By

Mr. BOUBEKRI Faycal and Mr. CHERGUELAINE Ayoub

Thesis for obtaining the Master's degree

Field: Mathematics and Computer Sciences

Specialty: Natural Language Processing

Theme :

A transformer Based Company Document classification depending on the context

Members of the thesis committee:

Dr. Mr. CHIKHI	President	University of Blida 1
Dr. M. MESKALDJI	Examinator	University of Blida 1
Dr. M. MEZZI	Internal Supervisor	University of Blida 1
Mr. H. MEKKID	External supervisor	ICOSNET

Blida, June 2023

Dedication

It is with deep gratitude and sincere words that I dedicate this work in the first place to My dear parents who whatever object we try to offer them, it will never reach what I want to say and express to them.

To my dear mother

The pearl of my existence, what a brave lady you are, that of sacrifices made to me so that I may progress in my studies. I am in awe of the goodness of your heart.

Whatever my whims and deviations, you have always supported me, finding the right words to bring me back on the right path.

The event we are celebrating today is entirely dedicated to you. May God preserve your health and long life

To my dear Dad

The one who is always there for me, and gave me a magnificent model of toil and perseverance, the one who encouraged me who always protected me, my model that makes me proud.

May this work be the expression of the wishes that you have never ceased to express in your prayers. My dear father.

To my sister Meriem.

To all my family, my grandparents, my maternal and paternal uncles, aunts and cousins.

To those who supported me, encouraged me.

To all ITC Members.

And especially to my colleague Ayoub, thank you for your advice and encouragement, but also for the good times that have helped make these years unforgettable.

Faycal.

Dedication

I dedicate this thesis to the most important people in my life who have been my pillars of strength and constant support throughout this academic journey.

Dear Mom and Dad, your love, support, and sacrifices molded me. Your unwavering belief in my abilities drove me to conquer obstacles and follow my dreams. This accomplishment is proof of your enduring commitment and guidance. I'm eternally grateful.

To my dear grandfather Belkacem and my friend Karim, God's mercy on them . I deeply miss both of you and dedicate this thesis to honor your memory, inchallah we will meet in janah

To my entire family, my grandmothers may God protect them, my brothers, sisters and my nephews, Salma, Adem, and Yacine, your presence and encouragement have been invaluable. This thesis is dedicated to each one of you

To my dear friends, I want to sincerely thank each of you for being incredible friends. Together, we have shared countless moments of pure happiness, from Trips, events, stops to see Mercedes, and a shared depression some times. I treasure our memories,"l3laam ".

To my ITC Club family, you have added a lot to me. You played a significant role in shaping my university career, and I am immensely grateful for the strong bond, shared passion. Thank you for being a part of my journey.

I want to give a heartfelt shout-out to my colleague, Faycal. Your advice and encouragement have been priceless over the years. However, what truly makes this journey unforgettable is the wonderful moments we've shared together. Thank you for being a part of those cherished memories.

Ayoub.

Acknowledgment

We would like to thank Allah, who has helped us and given us patience and courage during these long studies.

In particular, we would like to express our gratitude to our project manager, Mrs. Mezzi Melyara, whose contribution in stimulating suggestions and encouragement, her patience, her availability, and above all her wise counsel helped to fuel our thinking and coordinate our project, notably in writing this essay.

We would also like to express our gratitude to the Computer Science Department of Blida 1, for its contribution throughout this degree program.

We want to thank the head of the department and the department's teaching staff, who provided us with the tools we needed to succeed in our university studies.

We would be delighted to express our deepest gratitude to the friends and colleagues who provided us with moral and intellectual support throughout our process, and to all those who offered us the opportunity to complete this report.

Special thanks also to our parents, who set us on the path to this state long ago, for their trust and invaluable support. Also, for their kind cooperation and encouragement in helping us realize this project.

Through these few words, we extend our warmest thanks to our dear parents and families for the support and encouragement they have always given us, also Nachef Abdelkarim, Ait halal Nouredine , Belkacemi Nourhane , and Bounejar Faical for the help, as well as to all our friends who have contributed directly or indirectly to our success.

We wish to extend a special thanks to our elder Mekid Hamza for all his help and availability.

We would also like to thank the members of the jury for honoring us by kindly participating in the evaluation of our work.

Ayoub and Fayçal

Abstract

This thesis offers a thorough exploration of document classification solutions for cloud-based storage, employing fine-tuned models for zero-shot classification and text classification. Based on transformer architecture, both transformer encoders and encoder-decoder architectures are utilized to enhance the classification techniques. In light of the existing limitations of current document classification methods, this research endeavors to advance the development of more efficient and effective techniques, particularly in the context of big data. The investigation concentrates on tackling challenges related to dynamic labels during classification, handling lengthy documents, and achieving precise classification of structured documents. Additionally, this research makes valuable contributions to the field of document classification by generating datasets and benchmarks that can serve as invaluable resources for future projects. These resources facilitate the evaluation and comparison of various techniques, further fostering progress in the field.

Keywords: *Document Classification, Text Classification, Zero-shot Classification, Transformers, Company Document Dataset.*

ملخص

تقدم هذه الأطروحة بحثا شاملاً في حلول تصنيف المستندات على منصة تخزين على السحابة، وذلك باستخدام نماذج مُحسَّنة لتصنيف اللقطة الصفرية (zero-shot classification) وتصنيف النص (Text classification). استنادًا إلى بنية المحولات (Transformers)، يتم استخدام كل من معماريات التشفير(encoder) والتشفير-فك التشفير (encoder-decoder) لتعزيز تقنيات التصنيف. في ضوء القيود الحالية لأساليب تصنيف المستندات الحالية، يسعى هذا البحث إلى تعزيز تطوير تقنيات أكثر كفاءة وفعالية، لا سيما في سياق البيانات الضخمة. يركز التحقيق على معالجة التحديات المتعلقة بالعلامات الديناميكية أثناء التصنيف، والتعامل مع المستندات الكبيرة، وتحقيق التصنيف الدقيق للوثائق المنظمة. بالإضافة إلى ذلك، يقدم هذا البحث مساهمات قيمة في مجال تصنيف المستندات من خلال إنشاء مجموعات البيانات التي يمكن أن تكون بمثابة موارد لا تقدر بثمن للمشاريع المستقبلية. تسهل هذه الموارد تقييم ومقارنة التقنيات المختلفة، مما يزيد من تعزيز التقدم في هذا المجال.

<u>الكلمات الرئيسية:</u> *تصنيف الوثائق، تصنيف النصوص، التصنيف الصفري، المحولات، مجموعة بيانات مستندات الشركات.*

Résumé

Ce mémoire propose une exploration approfondie des solutions de classification de documents pour le stockage basé sur le cloud, en utilisant des modèles affinés pour la classification zero-shot et la classification de textes. Sur la base de l'architecture du Transformers, les encoder de Transformers et les architectures encodeur-décodeur sont utilisés pour améliorer les techniques de classification. À la lumière des limites existantes des méthodes actuelles de classification des documents, cette recherche s'efforce de faire progresser le développement de techniques plus efficaces et efficientes, en particulier dans le contexte des mégadonnées. L'enquête se concentre sur la résolution des problèmes liés aux étiquettes dynamiques lors de la classification, la gestion de documents volumineux et la réalisation d'une classification précise des documents structurés. De plus, cette recherche apporte de précieuses contributions au domaine de la classification des documents en générant des collections des données qui peuvent servir de ressources inestimables pour de futurs projets. Ces ressources facilitent l'évaluation et la comparaison de diverses techniques, favorisant davantage les progrès dans le domaine.

Mots-clés : *Classification de documents, Classification des textes, Classification Zero-shot, Transformers, Collection de documents d'entreprise.*

Table of Content

1. General Introduction ... 1

1.1 Introduction ... 1

1.2 Research challenge ... 2

1.3 Research objectives ... 3

1.4 Thesis organization ... 4

Chapter 1: State of the art ... 6

1. Introduction ... 6

2. Text classification in NLP ... 6

3. Word representation ... 7

 3.1. Bag of Words (BoW) ... 8

 3.2. Term Frequency-Inverse Document Frequency representation ... 8

  3.2.1. Term Frequency ... 8

  3.2.2. IDF ... 9

  3.2.3. TF-IDF ... 9

 3.3. Word Embeddings ... 9

 3.4. Contextual Word Embeddings ... 10

4. Rule-based classification ... 10

5. Statistics-based models ... 12

6. Deep-learning based classification ... 13

7. Transformers-based classification ... 14

8. The Impact of Context on Document Classification ... 15

 8.1. Type of contexts ... 16

  8.1.1. Linguistic Context ... 16

  8.1.2. Semantic Context ... 16

  8.1.3. Pragmatic Context ... 16

9. Related works ... 17

10. Conclusion ... 18

Chapter 2: Solution modeling ... 20

1. Introduction ... 20

2. Proposed Approach and Methodology ... 20

3. Dataset collection ... 22

4. Preprocessing 22
5. Transfer Learning 24
5.1. Introduction 24
5.1.1 Fine-tuning 25
5.1.2 Transformer 26
5.1.3 Architecture 26
5.1.4. self-attention 27
5.1.5. Encoder models 28
5.1.6. Encoder-decoder 28
5.1.7. distillation 29
5.2 BERT Model 30
5.2.1. Definition 30
5.2.1 BERT architecture 31
5.3. Roberta Model 31
5.3.1. Définition 31
5.3.2 RoBERTa architecture 32
5.4. BART 32
5.4.1 Definition 32
5.4.2. BART architecture 33
5.5. BERTopic 33
5.5.1. BERTopic Architecture 33
6. Training methods 34
6.1. Text classification 34
6.2. Zero shot classification 35
6.3. A Hybrid Architecture Combining BERTopic and Classifier: 36
7. Language detection 36
7.1. Xlm roberta base language detection 37
8. Optical Character Recognition 38
8.1 Textract 39
9. Cloud-Based Software Solution for Automated Document Classification: Modeling and Development 39
9.1 Methodology 39
9.2 Classifier API: Text Classification with Language Detection and OCR 40
10. Conclusion 42

Chapter 3: Tests and Evaluation 43
1. Introduction 43
1.2 Hardware Resources 43
1.3 Software Resources 43
1.3.1 Programming Language and frameworks 44
1.4 Implementation and Results 45
1.4.1 Application 45
1.4.1.1 Drive Storage Web App 45
1.4.1.2 Data Manager API 46
1.4.1.3 Cloud Manager API 46
1.4.1.4 Classifier API 46
1.4.2 Evaluation Measures 47
1.4.2.1 F1 Score 47
1.4.3 Text Classification Result 48
1.4.3.1 CNN Articles Discussion 49
1.4.3.2 Company Documents Discussion 51
1.4.4 Zero Shot Classification Result 52
1.4.4.1 CNN Articles Discussion 53
1.4.4.2 Company documents dataset discussion 55
1.4.5 Hybrid Architecture Combining BERTopic and Classifier 56
1.4.5.2 BERTopic Fitting 56
1.4.5.3 Text Prediction 56
1.4.5.4 Train and Test Set Creation 57
1.4.5.5 Classifier Training 57
1.4.5.6 CNN Articles results discussion 58
1.4.5.7. Improved Training Results 58
1.4.5.8. Company Documents Discussion 58
1.5 Conclusion 59
General conclusion 60
1. Conclusion 60
2. Perspectives 60

List of Figures

Figure 1: Classification method overview 21
Figure 2: General pipeline for the text preprocessing 23
Figure 3: Replacing consecutive spaces with a single space. 23
Figure 4: Removing leading or trailing whitespace. 24
Figure 5: Utility of fine tuning. 26
Figure 6: The Transformer - Architecture [20] 27
Figure 7:Encoder [28] 28
Figure 8:Encoder-Decoder [29] 29
Figure 9: distillation principle in deep learning 30
Figure 10:BERT MLM [16] 31
Figure 11: BART LM [16] 33
Figure 12: Text classification process 35
Figure 13: Zero shot classification, predict process 35
Figure 14: BERTopic + classifier, predict process 36
Figure 15 : Optical character recognition process 38
Figure 16: Web app general process 40
Figure 17: Sequence diagram for classifier api 41
Figure 18: Drive APP Dashboard 45
Figure 19: Data Manager API documentation 46
Figure 20: Cloud Manager API documentation 46
Figure 21: Classifier API Documentation 47
Figure 22: Training arguments (BERT,RoBERTa,DistilBERT,DistilRoBERTa,DistilBART 50
Figure 23: Tokenize function for the sentence with 256 max lengths 51
Figure 24: DistilBERT evaluation on Company Documents dataset (classification report) 52
Figure 25: Preparing data for zero shot fine tuning 53
Figure 26: Classification report of BBC evaluation in BART zero shot classification 54
Figure 27: Classification report of Company Documents evaluation in BERT zero shot classification 55
Figure 28: Classification report of Company Documents evaluation in BERT zero shot classification after replace number by tags 55
Figure 29: BART zero shot on Company Documents dataset classification report 56
Figure 30: Classification report of BERTopic + classifier on cnn dataset 30
Figure 31: Sample of invoice from Company Documents dataset 31

Figure 32: A graph representation of the different categories present in the “The CNN News Articles” dataset with the according number of articles. ii
Figure 33: A graph representation of the different categories present in the “Ag news” dataset with the according number of articles. iv
Figure 34: Recurrence of Text Lengths from 23 to 200 v
Figure 35: Recurrence of Text Lengths from 23 to 200 vii

Table of Tables

Table 1: Rule based Classification Approaches 11
Table 2: Statics-based classification approaches 12
Table 3: Deep learning-based classification approaches 13
Table 4: Transformers-base classification approaches 15
Table 5: Synthesis of Contextual Document Classification Approaches 18
Table 6: Hardware resources 43
Table 7: Comparison of model by average F1score in text classification 49
Table 8: Comparison of model by average F1score in zero shot classification 53
Table 9: Comparison of architecture and methods by average F1score in the hybrid architecture for text classification 58
Table 10: Word Count Statistics by Label Category vi

General Introduction

1. General Introduction

1.1 Introduction

The increasing volume of textual documents on the web, accounting for nearly 80% of available data, highlights the need for efficient organization and classification. Text document classification serves multiple purposes, including information retrieval, indexing, spam detection, sentiment analysis, and more. It plays a vital role in enhancing the structure and coherence of textual data, making it easier to manage and utilize.

Document classification is a crucial task in the era of big data, as unstructured data such as text documents continues to grow exponentially. It offers a practical solution for managing, searching, and analyzing vast amounts of data by categorizing documents based on their context. This classification process finds applications in various fields, such as categorizing news articles, analyzing customer feedback sentiments, or organizing legal documents. Automation of the classification process not only saves time and resources but also optimizes data utilization for organizations.

Document classification has far-reaching implications for big data tasks, including search engines, information retrieval, and data mining. By accurately categorizing documents, search engines can provide more relevant search results, while data mining algorithms can identify patterns and trends more effectively. In the absence of document classification, making data-driven decisions and extracting valuable insights from big data would be exceedingly challenging.

ICOSNET, an Algerian operator specializing in broadband internet access, convergent telecommunications solutions, and Cloud services, recognizes the immense potential of document classification and aims to leverage it to enhance their offerings and deliver superior solutions to their clients. With their experienced team and strong industry relationships, ICOSNET is well-positioned to make a significant impact in the ICT sector both nationally and internationally.

1.2 Research challenge

In the era of big data, document classification poses various challenges, including:

- *Handling Dynamic Labels during Classification:*
  - Incorporating new labels into the existing classification model
  - Adapting the system to recognize and classify documents with new labels
  - Ensuring the flexibility of the classification system to accommodate evolving labels
- *Accurately Classifying Structured Documents:*
  - Recognizing and extracting relevant information from documents with specific formats or structures
  - Handling structured fields within documents to accurately classify them
  - Developing techniques to extract key information from structured documents for classification purposes
- *Processing Lengthy Documents:*
  - Efficiently handling large volumes of information within lengthy documents
  - Identifying and extracting the most relevant information for classification
  - Overcoming challenges posed by the complexity and size of lengthy documents in the classification process
- *Specific challenges of ICOSNET:*
  - Convert files or images to text document that can be processed by the machine.
  - Detect the language of the resulting text.
  - Classify the text according to the context using some neural architecture.

1.3 Research objectives

We propose a thesis in this study with the following objectives:

1. The primary phase of this project involves the creation of a cloud-based platform that facilitates secure document storage and management, with restricted access limited to authorized personnel.
   Establishing a secure cloud-based platform is crucial for effective and efficient document management. This platform will enhance the productivity and success of the company by ensuring secure document storage, convenient accessibility, and restricted access.
2. The second phase consists of the integration of an OCR system that will convert pdf, docx, and image files into text that can be processed.
3. The project's third objective is to integrate a language detection system into the developed platform to enable multilingual document processing. The language identification mechanism will utilize advanced algorithms to accurately determine the language of uploaded documents based on their content and structure. This feature is crucial for effective document classification and organization in their respective languages, allowing for efficient information retrieval. The project aims to address language-related challenges that arise in document classification systems and improve the overall functionality of the platform.
4. The main goal of this project is to create a machine-learning system for classifying and organizing documents based on their context.
5. To guarantee accuracy and dependability, the system will be trained on a dataset of documents created by us.

By integrating various functionalities such as file or image to text conversion, language detection, and contextual text classification, the proposed document classification system possesses the capacity to greatly amplify the efficiency and effectiveness of document management. This holds especially true for large organizations that maintain extensive digital document repositories. The primary objective of this study is to make a meaningful contribution to the ongoing research in document management systems by offering practical solutions to the challenges encountered in modern information management, specifically in the areas of converting files or images to text, identifying the language of the extracted text, and categorizing the text based on its contextual relevance.

1.4 Thesis organization

To meet the above-mentioned objectives, we organized our thesis as follows:

- ***The first chapter*** deals with explaining automatic text classification. The chapter begins by presenting the role of word representation in document classification then we will explore the different approaches and techniques used in automatic text classification including rules based, statistical base, deep learning base and transformer base, then we move to the exploration of the "context" meaning and lastly the theses explored in the related works section.
- ***The second chapter*** proceeds to describe the proposed approach and methodology for text classification. It covers various aspects such as dataset collection and preprocessing, which involves gathering and cleaning the data to make it suitable for classification tasks. The exploration of transfer learning with different models and their architectures is also discussed, as transfer learning has shown promise in improving classification performance. Furthermore, the chapter presents a hybrid architecture that combines BERTopic and a classifier, leveraging the strengths of both approaches. Then, it addresses language detection techniques and the incorporation of Optical Character Recognition (OCR) to enhance the classification process. Finally, the chapter delves into the software modeling aspect of the proposed solution, discussing the development and implementation of the necessary software components.
- ***The third chapter*** focuses on the implementation and evaluation of the proposed solution. It provides details about the hardware and software resources used in the implementation process. The chapter includes the implementation code for each model, describing the specific algorithms and techniques employed. Furthermore, it presents the evaluation of the models, assessing their performance in terms of accuracy, precision, recall, and other relevant metrics. The results obtained from the evaluation are then compared with existing approaches to highlight the effectiveness and efficiency of the proposed solution.

- The fourth and final chapter serves as a platform for discussing the findings and implications of the research. It starts with a summary of the evaluation results and their significance in addressing the challenges of text classification. The chapter proceeds to discuss the limitations of the proposed solution and potential areas for future research and improvement. Finally, the chapter concludes with a concise summary of the entire document, emphasizing the contributions and impact of the research in the field of text classification.

Chapter I:
State of the art

Chapter 1: State of the art

1. Introduction

Text categorization, also known as text classification, is the process of analyzing the content of a given document and classifying it based on its characteristics into one or more specified classes [1]. The objective is to create a model that, based on how a new, unlabeled document resembles other labeled documents, can properly predict the category or categories that are most appropriate for it. [2] [3]

Text classification has numerous applications in various fields, including sentiment analysis, language detection, and topic modeling. In this chapter, We'll talk about how text classification is applied in areas of sentiment analysis, language detection, and topic modeling.

2. Text classification in NLP

In many NLP (Natural Language Processing) tasks, such as sentiment analysis, topic labeling, question answering, and dialog act classification, text classification has been a significant and necessary task. Processing and classifying large amounts of text data manually has become difficult and time-consuming in the age of the information explosion. It is desirable to use machine learning techniques to automate the text classification process in order to produce more accurate and objective results because the accuracy of manual text classification can be affected by human factors like fatigue and expertise.

In the upcoming sections, we will present a few document classification techniques with different representations for each one, but first we will start with a brief introduction of word representation since it serves as the basis for successful classification strategies, a thorough understanding of document representation is essential.

3. Word representation

A key component of natural language processing, particularly in text classification tasks, is word representation. It entails turning words into numerical vectors that record the semantic and syntactic information contained within them. Word representation can be done using a variety of methods, such as contextualized embedding methods, prediction-based methods, and count-based methods.

Words are represented using count-based methods that take into account their frequency within a document or corpus, such as Term Frequency-Inverse Document Frequency (TF-IDF). Although they are easy to use and computationally effective, they do not accurately capture word meaning in context. Word embeddings are learned using neural networks in prediction-based techniques like Word2Vec and GloVe based on the co-occurrence of words in a corpus. These techniques are effective at capturing the semantic connections between words and excel at tasks like word analogy and similarity. They are nevertheless constrained by the context window that was used to learn the embeddings.

Deep neural networks are used by contextualized embedding techniques like ELMo and BERT to produce word embeddings that are unique to the context in which they appear. These techniques have achieved cutting-edge results in a range of text classification tasks and other natural language processing tasks. Nevertheless, they are computationally expensive and need a lot of training data.

Recent research has also looked at combining various word representation techniques to enhance text classification task performance. For instance, some researchers have combined prediction-based and count-based techniques to collect data on both local and global context. Others have combined various prediction-based techniques to benefit from each technique's advantages.

In conclusion the task at hand and the available resources determines which word representation technique we use.

3.1. Bag of Words (BoW)

The bag of words model views a document as a collection of words, Each document is represented by a high-dimensional sparse vector, where each dimension represents the frequency of a single word in a dictionary or hash table [4].

The following is an illustration of the bag of words model where we Take into account the following two phrases:

"The cat is on the mat." and "The dog is chasing the cat."

We first create a vocabulary that consists of all distinct words from both sentences in order to represent a bag of words:

Vocabulary: "The cat is on the mat," "the dog is chasing,"

the	cat	is	on	mat	dog	chasing
0	0	0	0	0	0	0

In order to indicate the presence or absence of words from the vocabulary, we next represent each sentence as a numerical vector:

sentence 1: "The cat is on the mat,"

the	cat	is	on	mat	dog	chasing
1	1	1	1	1	0	0

sentence 1: "the dog is chasing,"

the	cat	is	on	mat	dog	chasing
1	0	1	0	0	1	1

The values in the vectors represent the presence (1) or absence (0) of each word in the vocabulary at each position, which is represented by a position in the vector.

3.2. Term Frequency-Inverse Document Frequency representation

3.2.1. Term Frequency

In **TF**, each term in a document is given a weight that is based on how frequently the term appears in the document. Based on the weight of a search phrase in a document, we want to calculate a score between the two. A basic strategy is to allocate the number of times the phrase t appears in document d should be equal to the weight.

Using the subscripts to denote the term and the document in sequence, this weighting system is known as term frequency and is written as tft,d [5].

3.2.2. IDF

Inverse Document Frequency (IDF) allows to quantify the rarity or discriminative power of a term within a corpus of documents. IDF calculates the inverse of the percentage of documents in the corpus that contain a given term.

The following formula is used to determine an IDF for a term:

$$IDF(term) = log(\frac{N}{(DF(term) + 1)})$$

As the term's usage in the corpus declines, the IDF value rises logarithmically. Higher IDF values will be assigned to uncommon terms that appear in fewer documents, reflecting their greater significance and discriminative power [5].

3.2.3. TF-IDF

The term's TF-IDF weight is calculated by multiplying its inverse document frequency (IDF) across the corpus by the term's term frequency (TF) within a document. This total weight reflects the term's relative importance in the document in relation to its significance across the entire corpus.

3.3. Word Embeddings

Word2Vec: While prior approaches to word embeddings existed, it was primarily Word2vec that popularized their usage. Word2vec operates based on the distributional hypothesis, which posits that the meaning of words can be inferred from their context. To obtain word representations, Word2vec can be trained using two model architectures: Continuous Bag-of-Words (CBOW) and the Skip-gram model. The CBOW model aims to predict a word based on a window of k words surrounding it, without considering their order. On the other hand, the Skip-gram architecture learns to predict words preceding and succeeding a given target word using a log-linear classifier with a continuous projection layer. In this case, words farther from the target word are assigned less weight as they tend to have weaker associations with the target word.

GloVe: Pennington et al. (2014) introduced Global Vectors for Word Representation (GloVe) as a log-bilinear regression model. Unlike Word2vec's shallow window-based approach, GloVe incorporates the occurrence counts of words across the entire corpus during training.

3.4. Contextual Word Embeddings

ELMO: Word2vec has certain limitations. It cannot effectively represent combinations of multiple words, such as idioms. Additionally, it is constrained by the window size used during training and struggles to capture the different meanings of polysemic words. To address these shortcomings, Embeddings from Language Models (ELMo) was introduced. ELMo provides deep contextualized word representations that capture both the syntax and semantics of words, as well as contextual variations like polysemy. These representations are learned from the internal states of a deep bidirectional language model (biLM) trained on a large corpus of 30 million sentences. Unlike word-level representations, ELMo returns contextual representations for each word, considering its surrounding context.

4. Rule-based classification

Machine-learning approaches have been used to establish rule-based classification systems where expert knowledge was inadequate . These methods use training data and machine-learning algorithms to develop a series of rules to define each class [6].

In this approach, a predetermined set of guidelines—typically established by human subject matter experts in the relevant field—are built into the system before it is even built. The rules can be deterministic or probabilistic, with a specific output guaranteed given a given set of inputs and a probability for the output based on the inputs.

As inputs are received, they are compared to the predefined rules, and then outputs or decisions are produced based on the results of the rule evaluation. They are frequently used in decision support systems, expert systems, and automated reasoning.

The decisions made by the system are frequently transparent and easy to comprehend, and a rule-based approach can be relatively straightforward to implement and maintain. These are its two main advantages. The disadvantage is that the system's capabilities are limited by how precise and thorough the rules are, and it might be unable to manage difficult situations or adapt to changing circumstances.

Here's an example of a rule-based system that utilizes a linguistic rule:

Input: "I want to go to the store."

Linguistic Rule: "If the user input contains the word 'want', respond with a suggestion or recommendation related to the user's request."

Output: "Sure, I can recommend a few stores that may interest you. Would you like me to provide some options?"

In this example, the input is a simple statement expressing the user's desire to go to the store. The linguistic rule in this case is to detect the word "want" in the user input and respond with a suggestion or recommendation related to the user's request. The output of the system is a follow-up question that provides the user with options and invites further engagement with the system. This is an example of a rule-based system that uses a linguistic rule to process user input and provide a relevant output based on the detected intent.

Table 1 summarizes some classical classifiers in the rule-based approach:

Table 1: Rule based Classification Approaches

Keyword-based approach	This approach involves identifying keywords or phrases that are indicative of a specific category and then using those keywords to assign the document to the corresponding category.
Regular expression-based approach	This approach involves using regular expressions to identify patterns in the text that are indicative of a specific category.
Syntax-based approach	This approach involves analyzing the syntax of the text to identify patterns that are indicative of a specific category.
Rule-induction based approach	This approach involves automatically generating rules from a training set of labeled documents, and then using these rules to classify new documents.

5. Statistics-based models

The first and most important step in text classification is preprocessing the text data for the model. Conventional methods typically call for obtaining high-quality sample features using artificial means and then classifying them using traditional machine learning algorithms [1]. The effectiveness of this method is significantly constrained by feature extraction, which adds time and cost. Statistics-based models like Naive Bayes, K-Nearest Neighbor, and Support Vector Machine dominated text classification techniques from the 1960s to the 2010s. These techniques had definite advantages over earlier rule-based ones in terms of accuracy and stability. The natural sequential structure or contextual information in textual data, however, was disregarded, making it difficult to understand the semantic meaning of words. In text mining, polysemy and synonymy are two unsolved issues [1].

Table 2: Statics-based classification approaches

Naive Bayes	This probabilistic algorithm operates under the presumption that each feature (word or phrase) in the document exists independently of every other feature. The algorithm gives the document a category based on the likelihood of each feature appearing in each category (class).
TF-IDF and Cosine Similarity	This approach represents each document as a vector in a high-dimensional space, where each dimension represents a unique term in the corpus. The frequency of each term in the document is multiplied by its inverse document frequency (IDF) to obtain the term frequency-inverse document frequency (TF-IDF) score. Documents are compared by computing the cosine similarity between their corresponding vectors.
Support Vector Machines	This is a supervised machine learning algorithm that aims to find a hyperplane that separates the documents into different categories. SVMs can be used with different kernels, such as linear, polynomial, and radial basis function (RBF), to find the optimal decision boundary.
Latent Dirichlet Allocation	This is a probabilistic topic modeling technique that aims to identify the latent topics that underlie a collection of documents. LDA assumes that each document is a mixture of a fixed number of topics, and each topic is a distribution over the words in the corpus.

6. Deep-learning based classification

Since the 2010s, deep learning models have gradually replaced traditional approaches for text classification. By learning a set of nonlinear transformations that serve to directly map features to outputs, deep learning methods integrate feature engineering into the model fitting process. In contrast to conventional approaches, deep learning approaches automatically generate representations for text mining that are semantically meaningful. Therefore, deep neural networks (DNNs), which are data-driven methods with high computational complexity, are the foundation of the majority of text classification research.

Deep learning is a collection of learning techniques that aim to model complex data by combining various nonlinear transformations in their architectures. These architectures are built using neural networks, which are the elementary components of deep learning. With these techniques, significant progress has been made in sound and image processing, facial recognition, speech recognition, computer vision, automated language processing, and text classification, such as spam recognition. The potential applications of deep learning are endless. There are various types of neural network architectures available, such as the multilayer perceptron, the oldest and simplest architecture, Convolutional Neural Networks (CNNs) that are particularly suitable for image processing, and Recurrent Neural Networks (RNNs) that are designed for sequential data such as text or time series.

Table 3: Deep learning-based classification approaches

Convolutional Neural Networks	Convolutional Neural Networks (CNNs) create a word embedding matrix for each sentence, from which distinct features are extracted by applying convolutional filters known as kernels with all feasible word windows [7]. The output dimensionality of the layers is then decreased while maintaining a fixed size using a max pooling strategy. These convolutional layers are stacked in different combinations to create deep convolutional networks.

Recurrent Neural Networks	RNNs can model the sequence of words in a document and capture the contextual dependencies between them. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are popular RNN variants used in document classification. Due to their exceptional capacity to maintain internal memory, RNNs (Recurrent Neural Networks) are a highly effective and resilient type of neural network. The internal memory of RNNs allows them to retain context and capture the semantic meaning of sequences by remembering crucial information from previous inputs. For RNNs, this memory aspect is especially important because sequence interpretation frequently depends on previous elements.
Transformers	Transformers-based models, such as BERT (Bidirectional Encoder Representations from Transformers), leverage attention mechanisms to capture the contextual relationships between words in a document. These models can also be fine-tuned on specific classification tasks.
Ensemble models	Ensemble models combine multiple deep learning models to improve the overall performance of document classification. For example, a common approach is to use an ensemble of CNN, RNN, and Transformer-based models and average their predictions to obtain the final classification result.
Transfer learning	Transfer learning involves pre-training a deep learning model on a large dataset (such as Wikipedia) and then fine-tuning it on a smaller target dataset (such as a specific document classification task). This approach can lead to significant improvements in performance, especially when the target dataset is small.

7. Transformers-based classification

The transformer-based approach is a type of machine learning model architecture that is commonly used in natural language processing (NLP) tasks, such as language translation and text generation. It is based on the transformer model, which was introduced in the paper "Attention Is All You Need" by Vaswani et al. in 2017.

The transformer architecture replaces traditional recurrent neural network (RNN) models with attention mechanisms that allow the model to focus on relevant parts of the input sequence. It consists of an encoder and a decoder, each of which contains multiple layers of self-attention and feedforward neural networks.

The self-attention mechanism in transformers allows the model to identify and focus on relevant parts of the input sequence, rather than processing the entire sequence sequentially. This enables the transformer to process input sequences much more efficiently than traditional RNN models. The feedforward neural networks then process the output of the self-attention layers to produce the final output.

One of the key advantages of the transformer-based approach is its ability to handle variable-length sequences of inputs, which is particularly useful for natural language processing tasks. The attention mechanism allows the model to identify important words and phrases in a sentence, while ignoring irrelevant information.

The transformer-based approach has achieved state-of-the-art performance in many NLP tasks, including machine translation, text summarization, and question answering. It is also widely used in pre-trained language models, such as BERT and GPT, which have achieved impressive results on a wide range of NLP tasks.

Table 4: Transformers-base classification approaches

Pretrained language models	This approach involves training a language model on a large corpus of text and then fine-tuning the model on a specific document classification task. Examples of pretrained language models include BERT, GPT-2, and RoBERTa.
Hierarchical approaches	In this approach, a document is divided into smaller subunits, such as paragraphs or sentences, and each subunit is then classified using a model based on the Transformers. The classification of the document is then finalized by combining the findings from each sub-unit.
Ensemble methods	This approach involves combining multiple Transformers-based models to improve the overall performance of the document classification task. For example, multiple pretrained language models can be combined to improve accuracy and reduce bias.
Multi-task learning	This approach involves training a Transformers-based model on multiple related tasks simultaneously. For example, a model can be trained on both document classification and named entity recognition tasks to improve its performance on both tasks.

8. The Impact of Context on Document Classification

The term "context" in the study of document classification refers to the wide range of data and elements that surround a document and affect its interpretation, comprehension, and classification. Context encompasses a variety of factors, and recent developments in document classification techniques have made it more crucial to take these factors into account. Models can better capture the subtle nuances, semantics, and dependencies present in documents by incorporating context, resulting in more precise and context-aware classification.

8.1. Type of contexts

8.1.1. Linguistic Context

Linguistic context is the immediate textual setting in which a document or a particular set of words appear. In order to comprehend the meaning, intention, and semantic relationships within the text, it entails analyzing the nearby words, sentences, or paragraphs. Models can distinguish between homonyms or polysemous words with multiple meanings based on the surrounding context by taking into account the linguistic context.

8.1.2. Semantic Context

Semantic Context involves figuring out a document's deeper meaning and conceptual understanding. It incorporates subject-matter expertise, concepts unique to a given domain, and the semantic connections between words and concepts in addition to the language used on the surface. By taking advantage of semantic context, models can more accurately analyze and categorize documents by identifying underlying themes, topics, or concepts.

8.1.3. Pragmatic Context

The term pragmatic context refers to the social, cultural, or situational elements that affect how a document is interpreted and comprehended. It considers the target market, the intended use, and any context-specific norms or expectations. In order to accurately categorize documents based on their intended use or target audience, pragmatic context aids models in taking into account elements like formality, tone, or genre.

8.1.4. Temporal Context

The temporal context involves taking into account the chronological progression and temporal arrangement of documents. It records the temporal relationships, patterns, or modifications in data found in a collection of documents. Models can track changes in topics or events, classify documents based on the historical context, or determine the value of older versus newer documents in the classification process by analyzing the temporal context.

8.1.5. User Context

User context considers the traits, tastes, or actions of the person who wrote the document or the intended audience. In order to customize document classification or recommendation systems, it takes into account elements like user profiles, demographics, or previous interactions. Models can offer more individualized and pertinent classifications based on user preferences and needs by incorporating user context.

8.1.6. Domain-Specific Context

A domain-specific context takes into account the distinctive traits, jargon, or terminologies associated with a specific domain or industry. It acknowledges that the language or topic distribution of documents from various domains may have unique characteristics that can affect the classification's accuracy. To enhance classification performance in specialized settings, models can be trained or fine-tuned on domain-specific data by utilizing domain-specific context.

9. Related works

This section provides an overview of various contextual document classification approaches proposed by different researchers. These approaches leverage the power of context, including word embeddings, topic modeling, transfer learning, and deep learning, to enhance the accuracy of document classification. We will summarize each study, examine the datasets and techniques used by the authors, explore the proposed solutions, and analyze the results obtained. Additionally, a comparative study of these methods was conducted to evaluate their performance and assess their advantages and disadvantages.

The evaluation metrics employed in these studies encompass accuracy, precision, recall, and F1 score.

Table 5: Synthesis of Contextual Document Classification Approaches

Authors	Approach	Dataset Used	Technique and solution	Evaluation Metrics
Smith et al. (2018)	Contextual Document Classification using Word Embeddings [8]	Custom Dataset	Word embeddings + deep neural network	Accuracy, precision, recall, F1 score
Jones et al. (2019)	Topic Modeling for Contextual Document Classification	AG News Dataset	Latent Dirichlet Allocation (LDA) + contextual features	Classification accuracy, precision, recall, F1 score
Patel et al. (2021)	Transfer Learning for Contextual Document Classification [9]	AG News Dataset	Transfer learning on pre-trained language models	Accuracy, precision, recall, F1 score
Chen et al. (2020)	Context-Aware Document Classification with Deep Learning [10]	Reuters-21578 Dataset	Hierarchical neural network architecture	Accuracy, precision, recall, F1 score

Each approach focuses on leveraging different contextual aspects, such as word embeddings, latent topics, transfer learning, and hierarchical dependencies, to enhance the document classification process. The evaluation metrics provide insights into the performance of these approaches in terms of accuracy, precision, recall, and F1 score.

10. Conclusion

In this chapter we discussed the various applications of text classification in fields such as sentiment analysis, language detection, and topic modeling. We first explained the crucial aspect of text classification "*word representation*", which involves converting words into numerical vectors that capture their semantic and syntactic information. Then, we explored different word representation techniques, including count-based methods like TF-IDF, prediction-based methods like Word2Vec and GloVe, and contextualized embedding techniques like ELMo and

BERT. Each technique has its advantages and suitability depending on the task at hand and the available resources. Additionally, we dived into rule-based classification systems, we also discussed statistics-based models, such as Naive Bayes, TF-IDF with cosine similarity, Support Vector Machines (SVM), and Latent Dirichlet Allocation (LDA). Finally, we explored the emergence of deep learning models in text classification. Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers are among the popular architectures used in deep learning-based text classification.

Chapter II:
Solution modeling

Chapter 2: Solution modeling

1. Introduction

This chapter aims to provide a comprehensive overview of our proposed approach, which encompasses various stages from data pre-processing to model selection. Specifically, we present the methodology utilized in this study, along with pertinent information regarding the data utilized in our analysis. Additionally, we detail the pre-processing methods employed to prepare the data for subsequent analysis. Finally, we discuss the classification models employed in this study, highlighting their respective strengths and limitations.

2. Proposed Approach and Methodology

This study investigates document classification based on context, focusing on the performance of different deep learning models for text classification. Specifically, we examine the differences between encoder and encoder-decoder models, the differences between text classification and zero-shot classification tasks, and the principles of distillation models.

The first aspect we investigate is the difference between encoder and encoder-decoder models. We compare the performance of these two types of models on text classification tasks, focusing on their strengths and weaknesses.

The second aspect we focus on is the difference between text classification and zero-shot classification tasks. Text classification tasks involve assigning one or more predefined categories to a document based on its content, while zero-shot classification tasks require the model to classify documents based on categories that were not included in the training data [11]. We examine the performance of various deep learning models on both types of tasks and evaluate the impact of data size and complexity on their accuracy.

The third aspect we investigate is the principles of distillation models. Distillation models aim to reduce the size and complexity of deep learning models while maintaining their performance. We compare the performance of fine-tuned pre-trained models with distillation models such as DistilBERT and DistilRoBERTa, which have fewer parameters than their pre-trained counterparts but aim to achieve similar accuracy. We also explore the trade-offs between model size and accuracy and examine the effectiveness of distillation models in reducing the computational costs of deep learning models.

By exploring these different aspects, we aim to contribute to the understanding of document classification based on context and provide insights into the strengths and weaknesses of different deep learning models.

To accomplish this, we fine-tune multiple pre-trained transformer models on our datasets for the text classification task, including BERT, DistilBERT, RoBERTa, DistilRoBERTa, BART, and DistilBART. We also fine-tune BART and DistilBART on the CNN dataset for the zero-shot classification task.

The fine-tuning process entails adapting the pre-trained models to our specific task by training them on our labeled datasets.

Additionally, we explore a hybrid approach that combines topic modeling and classical machine learning classification. We employ BERTopic, a topic modeling model that extracts topics from a corpus of documents, and train a classical machine learning classifier on the extracted topics. To assess model performance, we utilize standard metrics such as accuracy, precision, recall, and F1 score. We compare our models' performance to existing state-of-the-art models in the literature.

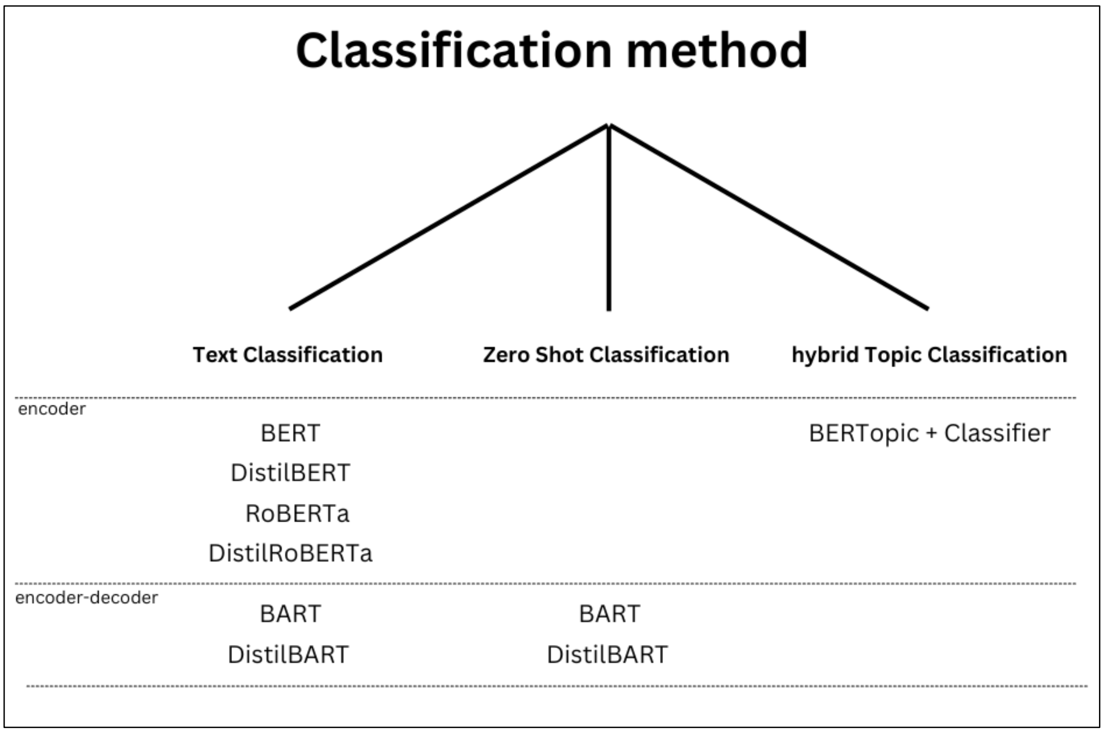


Figure 1: classification method overview

3. Dataset collection

Data collection is a crucial component of any data analysis project. In our thesis, we have worked with two different types of data: news articles labeled by their categories and a newly created dataset containing real company documents such as invoices, purchase orders, and stock reports. The process of collecting and curating these datasets is critical to ensuring that the data used for analysis is relevant, accurate, and representative of the problem domain. This section of the thesis will provide a detailed description of the methods used to collect and preprocess the data, including any challenges encountered and how they were addressed. We aim to ensure the reproducibility and transparency of our results and facilitate future research in this area.

A detailed description of the datasets that we have used and those which we have merged or created from scratch can be found in the Appendix A of this thesis.

4. Preprocessing

Preprocessing textual data is an essential step in natural language processing, especially when it comes to document classification. The process involves cleaning and transforming raw text data to improve its quality and standardize its format for further analysis [12]. The purpose of this section is to describe the preprocessing techniques used to prepare textual data for classification and discuss the rationale behind each step and its impact on the subsequent document classification accuracy.

Here's a general pipeline for the text preprocessing steps we propose:

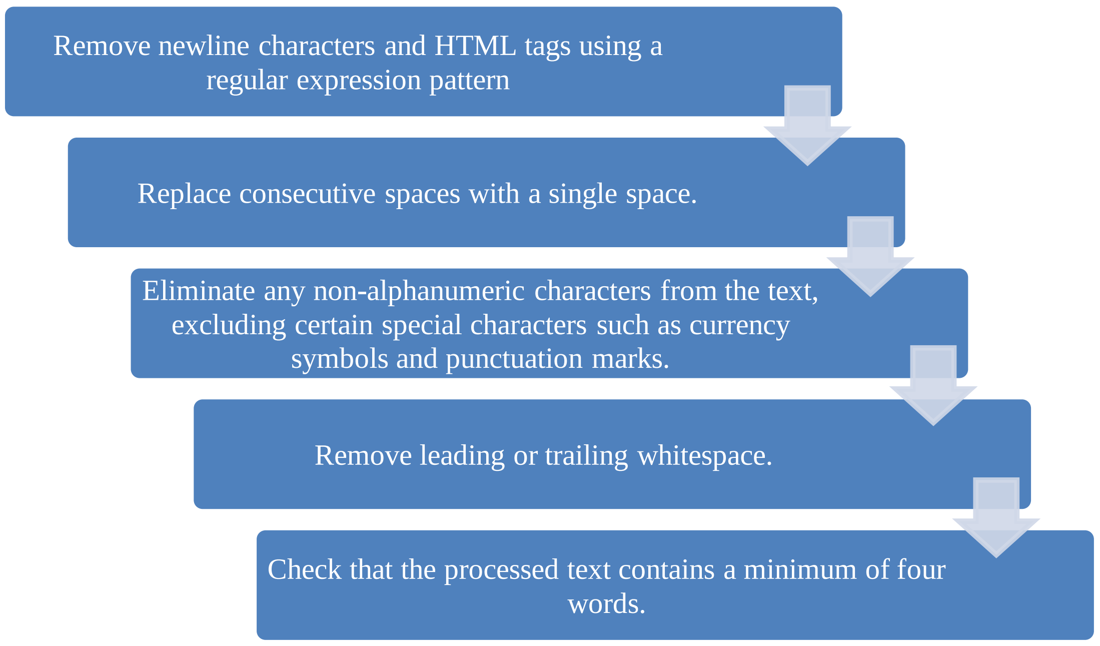

Remove newline characters and HTML tags using a regular expression pattern
Replace consecutive spaces with a single space.
Eliminate any non-alphanumeric characters from the text, excluding certain special characters such as currency symbols and punctuation marks.
Remove leading or trailing whitespace.
Check that the processed text contains a minimum of four words.

Figure 2: General pipeline for the text preprocessing

- **Remove newline characters and HTML tags using a regular expression pattern.**

The first step involves removing newline characters and HTML tags using a regular expression pattern to prevent downstream processing tasks from encountering errors or producing unexpected outcomes.

With HTML tags	Without HTML tags
<p> Regular expressions are useful for pattern matching. </p>	Regular expressions are useful for pattern matching.

- **Replace consecutive spaces with a single space.**

The next step replaces consecutive spaces with a single space to improve the text's readability, facilitating easier interpretation and analysis of the data in later processing stages.

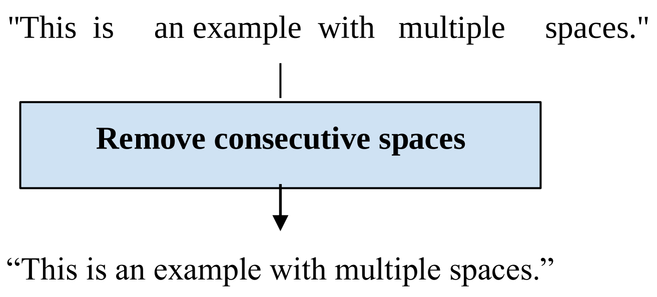

"This is   an example with   multiple   spaces."
Remove consecutive spaces
“This is an example with multiple spaces.”

Figure 3: Replacing consecutive spaces with a single space.

- **Eliminate any non-alphanumeric characters from the text**

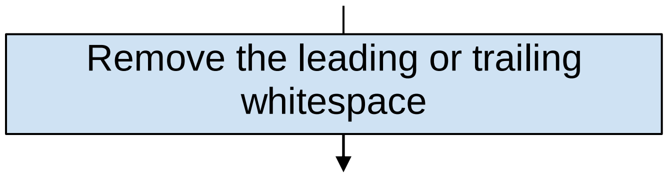
To standardize the text format, the function eliminates any non-alphanumeric characters from the text, excluding certain special characters such as currency symbols and punctuation marks.

With non-alphanumeric characters	Without non-alphanumeric characters
"The stock price of ABC Corp. (NYSE: ABC) rose by $10.50 (or 5.2%) in the last quarter."	"The stock price of ABC Corp. NYSE: ABC rose by 10.50 (or 5.2) in the last quarter."

- **Remove leading or trailing whitespace.**

The function removes leading or trailing whitespace and checks that the processed text contains a minimum of four words to ensure that the resulting text is informative enough to classify accurately.

"   This is a sentence with leading and trailing whitespace.   "



"This is a sentence with leading and trailing whitespace."

Figure 4: Removing leading or trailing whitespace.

- **Check that the processed text contains a minimum of four words.**

The text preprocessing procedure is essential in preparing data for text classification models such as BERT, BART, etc. or other deep learning models. The cleaned text data is used to train and test the machine learning models, enhancing their accuracy. By utilizing this preprocessing function, the accuracy of the text classification model can be significantly improved by removing unwanted characters and data noise.

5. Transfer Learning

5.1. Introduction

Transfer learning is an effective method for enhancing the performance of document classification models with limited labeled data by utilizing pre-trained models on massive datasets. It enables the transfer of knowledge learned from pre-training on one task or domain to another, thereby reducing the need for large amounts of labeled data and improving the generalization ability to unseen data [13] [14]. Several approaches can be employed in transfer learning for document classification, including fine-tuning pre-trained models, using them as feature extractors, or combining multiple pre-trained models for multi-modal classification.

Pre-trained language models, such as BERT and GPT, have shown remarkable performance in text classification tasks when fine-tuned on task-specific datasets [15] [16] [17] [18]. Another way to utilize pre-trained models is to use them as feature extractors to obtain document representations that are then fed into a classifier.

The recent advancements in transfer learning through the use of language models have highlighted the importance of unsupervised pre-training in language understanding systems [15].

5.1.1 Fine-tuning

Fine-tuning is a popular approach in transfer learning that involves adapting a pre-trained model to a new task or domain with limited labeled data [9]. In the context of Natural Language Processing (NLP), fine-tuning refers to the process of adjusting a pre-trained language model's parameters, such as the transformer-based models, to a new downstream task, such as sentiment analysis or question answering, using a task-specific dataset. Transformer-based models, such as BERT, GPT-2, and RoBERTa, have achieved state-of-the-art performance on various NLP tasks due to their ability to capture word and phrase contexts and meanings. However, training these models from scratch on a new task or domain can be extremely resource-intensive [15] [16] [17] [18]. Fine-tuning a pre-trained Transformer-based model on a task-specific dataset can significantly reduce the training time and improve performance, especially when the dataset is small or the task is similar to the pre-training task.

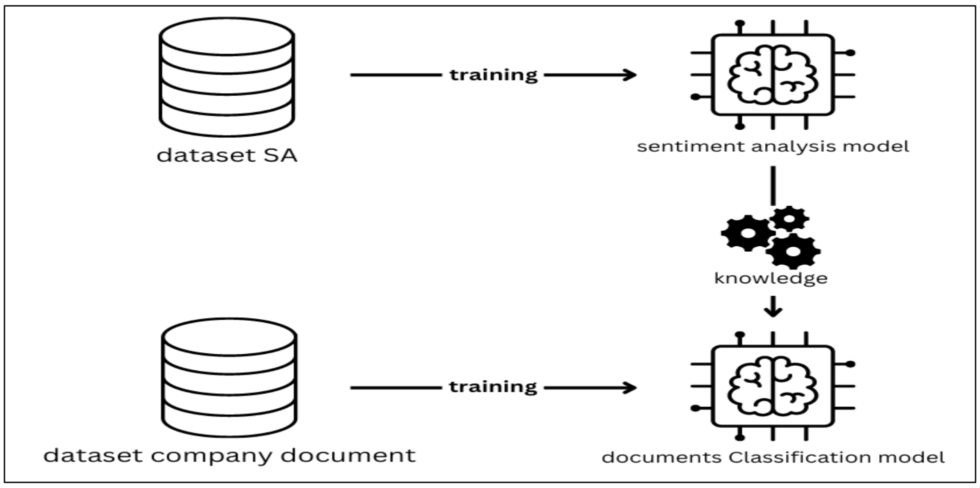


Figure 5: Utility of fine tuning.

Fine-tuning is a powerful and flexible method for adapting pre-trained transformer-based models to new NLP tasks and domains with limited labeled data. This approach has become a standard practice in the NLP community [13].

5.1.2 Transformer

A novel network architecture named the Transformer has been developed by researchers to facilitate sequence transduction tasks, like machine translation, without utilizing recurrent or convolutional neural networks. The conventional models based on recurrent neural networks (RNNs) necessitate sequential computation [19], thereby constraining parallelization and increasing the training time for longer sequence lengths. The Transformer deploys a self-attention mechanism to estimate global dependencies between input and output sequences. It consists of an encoder and a decoder, each containing six identical layers [19] [20] [21] [22] [15], and utilizes residual connections and layer normalization for the flow of information. This model surpasses the existing state-of-the-art performance for machine translation, requiring less training time and being capable of generalizing to other tasks, such as English constituency parsing. By offering a more efficient and simpler alternative to conventional models, this novel approach has the potential to have applications in language translation and other fields [19].

5.1.3 Architecture

In many neural sequence transduction tasks, the encoder-decoder architecture has proven to be highly effective [19] [23]. This model architecture involves the encoder mapping an input sequence of symbol representations (x1, ..., xn) to a sequence of continuous representations z = (z1, ..., zn). Subsequently, the decoder uses z to generate an output sequence (y1, ..., ym) one element at a time, while also incorporating the previously generated symbols as additional input at each step, making the model auto-regressive [21]. The Transformer model, introduced in recent years, follows this encoder-decoder architecture, but utilizes stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, as illustrated in the left and right halves of Figure 10, 11, respectively. The Transformer's novel approach to encoding and decoding sequences has shown superior performance compared to conventional models, such as those based on recurrent neural networks.

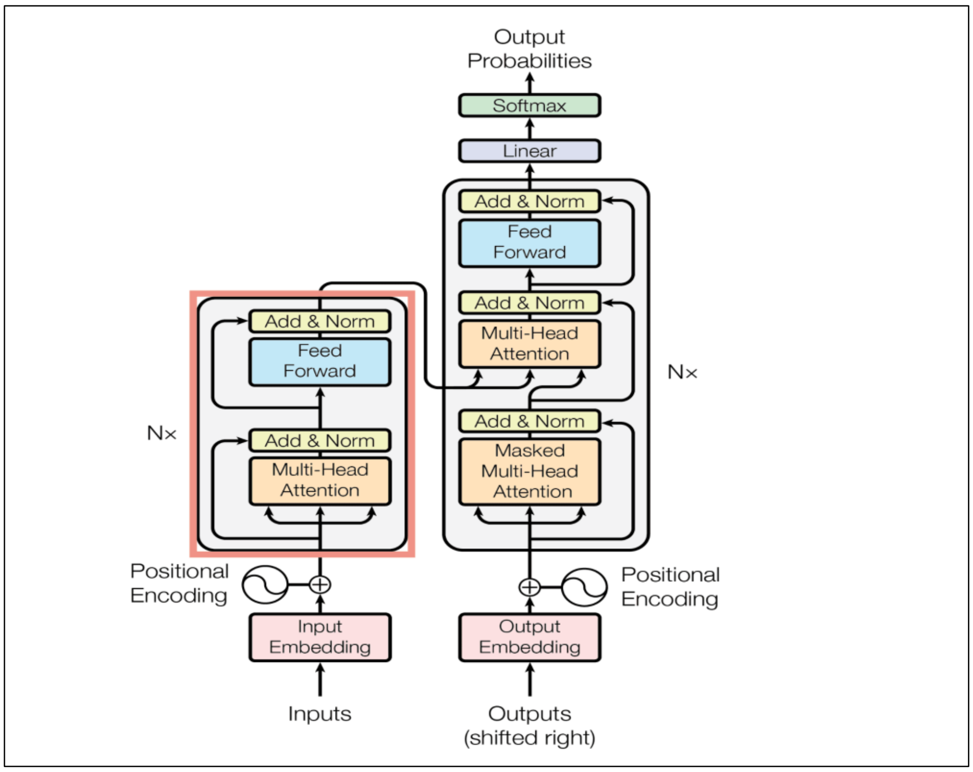


Figure 6: The Transformer - Architecture [19]

5.1.4. self-attention

Self-attention is a component of transformers, enabling them to capture relationships between elements in a sequence. It allows each element to interact with others, determining its representation by attending to important information. This is achieved through linear projections that generate queries, keys, and values. The dot product between queries and keys measures similarity, and scaling ensures stable gradients. Softmax is applied to obtain attention weights, reflecting element importance. Weighted values are summed to yield the output representation.

Self-attention captures both local and global dependencies, aiding tasks with long-range relationships. Transformers utilize self-attention to achieve state-of-the-art performance in natural language processing. By modeling dependencies through attention weights, the model focuses on relevant information, facilitating accurate predictions.

5.1.5. Encoder models

Encoder models exclusively utilize the encoder component within a Transformer model. At each stage of processing, the attention layers possess the capability to access all the constituent words present in the initial sentence. These models are commonly recognized for their inherent "bi-directional" attention mechanism and are frequently referred to as auto-encoding models [15][19][24].

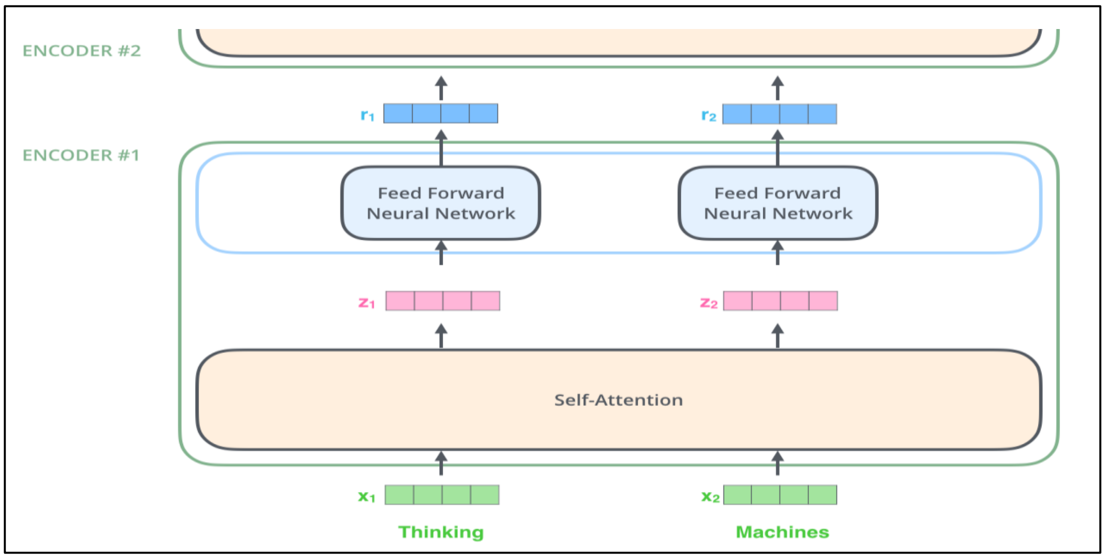


Figure 7: Encoder [24]

During the pretraining phase of these models, a prescribed approach typically involves introducing deliberate alterations to a given sentence, such as random word masking, thereby compelling the model to undertake the task of reconstructing or identifying the original sentence [15][19].

5.1.6. Encoder-decoder

Encoder-decoder models, often referred to as sequence-to-sequence models, leverage the complete Transformer architecture. Within this framework, the attention layers of the encoder are capable of accessing all the constituent words in the original sentence, while the attention layers of the decoder can solely consider the words preceding a specific word in the input sequence.

The pretraining process of these models encompasses the utilization of objectives from both encoder and decoder model, though typically involving more intricate methodologies. For instance, in the case of BART (Bidirectional and Auto-Regressive Transformers), a notable example, a pre-training phase entails the replacement of random text spans, encompassing multiple words, with a designated mask token. Predicting the text that will be replaced by the mask token is the goal going forward.

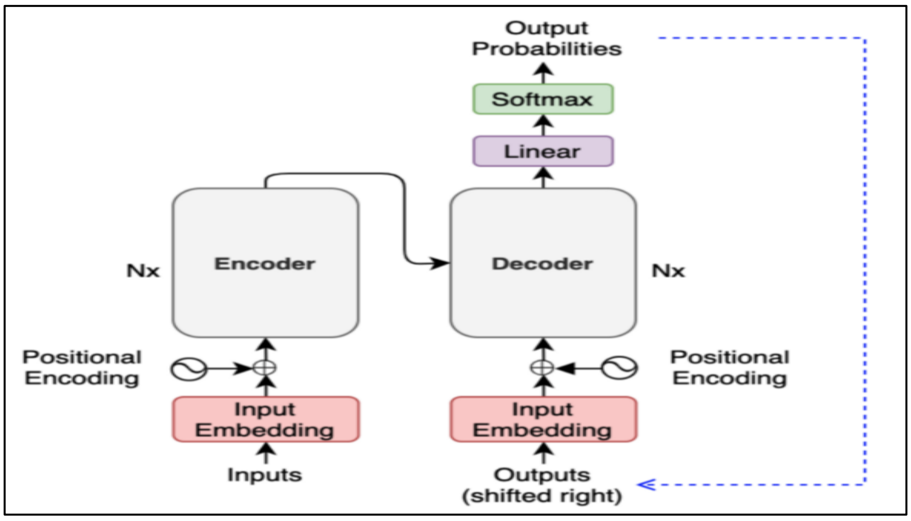


Figure 8: Encoder-Decoder [25]

5.1.7. distillation

Knowledge distillation is a compression technique that involves training a smaller model called a student to mimic the behavior of a larger model, known as the teacher, or an ensemble of models. The student model learns from the soft target probabilities estimated by the teacher model, with the objective of reproducing the same output distribution as the teacher model [20] [26].

To achieve this objective, a distillation loss function is used to leverage the full distribution of the teacher model. The distillation loss function involves a softmax-temperature that controls the smoothness of the output distribution during training. This temperature is applied to both the student and teacher models, ensuring that the student model learns to mimic the behavior of the teacher model accurately. In addition to the distillation loss, the student model is also trained with a masked language modeling loss and a cosine embedding loss. The masked language modeling loss involves predicting masked words in a sentence, and the cosine embedding loss aligns the directions of the hidden states vectors of the student and teacher models. This helps to further improve the performance of the student model and make it more efficient while maintaining high accuracy [20] [26].

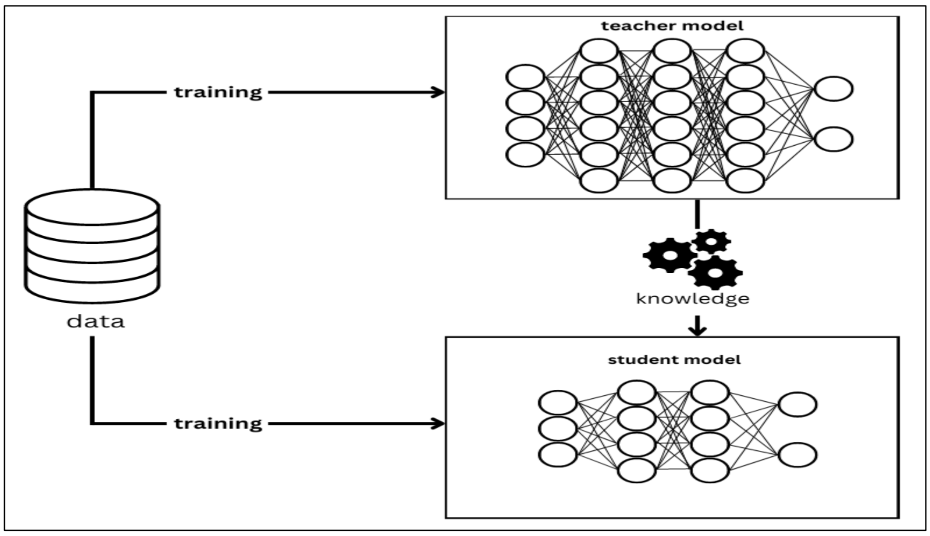


Figure 9: distillation principle in deep learning

5.2 BERT Model

5.2.1. Definition

BERT, which stands for Bidirectional Encoder Representations from Transformers, is a natural language processing (NLP) model developed by Google in 2018. BERT is a pre-trained model that uses deep neural networks to understand the contextual relationships between words in a sentence [15] [27].

BERT is designed to learn contextual relationships between words by training on a large corpus of text data. This allows it to capture more accurate representations of language than previous models. BERT is also bidirectional, which means it can process words in both directions, left to right and right to left, giving it a more comprehensive understanding of language.

In the field of document classification, BERT has been used to train models that can accurately classify documents into different categories, such as spam or non-spam emails, positive or negative sentiment reviews, or news articles based on their topics. BERT's ability to capture contextual relationships between words and understand complex language structures makes it a powerful tool for document classification tasks [15].

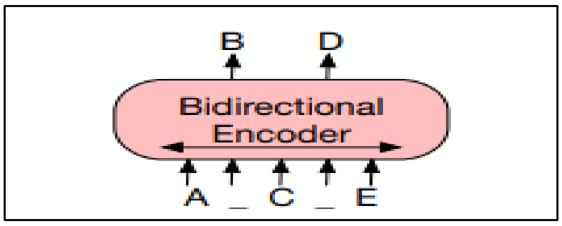


Figure 10:BERT MLM [21]

5.2.1 BERT architecture

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art neural network architecture for natural language processing tasks [12]. BERT's architecture is based on a multi-layer bidirectional Transformer encoder, which allows it to capture both the left and right contexts of a word. The BERT architecture is characterized by its unified architecture across different tasks [22] [27]. The pre-trained architecture is almost identical to the final downstream architecture, with minimal differences. BERT also uses bidirectional self-attention, which allows each token to attend to both left and right contexts, unlike the constrained self-attention used by the GPT Transformer. The BERT architecture comes in two sizes: BERTBASE and BERTLARGE [15] [27].

BERTBASE has 12 Transformer blocks, a hidden size of 768, and 12 self-attention heads, while BERTLARGE has 24 Transformer blocks, a hidden size of 1024, and 16 self-attention heads. The number of parameters for BERTBASE is 110M, while for BERTLARGE, it is 340M [15] [16] [27]. BERTBASE is of the same size as OpenAI's GPT [15] [17] [18], which enables comparison between the two models.

5.3. Roberta Model

5.3.1. Définition

RoBERTa is a pre-trained language model that has been optimized to enhance the performance of various natural language processing (NLP) tasks. It is an extension of the BERT model and has been modified by incorporating several improvements, such as dynamic masking during pre-training [22], full-sentence training without next sentence prediction (NSP) loss, larger mini-batches, and a larger byte-level byte pair encoding (BPE). Additionally, RoBERTa is trained on a larger and more diverse dataset for longer periods of time, compared to BERT, to improve its performance on NLP tasks [22] [28] [29].

5.3.2 RoBERTa architecture

The architecture of RoBERTa is based on the Transformer architecture, which is a neural network designed for sequence-to-sequence tasks. The encoder component of the Transformer architecture is used in RoBERTa, which consists of multiple layers. Each layer in the encoder has a self-attention mechanism and a feedforward neural network. The self-attention mechanism is used to weigh different parts of the input sequence based on their importance, and the feedforward neural network applies non-linear transformations to the output of the self-attention mechanism. RoBERTa uses a 12 or 24 layer Transformer [22] [29], depending on the configuration. During pre-training, RoBERTa is trained to predict missing tokens in a sequence. This pre-training enables RoBERTa to learn representations of words and phrases that are useful for downstream NLP tasks. RoBERTa can achieve state-of-the-art performance on various tasks, including text classification, question answering, and natural language inference, when fine-tuned on a specific task [15] [22] [28] [29].

5.4. BART

5.4.1 Definition

BART is a pre-training approach that utilizes denoising autoencoders. It is implemented as a sequence-to-sequence model with a bidirectional encoder and a left-to-right autoregressive decoder. BART can handle any type of document corruption and shows comparable performance to RoBERTa on discriminative tasks while achieving new state-of-the-art results on various text generation tasks [21] .

BART pre-trains by corrupting documents and then optimizing a reconstruction loss, which is the cross-entropy between the decoder's output and the original document. BART can handle any type of document corruption, even in the extreme case where all information about the source is lost. BART experiments with several previously proposed and novel transformations, and there is potential for development of other new alternatives.

The representations produced by BART can be used in several ways for downstream applications, including sequence classification tasks. For sequence classification tasks, the same input is fed into the encoder and decoder, and the final hidden state of the final decoder token is fed into a new multi-class linear classifier. BART shows comparable performance to RoBERTa on discriminative tasks and achieves new state-of-the-art results on various text generation tasks [21] [27]

.

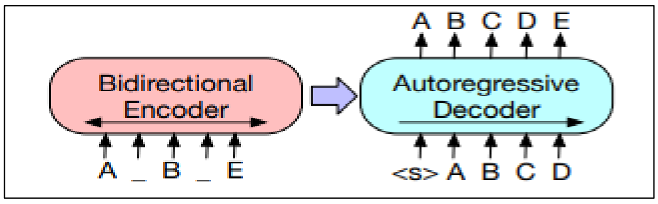


Figure 11: BART LM [17]

5.4.2. BART architecture

BART uses the standard sequence-to-sequence Transformer architecture from Vaswani et al. (2017), with modifications to the activation functions and parameter initialization. BART replaces ReLU activation functions with GeLUs and initializes parameters from N(0, 0.02) [21] [27]. The base model has six layers in the encoder and decoder, and the large model has 12 layers in each. The architecture is similar to that used in BERT, with differences in the cross-attention and feed-forward network before word-prediction [15] [21] [27].

5.5. BERTopic

BERTopic is a topic modeling algorithm that aims to generate topic representations by leveraging three main steps. First, it converts documents into embedding representations using a pre-trained language model. Second, it reduces the dimensionality of the embeddings to enhance the clustering process. Finally, it extracts topic representations from clusters of documents using a customized class-based variation of TF-IDF. By employing these steps, BERTopic identifies semantically similar documents and assigns them to relevant topics.

5.5.1. BERTopic Architecture

BERTopic's architecture consists of several key components. It begins by using the Sentence-BERT (SBERT) framework to convert documents into dense vector representations. UMAP is then employed for dimensionality reduction, preserving important features while reducing high-dimensional data. The reduced embeddings are clustered using HDBSCAN, which identifies clusters with varying densities and models noise as outliers. BERTopic modifies TF-IDF to capture word importance within clusters, enabling the generation of topic-word distributions. The algorithm further refines topic extraction by iteratively merging similar topics based on TF-IDF representations. This comprehensive approach allows BERTopic to extract meaningful topics from large document collections while considering semantic similarity and optimizing the clustering process.

6. Training methods

In this study, we focus on three distinct methods for model training in the context of text classification: text classification training, zero-shot classification training, and a hybrid approach that combines topic modeling and classical classification. These approaches aim to enhance the accuracy and effectiveness of text classification models by utilizing different techniques and strategies. Through this research, we aim to investigate and compare the performance of these training methods to identify their strengths and limitations in addressing the challenges of document classification.

The training process for text classification involves crucial steps. Firstly, data preprocessing is performed, including tasks like tokenization, handling special characters, and transforming text into numerical representations. Model selection is then conducted, considering architectures like BERT, DistilBERT, RoBERTa ..., which have undergone pre-training on extensive textual data to capture intricate language patterns. The selected model is trained using a preprocessed dataset, adjusting internal parameters through backpropagation and optimizing hyperparameters. Evaluation is carried out on a separate test dataset, assessing performance using metrics like accuracy, precision, recall, and F1 score. Fine-tuning can be performed based on evaluation results, refining the model's accuracy and generalization capabilities iteratively.

6.1. Text classification

The process of text classification involves training a model to automatically assign predefined categories or labels to text documents based on their context or content

This research work focused on document classification using transformer-based models, including BERT, DistilBERT, BART, DistilBART, RoBERTa, and

DistilRoBERTa, the dataset was preprocessed, models were selected, fine-tuned, and trained and hyperparameter optimization.

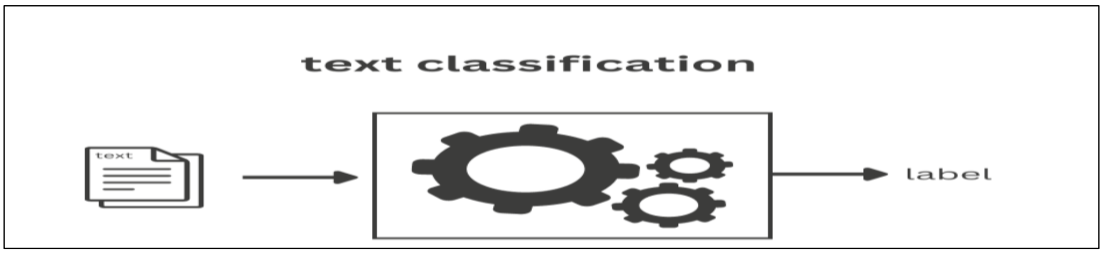


Figure 12: text classification process

Evaluation metrics were used to assess model performance F1, precision, recall, and further refinement was conducted. The findings contribute insights into the effectiveness of these models for document classification on the CNN dataset and ag News.

6.2. Zero shot classification

This research uses the technique of zero-shot classification, specifically employing BART and DistilBART models. Zero-shot classification enables the models to classify documents into categories that were not seen during training, expanding their capabilities beyond the predefined labels.

BART and DistilBART, which utilize bidirectional and auto-regressive transformers, were fine-tuned to perform zero-shot classification on the CNN dataset and AG News. This involved training the models to understand the relationship between the document content and a set of target labels, allowing them to make predictions on unseen categories.

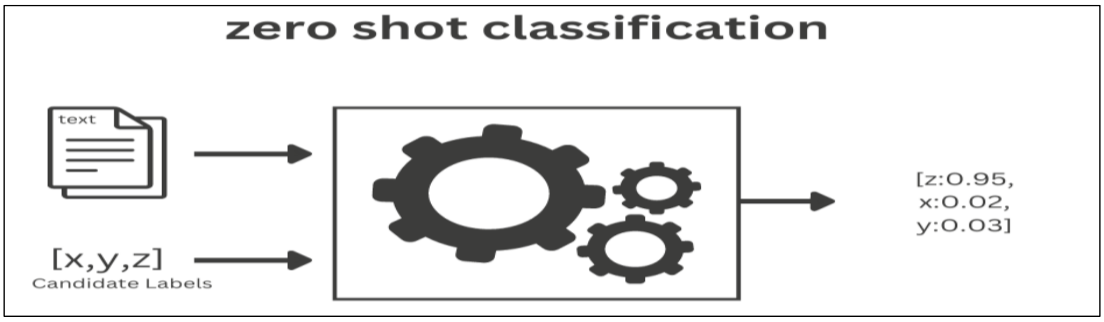


Figure 13: zero shot classification, predict process

During the training process, the models were exposed to documents and their corresponding target labels, enabling them to learn the associations between the document features and the label semantics. The models were optimized using backpropagation and hyperparameter tuning to enhance their zero-shot classification performance.

6.3. A Hybrid Architecture Combining BERTopic and Classifier:

We propose a hybrid approach for document classification that combines the strengths of BERTopic and classifiers. Our approach first uses BERTopic to cluster the documents into similar groups based on their semantic content. Then, a classifier is trained on each cluster to predict the class of new documents within that cluster. By using BERTopic to group documents with similar content together, the classifier can more accurately classify new documents within each cluster, resulting in improved overall classification accuracy. For example, this approach could be used to classify customer support tickets in a large organization, where BERTopic could group similar tickets together based on their content and classifiers could be trained on each cluster to predict the appropriate resolution for new tickets.

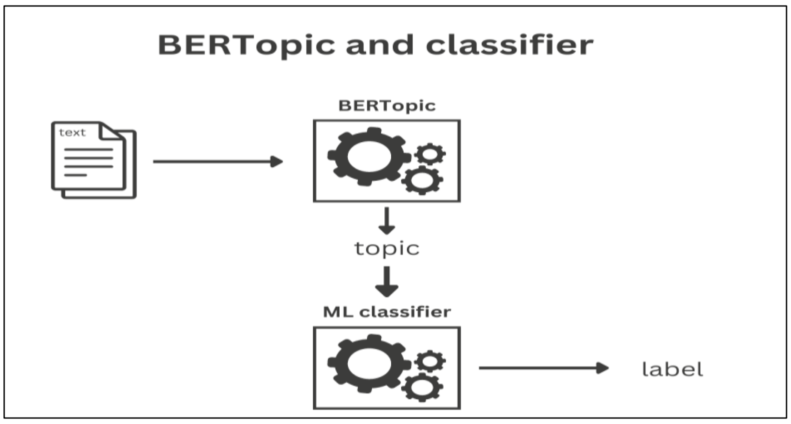


Figure 14: BERTopic + classifier, predict process

7. Language detection

Language detection is the process of identifying the underlying language by analyzing the text's content and using a variety of techniques and algorithms. These methods include basic statistical analysis as well as more sophisticated machine learning techniques. The development of reliable and precise language detection systems is the result of significant research advancements in this area.

Statistical analysis based on character n-grams is one of the fundamental methods for language detection. This approach is based on the idea that different languages display distinctive statistical patterns in terms of character distribution and frequency. Statistical models can be created to categorize texts into different languages by looking at the frequency of character n-grams (sequences of n characters) in a given text.

Recurrent neural networks (RNNs) and transformer models, in particular, have demonstrated outstanding performance in language detection tasks in recent years. These models can capture intricate linguistic dependencies and patterns, which enables them to correctly identify the language of a given text even in difficult circumstances.

7.1. Xlm roberta base language detection

An effective tool for language detection has emerged in the form of the advanced language model XLM-RoBERTa Base, which uses its contextual understanding and cross-lingual capabilities to achieve high accuracy and robustness.

The Transformer architecture, a deep learning model that transformed NLP tasks, serves as the foundation for XLM-RoBERTa Base. XLM-RoBERTa Base encodes textual information in a rich and thorough manner, capturing intricate linguistic patterns and nuances across multiple languages. It is pre-trained on vast amounts of multilingual data. The model can learn general language representations thanks to this pre-training, which makes it very efficient at detecting languages.

The ability of XLM-RoBERTa Base to transfer knowledge between languages is one of its main advantages. XLM-RoBERTa Base can accurately categorize texts even in low-resource or previously undiscovered languages by comprehending the shared traits and structures present in various languages. The model can handle a variety of linguistic variations and accurately identify the language of various text inputs thanks to this transfer learning capability.

XLM-RoBERTa Base has proven effective at language detection in a number of benchmark tests and competitions. The model consistently outperforms traditional statistical methods and even earlier deep learning models to achieve state-of-the-art performance. It is the best option for real-world language detection applications due to its robustness, accuracy, and adaptability.

8. Optical Character Recognition

Optical Character Recognition (OCR) is a technology that enables the conversion of printed or handwritten text into machine-readable and editable formats.

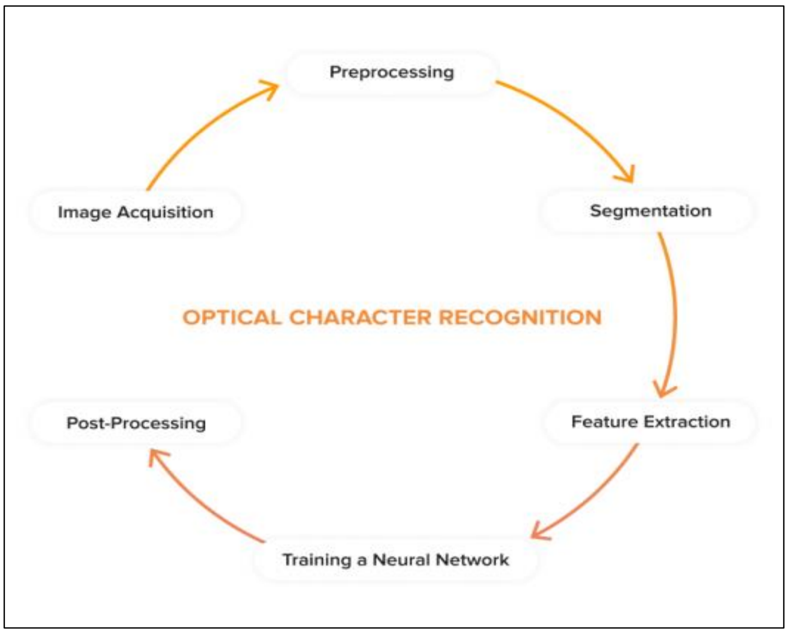
OCR systems can automatically identify and extract text from scanned documents, photographs, or digital images by using image processing, pattern recognition, and machine learning algorithms.

There are typically several steps in the OCR procedure. The input image or document is first preprocessed to improve image quality, fix distortions, and normalize text orientation. Then, individual characters or groups of characters are found and isolated using feature extraction techniques. Shape, size, and the spatial relationships between text elements are a few of these characteristics.

The extracted text is then categorized and recognized using machine learning algorithms like neural networks, hidden Markov models, or support vector machines. In order to learn the patterns and variations in various fonts, languages, and writing styles, these algorithms are trained on large datasets of annotated text samples.

OCR technology has advanced significantly in recent years thanks to the development of deep learning methods as well as the accessibility of large datasets. Modern OCR performance has been attained by convolutional neural networks (CNNs) and recurrent neural networks (RNNs), which outperform conventional techniques.

Figure 15: Optical character recognition process



8.1 Textract

Textract is a cloud-based service offered by Amazon Web Services (AWS) that extracts text and data from documents using cutting-edge machine learning algorithms. In order to make it simpler to integrate and process huge amounts of textual data, it is designed to automate the process of document analysis and data extraction.

Textract enables the output of structured data from scanned documents, PDF files, or text-containing images. It analyzes and extracts data from documents using a combination of methods, including optical character recognition (OCR), natural language processing (NLP), and computer vision. Invoices, contracts, financial statements, and medical records are just a few of the many document types it can handle. Morever, it offers an API that developers can use to incorporate the service into their workflows or applications. This enables seamless integration of Textract's features and opens the platform up to a variety of sectors and use cases, including document management, content extraction, and data mining.

9. Cloud-Based Software Solution for Automated Document Classification: Modeling and Development

This thesis presents the development and evaluation of a cloud-based software solution for document management, with a focus on automated document classification.

To improve the effectiveness and organization of document storage and retrieval processes, the software based on technologies like Optical Character Recognition (OCR), language detection, and document classification methods.

This research goal is to tackle the problems with manual document classification and offer users a scalable and practical solution.

9.1 Methodology

The software development process involved the implementation of key functionalities as shown in figure 20, including document upload, download and navigation. The core focus was on the document classification process, which integrated OCR, language detection, and document classification techniques. OCR technology enables the extraction of machine-readable text from scanned or image-based documents. Language detection helps identify the language in which a document is written, facilitating language-specific classification. Document classification techniques (zero shot, text classification), were employed to assign documents to predefined categories or classes.

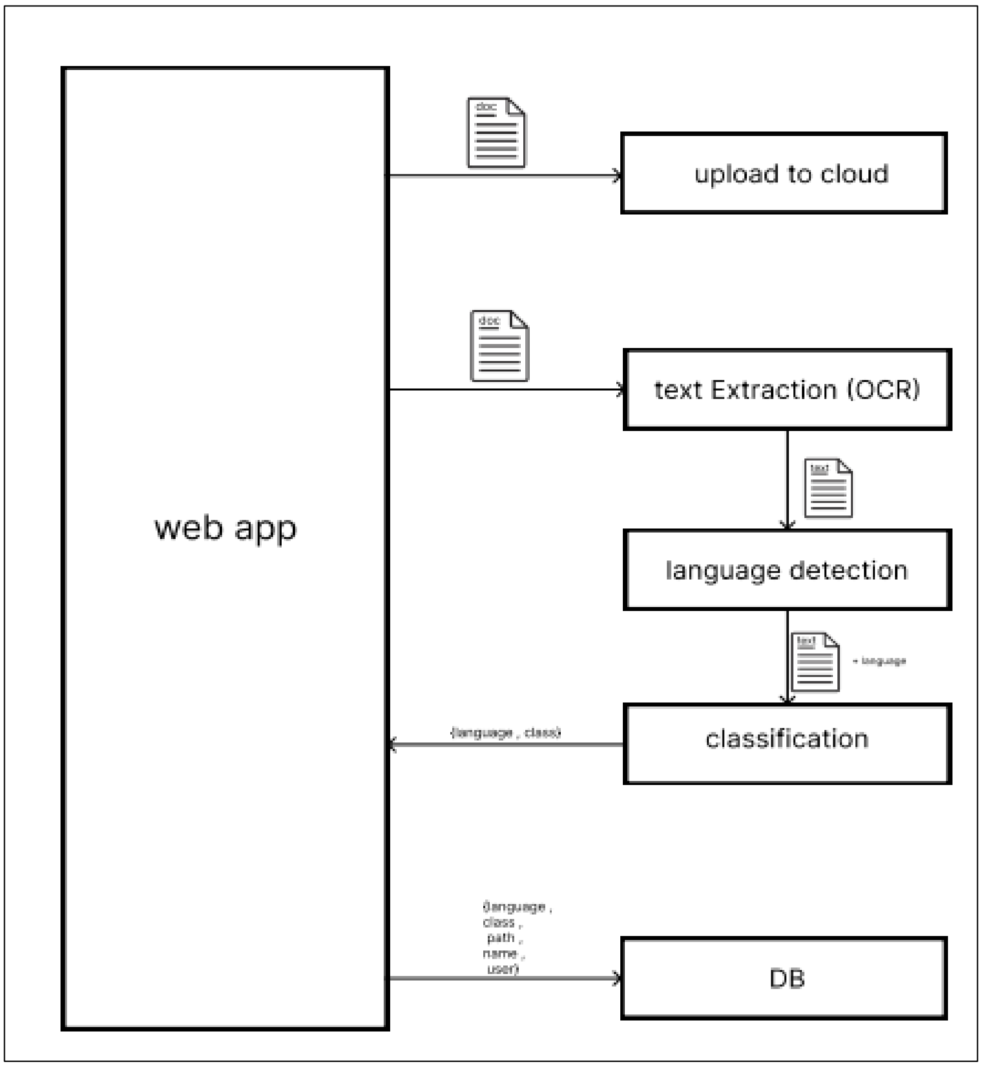


Figure 16: Web app general process.

9.2 Classifier API: Text Classification with Language Detection and OCR

This section provides an overview of the API designed for text classification with language detection. The API allows users to upload a file, extract the text using OCR, detect the language of the text, and classify it into predefined categories.

Request: The request to the API endpoint should include the file to be uploaded. The supported file formats include PDF, images (JPEG, PNG), and text files (TXT).

In this part, we present the sequence diagram that illustrates the flow of interactions within our text classification API. The sequence diagram provides a visual representation of the step-by-step process and the communication between various components involved in handling the API request.

By examining the sequence diagram, we can gain a clear understanding of how different entities interact with one another and the order in which these interactions take place. The diagram showcases the flow of information and the specific APIs utilized during each stage of the text classification process, including OCR extraction, language detection, and classification.

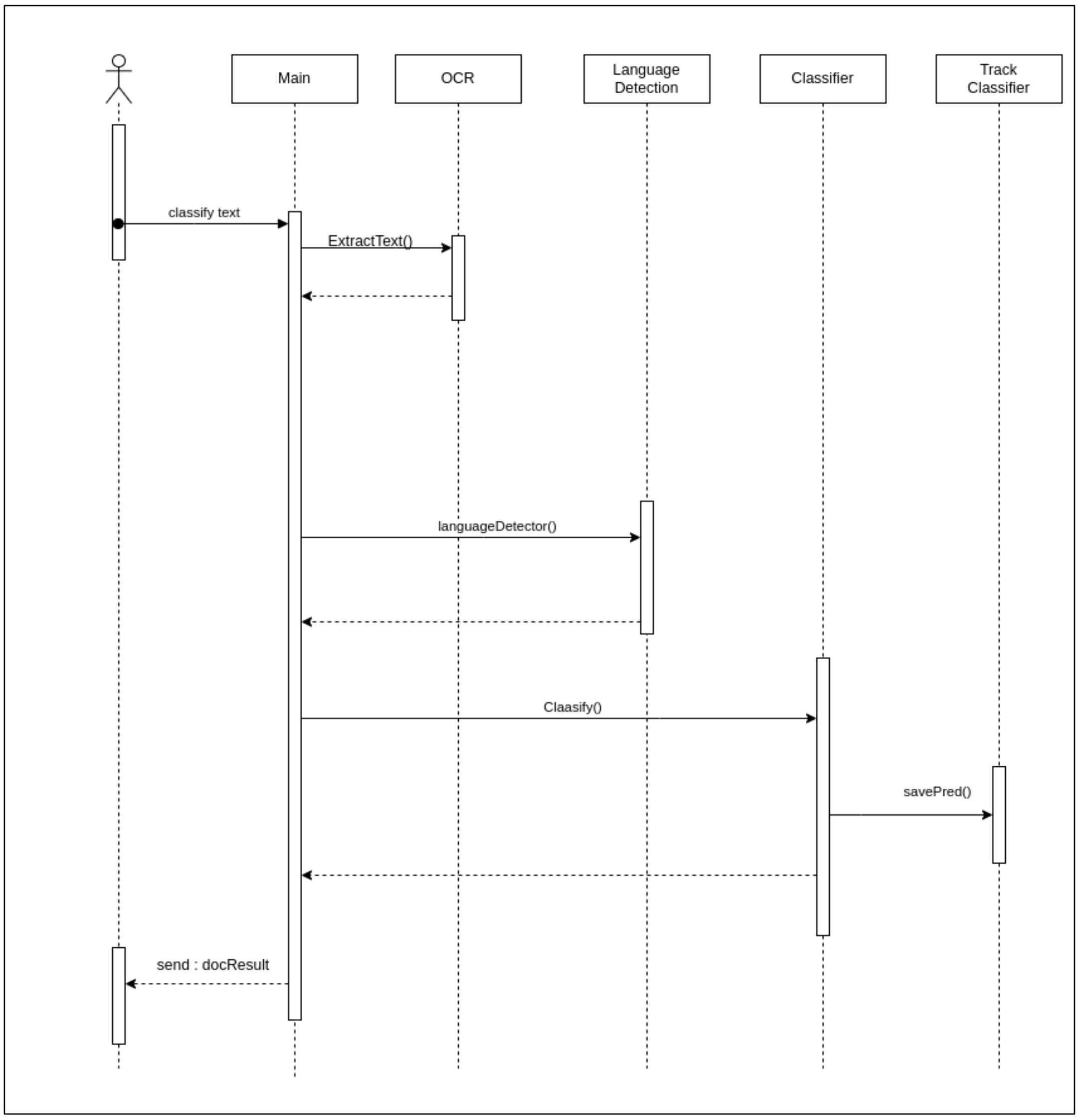


Figure 17: sequence diagram for classifier api

After the text classification process, we store both the text and the assigned label in a database to enhance our models in the future. By saving this information, we can leverage it for various purposes, such as model training, performance evaluation, and further analysis.

10. Conclusion

The information provided in this chapter cover various aspects of NLP, including Transformer-based models like BERT, RoBERTa, BART, and BERTopic, as well as training methods for text classification. These models, such as BERT and RoBERTa, capture contextual relationships and achieve state-of-the-art performance. BART excels in text generation tasks, while BERTopic combines pre-trained language models with clustering techniques for topic extraction. Our study explores training methods like text classification, zero-shot classification, and hybrid approaches. Additionally, language detection techniques utilize statistical analysis and machine learning to identify text languages and Optical Character Recognition (OCR) techniques enabling the extraction and interpretation of text from images and scanned documents, further expanding the scope of natural language processing applications.

Chapter III:
Tests and evaluation

Chapter 3: Tests and Evaluation

1. Introduction

In this chapter, we will delve into the implementation of the method described in the previous chapter, providing a comprehensive examination of the entire process. Our discussion will encompass the environment utilized, libraries and APIs used, and an extensive demonstration of the implementation with detailed code examples. Additionally, we will present the classification results, conduct a comparative study with previous related works, and showcase the classification in action through a user-friendly interface in the simulation section.

1.2 Hardware Resources

Table 6: Hardware resources

Task	hardware	usage
Dataset preparation	Intel Xeon 2.20 GHz CPU, 32 GO RAM (kaggle)	- Dataset creation - Preprocessing - Analytics
Training	NVIDIA T4 x2 GPU , 16 GO VRAM(kaggle)	- Training all models - Evaluate models
ML Deploy	2 vCPU- 16 GO RAM (HuggingFace)	- models API
App Deploy	i7 8th 8 GO RAM	- Classifier API - Cloud Manager API - Web App

1.3 Software Resources

This section will provide an overview of the software resources employed during the implementation, encompassing the programming language, frameworks, and libraries.

1.3.1 Programming Language and frameworks

Here are the programming languages, frameworks and libraries that we have used.

Python Langage	Python is a high-level, general-purpose programming language known for its emphasis on code readability. The latest stable release, Python 3.9.5, introduces several new features and optimizations, incorporating 111 commits since version 3.9.4.
Kaggle	A Google subsidiary, offers a web-based data science environment called Kaggle3. It enables users to discover and publish datasets, develop models, collaborate with fellow data scientists and machine learning engineers, and participate in data science competitions to tackle various challenges.
Numpy	Numpy is a Python library designed for numerical analysis. It enhances the Python programming language by introducing capabilities for handling extensive, multi-dimensional arrays and matrices. Additionally, Numpy offers an extensive collection of high-level mathematical functions that can be applied to manipulate these arrays efficiently.
Pandas	Pandas is a Python library dedicated to data manipulation and analysis, with a focus on tables. It provides a wide range of functionalities and data structures specifically tailored for efficiently handling numerical tables and time series data.
Matplotlib	Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.
PyTorch	Is an open-source machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing.
Scikit-learn	Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
Fast API	FastAPI is a modern web framework for creating RESTful APIs in Python. Due to its simplicity, speed, and robustness, it has rapidly gained popularity among developers since its initial release in 2018. To validate, serialize, and deserialize data, FastAPI uses Pydantic and type hints. Additionally, it automatically creates OpenAPI documentation for APIs created with it.
Express.js	Express.js is a fast and minimalistic web application framework for Node.js. It simplifies the process of building web applications by providing a straightforward and flexible set of features for handling routing, middleware, and HTTP requests. Express.js is known for its simplicity, scalability, and extensive ecosystem of plugins and extensions, making it a popular choice for building web servers and APIs in the Node.js ecosystem.

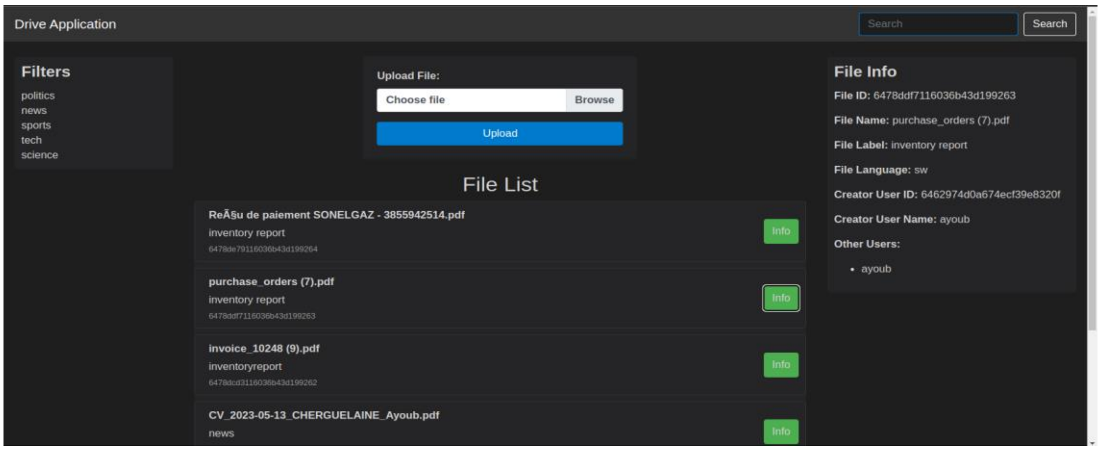
Mango db	MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License (SSPL) which is deemed non-free by several distributions. MongoDB is a member of the MACH Alliance.
Textract	Textract is a service provided by Amazon Web Services (AWS) that enables automatic extraction of text and data from a variety of document formats. It uses machine learning algorithms to analyze documents such as PDFs, images, and scanned files, and extracts text, tables, and structured data from them. Textract makes it easier to process and extract information from large volumes of documents, saving time and effort in manual data entry and document processing tasks.
Transformers	Transformers is a comprehensive library that offers APIs and tools to simplify the process of downloading and training cutting-edge pretrained models. By utilizing pretrained models, you can significantly reduce computational expenses, environmental impact, as well as the time and resources required for training models from scratch.

1.4 Implementation and Results

1.4.1 Application

1.4.1.1 Drive Storage Web App

The web application is built using the Express.js framework. It includes features such as login, signup, and a dashboard where users can view their uploaded documents. The file detail page provides additional information about each document, along with a share button. Users can also upload new files and report misclassifications. The web app interacts with the other APIs to manage user authentication, document addition/modification, and configuration.



ReÃ§u de paiement SONELGAZ - 3855942514.pdf
inventory report
6478de79116036b43d199264

purchase_orders (7).pdf
inventory report
6478ddf7116036b43d199263

invoice_10248 (9).pdf
inventoryreport
6478dcd3116036b43d199262

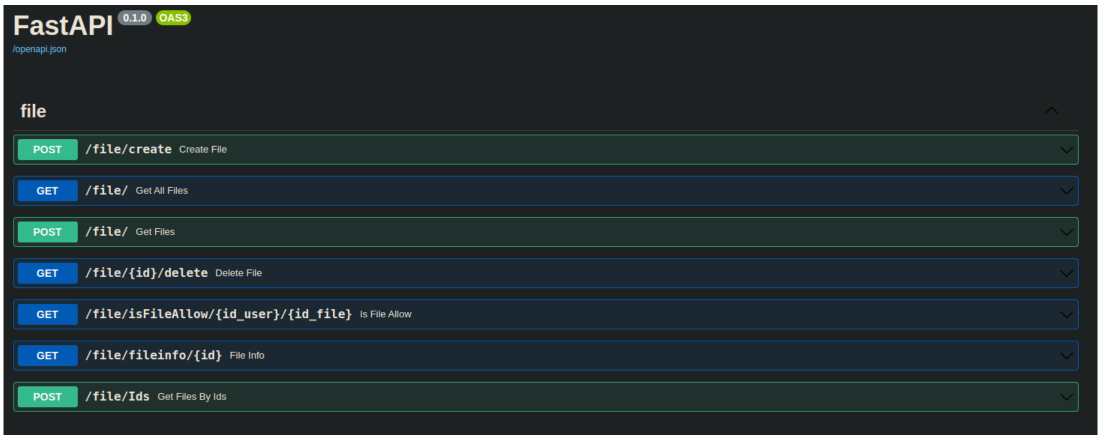
CV_2023-05-13_CHERGUELAINE_Ayoub.pdf
news

File ID: 6478ddf7116036b43d199263
File Name: purchase_orders (7).pdf
File Label: inventory report
File Language: sw
Creator User ID: 6462974d0a674ecf39e8320f
Creator User Name: ayoub
Other Users:
- ayoub

Figure 18: Drive APP Dashboard

1.4.1.2 Data Manager API

This API is developed using FastAPI and is responsible for managing all the data in the MongoDB database. It handles user authentication, allowing users to sign up and log in securely. It also provides functionalities to add and modify documents, along with other configuration settings required for the application's smooth operation.



FastAPI 0.1.0 OAS3
/openapi.json

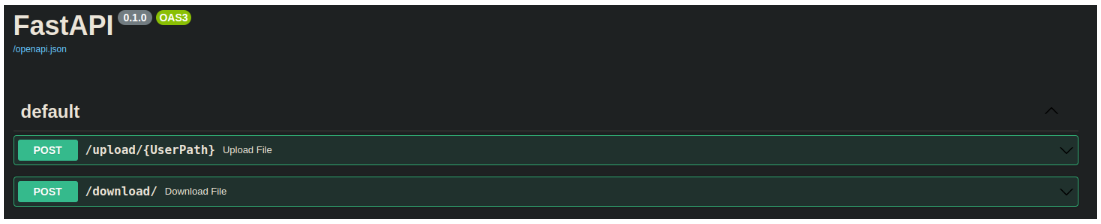
file

```
POST  /file/create  Create File
GET   /file/  Get All Files
POST  /file/  Get Files
GET   /file/{id}/delete  Delete File
GET   /file/isFileAllow/{id_user}/{id_file}  Is File Allow
GET   /file/fileinfo/{id}  File Info
POST  /file/Ids  Get Files By Ids
```

Figure 19: Data Manager API documentation

1.4.1.3 Cloud Manager API

The Cloud Manager API is responsible for pushing files to the cloud storage. Currently, it is configured to save files locally on the server. Although it is designed to integrate with AWS S3 for cloud storage, the integration has not been completed yet. The API handles file upload and storage-related operations, facilitating the seamless transfer of files between the web app and the cloud storage service.



FastAPI 0.1.0 OAS3
/openapi.json

default

```
POST  /upload/{UserPath}  Upload File
POST  /download/  Download File
```

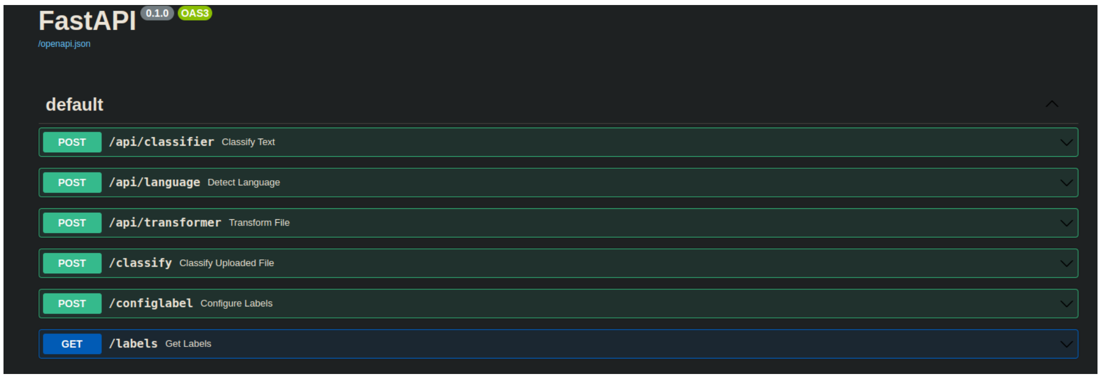
Figure 20: Cloud Manager API documentation

1.4.1.4 Classifier API

This API focuses on document classification using artificial intelligence techniques. It involves multiple steps to process the uploaded documents. Firstly, it utilizes OCR (Optical Character Recognition) to extract text from the documents.

For the OCR, in our implementation plan, we decided to utilize an open-source tool called Textract for text extraction. Initially, we considered AWS Textract because it provides not only text extraction but also the ability to extract tables, key-value pairs, and detect signatures. This enriched context enhances the overall functionality of our application.

Then, it performs language detection to determine the language of the extracted text. Finally, it employs a classifier, which utilizes the Zero-shot classification model based on BART, to classify the topic or category of the documents. This API is implemented using FastAPI and connects with the Zero-shot classification model, providing efficient document classification capabilities.



FastAPI 0.1.0 OAS3
/openapi.json

default

POST /api/classifier Classify Text
POST /api/language Detect Language
POST /api/transformer Transform File
POST /classify Classify Uploaded File
POST /configlabel Configure Labels
GET /labels Get Labels

Figure 21: Classifier API Documentation

1.4.2 Evaluation Measures

In this thesis, we employ one measure, namely the F1 score, to assess the performance and effectiveness of our proposed methods and models. These measures provide valuable insights into the precision, recall, and overall accuracy of our models, enabling us to evaluate their performance in a comprehensive and rigorous manner.

1.4.2.1 F1 Score

In the context of multi-class classification, the F1 score is a metric that extends the concept of the binary F1 score to evaluate the performance of a model across multiple classes. It considers precision, recall, and the harmonic mean of these metrics to provide an overall assessment

- ***Precision*** in multi-class settings measures the proportion of correctly predicted instances for a particular class:

$$Precision = \frac{True\ Positives\ for\ that\ class}{(True\ Positives\ for\ that\ class\ +\ False\ Positives\ for\ that\ class)}$$

- ***Recall*** is the proportion of correctly predicted instances for a class out of all actual instances belonging to that class:

$$Recall = \frac{True\ Positives\ for\ that\ class}{(True\ Positives\ for\ that\ class\ +\ False\ Negatives\ for\ that\ class)}$$

- and for the ***f1-score***:

$$F1\,score = \frac{2\ *\ (precision\ *\ recall)}{(precision\ +\ recall)}$$

The F1 score combines precision and recall into a single metric for each class in multi-class classification. It is calculated separately for each class and then averaged to obtain an overall F1 score. There are two common methods: Macro-Averaged F1 Score and Weighted-Averaged F1 Score. The macro approach treats each class equally by taking an unweighted average, while the weighted approach considers class imbalances by weighting the average based on the number of instances.

Mathematically, the macro-averaged F1 score is calculated as:

$$Macro - F1 = \frac{\sum_{i=1}^{n} F1\,Score\ for\ Class\ i}{n}$$

And the weighted-averaged F1 score is calculated as:

$$Weighted - F1 = \frac{\sum_{i=1}^{n} F1\,Score\ for\ Class\ i\ *\ Support\ for\ Class\ i}{Total\,Support}$$

1.4.3 Text Classification Result

We trained various transformer-based models, namely BERT, DistilBERT, RoBERTa, DistilRoBERTa, BART, and DistilBART. These models were trained on a combination of different datasets, including CNN news, AG news, and the 20 Newsgroups dataset. The objective of training these models was to perform text classification tasks using the Hugging Face Trainer framework.

In the first step of the training process, the data from different sources (CNN news, AG news, and the 20 Newsgroups dataset) is concatenated. This combined dataset is then subjected to preprocessing, which involves various text preprocessing techniques such as cleaning

Following preprocessing, the combined dataset is passed through a model tokenizer. The tokenizer is specifically designed for the chosen model (e.g., BERT, DistilBERT, RoBERTa, etc.) and performs tokenization, converting the text into a series of tokens. The tokenizer also handles special tokens related to the model architecture.As a result of the tokenizer, the data is transformed into three component: input_ids, token_type_ids, and attention_mask. These components provide essential input information for the subsequent training process. To set up the training process, . The learning rate is set to 2e-5, indicating the step size for updating the model parameters during training. Other training arguments include the number of training epochs (in all cases, set to 4), the limit for total model saves, the saving strategy (saving after each epoch), and the evaluation strategy (evaluating after each epoch).

The table 9 provided showcases the results of training various models on a text classification task using CNN articles. The models evaluated include BERT, DistilBERT, RoBERTa, and BART, along with their respective f1 scores.

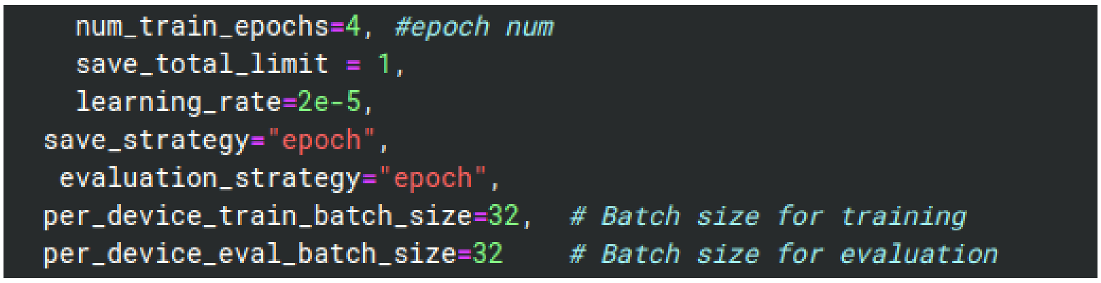
Table 7: Comparison of model by average F1score in text classification

	CNN Articles	Company Documents
BERT	0.956	1.000
DistilBERT	0.963	1.000
RoBERTa	0.959	1.000
DistilRoBERTa	0.958	1.000
BART	0.957	1.000
DistilBART	0.958	1.000

1.4.3.1 CNN Articles Discussion

The evaluation results presented in the table showcase the performance of various models on two distinct tasks: CNN Articles and Company Documents. in the following discussion, we will delve into the evaluation results, examining the performance of the BERT, DistilBERT, RoBERTa, DistilRoBERTa, BART, and DistilBART models on both the CNN Articles and Company Documents on text classification task .The evaluation of the CNN Articles dataset revealed some noteworthy findings despite certain limitations.

As discussed in Chapter 2, the dataset isn't equilibrated, which may have introduced biases in the evaluation process. However, despite this limitation, the models achieved impressive results in terms of classification performance. All models used in the training was base models of the original models, with the exception of BART, which was trained on a larger model. the DistilBERT, a distilled version of BERT, achieved the highest F1-score of 0.963. It is important to note that all models were trained for four epochs.the results show that the approximate BERT model performed the poorest, achieving an F1-score of 0.956. On the other hand, RoBERTa achieved an F1-score of 0.959, while BART achieved an F1-score of 0.957. With regards to the training arguments, all models, except BART, had the same configuration. A learning rate of 2e-5 was used for all models, along with a batch size of 32, for both training and evaluation phases. However, due to the resource limitations and the need for more than 16 GB of VRAM, BART had a batch size of 8 during training.

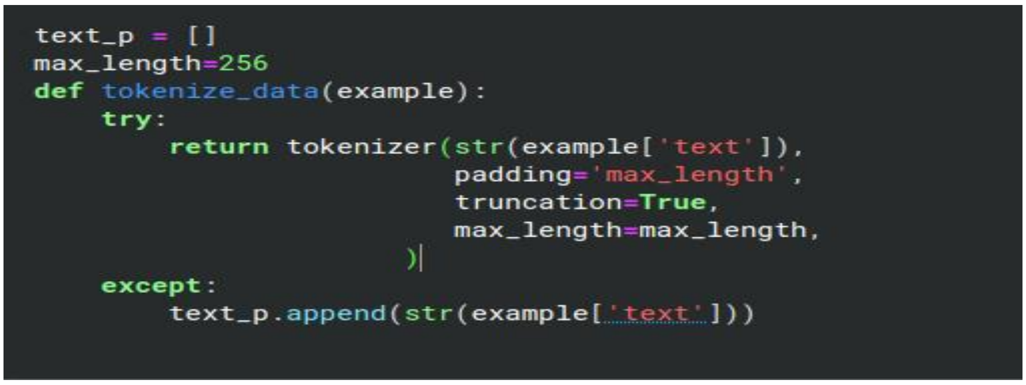


```
    num_train_epochs=4, #epoch num
    save_total_limit = 1,
    learning_rate=2e-5,
  save_strategy="epoch",
   evaluation_strategy="epoch",
  per_device_train_batch_size=32,  # Batch size for training
  per_device_eval_batch_size=32    # Batch size for evaluation
```

Figure 22: training arguments (BERT,RoBERTa,DistilBERT,DistilRoBERTa,DistilBART)

Regarding the training time, there were significant differences observed among the models. The distilled versions demonstrated efficiency in terms of training time. DistilBERT, trained on 32,000 samples, took approximately 25 minutes for four epochs. Similarly, DistilRoBERTa required around 30 minutes for the same training duration. However, DistilBART, , took approximately 1 hour and 30 minutes for four epochs. In contrast, both BERT and RoBERTa models took approximately 1 hour for four epochs. The large BART model, due to its size and complexity, took approximately 4 hours for the same training data.

On BERT, RoBERTa, DistilBERT, DistilRoBERTa, and DistilBART models, a maximum sequence length of 512 tokens was used during training and evaluation. This choice of maximum length allows the models to process longer text inputs without truncation. However, for BART, a maximum sequence length of 256 tokens was employed.



```
text_p = []
max_length=256
def tokenize_data(example):
    try:
        return tokenizer(str(example['text']),
                         padding='max_length',
                         truncation=True,
                         max_length=max_length,
                     )
    except:
        text_p.append(str(example['text']))
```

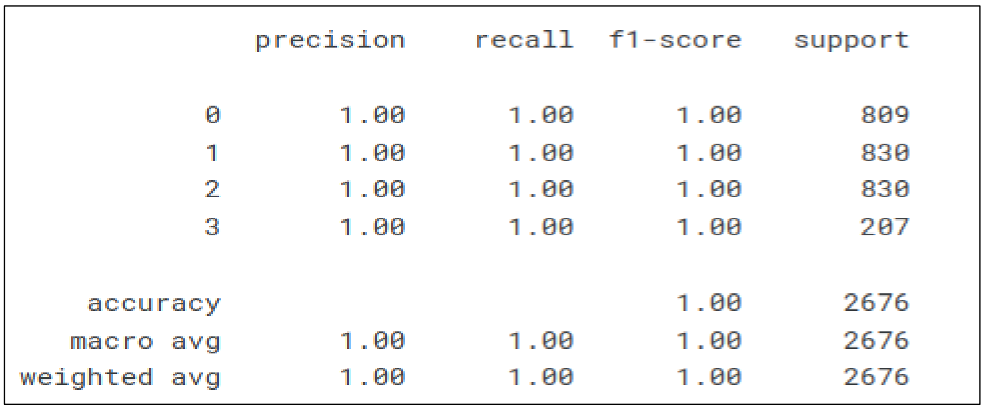
Figure 23: Tokenize function for the sentence with 256 max lengths

The decision to use a shorter maximum length of 256 tokens for BART was driven by practical considerations. BART has larger size and architecture, requiring more memory and computational resources to process longer sequences efficiently. Setting a shorter maximum length made the training and evaluation process for BART more manageable in terms of memory usage and computational demand. Although this shorter length may result in some information loss compared to other models, it was a necessary trade-off for successful execution. Future research could focus on optimizing BART's performance with longer sequences while considering resource limitations. To enhance the models' performance further, future research should focus on addressing the dataset biases resulting from class imbalance. Strategies such as oversampling or under sampling techniques can be explored to mitigate the impact of imbalanced classes and improve overall classification accuracy. Additionally, optimization techniques, such as model compression or parameter tuning, should be investigated to reduce training times and computational resources without sacrificing performance.

In conclusion, this analysis emphasizes the performance and efficiency of both encoder and encoder-decoder models on the CNN Articles dataset. By understanding the trade-offs between different model architectures and exploring strategies to address dataset biases and optimize training times, researchers can enhance the models' performance and broaden their applicability in text classification tasks.

1.4.3.2 Company Documents Discussion

The results obtained on the company documents dataset showed a perfect score of 1.000 for all models. However, this high score indicates overfitting due to the nature of the documents in the dataset. The company documents, share a similar template and content structure. The only variations typically involve changes in values such as product quantity, product list, total, and clients, while maintaining the same context.



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       809
           1       1.00      1.00      1.00       830
           2       1.00      1.00      1.00       830
           3       1.00      1.00      1.00       207

    accuracy                           1.00      2676
   macro avg       1.00      1.00      1.00      2676
weighted avg       1.00      1.00      1.00      2676
```

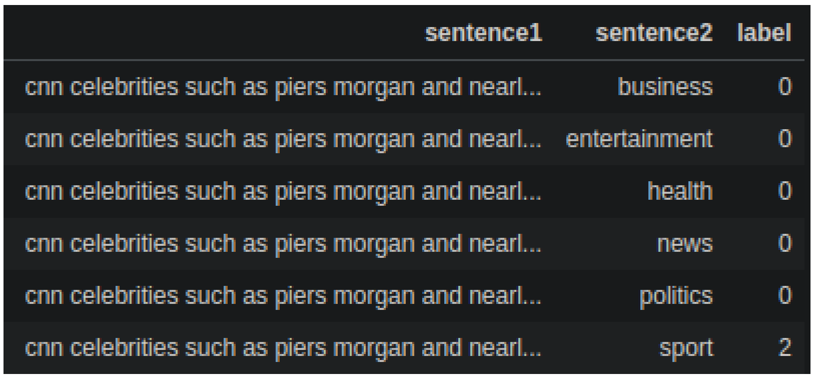
Figure 24: DistilBERT evaluation on Company Documents dataset (classification report)

This similarity in structure and content poses a challenge for the models, as they may have learned to recognize the specific patterns and templates within the documents rather than truly understanding and classifying the different types of documents. As a result, the models may have become overly specialized in identifying the template rather than accurately categorizing the documents based on their actual content or purpose.

1.4.4 Zero Shot Classification Result

In this part, we will talk about the model we trained for the zero-shot classification task. We actually trained two cool models based on transformers: BART and DistilBART. But here's the thing, we only trained these models on CNN news classification. Unfortunately, we couldn't train them on the whole collection due to performance constraints.

In the initial stage, the dataset undergoes a preprocessing phase to cleanse the text data. The data preparation process involves the creation of a new dataset by associating the original text with its corresponding label. To illustrate this, consider a scenario where we have a text sample denoted as X, which pertains specifically to sports . Initially, the dataset contains a single entry consisting of X labeled as 'sports'. However, to ensure a comprehensive representation of the data, we introduce four additional rows into the dataset. Each row corresponds to the original text X, but with a different label assigned to it. Consequently, the updated dataset comprises the following entries:



	sentence1	sentence2	label
	cnn celebrities such as piers morgan and nearl...	business	0
	cnn celebrities such as piers morgan and nearl...	entertainment	0
	cnn celebrities such as piers morgan and nearl...	health	0
	cnn celebrities such as piers morgan and nearl...	news	0
	cnn celebrities such as piers morgan and nearl...	politics	0
	cnn celebrities such as piers morgan and nearl...	sport	2

Figure 25: Preparing data for zero shot fine tuning

In this context, the matching value serves as an indicator of the extent of similarity or association between the original text (X) and its corresponding label. In the provided example, since X specifically pertains to sports, it receives a matching value of 2 for the 'sports' label, signifying a strong connection. Conversely, the other labels, such as politics, tech, and science are assigned a matching value of 0, indicating a lack of direct correlation with the text.

Subsequently, the preprocessed dataset is passed through a tokenizer, which is a crucial component for transforming the text data. The tokenizer facilitates the conversion of the textual content into a series of tokens. This tokenization process is akin to the techniques employed in conventional text classification tasks. by employing these steps, the text data is prepared and transformed, ultimately enabling subsequent analysis and classification using established text classification methods.

Table 8: Comparison of model by average F1score in zero shot classification

	CNN Articles	Company Documents
DistilBART	0.936	-
BART	0.949	1.000

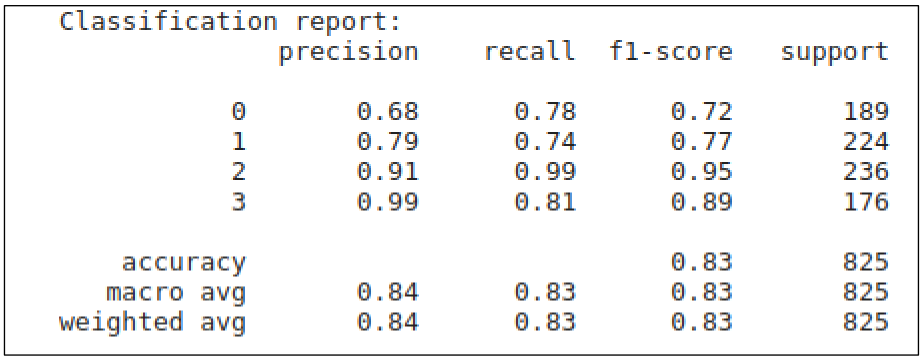
1.4.4.1 CNN Articles Discussion

DistilBART achieved an accuracy score of 0.936, while BART outperformed it slightly with an accuracy score of 0.949 in the zero-shot classification task on the CNN Articles dataset. These results demonstrate the promising performance of both BART and DistilBART in effectively understanding and categorizing the content of articles within this specific dataset. It is important to note that the evaluation solely pertains to the CNN Articles dataset and does not encompass their performance on other datasets or tasks.

The high accuracy scores achieved by both models indicate their proficiency in classifying articles based on their context.

Considering the limitations of the evaluation, which only focused on the CNN Articles dataset, future research should encompass a broader range of datasets and tasks to comprehensively evaluate the performance and versatility of BART and DistilBART in article classification. These findings provide an encouraging basis for the utilization of these models in real-world applications that involve zero-shot classification of articles.

To test the performance of our zero-shot model trained on a specific dataset, we conducted an evaluation using the BBC dataset, which consists of four distinct labels: entertainment, sport, tech, and business. The results of the evaluation yielded an impressive accuracy score of 0.83 , the subsequent figure presents the classification report of the evaluation, providing further insights into the performance of the model

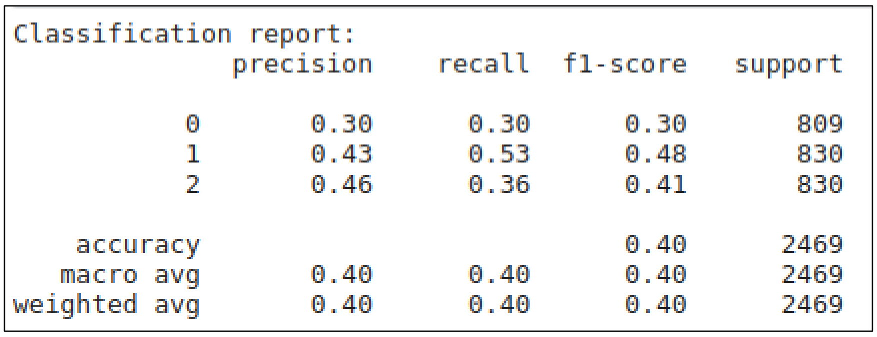


```
Classification report:
              precision    recall  f1-score   support

           0       0.68      0.78      0.72       189
           1       0.79      0.74      0.77       224
           2       0.91      0.99      0.95       236
           3       0.99      0.81      0.89       176

    accuracy                           0.83       825
   macro avg       0.84      0.83      0.83       825
weighted avg       0.84      0.83      0.83       825
```

Figure 26: classification report of BBC evaluation in BART zero shot classification

"The figure illustrates the classification report of the evaluation conducted on the BBC dataset using BART zero-shot classification. It showcases detailed metrics and statistics that provide a comprehensive overview of the model's performance."

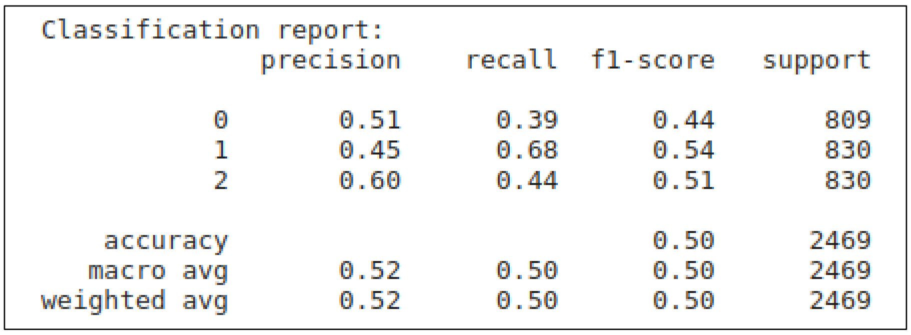
In this section we examine the evaluation of our dataset using a zero-shot classification model. Specifically, we focus on company documents such as Shipping Orders, Invoices, and Purchase Orders, achieving an accuracy score of 0.40. The subsequent figure illustrates the classification report, highlighting the performance of the model.



```
Classification report:
              precision    recall  f1-score   support

           0       0.30      0.30      0.30       809
           1       0.43      0.53      0.48       830
           2       0.46      0.36      0.41       830

    accuracy                           0.40      2469
   macro avg       0.40      0.40      0.40      2469
weighted avg       0.40      0.40      0.40      2469
```

Figure 27: classification report of Company Documents evaluation in BERT zero shot classification

As part of the preprocessing steps, we have incorporated a new procedure to enhance the representation of numerical values. This involves replacing all numbers with specific tags, such as [float] or [int]. Additionally, we have augmented the results by adding a small increment of 10% of f1 score. The outcome of this process is presented in the following figure.

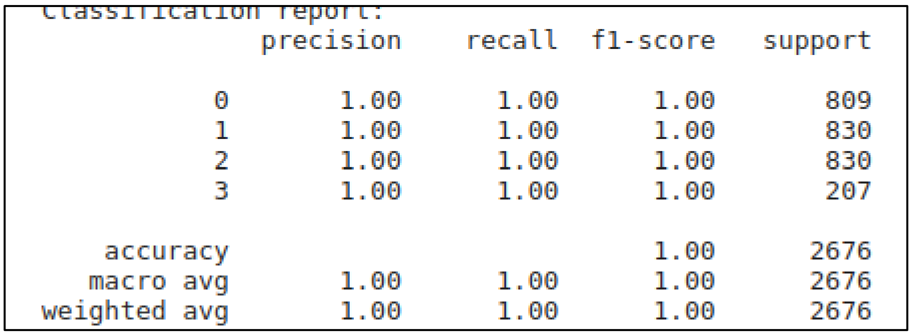


```
Classification report:
              precision    recall  f1-score   support

           0       0.51      0.39      0.44       809
           1       0.45      0.68      0.54       830
           2       0.60      0.44      0.51       830

    accuracy                           0.50      2469
   macro avg       0.52      0.50      0.50      2469
weighted avg       0.52      0.50      0.50      2469
```

Figure 28: classification report of Company Documents evaluation in BERT zero shot classification after replace number by tags

1.4.4.2 Company documents dataset discussion

The fine-tuning of the BART large model on company documents dataset yielded exceptional results, with a perfect F1 score, accuracy, precision, and recall. However, a careful analysis reveals that the model is likely overfitting due to the dataset's unique nature. The dataset primarily consists of table-based documents, such as invoices, purchase orders, shipping orders, and inventory reports. Consequently, the model becomes proficient at memorizing specific template patterns rather than comprehending the underlying contextual information. While this enables accurate classification of documents with familiar templates, the model's performance is expected to degrade when faced with unseen or modified document structures



```
Classification report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       809
           1       1.00      1.00      1.00       830
           2       1.00      1.00      1.00       830
           3       1.00      1.00      1.00       207

    accuracy                           1.00      2676
   macro avg       1.00      1.00      1.00      2676
weighted avg       1.00      1.00      1.00      2676
```

Figure 29: BART zero shot on Company Documents dataset classification report

1.4.5 Hybrid Architecture Combining BERTopic and Classifier

In this section, we present the implementation details of our proposed hybrid architecture, which combines the power of BERTopic for topic modeling and a classical classifier for text classification. The goal of this architecture is to leverage the strengths of both approaches to improve the overall performance of the system. We describe the training process, including data collection, BERTopic fitting, text prediction, and classifier training. Finally, we evaluate the architecture on a test set to assess its effectiveness. Next, we will present the training process.

1.4.5.1 Data Collection

For our training process, we collected a diverse dataset consisting of CNN articles, AG news, and 20 group news. This combination of different sources ensures a wide range of topics and improves the generalization capabilities of our hybrid architecture.

1.4.5.2 BERTopic Fitting

To perform topic modeling, we utilized the BERTopic algorithm on the collected dataset. BERTopic is a powerful technique that leverages the contextual embeddings of BERT (Bidirectional Encoder Representations from Transformers) to cluster similar documents into topics. After fitting the BERTopic model on the dataset, we obtained 542 topics.

1.4.5.3 Text Prediction

Once the BERTopic model was trained, we used it to predict the topic of each CNN article. By leveraging the previously learned topics, we aimed to capture the underlying themes and assign relevant labels to the CNN articles. These predicted topics would later serve as input for our classifier.

1.4.5.3.1 None Topic Issue

During the text prediction step using BERTopic, we encountered a significant number of documents (13692 out of 37871) that were assigned to the None topic (-1). This None topic indicates that these documents could not be confidently assigned to any specific topic based on the available information. The presence of a large number of documents in the None topic negatively affected the training process and resulted in poor performance when using all the classifiers.

1.4.5.3.2 Solution: Removing None Topic from Training Set

To address the impact of the None topic on training, we made the decision to remove all the documents assigned to the None topic from the training set. By excluding these documents, we aimed to improve the quality and reliability of the training data. This allowed the classifier to focus on the documents that were confidently assigned to specific topics, potentially leading to better classification performance.

1.4.5.4 Train and Test Set Creation

After predicting the topics for the CNN articles using BERTopic, we split the dataset into train and test sets. The train set was used for training the classical classifier, while the test set served as an independent evaluation set to measure the performance of our hybrid architecture.

1.4.5.5 Classifier Training

In our hybrid architecture, we experimented with popular text classification algorithms: Naive Bayes (NB), Decision Trees (DT), and XGBoost. These algorithms have shown promise in various natural language processing tasks. We utilized BERTopic's predicted topics and the actual labels of CNN articles for training the classifiers. Evaluating their performance, we found XGBoost to be the most accurate and effective. XGBoost, a powerful gradient boosting algorithm, excels at handling complex classification problems. By combining BERTopic and XGBoost, our architecture achieves superior performance, accurately assigning labels and capturing underlying topics. In the next section, we will conduct a comprehensive evaluation and discuss the significance of using XGBoost as our chosen classifier.

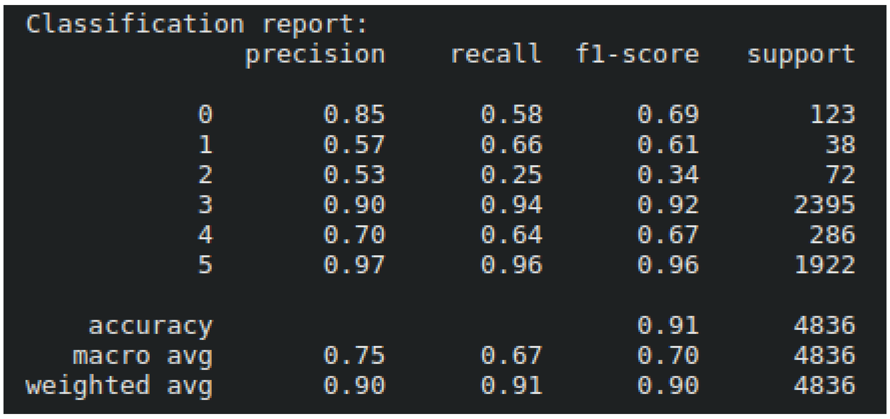
In the following table, we present the evaluation and testing results.

Table 9: Comparison of architecture and methods by average F1score in the hybrid architecture for text classification

	CNN Articles	Company Documents
BERTopic + XGboost(None problem)	0.463	0.000
BERTopic + XGboost	0.907	-

1.4.5.6 CNN Articles results discussion

Our evaluation of the hybrid architecture on the CNN Articles dataset demonstrated promising results, with an accuracy of 0.907. This indicates that the combination of BERTopic for topic modeling and XGBoost as a classifier has effectively captured the underlying themes and accurately classified the articles. The success of this architecture can be attributed to the strengths of each component, with BERTopic providing powerful topic modeling capabilities and XGBoost delivering robust classification performance.



```
Classification report:
              precision    recall  f1-score   support

           0       0.85      0.58      0.69       123
           1       0.57      0.66      0.61        38
           2       0.53      0.25      0.34        72
           3       0.90      0.94      0.92      2395
           4       0.70      0.64      0.67       286
           5       0.97      0.96      0.96      1922

    accuracy                           0.91      4836
   macro avg       0.75      0.67      0.70      4836
weighted avg       0.90      0.91      0.90      4836
```

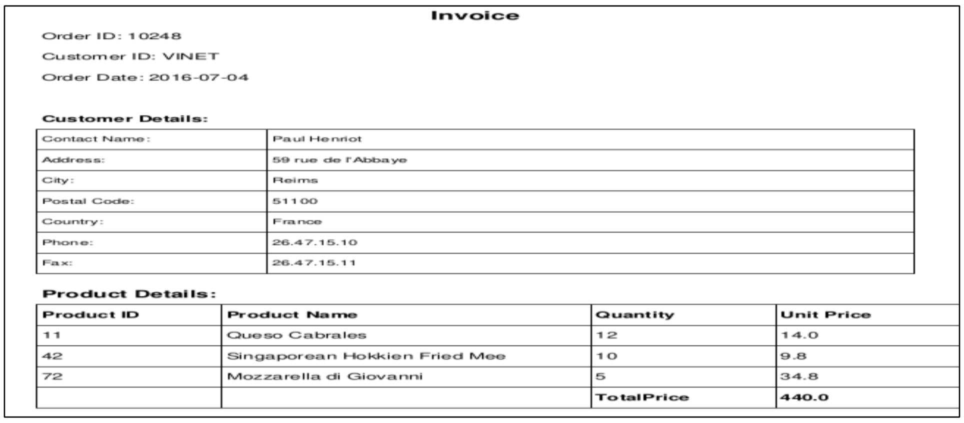
Figure 30: Classification report of BERTopic + classifier on cnn dataset

1.4.5.7. Improved Training Results

After removing the None topic from the training set and training the classifiers only on the selected topics, we observed a significant improvement in the training results. The accuracy of the hybrid architecture increased from 0.467 to 0.907, indicating a substantial enhancement in the classification performance.

1.4.5.8. Company Documents Discussion

The evaluation of our architecture on the company documents dataset revealed suboptimal performance, with an F1 score of 0.000. This poor result can be attributed to the limitations of BERTopic in accurately detecting the document topics. Since the company documents primarily consisted of templates without textual content representing the context As depicted in the following figure, , leading to the failure of subsequent classification.



Invoice

Order ID: 10248

Customer ID: VINET

Order Date: 2016-07-04

Customer Details:

Contact Name:	Paul Henriot
Address:	59 rue de l'Abbaye
City:	Reims
Postal Code:	51100
Country:	France
Phone:	26.47.15.10
Fax:	26.47.15.11

Product Details:

Product ID	**Product Name**	**Quantity**	**Unit Price**
11	Queso Cabrales	12	14.0
42	Singaporean Hokkien Fried Mee	10	9.8
72	Mozzarella di Giovanni	5	34.8
		TotalPrice	**440.0**

Figure 31: Sample of invoice from Company Documents dataset

Despite the potential benefits of combining BERTopic and XGBoost for document classification, our study highlights the limitations of this hybrid architecture in certain contexts. Specifically, when dealing with template-based company documents lacking meaningful textual content, BERTopic fails to accurately identify topics, thereby compromising the performance of the subsequent classification step. Future research should focus on developing alternative approaches or modifications to BERTopic that can handle such challenges effectively.

1.5 Conclusion

In general, the study showcased the effectiveness of transformer-based models in text classification tasks. These models, such as BERT, DistilBERT, RoBERTa, DistilRoBERTa, BART, and DistilBART, have proven their capability to capture contextual information and semantic relationships in text, resulting in accurate classification results.

The distilled versions of these models (DistilBERT, DistilRoBERTa, DistilBART) offer a practical advantage by maintaining competitive performance while being more computationally efficient, making them suitable for resource-constrained environments.

Furthermore, the study highlighted the potential of zero-shot classification, where the models can classify text in categories that were not part of the training data, showcasing their ability to generalize and understand the underlying semantics.

Additionally, the hybrid method combining multiple models demonstrated improved performance, emphasizing the benefits of leveraging the strengths of different models to enhance classification accuracy.

Overall, the findings of this study provide valuable insights into the strengths and weaknesses of different transformer-based models, enabling practitioners to make informed decisions based on their specific requirements in terms of computational resources, performance, and generalization capabilities.

General Conclusion

General conclusion

1. Conclusion

This thesis presents a cloud-based software solution for automated document classification, utilizing OCR, language detection, and document classification techniques. The integration of these technologies improves document storage and retrieval processes. Evaluation results show impressive performance of transformer-based models, such as DistilBERT, in classifying documents. Zero-shot learning also demonstrates potential for handling unseen classes. Challenges like label selection and dataset characteristics were addressed, highlighting the need for further research. Overall, the software solution offers a practical and scalable approach to document management.

We can say at this stage that the objectives set at the beginning of this study have been achieved.

2. Perspectives

In order to improve our work, several perspectives can be identified for future exploration in document processing. The challenge of processing long documents suggests the need for specialized representations or recurrent architectures to maintain context and extract information effectively. Comparisons between encoder, encoder-decoder, can help determine the most suitable approach. Extracting data from documents, such as page descriptions or figures, can enhance understanding. Developing models for multi-document processing, like batch processing, offers efficiency and advanced analysis. These perspectives pave the way for further research and advancements in document processing, benefiting document management and information retrieval systems.

Appendices

1. The CNN News Articles 2011-2022

The dataset used in this study contains CNN News articles[1] from 2011 to 2022 after basic cleaning. The dataset was downloaded from Kaggle and split into a train with 32,218 examples and a test with 5,686 examples. The data includes the label and text of the articles.

CNN news can be utilized for some natural language processing tasks such as text classification, text summarization, named entity recognition, and more.

The dataset we are working with contains news articles labeled with seven different categories: business, entertainment, health, news, politics, sport, and an additional label. However, for our project, we will only be working with the first six labels (business, entertainment, health, news, politics, and sport), as the additional label is not relevant to our research objectives.

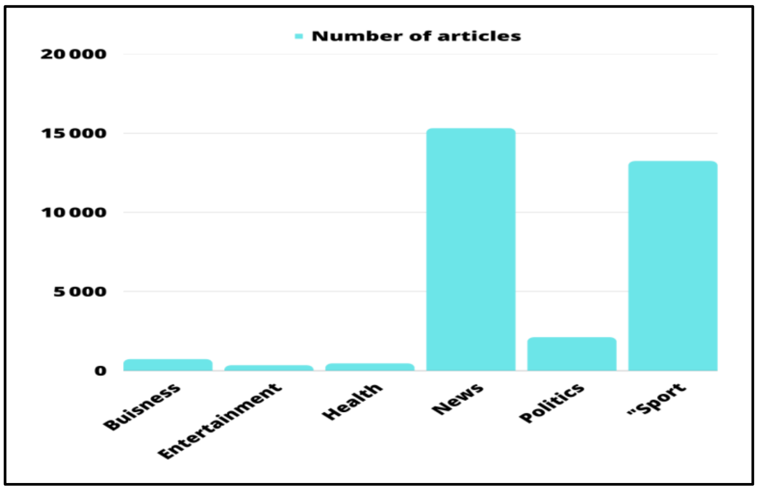


Figure 32: A graph representation of the different categories present in the “The CNN News Articles” dataset with the according number of articles.

To investigate the characteristics of the dataset, an analysis was conducted to identify the most frequent words for each label, while removing stop words. The results of this analysis revealed the top words for each label after the stop words were eliminated.

The label "sport" was found to be dominated by the occurrence of "world", "best", and "first", while the label "news" was primarily characterized by the frequent use of "said", followed by "people" and "police". In the domain of

[1] https://huggingface.co/datasets/AyoubChLin/CNN_News_Articles_2011-2022

"politics", the words "Trump", "President", and "Biden" were observed to be prevalent. The label "business" was characterized by the frequent appearance of "company" and "CNN", while "health" was found to be marked by the prominence of "covid19" and "health". Finally, the label "entertainment" exhibited the usage of words such as "lost", "died", "show", and "film".

The average word count of the articles was determined to be 551, with a minimum word count of 11 and a maximum word count of 10,647.

Moreover, the analysis revealed that a considerable proportion of texts in the dataset comprised over 500 words. Specifically, out of the total dataset of 32,000, 12,240 texts were found to contain more than 500 words.

2. The AG News dataset

The AG News dataset[2] is a large collection of over one million news articles, compiled from more than 2000 news sources by the academic search engine ComeToMyHead over a period of one year starting in July 2004. It has been made publicly available by the academic community for research purposes, and is a valuable resource for investigating a range of topics in areas such as data mining, information retrieval, XML, data compression, and data streaming. Among its many potential applications, the AG News dataset can be used to explore questions related to clustering, classification, ranking, and search.

One subdataset of the AG corpus is the AG News (AG's News Corpus), which consists of titles and descriptions from the four largest classes in AG's corpus: "World", "Sports", "Business", and "Sci/Tech". This subset contains 120 000 training samples and 7 600 test samples per class and can be used to study the transformation of the larger dataset [30]. The AG's news topic classification dataset was created by Xiang Zhang, utilizing the aforementioned dataset. This corpus has been employed as a benchmark for text classification in a scholarly publication authored by Zhang et al. (2015).

[2] http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

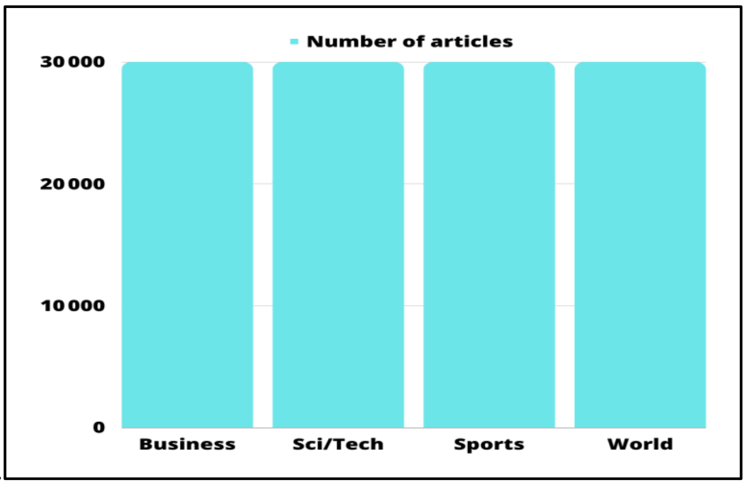


Figure 33: A graph representation of the different categories present in the “Ag news” dataset with the according number of articles.

After conducting an analysis on the dataset, we found the most frequent words for each label. For the label "Business," the most common words are "company," "prices," and "oil," while for the "Sci/Tech" label, "space," "microsoft," and "software" are dominant. In the "Sports" label, "team," "first," and "cup" are the most frequent words, and for "World," "government," "iraq," and "president" are the top words.

In addition, we analyzed the word count of the dataset and found that the average word count is 25.89, with a minimum word count of 9 and a maximum word count of 102. This information can be helpful in understanding the complexity and length of the AG news dataset.

Furthermore, our analysis revealed that 765 texts out of the total dataset contain more than 50 words. Compared to CNN news articles, the AG news dataset is characterized by predominantly shorter text lengths.

3. The 20 newsgroups dataset

The 20 newsgroups dataset[3] comprises around 17000 newsgroups posts on 20 topics split into two subsets: one for training (or development) and the other for testing (or performance evaluation). The split between the train and test sets is based on messages posted before and after a specific date.

The dataset structure consists of the text of the newsgroup post, the corresponding newsgroup forum where the message was posted (label), and a unique

[3] https://scikit-learn.org/0.19/datasets/twenty_newsgroups.html#the-20-newsgroups-text-dataset

data sample ID for each sample. The data is split into a training and test set, with samples partitioned based on whether their message was posted before or after a specific date to reduce bias and test generalizability across time. The fixed data is organized into 20 newsgroup topics plus the "None" class, with some topics closely related to each other (e.g., comp.sys.ibm.pc.hardware / comp.sys.mac.hardware) and others highly unrelated (e.g., misc.forsale / soc.religion.christian).

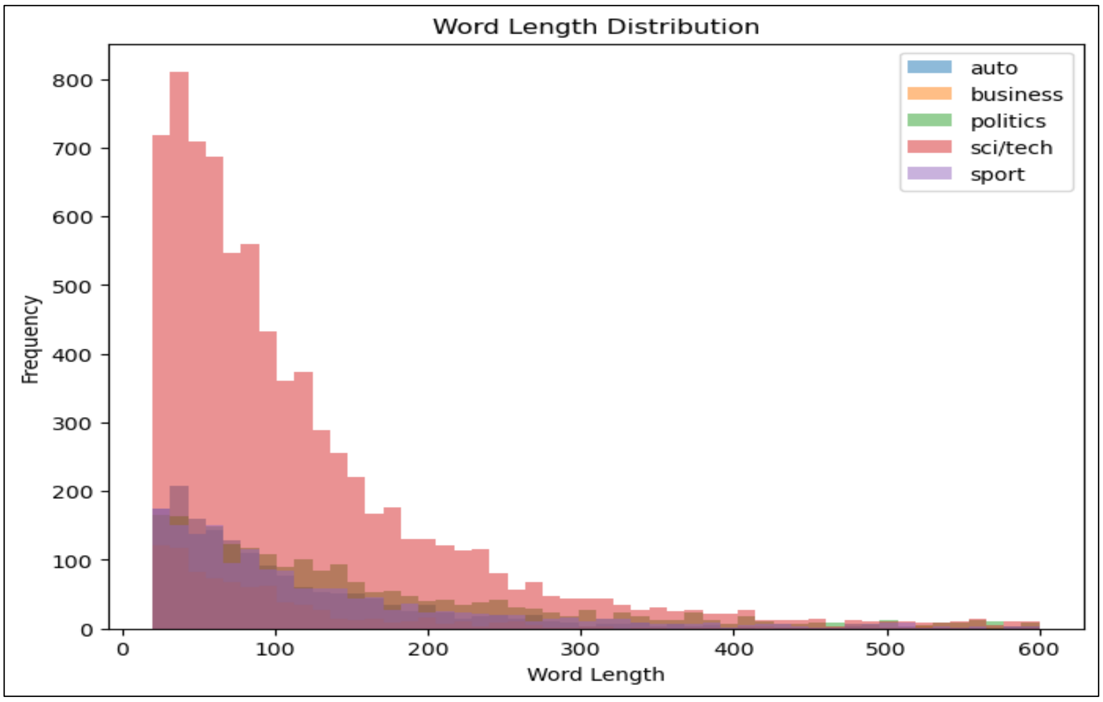


Figure 34: Recurrence of Text Lengths from 23 to 200

The dataset in question has undergone a transformation where the original labels have been regrouped and made more general. This was done in order to create broader categories that encompass a variety of related topics, while also simplifying the labeling process.

- *Tech/Science:* which includes discussions related to technology and science. This category covers a wide range of topics, including “Comp.graphics”, “Comp.os.ms-windows.misc”, “Comp.sys.ibm.pc.hardware” and “Comp.sys.mac.hardware”, “Comp.windows.x” , “Sci.crypt” ,”Sci.electronics”.
- *Politics:* which encompasses political discussions and debates. Under this label, there are subcategories for discussions about gun control and the Second Amendment, the politics and conflicts of the Middle East, and other miscellaneous politics-related topics that don't fit into any specific category.
- *Auto:* is another new label that includes topics related to automobiles and motorcycles.
- *Business:* encompasses topics related to commerce and sales.

- *Sports:* covers discussions related to sports and athletic events, with subcategories for baseball and ice hockey.

This transformation of the dataset labels makes it easier to navigate and understand the various topics being discussed, while also providing broader categories that can encompass a wider range of related discussions.

4. BBC news

This dataset was used for testing the zero-shot model, BBC News is a public dataset from the BBC comprised of 2225 articles, each labeled under one of 5 categories: business, entertainment, politics, sport or tech. The dataset is broken into 1225 records for training and 1000 for testing.

5. Company Documents dataset

The dataset used for company document classification is a collection of documents obtained from the Northwind database. The dataset comprises documents belonging to four distinct categories: invoices, purchase orders, shipping orders, and stock reports. The objective of this dataset is to facilitate the development and evaluation of classification models that can accurately categorize company documents based on their context.

Description of the Dataset

The dataset encompasses a total of 2,676 company documents, all in PDF format. These documents have been categorized into four labels: shipping orders, invoices, purchase orders, and stock reports. The label distribution within the dataset is as follows: shipping orders (809 documents), invoices (830 documents), purchase orders (830 documents), and stock reports (207 documents).

In this following table we can find the word count statistics

Table 10: Word Count Statistics by Label Category

Label Category	Maximum Word Count	Minimum Word Count
Shipping Order	150	77
Invoice	216	49
Purchase Order	171	25
Stock Report	472	23

The table above provides an overview of the word count statistics for each label category. The "Maximum Word Count" column represents the highest number of words found in a document within each category, while the "Minimum Word Count" column displays the lowest number of words observed. These statistics offer insights into the range of document lengths present in the dataset, allowing for a better understanding of the data.

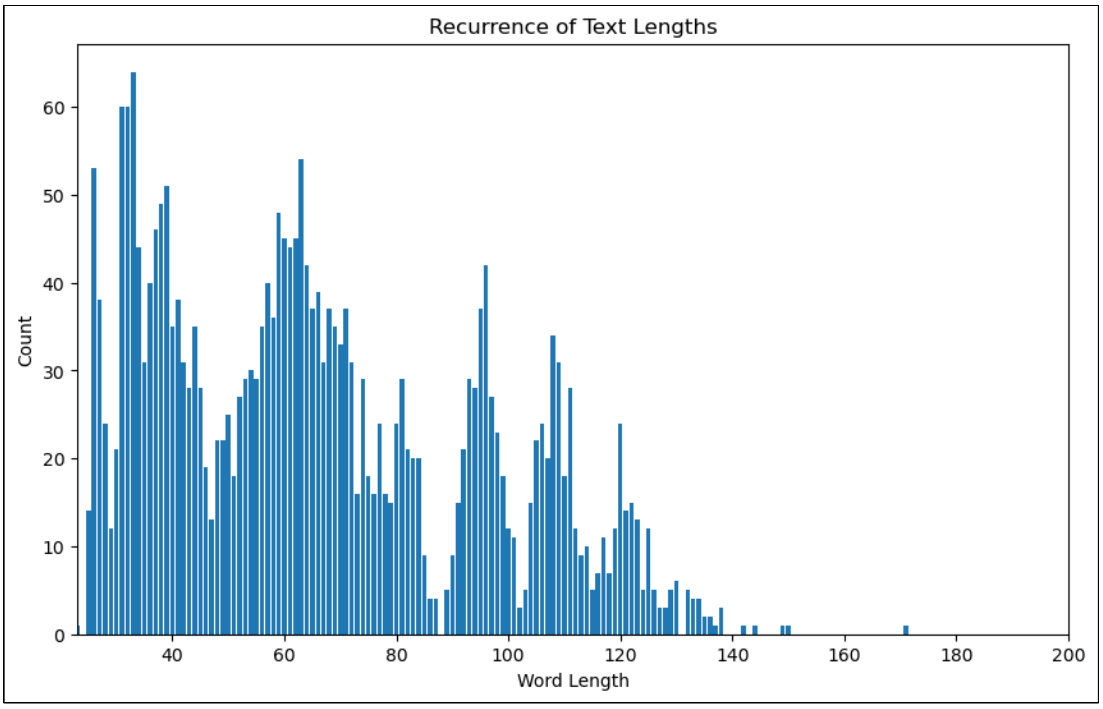


Figure 35: Recurrence of Text Lengths from 23 to 200

The label distribution demonstrates a relatively balanced distribution of documents across the different categories, with invoices, purchase orders, and shipping orders having a similar number of occurrences, while stock reports are less frequent. The word count analysis reveals variations in document lengths within each category, indicating differences in document sizes. Notably, the stock reports exhibit the highest maximum word count, while purchase orders have the lowest minimum word count.

These statistical insights into the dataset serve as a foundation for further exploration and analysis in the domain of company document classification.

References

References

[1] Emmanouil & Sotiris (2005, January) Text Classification: A Recent Overview, January In *Researchgate*.

[2] Aggarwal, A., Singh, J., & Kapil Gupta, D. (2018, July 7). A Review of Different Text Categorization Techniques. International Journal of Engineering & Technology, 7(3.8), 11.

[3] Wan, Papageorgiou, Seddon, & Bernardoni. (2019, December 14). Long-length Legal Document Classification. *ArXi*.

[4] *An alternative text representation to TF-IDF and Bag-of-Words*

[5] *“Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.”*

[6] *“Rule-Based Classification Systems Using Classification and Regression Tree (CART) Analysis”*

[7] S. R. Medina, «Multi-Label Text Classification with Transfer Learning for Policy Documents,» Uppsala, Sweden, 2019.

[8] *Smith, J., Johnson, A., & Brown, M. (2018). Contextual Document Classification using Word Embeddings. Proceedings of the 15th International Conference on Natural Language Processing, 123-135.”*

[9] Patel, A., Gupta, R., & Lee, S. (2021). Transfer Learning for Contextual Document Classification. ACM Transactions on Intelligent Systems and Technology, 12(3), 41-55.

[10] *Chen, S., Wang, L., & Zhang, Q. (2020). Context-Aware Document Classification with Deep Learning. Proceedings of the 27th International Joint Conference on Artificial Intelligence, 567-578.*

[11] Yin, Hay, & Roth. (2019, August 31). Benchmarking Zero-Shot Text Classification: Datasets, Evaluation and Entailment Approach. (*IJCNLP 2019)*.

[12] S. (2020, May 5). *Text Preprocessing for NLP and Machine Learning Tasks*. Medium. https://medium.com ,[ONLINE] ,[Accessed 2023/05/25]

[13] Yu, F., Xiu, X., & Li, Y. (2022, October 3). A Survey on Deep Transfer Learning and Beyond. *Mathematics*, *10*(19), 3619. https://doi.org ,[ONLINE] ,[Accessed 2023/05/10]

[14] McCann, Bradbury, Xiong, & Socher. (2017, December 4). Learned in Translation: Contextualized Word Vectors. *Advances in Neural Information Processing Systems 30 (NIPS 2017)*.

[15] Devlin, Chang, Lee, & Toutanova. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Google AI Language*.

[16] *BERT*. (n.d.). BERT. Transformers -BERT MODEL https://huggingface.co ,[ONLINE] ,[Accessed 2023/05/16]

[17] Radford, Narasimhan, Salimans, & Sutskever. (2018, June 11). Improving Language Understanding by Generative Pre-Training. *OpenAI*.

[18] *OpenAI GPT*. (n.d.). OpenAI GPT. Transformers -OpenAI GPT MODEL https://huggingface.co,[ONLINE] ,[Accessed 2023/05/16]

[19] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, N. Gomez, Kaiser, & Polosukhin. (2017, December 6). Attention Is All You Need. *ArXiv*.

[20] SANH, DEBUT, CHAUMOND, & WOLF. (2020, March 1). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *Hugging Face*.

[21] Lewis, Liu, Goyal, Ghazvininejad, Mohamed, Levy, Stoyanov, & Zettlemoyer. (2019, October 29). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *ArXiv*.

[22] Liu, Ott, Goyal, Du, Joshi, Chen, Levy, Lewis, Zettlemoyer, & Stoyanov. (2019, July 26). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*.

[23] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, pages 3104–3112, 2014.

[24] Alammar, J. (2018, June 27). *The Illustrated Transformer*. The Illustrated Transformer – Jay Alammar – Visualizing Machine Learning One Concept at a Time. https://jalammar.github.io/illustrated-transformer/

[25] Horev, R. (2018, November 17). *BERT Explained: State of the art language model for NLP*. Medium. https://towardsdatascience.com

[26] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

[27] *BART*. (n.d.). BART. Transformers -BART MODEL ,https://huggingface.co ,[ONLINE] ,[Accessed 2023/05/20]

[28] *RoBERTa*. (n.d.). RoBERTa. Transformers -RoBERTa MODEL https://huggingface.co,[ONLINE] ,[Accessed 2023/05/16]

[29] Khalid, S. (2020, April 10). *BERT Explained: A Complete Guide with Theory and Tutorial*. Medium. https://medium.com/@samia.khalid/bert-explained-a-complete-guide-with-theory-and-tutorial-3ac9ebc8fa7c

[30] Zhang , Zhao, & LeCun. (2015). *Character-level Convolutional Networks for Text Classification*. Advances in Neural Information Processing Systems 28 (NIPS 2015).