of three tasks:

- **Task 1** - Scanned Receipt Text Localization
- **Task 2** - Scanned Receipt OCR
- **Task 3** - Key Information Extraction from Scanned Receipts

For the Scanned Receipt Text Localization task (Task 1), participants had to locate text regions ([bounding boxes](#)), accurately enough to identify individual words within them [41].

To deal with the challenge of text correspondences, the evaluation is based on the DetVal methodology, which involves calculating [Mean Average Precision \(mAP\)](#) and the average recall, and the F1 score is used as the ranking measurement.

For the Scanned Receipt OCR task (Task 2), participants had to accurately recognize the text in a receipt image, and submit the list of words found. The task was restricted to words comprising Latin characters and numbers only [41].

For the ground truth, all strings had to be tokenized by splitting on spaces. For example, the string "Date: 12/3/56" should be tokenized as "Date:", "12/3/56", while the string "Date: 12 / 3 / 56" should be tokenized as "Date:" "12", "/", "3", "/", "56".

Regarding evaluation, words are compared to the ground truth. If a word is repeated in an image, it should also be repeated in the submission. The Precision and Recall are calculated, and the F1 score is used as the ranking measurement.

For the Key Information Extraction from Scanned Receipts task (Task 3), participants had to extract texts of a number of key fields (key-value pairs) from each receipt, and save the results in [JSON](#) files [41].

For each receipt, the extracted text is compared with the ground truth. The precision is calculated over all the extracted texts of all the test receipt images. The Precision and Recall are calculated, and the F1 score is used as the ranking measurement.

4.2. Processing Pipeline

In this section we are going to present a schematization of the work developed on [Section 4.3](#) and [Section 4.4](#). As shown in [Figure 4.2](#), we start by read each scanned receipt image and extracting its content using an [OCR](#) service, we then store the text results into a file. After gathering and storing all the [OCR](#) results, we then proceed by performing [KIE](#) on each text file, either by applying the custom rules mentioned in [Section 4.3](#) or by applying the writing a set of instructions and sending them to the generative model mentioned in [Section 4.4](#).

4.3. Approach: Custom Rules

This section aims to make an heuristic approach, in order to help with the decision-making of the next steps towards a final solution. The goal is to gain a better understanding of the [SROIE dataset](#) and its specifications, while at the same time exploring hand-craft features based methods for [KIE](#), such as [Regular Expression \(RegEx\)](#), without the need of introducing more complex solutions at this stage.

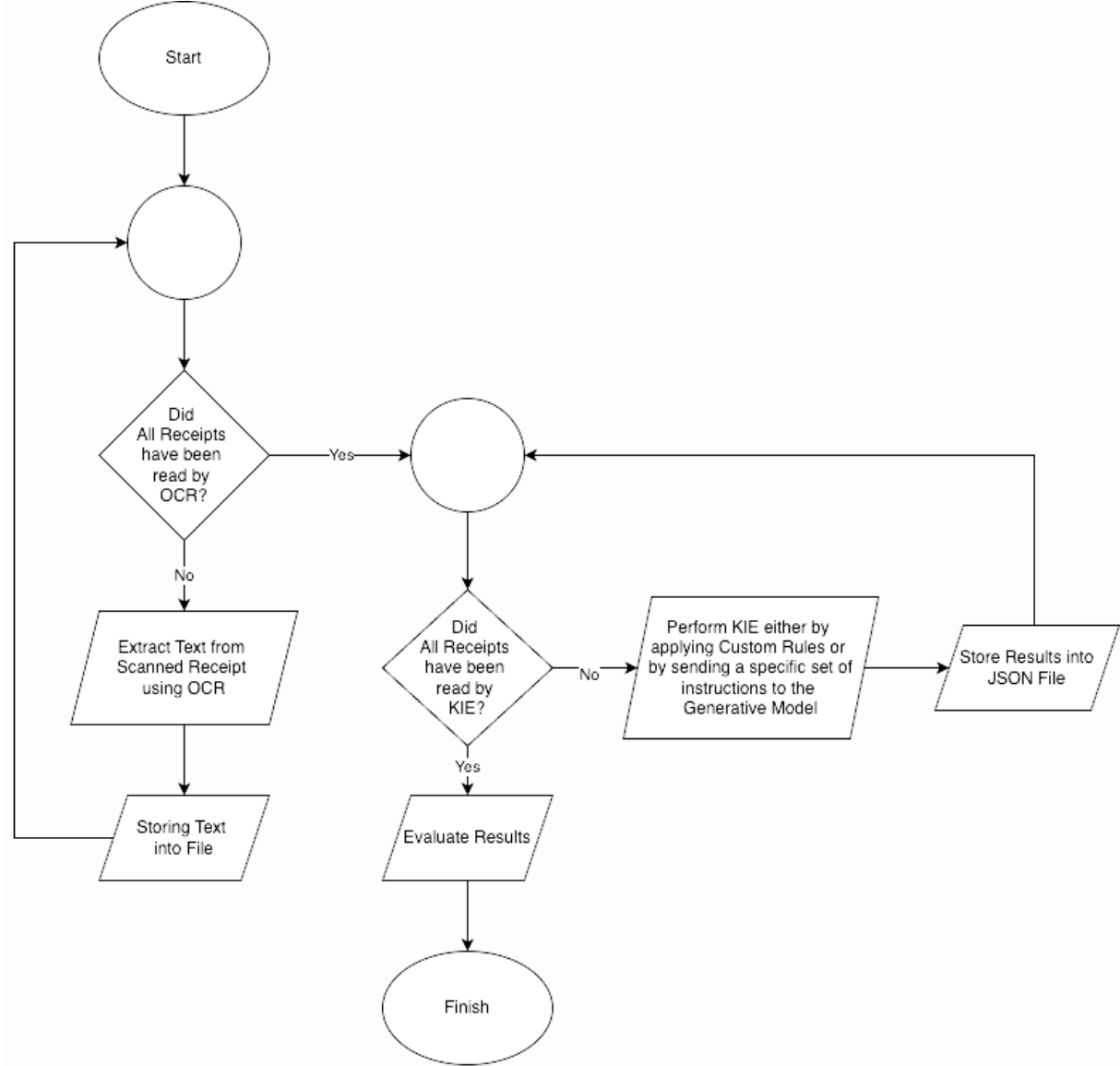


FIGURE 4.2. Approach Workflow

4.3.1. Setup

This report discusses the tests conducted for **KIE** on the **SROIE dataset**, which contains 626 receipts. The objective was to extract four key-value pairs from each receipt: Company, Address, Date and Total. To perform **OCR** we employed the Tesseract⁹ **OCR** service using two different configurations: the default mode (Tesseract Default) and a custom configuration (Tesseract Configured) with the parameters `'-oem 3 -psm 6'`, this is one of the most commonly used custom configurations. The tests aimed to understand how these configurations affected the results, using **RegEx** for extracting the values of each key.

4.3.2. Tesseract Setup

Regarding the Tesseract Custom Configuration [42]:

⁹<https://pypi.org/project/pytesseract/>

- **Page Segmentation Mode (-psm)**: This parameter determines how Tesseract processes and separates text lines and words within an image.
- **Engine Mode (-oem)**: This parameter determines which engine mode to use since Tesseract offers various engine modes with different performances and speeds.

4.3.3. Defining Custom Rules

There are four fields to extract from each receipt which are Company, Date, Address and Total. The Company field stands for the company name where the purchase was made. The Date field indicates the date of the transaction. The Address field is the local address of the store. The Total field is the amount of monetary value exchanged. For the rules applied on the field Company, we go through each line of text and search for the first one that matches the following [RegEx](#):

```
^[A-Z,&.( )]*$
```

For the rules applied on the field Address, we tried to extract this key from receipts by looking between lines 2 to 5, which is where the Address information is usually located. However, because the value of this field is lengthy and extends through multiple lines, Tesseract had a higher chance of failing to recognize and extract every single character from the text. Due to these challenges, we could not extract the Address key accurately.

For the rules applied on the field Date, we first tokenized every word in the text, and then performed a match with the following [RegEx](#):

```
^((( [0-3] [0-9] | [1-9] ) [-|.|\|/] ( [0-3] [0-9] | [1-9] ) [-|.|\|/] ( [0-9]{4} | [0-9]{2} ))$ |  
^((( [0-9]{4} | [0-9]{2} ) [-|.|\|/] ( [0-3] [0-9] | [1-9] ) [-|.|\|/] ( [0-3] [0-9] | [1-9] ))$
```

The goal of the [RegEx](#) is to identify specific date formats separated by "-", ".", "/", or joined (string concatenation), ensuring that regarding joined dates it must have 4 digits for the year:

- DD MM AA
- DD MM AAAA
- MM DD AA
- MM DD AAAA
- AA MM DD
- AAAA MM DD
- AA DD MM
- AAAA DD MM

If no result is found, we then try a second extraction using a different [RegEx](#), for dates containing the month's abbreviation, such as JAN, FEB and MAR:

```
( [0-3] [0-9] | [1-9] ) (\s) (Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec|  
JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC) (\s) ( [0-9]{4} | [0-9]{2} )
```

For the rules applied on the field Total, we first try to find the word "Total" in the document, we then create a substring with the text located after the word "Total", and finally perform the following [RegEx](#) on the newly substring:

`([0-9]{1}[,.][0-9]{1,2})? $`

4.3.4. Results

We then performed the first set of tests on [KIE](#), based on the custom rules previously mentioned, as shown in [Table 4.1](#) and [Table 4.2](#), the results are very similar when comparing the Default Configuration and the Custom Configuration.

TABLE 4.1. Tesseract default - first set of tests

Key	Precision	Recall	F1
Address	0.00	0.00	0.00
Company	26.68	26.68	26.68
Date	66.93	66.93	66.93
Total	33.55	33.55	33.55

TABLE 4.2. Tesseract configured - first set of tests

Key	Precision	Recall	F1
Address	0.00	0.00	0.00
Company	22.20	22.20	22.20
Date	67.89	67.89	67.89
Total	33.55	33.55	33.55

For the second set of tests on [KIE](#), we are attempting to improve the results of document scanning by making adjustments to the scanned receipts. Specifically, we are cropping the white margins and eliminating small dots and shadows within the receipts. We are trying this approach to see if it improves the ability of the Tesseract [OCR](#) to extract text from these documents.

TABLE 4.3. Tesseract default - second set of tests

Key	Precision	Recall	F1
Address	0.00	0.00	0.00
Company	24.76	24.76	24.76
Date	59.90	59.90	59.90
Total	27.96	27.96	27.96

TABLE 4.4. Tesseract configured - second set of tests

Key	Precision	Recall	F1
Address	0.00	0.00	0.00
Company	22.04	22.04	22.04
Date	68.21	68.21	68.21
Total	32.91	32.91	32.91

As shown in [Table 4.3](#) and [Table 4.4](#), removing the margins around the text in the image leads to worse results, possibly because the image quality degrades during processing. Since we are applying specific rules based on our sample size it's normal to have limitations extracting specific fields specially alphanumeric, for numbers only it's easier since we are looking for small pieces of data with some limited values.

4.4. Approach: Generative Models

In this section we explore a different approach by using GPT model 3.5 from OpenAI, in order to perform [KIE](#) with the generated texts from Tesseract [OCR](#) and also with the ground-truth texts, and compare both results.

4.4.1. Setup

We start by organizing and structuring all folders, for the project we used a refactored version of the [SROIE dataset](#), where all files related to the 626 receipts have been renamed from "000" to "625", and the main folder (data) has been divided into the following subfolders:

- **key**: Contains the [JSON](#) files with all ground truth results of the structured data from the [KIE](#) task. [Figure 4.3](#) shows an example of a [JSON](#) object from the [SROIE dataset](#), where we can see the four key-value pairs from this specific example: company, date, address, total.

```
{
  "company": "LIAN HING STATIONERY SDN BHD",
  "date": "27/03/2018",
  "address": "NO.32 & 33, JALAN SR 1/9, SEKSYEN 9, TAMAN SERDANG RAYA,
              43300 SERI KEMBANGAN, SELANGOR DARUL EHSAN",
  "total": "12.00"
}
```

FIGURE 4.3. [SROIE](#) Key for Receipt 625 [\[41\]](#)

- **img**: Contains the image files in [JPG](#) of the actual scanned images for the [OCR](#) task. [Figure 4.4](#) shows an example of a scanned receipt from the [SROIE dataset](#), we can see the specifications of this receipt, such as the existence of a table, the usage of different fonts, as well as the presence of green marks possibly from a pen. We can also identify the set of keys and their respective values for the [ICDAR](#) challenge: company, date, address, total.
- **box**: Contains the [Comma-Separated Values \(CSV\)](#) files of the [bounding boxes](#) and its corresponding texts for the [OCR](#) task. [Figure 4.5](#) depicts a scanned receipt from a metadata point of view, the first eight columns (A to H) always indicate the coordinates from the [bounding boxes](#) of the texts present in the columns after the H column, in this specific case I, J, K, but a scanned receipt can have more or less of these columns, these columns try to depict the spaces that exist between texts of the same [bounding box](#). A and B represent the X and Y coordinates of the top-left corner of the scanned receipt image text, respectively.

3180303

LIAN HING STATIONERY SDN BHD
(162761-M)
NO.32 & 33, JALAN SR 1/9, SEKSYEN 9,
TAMAN SERDANG RAYA,
43300 SERI KEMBANGAN, SELANGOR
DARUL EHSAN
GST ID : 002139201536

Tax Invoice

27/03/2018 No : CS-20243

	Qty	Tax	RM
F/Castell 187057-75 Tack-It 75g- White (new) @ 5.6600	2	SR	12.00

Total Amt Incl. GST @ 6% : 12.00

Rounding Adjustment :

Total Amt Payable : 12.00

Paid Amount : 20.00

Change : 8.00

Total Qty Tender : 2

GST Summary	Amount (RM)	Tax (RM)
SR @ A	11.32	0.68
Total	11.32	0.68

THANK YOU
For any enquiry, please contact us:

FIGURE 4.4. SROIE Key for Receipt 625 [41]

C and D represent the X and Y coordinates of the top-right corner of the scanned receipt image text, respectively. E and F represent the X and Y coordinates of the bottom-right corner of the scanned receipt image text, respectively. G and H represent the X and Y coordinates of the bottom-left corner of the scanned receipt image text, respectively.

We also created an evaluation folder (evaluation) which contains all the evaluation scripts and includes the following subfolders:

- **gt**: To store the Ground Truth results in zip files.
- **submit**: To store the Generated Outputs in zip files to be evaluated.

The last folder (generated_data) we created to store all generated data based on the outputs of Tesseract OCR, OpenAI GPT API and also converted files from JSON to TXT.

Bounding Boxes									Text Extracted		
Top-Left		Top-Right		Bottom-Right		Bottom-Left					
X	Y	X	Y	X	Y	X	Y				
18	17	132	17	132	41	18	41	3180303			
154	134	471	134	471	154	154	154	LIAN HING STATIONERY SDN BHD			
262	158	362	158	362	179	262	179	(162761-M)			
133	182	491	182	491	201	133	201	NO.32 & 33	JALAN SR 1/9	SEKSYEN 9	
195	203	430	203	430	223	195	223	TAMAN SERDANG RAYA			
132	225	489	225	489	247	132	247	43300 SERI KEMBANGAN	SELANGOR		
245	250	383	250	383	268	245	268	DARUL EHSAN			
188	274	438	274	438	296	188	296	GST ID : 002139201536			
251	308	373	308	373	330	251	330	TAX INVOICE			
115	337	212	337	212	355	115	355	27/03/2018			
369	335	502	335	502	356	369	356	NO : CS-20243			
379	382	410	382	410	403	379	403	QTY			
419	382	454	382	454	403	419	403	TAX			
493	383	525	383	525	403	493	403	RM			
110	405	348	405	348	423	110	423	F/CASTELL 187057- 75 TACK-LT			
393	405	404	405	404	423	393	423	2			
425	405	454	405	454	423	425	423	SR			
477	406	525	406	525	424	477	424	12.00			
109	427	344	427	344	450	109	450	75G- WHITE (NEW) @ 5.6600			
127	475	373	475	373	497	127	497	TOTAL AMT INCL. GST @ 6% :			
456	475	503	475	503	494	456	494	12.00			
179	510	373	510	373	529	179	529	ROUNDING ADJUSTMENT :			
199	541	374	541	374	562	199	562	TOTAL AMT PAYABLE :			
451	542	502	542	502	559	451	559	12.00			
253	574	374	574	374	593	253	593	PAID AMOUNT :			
454	575	503	575	503	593	454	593	20.00			
290	607	373	607	373	629	290	629	CHANGE :			
464	610	501	610	501	627	464	627	8.00			
212	642	373	642	373	664	212	664	TOTAL QTY TENDER :			
490	641	502	641	502	660	490	660	2			
127	692	254	692	254	715	127	715	GST SUMMARY			
288	691	359	691	359	710	288	710	AMOUNT			
443	691	479	691	479	710	443	710	TAX			
314	713	357	713	357	735	314	735	(RM)			
433	712	478	712	478	734	433	734	(RM)			
125	747	197	747	197	766	125	766	SR @ A			
318	745	366	745	366	763	318	763	11.32			
437	745	478	745	478	762	437	762	0.68			
124	772	170	772	170	790	124	790	TOTAL			
318	773	366	773	366	790	318	790	11.32			
439	772	478	772	478	789	439	789	0.68			
242	810	362	810	362	828	242	828	THANK YOU			
159	837	461	837	461	853	159	853	FOR ANY ENQUIRY	PLEASE CONTACT US:		

FIGURE 4.5. SROIE bounding boxes for Receipt 625 [41]

There were some specifications regarding the evaluation scripts, in order to address them we had to pre-process the data by making the following changes:

- **Remove Key (Address) from Record 104:** It ensures that Record 104 aligns with the ground truth by removing the Key (Address) since it is not present in the ground truth.
- **Convert Key (Total) Values to String:** We convert the data type of values associated with Key (Total) from Number to String. This standardizes the data type.
- **Convert OpenAI Output Files from JSON to TXT:** We transform the format of OpenAI output files from JSON to TXT. This is required since it is the only file type accepted.

- **Zip Files Without Subfolders:** We zip all files from the folder without creating additional subfolders, since the evaluation script is expecting that.

TABLE 4.5. Results using gpt on tesseract ocr outputs

Key	Precision	Recall	F1
Address	6.72	6.72	6.72
Company	36.42	36.42	36.42
Date	52.08	52.08	52.08
Total	66.45	66.45	66.45
Overall	40.43	40.43	40.43

4.4.2. Data Analysis

We randomly chose 20 records and sort them by their IDs. Then analyzed each record in terms of the outputs provided by Tesseract and OpenAI, regarding OpenAI the prompts given were based on the Tesseract outputs. The records were the following: 021, 101, 108, 163, 175, 189, 236, 281, 283, 289, 324, 350, 393, 405, 510, 524, 543, 550, 609, 622. Figure 4.6 shows the output results from the randomly picked records using Tesseract and OpenAI.

Figure 4.13 contains a subset of 3 records so that we can analyze more in depth:

- **Record 189:** All keys were extracted correctly, we can see it by comparing the key-value pairs of both JSON objects from the ground-truth data, and the generated data from the combined usage of Tesseract and OpenAI, as shown in Figure 4.8. Figure 4.7 is the original image of the scanned receipt 189.
- **Record 510:** OCR wrongly extracted the texts of all keys, we can see it by comparing the key-value pairs of both JSON objects from the ground-truth data, and the generated data from the combined usage of Tesseract and OpenAI, as shown in Figure 4.10. Figure 4.9 is the original image of the scanned receipt 510.
- **Record 543:** We were able to extract the values of Company and Total, but missed the values of Address and Date due to Tesseract and OpenAI respectively. We can see it by comparing the key-value pairs of both JSON objects from the ground-truth data, and the generated data from the combined usage of Tesseract and OpenAI, as shown in Figure 4.12. Figure 4.11 is the image of the scanned receipt 543, with a specific difference from the original, a border was manually applied to show the actual limits of the original image from the scanned receipt, in order clearly identify the presence of a large blank area, this can affect OCR and KIE processes, specially if they use rules for text extraction based on absolute positions.

4.4.3. Results

As shown in Table 4.5 we can see an improvement in results compared to the methods used in Section 4.3, these affected all fields but one, the key Date had better results when

Record ID	Key	Tesseract	OpenAI	Observations
021	Address	passed	failed	OCR wasn't able to extract Address, after looking at the invoice we noticed that a stamp was marked on top of its letters
	Company	failed	---	OCR wasn't able to extract Company name, after looking at the invoice we noticed that a stamp was marked on top of its letters
	Date	passed	passed	
	Total	passed	passed	
101	Address	failed	---	OCR extracted a comma instead of a period and also extracted "\$3200" instead of "53200"
	Company	passed	passed	
	Date	passed	passed	
	Total	passed	passed	
108	Address	failed	---	OCR needed to capitalize some words and use whitespaces instead of newline characters, there was also a comma that was suppose to be a period
	Company	passed	passed	
	Date	passed	failed	GPT extracted hours and minutes as well
	Total	passed	passed	
163	Address	failed	---	OCR extracted "1342" instead of "1362"
	Company	failed	---	GPT extracted more data than needed
	Date	passed	passed	
	Total	failed	---	OCR extracted "PH108.S0" instead of "PH 108.50"
175	Address	failed	---	OCR needed to capitalize some of the words and GPT missed the extraction of the last part
	Company	passed	passed	
	Date	passed	failed	GPT extracted hours and minutes as well
	Total	passed	passed	
189	Address	passed	passed	
	Company	passed	passed	
	Date	passed	passed	
	Total	passed	passed	
236	Address	failed	---	OCR failed to recognize some of the words
	Company	passed	passed	OCR didn't extracted the "&"
	Date	passed	passed	
	Total	failed	---	OCR extracted "14.95" instead of "14.85"
281	Address	failed	---	OCR didn't add commas
	Company	passed	passed	
	Date	passed	passed	
	Total	failed	---	OCR extracted "38.00" instead of "35.00"
283	Address	failed	---	OCR needed to capitalize some of the words and missed some white spaces
	Company	passed	passed	
	Date	failed	---	OCR needed to capitalize the month and GPT extracted more data than needed
	Total	passed	passed	
289	Address	failed	---	OCR needed to capitalize some of the words and GPT added extra commas
	Company	failed	---	OCR missed a few words
	Date	failed	---	OCR extracted "10-04-2016" instead of "10-04-2018"
	Total	passed	failed	GPT extracted more data than needed
324	Address	failed	---	OCR extracted "R-B, JALAN S515/41" instead of "B-8, JALAN SS15/4D"
	Company	failed	---	OCR extracted "SIN" instead of "SDN" and GPT extracted the very first line of text which didn't correspond at all
	Date	failed	---	OCR extracted "27/10/2017" instead of "29/10/2017"
	Total	passed	passed	
350	Address	failed	---	OCR only missed a dot at the end
	Company	passed	passed	
	Date	passed	passed	
	Total	passed	passed	
393	Address	failed	---	OCR needed to capitalize some of the words
	Company	failed	---	OCR needed to capitalize some of the words
	Date	passed	passed	
	Total	passed	passed	
405	Address	failed	---	OCR needed to capitalize some of the words and GPT missed the extraction of the last part
	Company	passed	failed	GPT extracted more data than needed
	Date	passed	passed	
	Total	failed	---	OCR added a white space and GPT retrieved incorrect field
510	Address	failed	---	OCR missed a few words
	Company	failed	---	OCR extracted "HARDUARE" instead of "HARDWARE" and also extracted "SETI" instead of "SETIA"
	Date	failed	---	OCR extracted "14/12/2017" instead of "13/12/2017"
	Total	failed	---	OCR extracted "24.80" instead of "24.00"
524	Address	failed	---	OCR needed to capitalize some of the words
	Company	passed	passed	
	Date	passed	passed	
	Total	failed	---	OCR extracted "RM15.39" instead of "RM15.40"
543	Address	failed	---	OCR missed whitespace
	Company	passed	passed	
	Date	passed	failed	GPT extracted hours and minutes as well
	Total	passed	passed	
550	Address	failed	---	OCR needed to capitalize some of the words
	Company	failed	---	OCR needed to capitalize some of the words
	Date	passed	passed	
	Total	passed	passed	
609	Address	passed	failed	GPT added an extra comma
	Company	passed	passed	
	Date	failed	---	OCR didn't extract date
	Total	passed	passed	
622	Address	failed	---	OCR missed a white space, a comma and a period
	Company	passed	passed	
	Date	passed	failed	GPT extracted hours and minutes as well
	Total	passed	passed	

FIGURE 4.6. Tesseract OCR and OpenAI GPT Outputs

(P)

YONG CEN ENTERPRISE
 9, JALAN SUBANG JASA 3,
 40150 SHAH ALAM, SELANGOR.
 TEL: 012-9719498
 GST NO: 001147981824

CASH

RECEIPT #: CS00677115 DATE: 02/03/2018
 SALESPERSON: TIME: 09:43:00
 COUNTER:

ITEM	QTY	U/P	AMOUNT
9555223500307	24	2.33	55.99
SR:FEI YAN BRAND YOUNG CORN 425G (2.333X			
9557166110018	2	2.00	4.00
ZRL:ASAM BOI 65G			

TOTAL QUANTITY 26
 SUB-TOTAL 59.99
 DISC 0.00
 GST 3.36 ✓
 TAX 0.00
 ROUNDING 0.00
 TOTAL 63.35 ✗

63.35

CASH 63.35
 CHANGE 0.00
 GOODS SOLD ARE NOT RETURNABLE,
 THANK YOU.

FIGURE 4.7. Receipt 189

<pre>{ "company": "YONG CEN ENTERPRISE", "date": "02/03/2018", "address": "9, JALAN SUBANG JASA 3, 40150 SHAN ALAM, SELANGOR.", "total": "63.35" }</pre>
<pre>{ "company": "YONG CEN ENTERPRISE", "date": "02/03/2018", "address": "9, JALAN SUBANG JASA 3, 40150 SHAH ALAM, SELANGOR.", "total": "63.35" }</pre>

FIGURE 4.8. Receipt 189. Ground truth keys (top) vs Tesseract+OpenAI keys (bottom)

using custom rules, after further analysis we noticed that in some cases OpenAI extracted more data than needed, such as hours and minutes, therefore leading to a lower score. Although the score is 6.72%, we now have results for the field Address.

AIK HUAT HARDWARE
ENTERPRISE (SETIA
ALAM) SDN BHD
822737-X
NO. 17-G, JALAN SETIA INDAH
(X) U13/X, SETIA ALAM,
SEKSYEN U13, 40170 SHAH ALAM,
TEL: 012 - 6651783 FAX: 03 - 33623608
GST NO: 000394528768

SIMPLIFIED TAX INVOICE

CASH

RECEIPT #: CSP0420207 DATE: 13/12/2017
SALESPERSON : AH019 TIME: 17:58:00

ITEM	QTY	U/P (RM)	AMOUNT (RM)
8710163220787	2	12.00	24.00 S
PHILIPS 18W/E27/827 ESSENTIAL BULB W/WHI			
TOTAL QUANTITY		2	

SUB-TOTAL (GST) 24.00
DISC 0.00
ROUNDING 0.00

TOTAL 24.00
CASH 100.00
CHANGE 76.00

*GST @ 6% INCLUDED IN TOTAL

GST SUMMARY			
CODE	AMOUNT	%	TAX/AMT
SR	22.64	6	1.36
TAX TOTAL:			1.36

GOODS SOLD ARE NOT REFUNDABLE.
THANK YOU FOR CHOOSING US.
PLS PROVIDE ORIGINAL BILL FOR GOODS
EXCHANGE WITHIN 1 WEEK FROM TRANSACTION
GOODS MUST BE IN ORIGINAL STATE TO BE
ENTITLED FOR EXCHANGE.

FIGURE 4.9. Receipt 510

4.4.4. Error Analysis

Figure 4.13 shows a subset of three different receipts and their respective output results from the tests performed using Tesseract and OpenAI combined. We can see that for the receipt 189 all keys were correctly extracted, but for the receipt 510 Tesseract was not able to extract the correct sets of text and therefore OpenAI was not able to continue the KIE task, lastly for the receipt 543 the keys Company and Total were correctly identified,

<pre> { "company": "AIK HUAT HARDWARE ENTERPRISE (SETIA ALAM) SDN BHD", "date": "13/12/2017", "address": "NO. 17-G, JALAN SETIA INDAH (X) U13/X, SETIA ALAM, SEKSYEN U13, 40170 SHAH ALAM," , "total": "24.00" } </pre>
<pre> { "company": "AIK HUAT HARDWARE ENTERPRISE (SETI ALAM) SDN BHo", "date": "14/12/2017", "address": "NO. 17-G, JALAN SETIA INDAH, Went uta/X, SETIA ALAM, SEKSYEN 13, 40170 SHAH Lit", "total": "24.80" } </pre>

FIGURE 4.10. Receipt 510. Ground truth keys (top) vs Tesseract+OpenAI keys (bottom)

the key Address was not completely extracted from the [OCR](#) process, and the key Date had more data than the required one, in this case not only extracted day, month and year, but also hours and minutes as well.

Out of the 4 keys present across the 20 files processed, we have a total of 80 keys. From those 80 keys Tesseract OCR correctly identified 45 of them, corresponding to a 56.25% success rate.

- **Address:** Correctly identified 3 out of 20, corresponding to a 15% success rate. The lowest result.
- **Company:** Correctly identified 13 out of 20, corresponding to a 65% success rate.
- **Date:** Correctly identified 15 out of 20, corresponding to a 75% success rate. The highest result.
- **Total:** Correctly identified 14 out of 20, corresponding to a 70% success rate.

Since OpenAI depends on Tesseract results, we can only consider the 45 keys that were correctly identified by Tesseract OCR, from those 45 keys OpenAI correctly identified 37 of them, corresponding to a 82.22% success rate.

- **Address:** Correctly identified 1 out of 3, corresponding to a 33.33% success rate. The lowest result.
- **Company:** Correctly identified 12 out of 13, corresponding to a 92.31% success rate.
- **Date:** Correctly identified 11 out of 15, corresponding to a 73.33% success rate.
- **Total:** Correctly identified 13 out of 14, corresponding to a 92.86% success rate. The highest result.

We perform the same observations but now with the ground truth results of [OCR](#), so that we are able to analyze the OpenAI GPT. [Figure 4.14](#) shows the output results from the randomly picked records using the ground-truth [OCR](#) data and OpenAI.

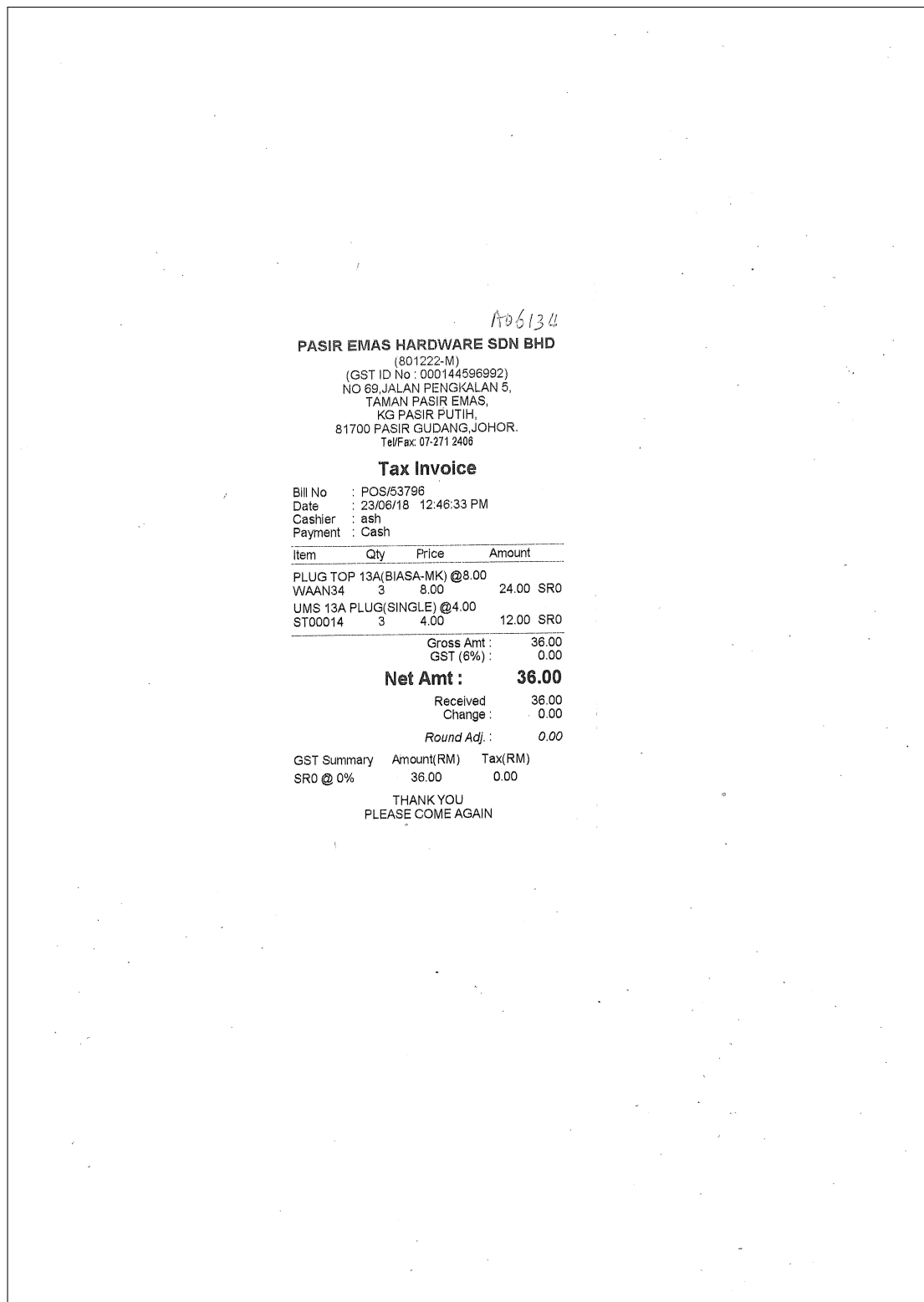


FIGURE 4.11. Receipt 543, the border shows the amount of white space that exists on the original image

As shown in Table 4.6 all fields scored better results compared to the previous tests made.

Figure 4.15 shows the same subset of receipts as in Figure 4.13 but now regarding the output results of the tests performed using OpenAI on the ground-truth OCR. We can see

<pre> { "company": "PASIR EMAS HARDWARE SDN BHD", "date": "23/06/18", "address": "NO 69, JALAN PENGKALAN 5, TAMAN PASIR EMAS, KG PASIR PUTIH, 81700 PASIR GUDANG, JOHOR.", "total": "36.00" } </pre>
<pre> { "company": "PASIR EMAS HARDWARE SDN BHD", "date": "23/06/18 12:46:33 PM", "address": "NO 69, JALAN PENGKALAN 5, TAMAN PASIR EMAS, KG PASIR PUTIH, 81700 PASIR GUDANG, JOHOR.", "total": "36.00" } </pre>

FIGURE 4.12. Receipt 543. Ground truth keys (top) vs Tesseract+OpenAI keys (bottom)

Record ID	Key	Tesseract	OpenAI	Observations
189	Address	passed	passed	
	Company	passed	passed	
	Date	passed	passed	
	Total	passed	passed	
510	Address	failed	---	OCR missed a few words
	Company	failed	---	OCR extracted "HARDUARE" instead of "HARDWARE" and also extracted "SETI" instead of "SETIA"
	Date	failed	---	OCR extracted "14/12/2017" instead of "13/12/2017"
	Total	failed	---	OCR extracted "24.80" instead of "24.00"
543	Address	failed	---	OCR missed whitespace
	Company	passed	passed	
	Date	passed	failed	GPT extracted hours and minutes as well
	Total	passed	passed	

FIGURE 4.13. Tesseract and OpenAI Outputs

TABLE 4.6. Results using gpt on ground truth texts

Key	Precision	Recall	F1-Score	Difference
Address	48.32	48.32	48.32	+41.6
Company	69.01	69.01	69.01	+32.59
Date	59.58	59.58	59.58	+7.5
Total	84.82	84.82	84.82	+18.37
Overall	65.44	65.44	65.44	+25.01

that the ground-truth **OCR** data does not have all correct values from their receipt images, overall OpenAI shows significant results when dealing with the correct text extraction.

Out of the 4 keys present across the 20 files processed, we have a total of 80 keys. From those 80 keys, OpenAI was not able to identify 9 of them, since there were differences between the Ground Truth texts from the CSV file and the Ground Truth results from the JSON file, so we manually corrected them. Regarding the 80 keys, OpenAI correctly identified 64 of them, corresponding to a 80% success rate.

- **Address:** Correctly identified 14 out of 20, corresponding to a 70% success rate.
- **Company:** Correctly identified 18 out of 20, corresponding to a 90% success rate.

Record ID	Key	OpenAI	Observations
021	Address	passed	
	Company	passed	
	Date	passed	
	Total	passed	
101	Address	---	Ground Truth CSV has a comma in the end (GPT correctly ignored it) but Ground Truth JSON has a dot at the end of the field
	Company	passed	
	Date	passed	
	Total	passed	
108	Address	---	Ground Truth CSV has two dots (GPT correctly extracted them) but Ground Truth JSON has two commas instead
	Company	passed	
	Date	failed	Extracted hours and minutes as well
	Total	passed	
163	Address	failed	Didn't extract last part of the field and also added an extra comma that didn't exist
	Company	failed	Extracted more data than needed
	Date	passed	
	Total	passed	
175	Address	failed	Extracted "in" instead of a whitespace and also needed to extract last part of the field
	Company	passed	
	Date	failed	Extracted hours and minutes as well
	Total	passed	
189	Address	---	Ground Truth CSV has "SHAH" (GPT correctly extracted it) but Ground Truth JSON has "SHAN"
	Company	passed	
	Date	passed	
	Total	passed	
236	Address	---	Ground Truth CSV doesn't have all text from Ground Truth JSON and also extracted "in" instead of a whitespace
	Company	passed	
	Date	passed	
	Total	passed	
281	Address	passed	
	Company	passed	
	Date	passed	
	Total	passed	
283	Address	passed	
	Company	passed	
	Date	failed	Extracted hours and minutes as well
	Total	passed	
289	Address	---	Ground Truth CSV has "8175" (GPT correctly extracted it) but Ground Truth JSON has "81750" and also extracted two "in" instead of whitespaces
	Company	passed	
	Date	failed	Extracted hours, minutes and seconds as well
	Total	failed	
324	Address	passed	
	Company	---	Last part of the field was separated from the rest and stored in the last line of the CSV and also extracted the wrong value although it's a company as well
	Date	passed	
	Total	passed	
350	Address	passed	
	Company	passed	
	Date	passed	
	Total	passed	
393	Address	failed	Added two commas that didn't exist
	Company	passed	
	Date	passed	
	Total	failed	Extracted "71.8" instead of "28.20"
405	Address	passed	
	Company	failed	Extracted more data than needed
	Date	passed	
	Total	passed	
510	Address	failed	Missed a comma at the end of the field
	Company	passed	
	Date	passed	
	Total	passed	
524	Address	---	Added a comma that doesn't exist and also Ground Truth CSV doesn't have a dot at the end of the field (GPT correctly ignored it) but Ground Truth JSON does
	Company	passed	
	Date	passed	
	Total	passed	
543	Address	---	Ground Truth CSV doesn't have two whitespaces (GPT correctly ignored them) but Ground Truth JSON does
	Company	passed	
	Date	failed	Extracted hours, minutes and seconds as well
	Total	passed	
550	Address	failed	Added two commas that didn't exist
	Company	passed	
	Date	passed	
	Total	passed	
609	Address	---	Added a comma didn't exist and also Ground Truth CSV has a dot (GPT correctly extracted it) but Ground Truth JSON doesn't
	Company	passed	
	Date	passed	
	Total	passed	
622	Address	failed	Added a comma that didn't exist
	Company	passed	
	Date	failed	Extracted hours, minutes and seconds as well
	Total	passed	

FIGURE 4.14. OpenAI GPT Outputs on Ground Truth Texts

Record ID	Key	OpenAI	Observations
189	Address	---	Ground Truth CSV has "SHAH" (GPT correctly extracted it) but Ground Truth JSON has "SHAN"
	Company	passed	
	Date	passed	
	Total	passed	
510	Address	failed	Missed a comma at the end of the field
	Company	passed	
	Date	passed	
	Total	passed	
543	Address	---	Ground Truth CSV doesn't have two whitespaces (GPT correctly ignored them) but Ground Truth JSON does
	Company	passed	
	Date	failed	Extracted hours, minutes and seconds as well
	Total	passed	

FIGURE 4.15. OpenAI Outputs from Ground Truth Texts

- **Date:** Correctly identified 14 out of 20, corresponding to a 70% success rate.
- **Total:** Correctly identified 18 out of 20, corresponding to a 90% success rate.

After analysing the experiments and results from both approaches we can now make the following assumptions.

The most difficult field to extract information is Address, this has to do with several aspects:

- **Length:** Based on the ground truth of 20 analysed records, this field ranged from 45 to 100 characters, if **OCR** or GPT miss extracting a single character we fail the evaluation.
- **Punctuation:** There are multiple punctuation marks throughout this field, which means that if **OCR** incorrectly identifies a character, such as a comma instead of a dot, it is already enough for GPT to not have an exact match with the ground truth.
- **Capital Letters:** There was several times that GPT changed the capitalization of letters. We can improve this by prompting that we do not want GPT to change capitalization.
- **Break lines:** This field can extend through multiple lines, therefore GPT has to understand which of them belong to the field, and then concatenate them into a single string. In some cases GPT added a specific character that did not exist in the original text, such as commas, when performing this concatenation. These factors increase the chances of failing in the evaluation if at least one line was not correctly identified or a character was randomly added. We can improve the evaluation results by informing GPT to not add new characters when concatenating lines for this field.

Regarding the field Company, there are two points to mention as well:

- **Multiple Companies:** In some cases the receipt contains more than one company name.

- **More data than required:** In the examples analyzed GPT tends to add more data than needed, sometimes if it text with brackets it counts as part of the company. We can try to improve the results by indicating that we do not need text between brackets.

For the field Date, the main reason why we were not able to recognize more examples was due to GPT adding hours, minutes and sometimes even seconds. We can improve this by changing the GPT prompt to only extract the day, month and year.

The easiest field to extract information is Total, this has to do with the fact that we are only extracting number, although sometimes we receive the currency as well, such as €, nevertheless we can still have the following problems:

- **Punctuation:** OCR might incorrectly identify the comma or dot between the numbers.
- **Other Fields:** In a receipt there are multiple fields that only have numbers, such as the individual price of each item in the receipt, or even the subtotal, because of this GPT might extract the wrong value for the field Total.

4.5. Summary

The ICDAR 2019 Challenge, focuses on two main subjects, OCR and KIE. For the OCR tasks we had to take into consideration that there were some small discrepancies from the ground-truth files and the actual text in the images, this affected evaluation since the results are compared with the ground-truth files, and also the downstream task of KIE, since we require the OCR results to perform KIE. We also had to perform some pre-processing of data, in order to solve conflicts when running the evaluations.

Nevertheless, this a very robust challenge that focuses on the essentials of OCR and KIE, and provides a dataset that proved to be very useful to validate different approaches regarding both subjects.

On the subject of OCR we can analyse that there are some constraints that can affect results, such as having long sets of text to extract, image quality, font size and style, and also document orientation, yet Tesseract OCR presented good results.

The custom rules applied showed inefficient results when dealing with alphanumeric fields, but presented significant results in numerical and date type fields, specifically in the last type where there's less room for ambiguity.

For the generative model ChatGPT, the results improve significantly compared to the custom rules, although there's still some limitations, such as identifying and extracting fields with long sets of characters, using different punctuation marks, and breaking lines when there's no evidence of a line break.

CHAPTER 5

Conclusions and Future Work

This thesis explores different approaches on the topics related to [OCR](#), [TA](#) and [KIE](#). We've seen the way of performing [KIE](#) has changed throughout the years, starting by using simpler methods such as, rule-based and template-based approaches, and then more recently using [DL](#) approaches that outperformed the tradition methods. We also seen that although traditional approaches tend to be outdated, they show significant results regarding specific field types, such as numbers and dates, where a regex or set of keywords is enough to obtain the correct output results.

We also identified several challenges regarding [OCR](#), [TA](#) and [KIE](#), such as having to deal with documents that contain extensive lines of text to extract, poor image quality, several font sizes and styles on the same document, and also different orientations as well, this last constraint might invalidate some methods used for text extraction based on its position in the document, for these cases we can perform image manipulation, in order to rotate the document itself but we also need extra validations to identify the correct orientation of the document.

There are three main types of document representation in [KIE](#), these are sequence-based, grid-based and graph-based. Sequence-based serializes a document into a 1D text sequence, grid-based generates a 2D grid of token embeddings, graph-based model a document into a graph and turn each text segment into a node. The approaches that present better results regarding document representation, tend to be based on graph-based models, and the usage of transformers. Transformers have become a standard when dealing with this type of models.

There are several goals that recent solutions have worked towards, such as removing ambiguity and reducing computational complexity by implementing different model architectures, and also different pre-trained objectives, such as [MLM](#), [MIM](#) and [WPA](#). These objectives seem to have become the standard for pre-trained models on [OCR](#), [TA](#) and [KIE](#).

For [OCR](#) we worked with Tesseract OCR which proved very effective on extracting text from digital documents, as well as configurable to the type of documents we were using, in our case scanned receipts.

For [TA](#) and [KIE](#) we first used heuristic methods with custom-rules, such as regex and keywords, we then explored a more refined solution using OpenAI GPT, these proved to be a very effective solution regarding [KIE](#), where almost all the fields showed an improvement in terms of results comparing to the previous approach.

For the future work one should focus on addressing the constraints identified in [Chapter 4](#), we could potentially use the code developed on [Chapter 4](#) and implement a software solution, such as an [Application Programming Interface \(API\)](#) to streamline all the tasks of [OCR](#), [TA](#) and [KIE](#) and present a final result to an end user, regarding an improvement in terms of results we could refine our prompts on OpenAI GPT, in order to not add or replace characters that do not exist in the text, this happened specially on the field Address, and to not extract hours and minutes for the field Dates, as we have seen doing that in multiple occasions.

References

- [1] R. Khandelwal, *An Introduction to Optical Character Recognition for Beginners*, en, May 2020. [Online]. Available: <https://towardsdatascience.com/an-introduction-to-optical-character-recognition-for-beginners-14268c99d60> (visited on 09/13/2022).
- [2] IBM Cloud Education, *What is Text Mining?* en-us, Jan. 2021. [Online]. Available: <https://www.ibm.com/cloud/learn/text-mining> (visited on 09/13/2022).
- [3] Zheng Huang, Kai Chen, Jianhua He, *et al.*, *Overview - ICDAR 2019 Robust Reading Challenge on Scanned Receipts OCR and Information Extraction - Robust Reading Competition*. [Online]. Available: <https://rrc.cvc.uab.es/?ch=13> (visited on 09/14/2022).
- [4] K. Peffers, T. Tuunanen, C. E. Gengler, M. Rossi, and W. Hui, “THE DESIGN SCIENCE RESEARCH PROCESS: A MODEL FOR PRODUCING AND PRESENTING INFORMATION SYSTEMS RESEARCH,” en, *SYSTEMS RESEARCH*, p. 24, 2020.
- [5] V. Jayaswal, *Performance metrics: Confusion matrix, precision, recall, and f1 score*, Sep. 2020. [Online]. Available: <https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262>.
- [6] S. Narkhede, *Understanding confusion matrix*, Jun. 2021. [Online]. Available: <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>.
- [7] J. Nabi, *Machine learning — fundamentals*, May 2019. [Online]. Available: <https://towardsdatascience.com/machine-learning-basics-part-1-a36d38c7916>.
- [8] *Basic concepts in machine learning*. [Online]. Available: <https://www.javatpoint.com/basic-concepts-in-machine-learning>.
- [9] *Decision guide for machine learning on aws*, Jul. 2023. [Online]. Available: <https://aws.amazon.com/getting-started/decision-guides/machine-learning-on-aws-how-to-choose/>.
- [10] *Machine learning algorithms*. [Online]. Available: <https://www.javatpoint.com/machine-learning-algorithms>.
- [11] *Machine learning glossary | google for developers*. [Online]. Available: <https://developers.google.com/machine-learning/glossary>.
- [12] GeeksforGeeks, *Bayes’ theorem*, en-us, Mar. 2024. [Online]. Available: <https://www.geeksforgeeks.org/bayes-theorem/>.

- [13] K. E. Koech, *The basics of neural networks (neural network series) - part 1*, May 2022. [Online]. Available: <https://towardsdatascience.com/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b>.
- [14] *What is deep learning?* [Online]. Available: <https://www.ibm.com/topics/deep-learning>.
- [15] R. Wolff, *Natural language processing (nlp): 7 key techniques*, Oct. 2021. [Online]. Available: <https://monkeylearn.com/blog/natural-language-processing-techniques/>.
- [16] pawangfg, *Named entity recognition*, Jan. 2024. [Online]. Available: <https://www.geeksforgeeks.org/named-entity-recognition/>.
- [17] M. D. Pietro, *Text summarization with nlp: Textrank vs seq2seq vs bart*, Mar. 2022. [Online]. Available: <https://towardsdatascience.com/text-summarization-with-nlp-textrank-vs-seq2seq-vs-bart-474943efeb09>.
- [18] F. Pascual, *Topic modeling: An introduction*, Sep. 2019. [Online]. Available: <https://monkeylearn.com/blog/introduction-to-topic-modeling/>.
- [19] M. Ali, *Understanding text classification in python*, Nov. 2022. [Online]. Available: <https://www.datacamp.com/tutorial/text-classification-python>.
- [20] M. Grootendorst, *Keyword extraction with bert*, Oct. 2020. [Online]. Available: <https://towardsdatascience.com/keyword-extraction-with-bert-724efca412ea>.
- [21] A. Beri, *Stemming vs lemmatization*, May 2020. [Online]. Available: <https://towardsdatascience.com/stemming-vs-lemmatization-2daddabcb221>.
- [22] W. Lin, Q. Gao, L. Sun, *et al.*, “Vibertgrid: A jointly trained multi-modal 2d document representation for key information extraction from documents,” in *Document Analysis and Recognition-ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I 16*, Springer, 2021, pp. 548–563.
- [23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [24] T. I. Denk and C. Reisswig, “Bertgrid: Contextualized embedding for 2d document representation and understanding,” *arXiv preprint arXiv:1909.04948*, 2019.
- [25] W. Yu, N. Lu, X. Qi, P. Gong, and R. Xiao, “Pick: Processing key information extraction from documents using improved graph learning-convolutional networks,” in *2020 25th International Conference on Pattern Recognition (ICPR)*, IEEE, 2021, pp. 4363–4370.
- [26] T. Hong, D. Kim, M. Ji, W. Hwang, D. Nam, and S. Park, “Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 10 767–10 775.

- [27] M. Zhai, Y. Li, X. Qin, *et al.*, “Fast-structext: An efficient hourglass transformer with modality-guided dynamic token merge for document understanding,” *arXiv preprint arXiv:2305.11392*, 2023.
- [28] C.-Y. Lee, C.-L. Li, H. Zhang, *et al.*, “Formnetv2: Multimodal graph contrastive learning for form document information extraction,” *arXiv preprint arXiv:2305.02549*, 2023.
- [29] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, “Layoutlmv3: Pre-training for document ai with unified text and image masking,” in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 4083–4091.
- [30] Z. Chi, H. Huang, H.-D. Xu, H. Yu, W. Yin, and X.-L. Mao, “Complicated table structure recognition,” *arXiv preprint arXiv:1908.04729*, 2019.
- [31] S. Park, S. Shin, B. Lee, *et al.*, “Cord: A consolidated receipt dataset for post-ocr parsing,” 2019.
- [32] M. Assefi, “OCR as a Service: An Experimental Evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym,” *ISCV*, Dec. 2016.
- [33] J.-P. T. Guillaume Jaume Hazim Kemal Ekenel, “Funsd: A dataset for form understanding in noisy scanned documents,” in *Accepted to ICDAR-OST*, 2019.
- [34] J. Wang, C. Liu, L. Jin, *et al.*, “Towards robust visual information extraction in real world: New dataset and novel solution,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.
- [35] B. Majumder, N. Potti, S. Tata, J. B. Wendt, Q. Zhao, and M. Najork, “Representation learning for information extraction from form-like documents,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, 2020, pp. 6495–6504.
- [36] A. W. Harley, A. Ufkes, and K. G. Derpanis, “Evaluation of deep convolutional nets for document image classification and retrieval,” in *International Conference on Document Analysis and Recognition (ICDAR)*, 2015.
- [37] X. Zhong, J. Tang, and A. J. Yepes, “Publaynet: Largest dataset ever for document layout analysis,” in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, Sep. 2019, pp. 1015–1022. DOI: [10.1109/ICDAR.2019.00166](https://doi.org/10.1109/ICDAR.2019.00166).
- [38] M. Mathew, D. Karatzas, R. Manmatha, and C. V. Jawahar, “Docvqa: A dataset for VQA on document images,” *CoRR*, vol. abs/2007.00398, 2020. arXiv: [2007.00398](https://arxiv.org/abs/2007.00398). [Online]. Available: <https://arxiv.org/abs/2007.00398>.
- [39] H. Guo, X. Qin, J. Liu, J. Han, J. Liu, and E. Ding, “EATEN: entity-aware attention for single shot visual text extraction,” *CoRR*, vol. abs/1909.09380, 2019. arXiv: [1909.09380](https://arxiv.org/abs/1909.09380). [Online]. Available: <http://arxiv.org/abs/1909.09380>.
- [40] P. Farley, N. Mehrotra, and E. Urban, *Ocr - optical character recognition*, Jul. 2023. [Online]. Available: <https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-ocr>.

- [41] Z. Huang, K. Chen, J. He, *et al.*, “Icdar2019 competition on scanned receipt ocr and information extraction,” in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, 2019, pp. 1516–1520.
- [42] Ivan, *Tesseract ocr best practices*, Feb. 2019. [Online]. Available: <https://ai-facets.org/tesseract-ocr-best-practices/>.

Glossary

algorithm: It is a set of rules and procedures, to solve a specific problem or task. 2, 3, 8–11, 13

bias: In ml, bias or bias term is a parameter that represents an offset or starting point different from the origin (0,0). This adjustment is necessary because not all models begin at the origin. 11, 50

binary classification: A type of classification that predicts one of two possible classes (positive and negative). 7

bounding box: The coordinates (x, y) of a rectangle surrounding an area of interest, in an image. 17, 18, 20, 24, 25, 29, 31

dataset: Dataset or data set, is a collection of raw data, commonly organized in a spreadsheet or in a csv file. v, vii, 2, 8–10, 12, 16–20, 23–26, 29, 41, 50

document analytics: Research field that uses NLP to automate tasks related to documents and gain a better understanding of their content. 2, 23

efficient archiving: Involves moving infrequently used data to low-cost storage repositories. 2, 23

fast indexing: Involves using indexing techniques that ensure quick and flexible access to content. 2, 23

fine-tuning: The process of refining a pre-trained model through additional training for a specific task or use case. 50

framework: A pre-built and reusable structure or set of tools developed by someone in order to facilitate the development process. 2–4

harmonic mean: It is a numerical average calculated by dividing the number of entries in a series, by the reciprocal of each number in the series. 8

layer: A set of neurons in a nn. 11, 12

library: A collection of reusable code modules that simplify programming by eliminating the need to write the same code repeatedly for different programs. 2

model: A mathematical construct that processes input data to produce an output. 2, 3, 7–11, 16, 49, 50

overfitting: Developing a model that is so closely linked to the training data, that it is unable to accurately predict new data. 10

parameter: The [weights](#) and [biases](#) a model learns during training. 11

pre-training: The initial [training](#) of a [model](#) on a large-scale [dataset](#), which can then be used to train other [models](#), and fine-tuned. 18

ResNet: A type of cnn often used for image classification. Usually has a number following its name, which indicates the depth of the network, or how many layers it has. 16

sigmoid function: A mathematical function that transforms a value from an input into a restricted range, usually from 0 to 1 or from -1 to +1. 9

training: In ml, training is the process of [fine-tuning](#) a model by adjusting its parameters ([weights](#) and [biases](#)) using examples. This adjustment can occur repeatedly, ranging from a few times to billions of times, to find the best configuration for the model. 10, 20, 49, 50

weight: In ml, weight is a value that multiplies with another value. Training involves finding the best weights for a model, and inference uses these learned weights to make predictions. 11, 50