Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Service Quality AI-Based System

July 2023

Ain Shams University
Faculty of Computer & Information Sciences
Computer Science Department

Service Quality AI-Based System
:

By:

Amr Ayman Essawi	Computer Science
Ahmed Samir Elsayed	Computer Science
Ziad Elsayed Abdelhamed	Computer Science
Yousef Mohamed Eid	Computer Science
Ali Fathy Ali	Computer Science
Moustafa Gamal Abdelsanea	Computer Science

Under supervision of:
Dr. Ahmed Salah,
Computer Science Department,
Faculty of Computer and Information Sciences,
Ain Shams University

T.A. Moataz Mohamed,
Computer Science Department,
Faculty of Computer and Information Sciences,
Ain Shams University

Acknowledgement

We would like to use this opportunity to express our gratitude to everyone who supported this project.

We are thankful for their indispensable guidance, invaluably constructive criticism, friendly advice, and the most obliged provision of their genuine and illuminating views.

We would like to offer our special thanks to our supervisor Dr. Ahmed Salah for his help and support.

We would also like to extend our thanks to T.A. Moataz Mohamed for guiding and helping us throughout the whole course of the project; his help was invaluable to us.

Abstract

The concept of smart government has been gaining momentum in recent years, as governments around the world strive to improve service delivery and enhance citizen engagement with technology.

In this context, the Smart Government Service Quality Project represents an innovative and ambitious initiative aimed at improving the quality of government services and enhancing citizen satisfaction.

Our project has applied the target desired from smart government service quality idea as we have deployed many AI NLP-based techniques as topic classification, sentiment analysis, text generation and information retrieval, these techniques allowed us to analyze reviews, complains and feedbacks of citizens in order to get desired outputs automatically without need of human participating, these outputs includes: the ministry responsible for the review, the sentiment of the citizen posting the review, the specific governmental sector responsible for the problem in the review and the recommended solution for the problem in the review.

We have deployed a website to make it easy for every citizen to use it from his house, work, or any place with the internet.

We have applied fine tuning for transformer models like MARBERT [1] , Arabert [2], Arat5 [4] , MT5 [3]on our downstream tasks, we have reached **0.96** f1 score in sentiment analysis task, **0.957** f1 score in topic classification task and **7.8** BLEU score in text generation task.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	What the abbreviation stands for	
AI	Artificial Intelligence	
ARABERT	Arabic Bidirectional Encoder Representations from Transformers	
ARAGPT2	Arabic Generative Pre-Trained Transformer 2	
AUB	American University of Beirut	
BELU	Bilingual Evaluation Understudy	
BERT	Bidirectional Encoder Representations from Transformers	
DNNs	Deep Neural Networks	
GPUs	Graphics Processing Units	
GPT	Generative Pre-Trained Transformer	
GPT-2	Generative Pre-Trained Transformer 2	
GOOGLE/MT5	Multilingual Text-to-Text Transfer Transformer by Google	
IR	Information Retrieval	
LSTM	Long Short-Term Memory	
MVC	Model-View-Controller	
MT5	Multilingual Text-to-Text Transfer Transformer	
NLP	Natural Language Processing	
ORM	Object-Relational Mapping	
RNN	Recurrent Neural Network	
ROBERT	A Robustly Optimized BERT Pretraining Approach	
SGD	Stochastic Gradient Descent	
SVMs	Support Vector Machines	
TF-IDF	Term Frequency-Inverse Document Frequency	
TLU	Threshold Logic Unit	
TPUs	Tensor Processing Units	
UBC/NLP	Group at the University of British Columbia	

1- Introduction

The Smart Government Service Quality Project is an innovative initiative aimed at improving the quality of government services through the use of advanced technologies.

The project leverages data analytics, artificial intelligence, and other smart technologies to optimize government service delivery processes, reduce bureaucracy, enhance transparency, and increase citizen satisfaction.

The project is based on a customer-centric approach that puts citizen needs and preferences at the center of service delivery. Through this approach, the project seeks to transform the way governments deliver services, making them more efficient, effective, and responsive to the needs of citizens.

1.1 Motivation

The purpose of creating an Operating e-services application is to modify and enhance the connection between the government and the people.
So, we are motivated to do such an application to help the people to communicate easily with the responsible men working in the government organizations and transform their opinions complains and even their gratefulness to avoid miss understanding and the feelings of being forgotten so we are enthusiastic about achieving this aims to help the government and the people to make a real progress in this field.

Improving Mutual benefit between citizens and government organizations by the vastly improved flow of the information from citizens to governments, governments to citizens, and within government itself.
Significantly improves communication between citizens and governments. Reduce Bureaucracy during the operations.

Instead of the lots of paperwork operations the citizen or the government employee does it can be with a few clicks on the system to reduce a lot of wasted time.

1.2 Problem Definition

The Smart Government Service Quality Project aims to address several challenges that currently exist in government service delivery. These challenges include long wait times, inefficient service delivery processes, lack of transparency and accountability, and inadequate citizen feedback mechanisms. These issues can lead to low citizen satisfaction with government services and a lack of trust in government institutions. The project seeks to overcome these challenges by leveraging data analytics, artificial intelligence, and other smart technologies to optimize service delivery processes, enhance transparency and accountability, and improve citizen satisfaction.

1.3 Objective

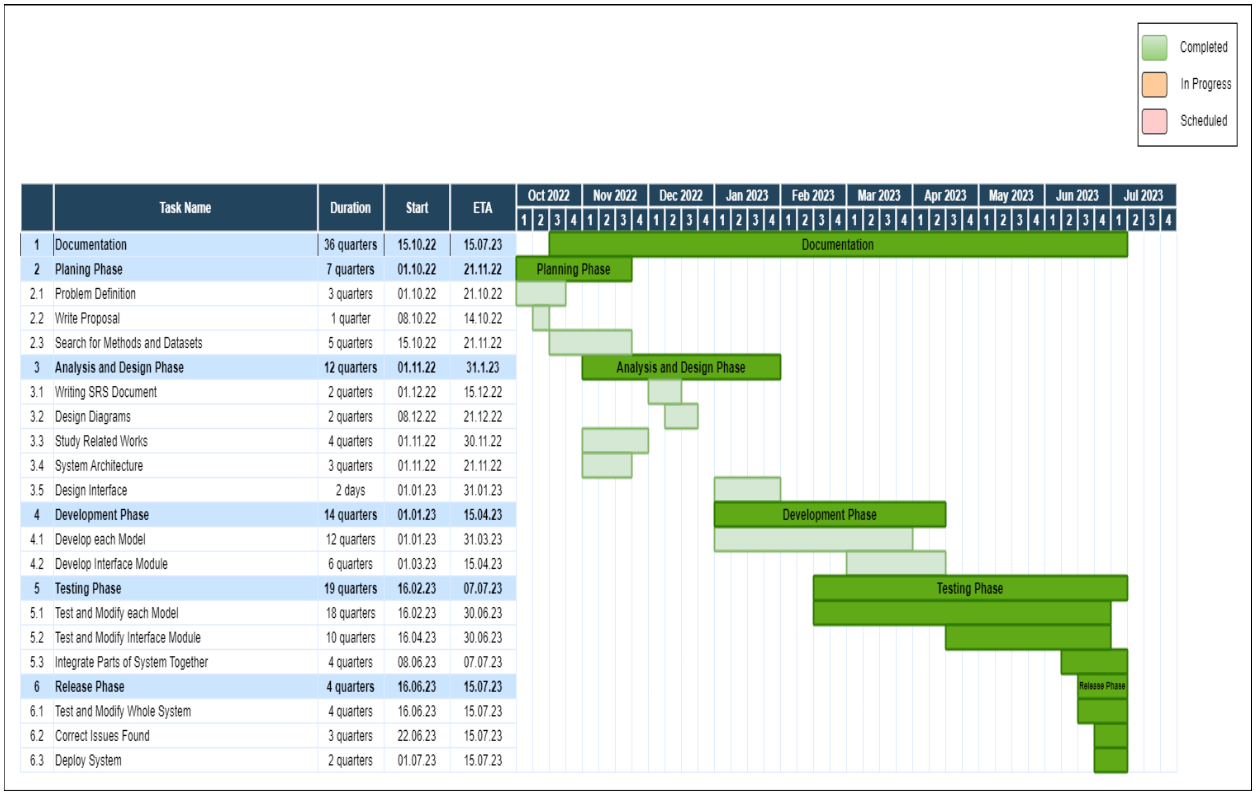
- Topics classification:
  Analyzing the review text and predicting its topic(ministry) to know which ministry to send the review.

- Sentiment analysis:
  Classifying the text into negative, positive, and neutral to make it easy to know the good and bad opinions.

- Generate Recommended Solutions:
  After the text is classified and we know its target sector, if the problem or opinion is like the known one before it will recommend actions to the problem.

- Classifying the Governmental Sector:
  Classify the review text to its specified governmental sector to make it easy to solve the problem in the review in its specified sector.

1.4 Time Plan



	Task Name	Duration	Start	ETA
1	Documentation	36 quarters	15.10.22	15.07.23
2	Planing Phase	7 quarters	01.10.22	21.11.22
2.1	Problem Definition	3 quarters	01.10.22	21.10.22
2.2	Write Proposal	1 quarter	08.10.22	14.10.22
2.3	Search for Methods and Datasets	5 quarters	15.10.22	21.11.22
3	Analysis and Design Phase	12 quarters	01.11.22	31.1.23
3.1	Writing SRS Document	2 quarters	01.12.22	15.12.22
3.2	Design Diagrams	2 quarters	08.12.22	21.12.22
3.3	Study Related Works	4 quarters	01.11.22	30.11.22
3.4	System Architecture	3 quarters	01.11.22	21.11.22
3.5	Design Interface	2 days	01.01.23	31.01.23
4	Development Phase	14 quarters	01.01.23	15.04.23
4.1	Develop each Model	12 quarters	01.01.23	31.03.23
4.2	Develop Interface Module	6 quarters	01.03.23	15.04.23
5	Testing Phase	19 quarters	16.02.23	07.07.23
5.1	Test and Modify each Model	18 quarters	16.02.23	30.06.23
5.2	Test and Modify Interface Module	10 quarters	16.04.23	30.06.23
5.3	Integrate Parts of System Together	4 quarters	08.06.23	07.07.23
6	Release Phase	4 quarters	16.06.23	15.07.23
6.1	Test and Modify Whole System	4 quarters	16.06.23	15.07.23
6.2	Correct Issues Found	3 quarters	22.06.23	15.07.23
6.3	Deploy System	2 quarters	01.07.23	15.07.23

Figure 1 Gantt Chart

These are the days distribution of tasks as shown in Figure 1

1. 49 days for planning phase.
2. 84 days for analysis and design phase.
3. 98 days for Development and implementation Phase.
4. 133 days for Testing phase , while working in parallel with development.
5. 21 days for Release phase.
6. Working on the documentation in parallel with the whole project

1.5 Document Organization

- **chapter 2: Background**
  This chapter includes background about the project, basic concepts, and the related work according to our research.

- **chapter 3: Analysis & Design**
  This chapter includes an overview of the whole system along with the intended user, system architecture, analysis, and design.

- **chapter 4: Dataset**
  This chapter includes an overview of the dataset used in the system.

- **chapter 5: Implementation and Testing**
  This chapter describes the system functions with details about the implementation of the project's modules.

- **chapter 6: User Manual**
  This chapter talks about the needed packages and libraries that must be installed before using the application and shows the user how to use it**.**

- **chapter 7: Conclusion & Future work**
  This chapter includes the conclusion and results of our work and the future work that may be done on the project to improve its performance.

- **References**
  This chapter includes all the papers and resources that we used and studied from through our work.

2- Background

2.1 Neural Networks

Neural networks are a class of machine learning models inspired by the functioning of the human brain. They consist of interconnected nodes, or artificial neurons, organized into layers that process and transform input data.
Each node in a neural network performs a weighted sum of its inputs, applies an activation function, and passes the result to the next layer.

Neural networks can learn and adapt through a process called training, where they adjust the weights of their connections based on observed data. They excel at tasks such as pattern recognition, image and speech recognition, natural language processing, and predictive modeling.

We can see the architecture of the simplest neural network in Figure 2.

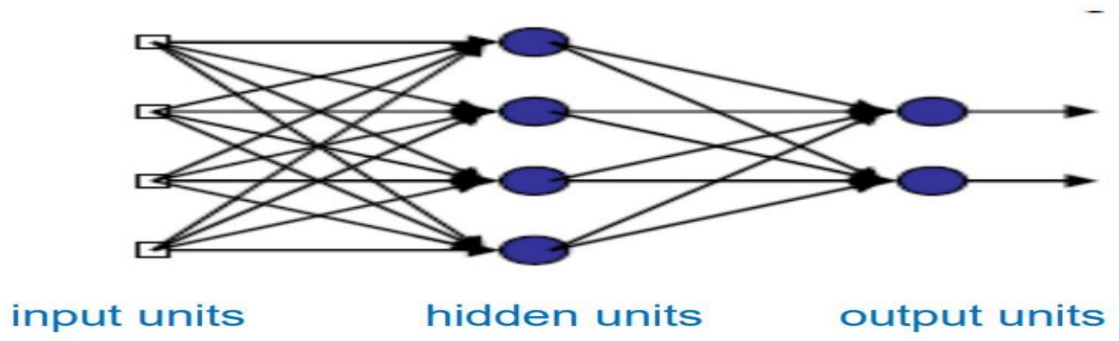


Figure 2 Simple Neural Network Diagram

Neural networks require enormous amounts of labeled training data to generalize well and make accurate predictions.
They are computationally intensive models and often benefit from being trained on powerful hardware, such as GPUs or TPUs.

Neural networks have been successfully applied in various fields, including computer vision, natural language processing, robotics, and healthcare.

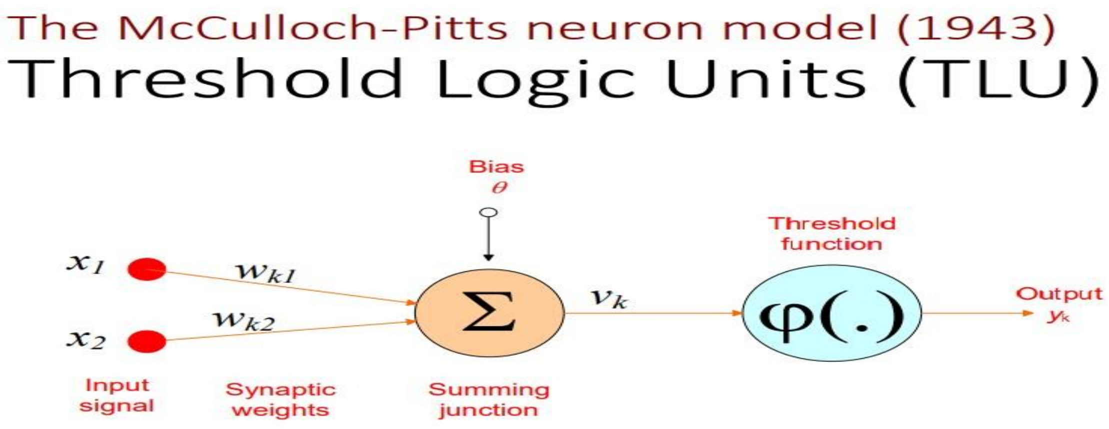


Figure 3 The McCulloch-Pitts neuron model

In Figure 3 we can see the architecture of the McCulloch-Pitts neuron model.

Despite their effectiveness, neural networks can be prone to overfitting, where they memorize the training data and fail to generalize to new examples.

Ongoing research in neural networks focuses on improving model interpretability, reducing training time, and addressing ethical concerns surrounding bias and fairness.

2.2 Deep Neural Networks

Deep neural networks (DNNs) are a class of machine learning models widely adopted for various tasks, including image and speech recognition, natural language processing, and even game playing.

They are modeled after the structure of the human brain, with layers of interconnected nodes, or neurons, that process and transmit information.

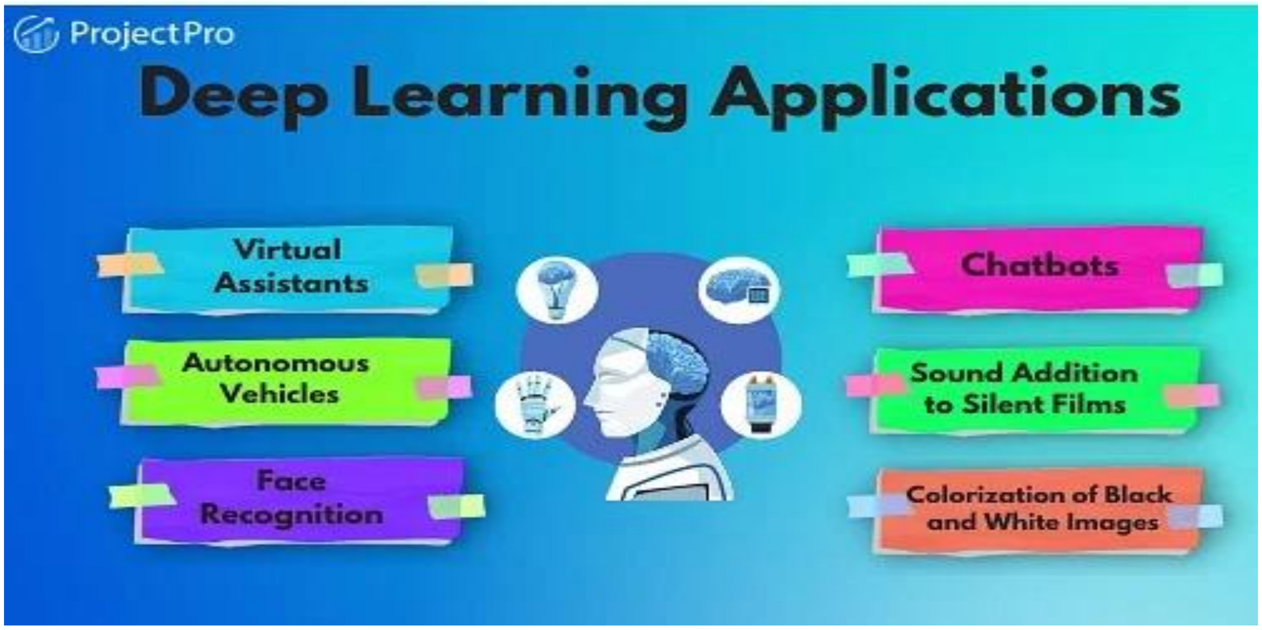

ProjectPro
Deep Learning Applications
Virtual Assistants
Autonomous Vehicles
Face Recognition
Chatbots
Sound Addition to Silent Films
Colorization of Black and White Images

Figure 4 Deep Learning Applications

We can see some of the applications of deep learning in Figure 4.

One of the key advantages of DNNs is their ability to automatically learn features from raw data, without the need for manual feature engineering.

This has led to significant improvements in many areas of artificial intelligence, especially in computer vision and natural language processing.

DNNs are typically trained using large datasets and backpropagation, a method for adjusting the weights of the connections between neurons to minimize the error between the predicted output and the true output.

This training process can be computationally intensive and requires specialized hardware, such as GPUs or TPUs.

One of the challenges of DNNs is the risk of overfitting, where the model becomes too specialized to the training data and performs poorly on new data.
To address this, various regularization techniques, such as dropout and weight decay, are used.

Another challenge is the interpretability of DNNs, as the complex interactions between the neurons can make it difficult to understand how the model is making its predictions.

Recent research has focused on developing methods for interpreting and explaining DNNs, such as feature visualization and attribution techniques.

Despite their challenges, DNNs have achieved remarkable success in many applications, including image and speech recognition, machine translation, and autonomous driving.

As research in this area continues, it is likely that DNNs will continue to play a pivotal role in the development of artificial intelligence.

There are many architectures of deep neural networks:

Recurrent Neural Networks (RNN)

RNNs are a type of neural network that can process sequential data, making them well-suited for NLP tasks such as language modeling, speech recognition, and sentiment analysis.
RNNs have the ability to process temporal information — data that comes in sequences, such as a sentence. Recurrent neural networks are designed for this very purpose.
We can see the architecture diagram of recurrent neural networks in Figure 5

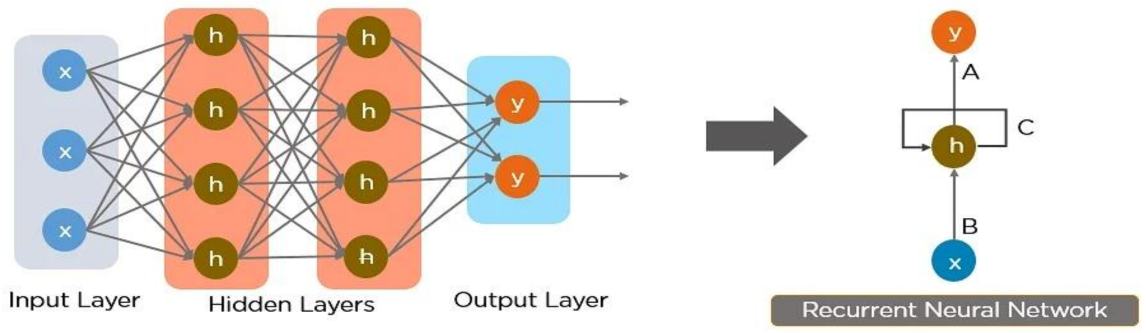


Figure 5 Recurrent Neural Network Architecture

Long Short-Term Memory (LSTM) Networks

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture that is designed to handle the vanishing gradient problem in traditional RNNs.

LSTMs use a memory cell and three gating mechanisms (input, output, and forget) to selectively retain and forget information over time, allowing them to better handle long-term dependencies in sequential data. LSTMs have achieved state-of-the-art performance in a variety of NLP tasks, such as language modeling, machine translation, and sentiment analysis, and have also been applied to other domains, such as speech recognition and music composition.

The architecture diagram of LSTM is shown in Figure 6

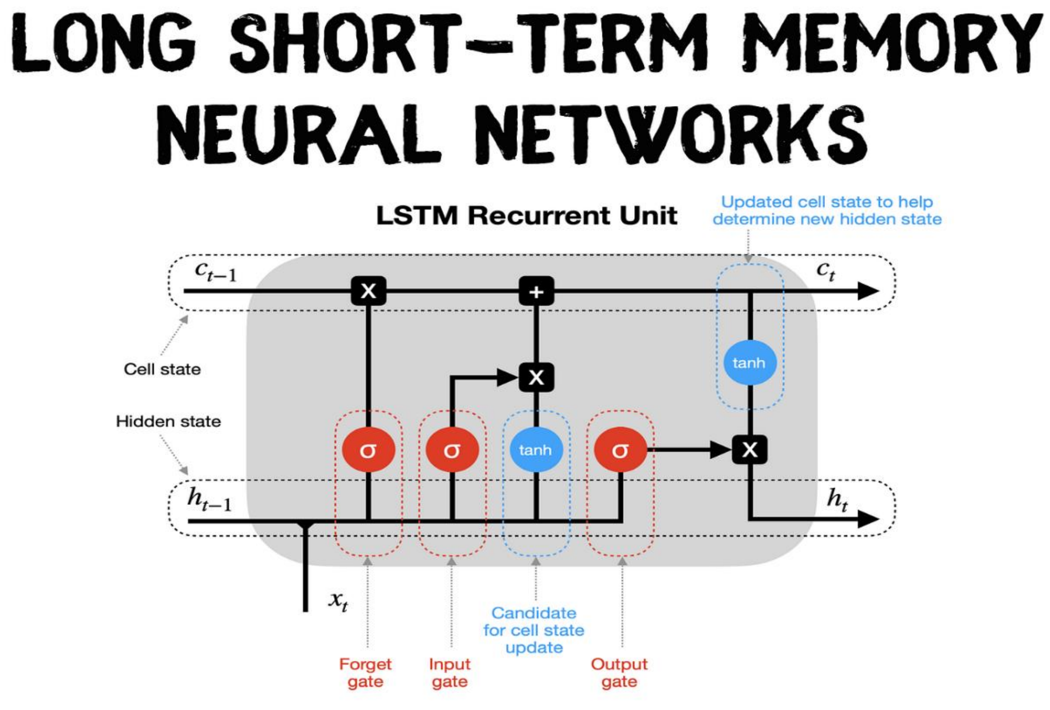


Figure 6 LSTM Architecture

There are many other architectures in deep neural networks but in our models, we were concerned with the previous architectures only.

2.3 Natural Language Processing

Natural Language Processing (NLP) is a field of artificial intelligence concerned with the interaction between computers and human language. It involves the development of algorithms and models that can analyze, understand, and generate natural language text. NLP has many practical applications, including language translation, sentiment analysis, and chatbots.

One of the key challenges in NLP is the ambiguity and variability of natural language, which can make it difficult to extract meaning accurately. To address this, NLP researchers use machine learning techniques, such as deep neural networks, to automatically learn patterns in text data.
NLP models often involve preprocessing the text, such as

- Tokenization
- Stemming
- Lemmatization
- Stop word Removal.

To reduce the complexity of the input. They may also use techniques such as part-of-speech tagging, named entity recognition, and sentiment analysis to extract specific information from the text.

There are many other sectors included with Natural Language Processing as Artificial Intelligence, Human Language and Computer Science, these sectors are combined to help in NLP tasks, they are shown in Figure 7

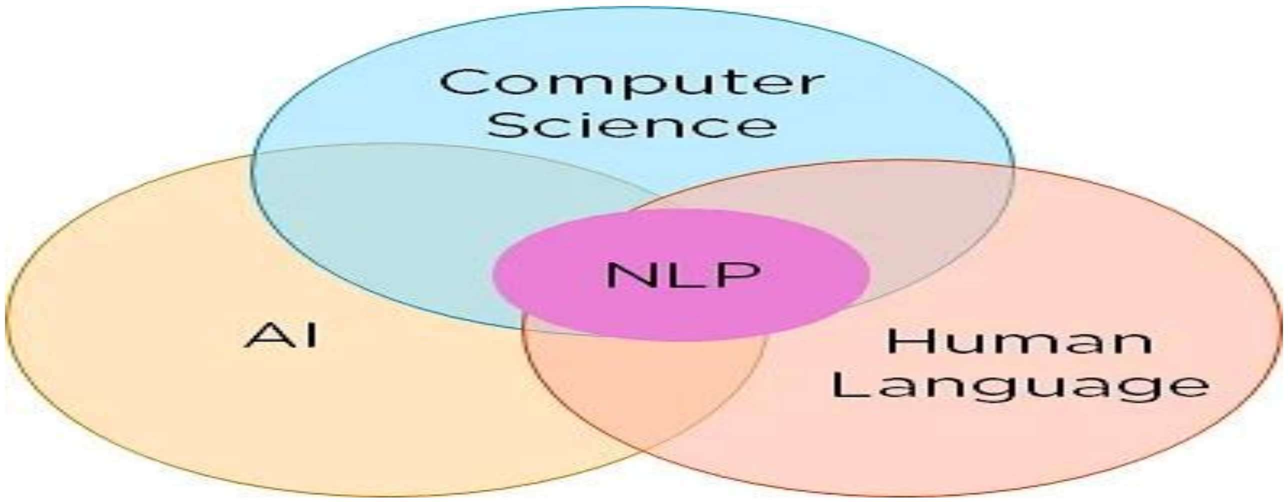


Figure 7 Natural Language Processing Sectors

2.4 Transformers Models

Transformers are a type of deep neural network architecture that has gained widespread popularity in natural language processing (NLP) due to their ability to handle long-range dependencies in text. The transformer architecture was first introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017 and has since become the basis for many state-of-the-art NLP models.

The key innovation of transformer models is the use of self-attention mechanisms, which allow the model to selectively attend to various parts of the input sequence when generating the output. This allows the model to process long sequences more efficiently than traditional recurrent neural networks (RNNs), which can suffer from the vanishing gradient problem when processing long sequences.

We can see the architecture of Encoder-Decoder Transformer models in Figure 8

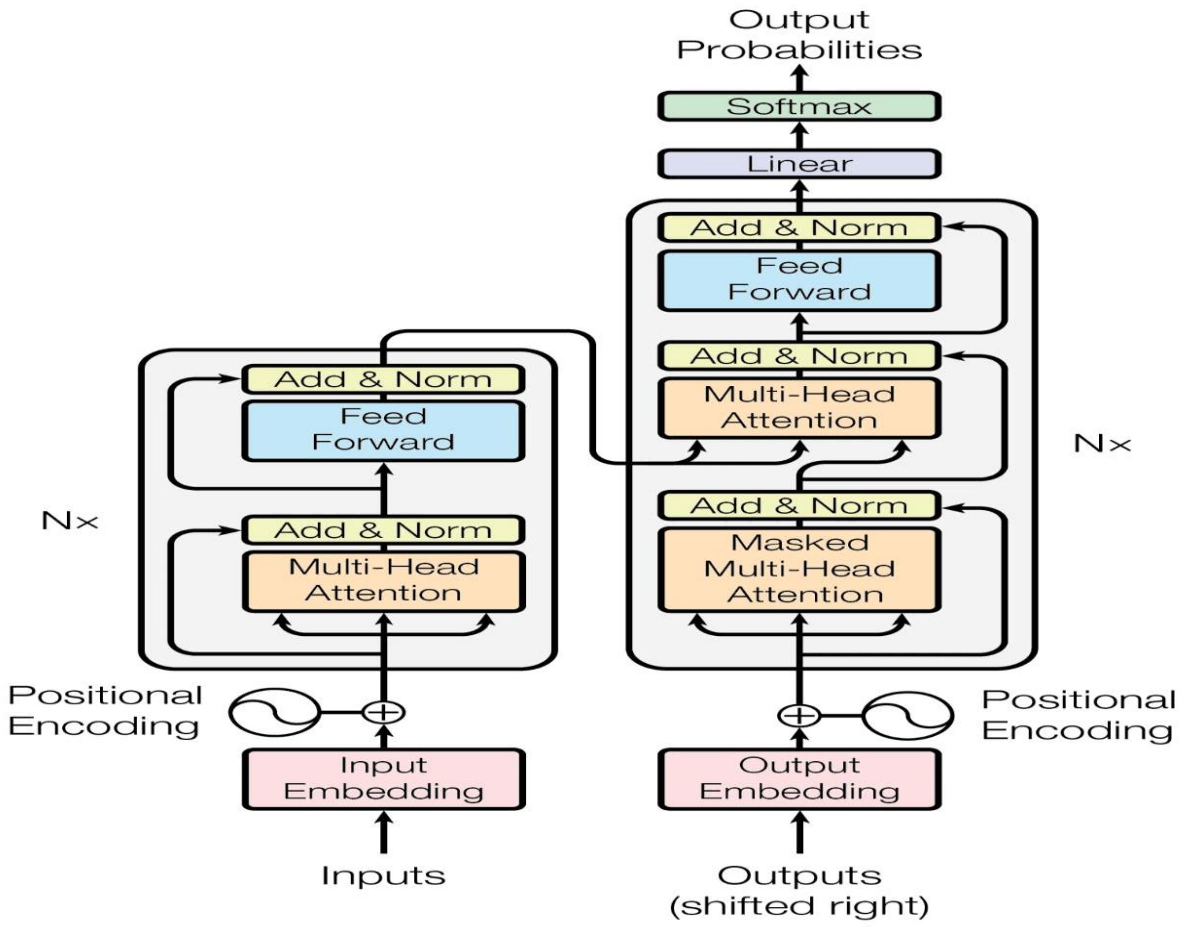


Figure 8 Encoder-Decoder Transformers Architecture

Transformers Architecture Phases:

- **Input Embedding:** The input sequence is first converted into a sequence of embeddings, which are learned representations of the input tokens.
- **Multi-Head Self-Attention:** This component uses self-attention mechanisms to allow the model to attend to different parts of the input sequence when generating the output. This is done through multiple heads, which each attend to different parts of the sequence and are then concatenated.
- **Feedforward Layers:** After the self-attention layer, the output is passed through one or more feedforward layers that apply non-linear transformations to the output.
- **Layer Normalization:** This component normalizes the output of each layer, improving the stability and performance of the model.
- **Encoder and Decoder Stacks:** The transformer architecture includes both an encoder and a decoder, each consisting of multiple layers of the above components.
- **Masking:** In the decoder stack, masking is used to prevent the model from attending to future tokens in the input sequence.
- **Positional Encoding:** Since the transformer architecture does not use recurrent connections, it requires a way to encode the position of each token in the sequence. This is done through the use of positional encodings, which are added to the input embeddings.

Different Architectures of Transformer models:

1. Encoder Architecture

The encoder architecture of Transformers consists of stacked identical layers.
Each layer includes a multi-head self-attention mechanism and a position-wise feed-forward network.
Residual connections are used to facilitate information flow and mitigate the vanishing gradient problem.
Layer normalization is applied after each sub-layer for normalization.
Positional encoding is added to incorporate positional information into the model.

This Architecture is illustrated in Figure 9

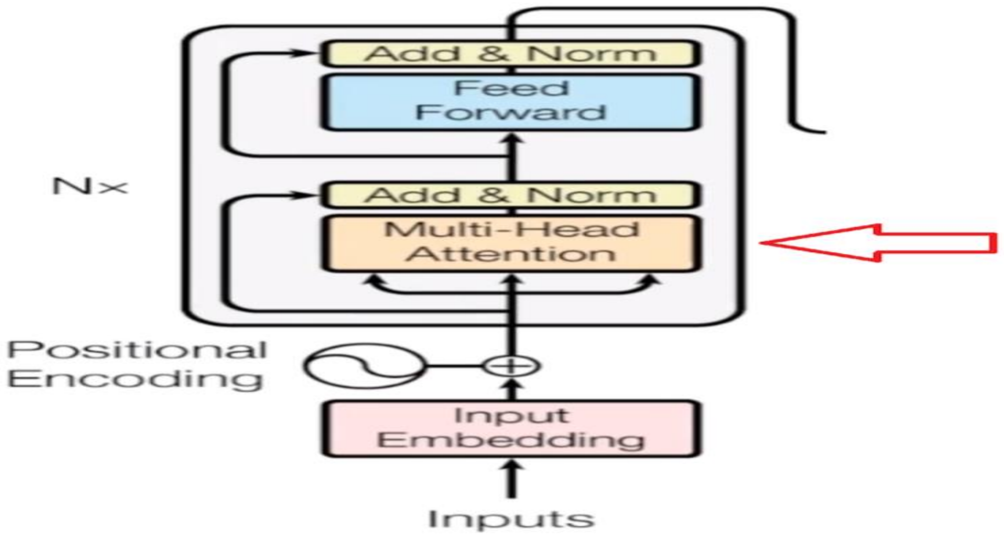


Figure 9 Encoder Transformers Architecture

Encoder Models

- **Bidirectional Encoder Representations from Transformers (BERT):**
  BERT [13] is a pre-trained language model developed by Google that uses a transformer encoder to generate representations of text that can be fine-tuned for a variety of downstream NLP tasks.

2. Decoder Architecture

Decoder transformer models are a type of deep neural network architecture used in natural language processing (NLP) that use self-attention mechanisms to generate an output sequence from an input sequence.

The decoder transformer architecture consists of multiple layers of self-attention and feedforward neural networks and includes masking to prevent the model from attending to future tokens in the input sequence. Overall, decoder transformer models have proven to be highly effective for sequence-to-sequence tasks in NLP and continue to be an active area of research.

This Architecture is illustrated in Figure 10

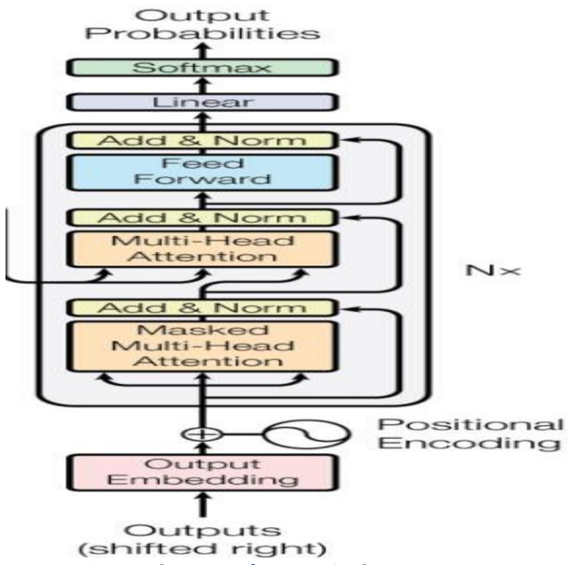


Figure 10 Decoder Transformers Architecture

Decoder Models

- **GPT-2 (Generative Pre-trained Transformer 2):**
  GPT-2 [5] is a transformer-based language model developed by OpenAI that has achieved state-of-the-art performance on a range of language modeling tasks.

3. Encoder Decoder Architecture

The encoder-decoder transformer model architecture is a type of deep neural network used in natural language processing (NLP) that combines the transformer architecture with both an encoder and a decoder component. This architecture is primarily used for sequence-to-sequence tasks, such as machine translation, where the goal is to generate an output sequence that is a translation of the input sequence.

The encoder component of the architecture processes the input sequence and generates a sequence of hidden states, while the decoder component uses these hidden states to generate the output sequence. Both the encoder and decoder components contain multiple layers of self-attention and feedforward neural networks.

The encoder-decoder transformer model architecture also includes masking to prevent the model from attending to future tokens in the input sequence during the decoding process. Additionally, positional encoding is used to encode the position of each token in the input sequence.

Overall, the encoder-decoder transformer model architecture has demonstrated powerful performance on a variety of sequence-to-sequence tasks in NLP and has become a popular choice for many researchers and practitioners in the field. Ongoing research is focused on improving this architecture and developing new applications and use cases for encoder-decoder transformer models.

Encoder Decoder Models Examples:

- **The T5 (Text-to-Text Transfer Transformer):**
  transformer-based language model developed by Google that can be fine-tuned for a wide range of natural language processing (NLP) tasks.

- **MT5 (Multilingual Translation Transformer):**
  MT5 [3] transformer-based language model developed by Google for multilingual machine translation tasks.

2.5 Optimizers & Loss Functions

Optimizers and loss functions are important components of training transformer models in natural language processing (NLP).
Here are some commonly used optimizers and loss functions for transformer models:

Optimizers:

- **Adam:**
  Adam is a popular optimizer used for training neural networks, including transformer models. It uses adaptive learning rates to update the model parameters during training.

- **Adadelta:**
  Adadelta is another optimizer commonly used in transformer models. It is designed to adaptively adjust the learning rate based on the gradients of the parameters.

Loss functions:

- **Cross-entropy loss:**
  Cross-entropy loss is a commonly used loss function in transformer models for classification tasks. It measures the difference between the predicted output and the true label and is often used with SoftMax activation functions.

- **Binary cross-entropy loss:**
  Binary cross-entropy loss is a variant of cross-entropy loss used for binary classification tasks. It measures the difference between the predicted output and the true label and is often used with sigmoid activation functions.

2.6 Topic Classification

Topic classification is a natural language processing (NLP) task that involves categorizing text data into one or more predefined topics or categories.

The goal of topic classification is to automatically assign a label or category to a given piece of text, such as an article or a social media post.

Topic classification has a wide range of applications in NLP, including content filtering, information retrieval, and text recommendation systems. It is typically performed using supervised machine learning algorithms, such as support vector machines (SVMs), neural networks, or decision trees, that are trained on labeled data.

Topic classification can also be performed using transformer models, which have achieved state-of-the-art performance on several NLP tasks, including topic classification. Transformer models, such as BERT [13] and RoBERTa [14] , can be fine-tuned for topic classification tasks by training them on labeled data and using a SoftMax classifier to predict the topic label for a given piece of text.

Transformer models have the advantage of being able to capture contextual information and long-range dependencies in the text data, which can improve the accuracy of the topic classification task. Overall, transformer models have proven to be a powerful tool for topic classification and continue to be an active area of research in NLP.

2.7 Sentiment Analysis

Sentiment analysis is a natural language processing (NLP) task that involves finding the sentiment or emotion expressed in each piece of text.
It aims to analyze and understand the subjective information conveyed by the text, such as whether it expresses a positive, negative, or neutral sentiment.

The process of sentiment analysis involves several steps. Firstly, the text is preprocessed to remove irrelevant information, such as punctuation and stop words.
Then, the text is analyzed using various techniques, such as machine learning algorithms or lexicon-based approaches.

Machine learning models are often used in sentiment analysis to classify text into different sentiment categories.
These models are trained on labeled datasets, where human annotators assign sentiment labels to text samples.

The models learn patterns and features from the labeled data to predict the sentiment of new, unseen text.
Lexicon-based approaches, on the other hand, rely on sentiment lexicons or dictionaries that contain words or phrases associated with specific sentiments.

The sentiment of a text is determined by calculating the presence or frequency of sentiment-bearing words in the text.

2.8 Text Generation

Text generation is a natural language processing (NLP) task that involves creating new text based on a given prompt or context.
It aims to generate coherent and contextually relevant text that resembles human-written content.

Text generation can be approached using different techniques. Rule-based methods involve defining a set of grammar rules and using them to generate text based on the given input. However, these methods often produce rigid and predictable results.

Statistical approaches, such as n-gram models or Hidden Markov Models (HMMs), learn the statistical patterns and relationships between words in a given text corpus. These models generate text by sampling from the learned probability distributions.

More recently, deep learning models, particularly recurrent neural networks (RNNs) and transformer-based architectures like GPT (Generative Pre-trained Transformer), have shown remarkable progress in text generation. These models learn from large amounts of text data and are capable of capturing long-range dependencies, generating coherent and contextually relevant text.

Text generation has numerous applications across various domains. It can be used for creative writing, automated storytelling, chatbots, machine translation, and even code generation. In addition, text generation models can be fine-tuned for specific tasks such as summarization, dialogue generation, or poetry generation

2.9 Information Retrieval

Information retrieval (IR) is a field of study in computer science that deals with the process of retrieving relevant information from large collections of unstructured data, such as documents, web pages, or databases. The goal of information retrieval is to provide users with accurate and meaningful results in response to their queries or information needs.

The process of information retrieval involves several steps. Firstly, the system indexes the documents or data by creating an inverted index, which maps words or phrases to the documents that contain them. This indexing enables efficient and fast retrieval of relevant documents during the search process.

When a user submits a query, the information retrieval system matches the query terms against the indexed documents. Various techniques are used to rank the documents based on their relevance to the query. These techniques often involve analyzing the textual similarity between the query and the documents, considering factors such as term frequency, inverse document frequency, and relevance feedback.

Information retrieval systems can employ different retrieval models, such as Boolean retrieval, vector space models, probabilistic models, or neural network-based models. Each model has its strengths and limitations, and the choice of model depends on the specific requirements and characteristics of the data and the search task.

Information retrieval has a wide range of applications, including web search engines, document retrieval systems, recommendation systems, and question-answering systems. It plays a crucial role in helping users find relevant information efficiently and effectively.

Challenges in information retrieval include handling large-scale datasets, dealing with ambiguity and context-dependent queries, and addressing the problem of information overload. Ongoing research in the field focuses on developing advanced techniques to improve the accuracy, efficiency, and user experience of information retrieval systems, including the incorporation of machine learning and natural language processing approaches.

2.10 Related Work

2.10.1 Topic Classification

There are many approaches for topic classification [10] :

- **Early Approaches to Topic Classification:**

  involved manual categorization and rule-based systems, where experts manually assigned topics to documents or predefined rules were used to classify texts. These methods had limitations in terms of scalability and reliance on domain-specific knowledge.

- **Statistical Methods for Topic Classification:**

  Statistical methods revolutionized topic classification by introducing machine learning algorithms like Naive Bayes, Support Vector Machines (SVM), and decision trees. These methods utilized feature extraction techniques such as bag-of-words representation and n-grams to classify texts into predefined topic categories. Several studies have demonstrated the effectiveness of statistical methods in topic classification, although their performance can be limited by the quality and representativeness of the features used.

- **Neural Network Approaches for Topic Classification:**

  Neural network approaches have emerged as powerful methods for topic classification. Models like recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformer-based architectures capture complex patterns and dependencies in textual data, enabling accurate and robust topic classification. These models have shown significant improvements over traditional methods, achieving state-of-the-art performance on various benchmark datasets. However, their success often relies on the availability of large labeled datasets and computational resources for training and fine-tuning.

- **Transfer Learning and Pretrained Models:**
  Transfer learning has been widely adopted in topic classification, leveraging pretrained models trained on large-scale language tasks. By fine-tuning these models on topic classification data, they exhibit improved performance even with limited labeled data. Pretrained models like BERT [13], GPT , and

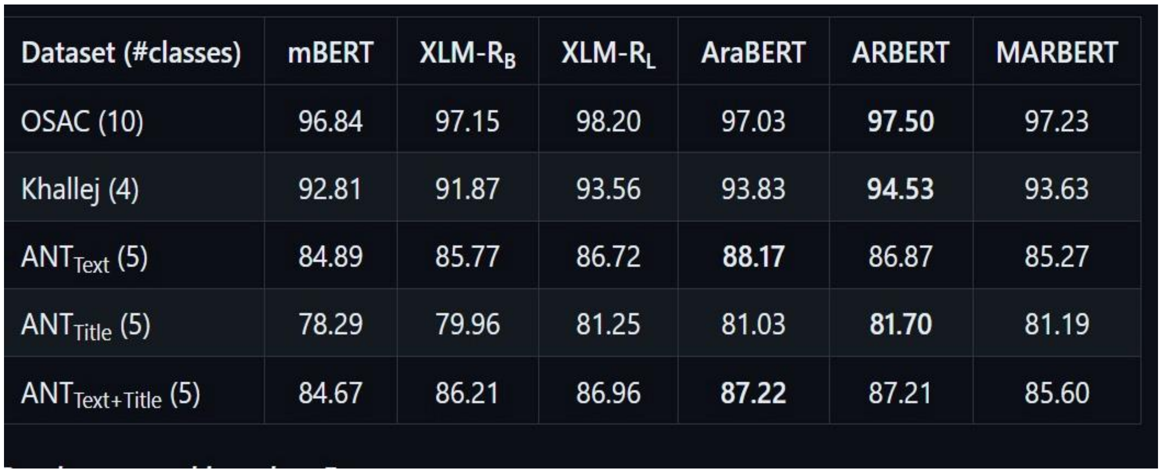
RoBERTa [14] have been applied in topic classification, capturing contextual information and semantic relationships. Transfer learning enables the utilization of preexisting linguistic knowledge, empowering topic classification models to achieve higher accuracy and robustness in various domains and languages.

- **Domain-Specific Topic Classification:**

  in the context of a service quality project in the Arabic language involves addressing unique challenges. These challenges include the scarcity of labeled data in Arabic, the complexity of the language's morphology and syntax, and the need for domain-specific knowledge and vocabulary. To achieve accurate topic classification, researchers have explored techniques such as domain adaptation, linguistic feature engineering, and leveraging pre-existing resources like Arabic lexicons and ontologies. These efforts aim to improve the performance and applicability of topic classification models specifically for the Arabic language in the context of service quality assessment.

- We can say that transformer models have made the best scores and best outputs in this task as shown in Table 1
  [1]

Table 1 f1 scores for different transformer models on topic classification task



Dataset (#classes)	mBERT	XLM-R$_B$	XLM-R$_L$	AraBERT	ARBERT	MARBERT
OSAC (10)	96.84	97.15	98.20	97.03	**97.50**	97.23
Khallej (4)	92.81	91.87	93.56	93.83	**94.53**	93.63
ANT$_{Text}$ (5)	84.89	85.77	86.72	**88.17**	86.87	85.27
ANT$_{Title}$ (5)	78.29	79.96	81.25	81.03	**81.70**	81.19
ANT$_{Text+Title}$ (5)	84.67	86.21	86.96	**87.22**	87.21	85.60

2.10.2 Sentiment Analysis

Sentiment analysis is a natural language processing task that involves identifying and extracting the sentiment or opinion expressed in a piece of text. There are several approaches to sentiment analysis, each with its own strengths and weaknesses [9] .

- One approach is the **lexicon-based approach,** which involves using a pre-defined set of words with assigned sentiment scores to determine the sentiment of a piece of text. This approach is relatively simple and fast, but it may not be accurate in cases where the sentiment of a word is context dependent.

- Another approach is the **machine learning-based approach,** which involves training a model on a labeled dataset to predict the sentiment of a piece of text. This approach can be more accurate than the lexicon-based approach, but it requires a large amount of labeled data and can be computationally expensive.

- A third approach is the **hybrid approach**, which combines both the lexicon-based and machine learning-based approaches. This approach aims to leverage the strengths of both approaches while minimizing their weaknesses. For example, a hybrid approach might use lexicons to identify sentiment-bearing words and then use machine learning to determine the overall sentiment of a sentence based on the context in which those words appear.

- **Transfer learning and transformers Approach** have been successfully applied to sentiment analysis tasks, leading to significant improvements in performance over traditional approaches.

    - One popular approach is to use a pre-trained transformer-based language model, such as BERT or RoBERTa, as a feature extractor for sentiment analysis tasks. The pre-trained model is fine-tuned on a large dataset, such as IMDb or Amazon reviews, to capture the general linguistic patterns and relationships that are relevant to sentiment analysis. Then, the pre-trained model is used to extract features from a smaller labeled dataset for a specific sentiment analysis task.

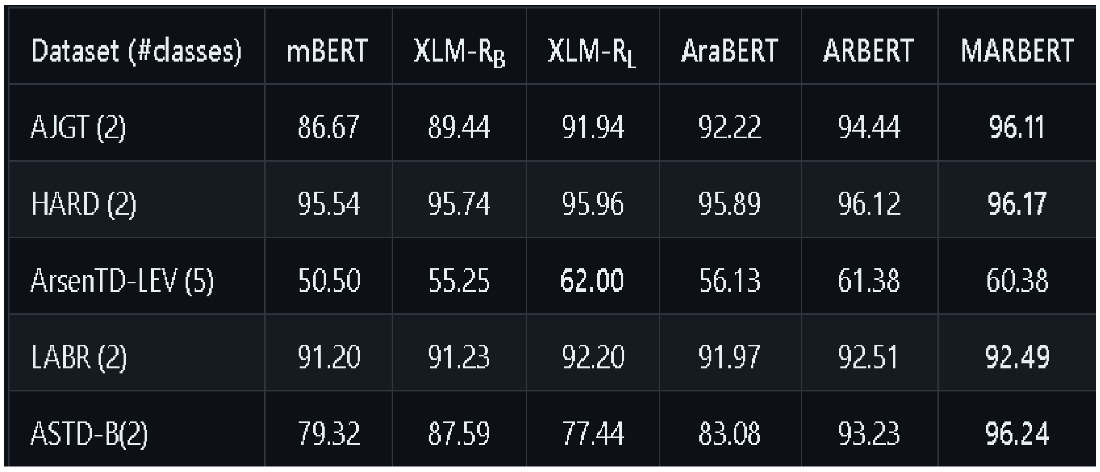
- These extracted features are then fed into a classifier, such as a support vector machine or a neural network, which is trained on the labeled dataset to predict the sentiment of a given piece of text. The pre-trained transformer-based language model acts as a feature extractor, providing the classifier with rich and contextually relevant representations of the text, which can capture the nuances of sentiment expression in natural language.

- This approach has been shown to be highly effective, achieving state-of-the-art performance on several benchmark sentiment analysis datasets. Additionally, transfer learning with transformers can also be used to address domain adaptation issues, where the target domain may be different from the source domain used for pre-training. By fine-tuning on a domain-specific dataset, the pre-trained model can adapt its learned representations to the specific nuances and requirements of the target domain.

- Overall, transfer learning with transformers has proven to be a powerful approach for **sentiment analysis**, improving accuracy and reducing the need for large amounts of labeled data. This approach is likely to continue to be a key area of research in the field of natural language processing and sentiment analysis, as shown in Table 2

  [1]

Table 2 accuracy scores for different transformer models on sentiment analysis



Dataset (#classes)	mBERT	XLM-R_B	XLM-R_L	AraBERT	ARBERT	MARBERT
AJGT (2)	86.67	89.44	91.94	92.22	94.44	**96.11**
HARD (2)	95.54	95.74	95.96	95.89	96.12	**96.17**
ArsenTD-LEV (5)	50.50	55.25	**62.00**	56.13	61.38	60.38
LABR (2)	91.20	91.23	92.20	91.97	92.51	**92.49**
ASTD-B(2)	79.32	87.59	77.44	83.08	93.23	**96.24**

2.10.3 Text Generation

There are several approaches to text generation, each with its own strengths and weaknesses. Some of the most common approaches [11] include:

- **Rule-based generation:** This approach involves creating a set of rules or templates that dictate how the generated text should be structured. For example, a rule-based generator might use templates to generate product descriptions or weather reports. While this approach is relatively simple and interpretable, it can be limited by the complexity of the task and the need for manual rule creation.

- **Markov chain generation:** This approach involves using a statistical model to generate text based on the probability of each word or sequence of words appearing in the training data. Markov chain generators can be simple to implement, but they may struggle with capturing long-term dependencies and producing coherent text.

- **Recurrent neural network (RNN) generation**: This approach involves using a neural network architecture, such as a long short-term memory (LSTM) or gated recurrent unit (GRU), to generate text. RNNs can capture long-term dependencies and produce coherent text, but they may struggle with modeling long-range dependencies and suffer from the vanishing gradient problem.

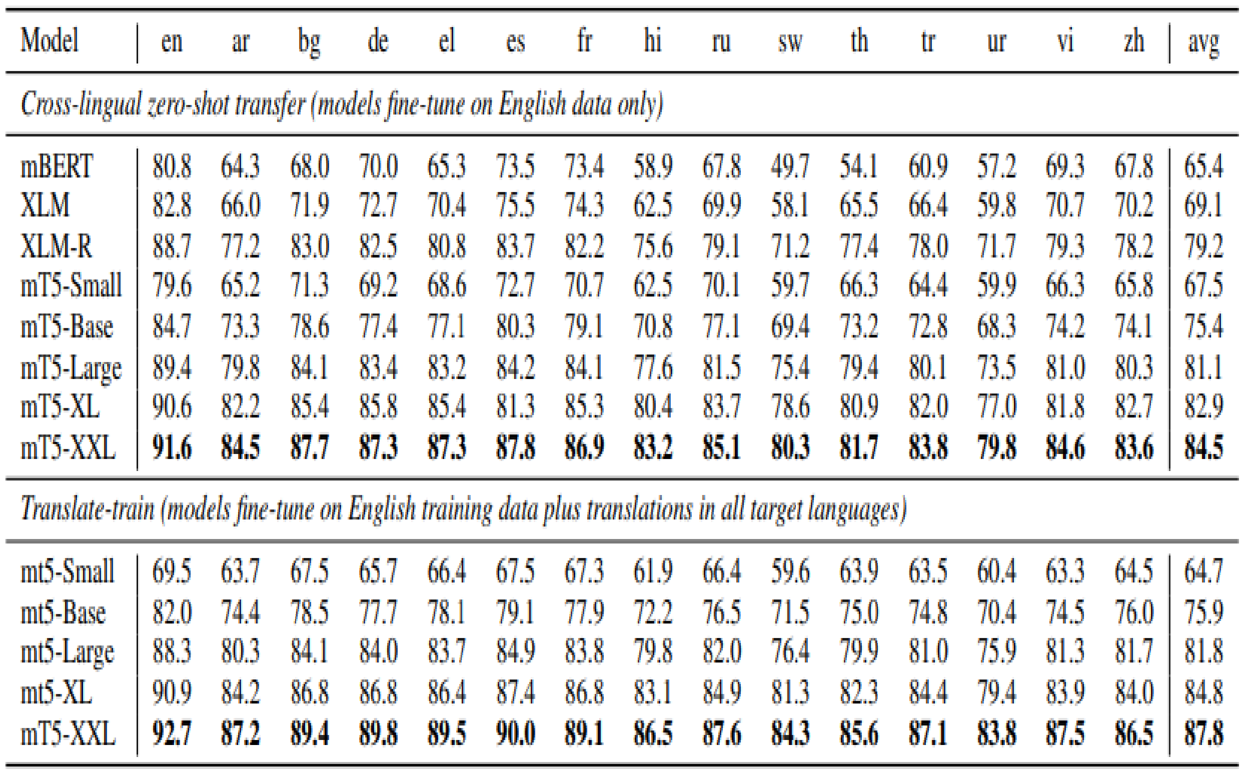
- **Transformer-based generation:** This approach involves using a transformer-based language model, such as AraGPT-2 [5] or MT5 [3], to generate text. These models are pre-trained on large amounts of text data and can generate high-quality text that is contextually relevant and coherent. Additionally, fine-tuning these models on a specific task can further improve their performance.

Using pre-trained models for text generation tasks can be highly effective, as these models have already learned to capture the general linguistic patterns and relationships that are relevant to generating coherent and contextually relevant text.

Pre-trained models can be fine-tuned on a specific task or domain to further improve their performance. For example, a pre-trained model like AraGPT-2 [5]can be fine-tuned on a dataset of news articles to generate high-quality news articles that are relevant to a particular topic or audience.

We can see the scores of different transformer models and different languages in Table 2 [3]

Table 3 Accuracy scores of different transformer models on text generation



Model	en	ar	bg	de	el	es	fr	hi	ru	sw	th	tr	ur	vi	zh	avg
Cross-lingual zero-shot transfer (models fine-tune on English data only)																
mBERT	80.8	64.3	68.0	70.0	65.3	73.5	73.4	58.9	67.8	49.7	54.1	60.9	57.2	69.3	67.8	65.4
XLM	82.8	66.0	71.9	72.7	70.4	75.5	74.3	62.5	69.9	58.1	65.5	66.4	59.8	70.7	70.2	69.1
XLM-R	88.7	77.2	83.0	82.5	80.8	83.7	82.2	75.6	79.1	71.2	77.4	78.0	71.7	79.3	78.2	79.2
mT5-Small	79.6	65.2	71.3	69.2	68.6	72.7	70.7	62.5	70.1	59.7	66.3	64.4	59.9	66.3	65.8	67.5
mT5-Base	84.7	73.3	78.6	77.4	77.1	80.3	79.1	70.8	77.1	69.4	73.2	72.8	68.3	74.2	74.1	75.4
mT5-Large	89.4	79.8	84.1	83.4	83.2	84.2	84.1	77.6	81.5	75.4	79.4	80.1	73.5	81.0	80.3	81.1
mT5-XL	90.6	82.2	85.4	85.8	85.4	81.3	85.3	80.4	83.7	78.6	80.9	82.0	77.0	81.8	82.7	82.9
mT5-XXL	**91.6**	**84.5**	**87.7**	**87.3**	**87.3**	**87.8**	**86.9**	**83.2**	**85.1**	**80.3**	**81.7**	**83.8**	**79.8**	**84.6**	**83.6**	**84.5**
Translate-train (models fine-tune on English training data plus translations in all target languages)																
mt5-Small	69.5	63.7	67.5	65.7	66.4	67.5	67.3	61.9	66.4	59.6	63.9	63.5	60.4	63.3	64.5	64.7
mt5-Base	82.0	74.4	78.5	77.7	78.1	79.1	77.9	72.2	76.5	71.5	75.0	74.8	70.4	74.5	76.0	75.9
mt5-Large	88.3	80.3	84.1	84.0	83.7	84.9	83.8	79.8	82.0	76.4	79.9	81.0	75.9	81.3	81.7	81.8
mt5-XL	90.9	84.2	86.8	86.8	86.4	87.4	86.8	83.1	84.9	81.3	82.3	84.4	79.4	83.9	84.0	84.8
mT5-XXL	**92.7**	**87.2**	**89.4**	**89.8**	**89.5**	**90.0**	**89.1**	**86.5**	**87.6**	**84.3**	**85.6**	**87.1**	**83.8**	**87.5**	**86.5**	**87.8**

2.10.4 Information Retrieval

There are several approaches to information retrieval, each with its own strengths and weaknesses. Some common approaches [8] include:

- **Keyword-based retrieval:** This approach involves matching user queries with documents that contain specific keywords or phrases. Keyword-based retrieval can be simple and effective, but it may miss relevant documents that do not contain the exact keywords or phrases used in the query.

- **Boolean retrieval:** This approach involves using Boolean operators, such as AND, OR, and NOT, to combine keywords and phrases in a query and retrieve documents that match the combination of terms. Boolean retrieval allows for more complex queries, but it requires users to have a good understanding of Boolean logic.

- **Vector space model:** This approach involves representing documents and queries as vectors in a high-dimensional space and calculating the similarity between them using measures such as cosine similarity. Vector space models can capture the semantic meaning of words and can be effective for retrieving relevant documents, but they require significant computational resources to process large datasets.

- **Probabilistic retrieval:** This approach involves using probabilistic models, such as the Okapi BM25 algorithm, to rank documents based on their relevance to a query. Probabilistic retrieval can be effective for retrieving relevant documents, but it requires significant computational resources to process large datasets.

- **Neural network-based retrieval:** This approach involves using neural network architectures, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), to learn representations of text and retrieve relevant documents. Neural network-based retrieval can be effective for capturing complex relationships between words and documents, but it requires significant computational resources and large amounts of training data.

3- Analysis and Design

3.1 System Overview

3.1.1 System Architecture

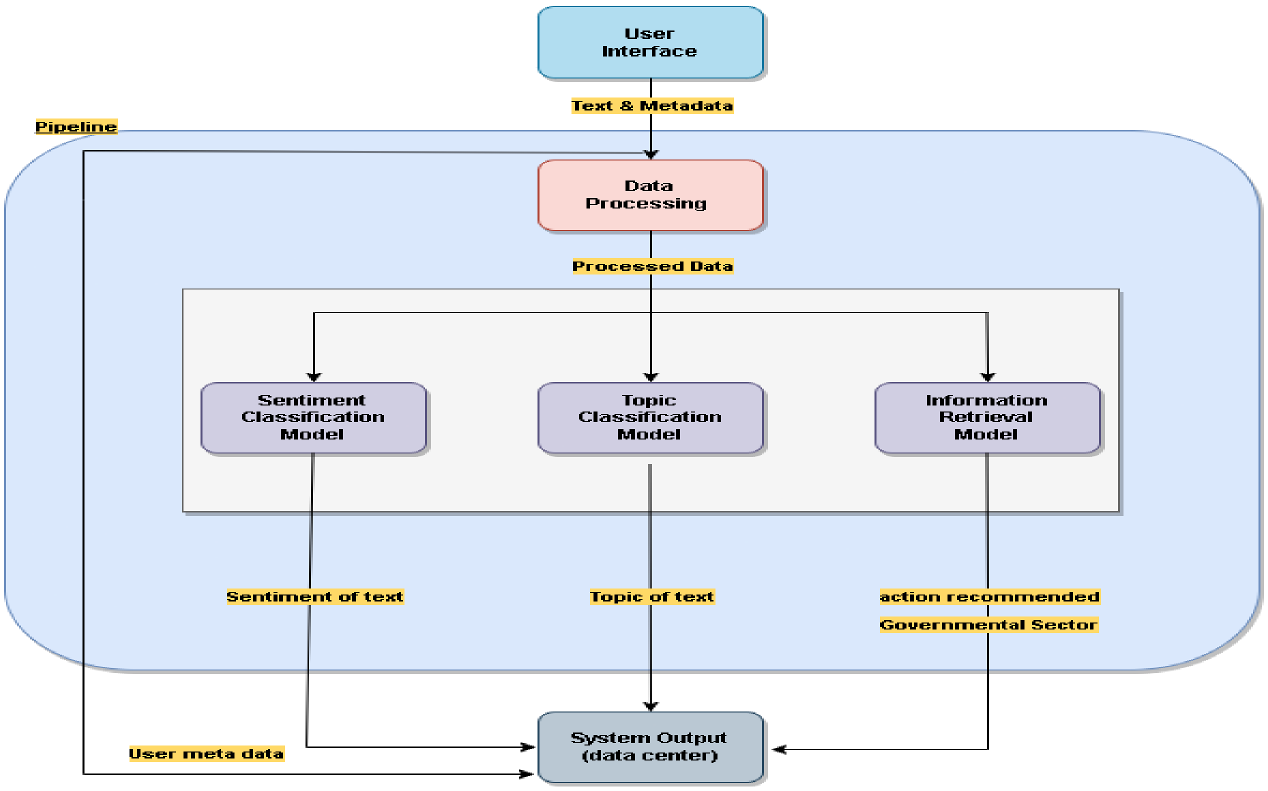


Figure 11System Architecture Diagram

These are the phases description of system architecture as shown in Figure 11

- **User Interface:**
  The user logs in to his account and provides his review text in the text Area.
- **Data Preprocessing:**
  The text data is then preprocessed and tokenized to be used in the models' pipeline.
- **Models pipeline:**
  The processed data is then sent to all models to work on it and provide the output in parallel.
- **System Output:**
  The output of the models is then passed to the database to be stored with its corresponding record.

3.1.2 System Users

A. Intended Users:

The system is made for all people who use android, iPhone, or pc.

B. User Characteristics

Users must have experience in using websites and browsers.

3.2 System Analysis & Design

3.2.1 Use Case Diagram

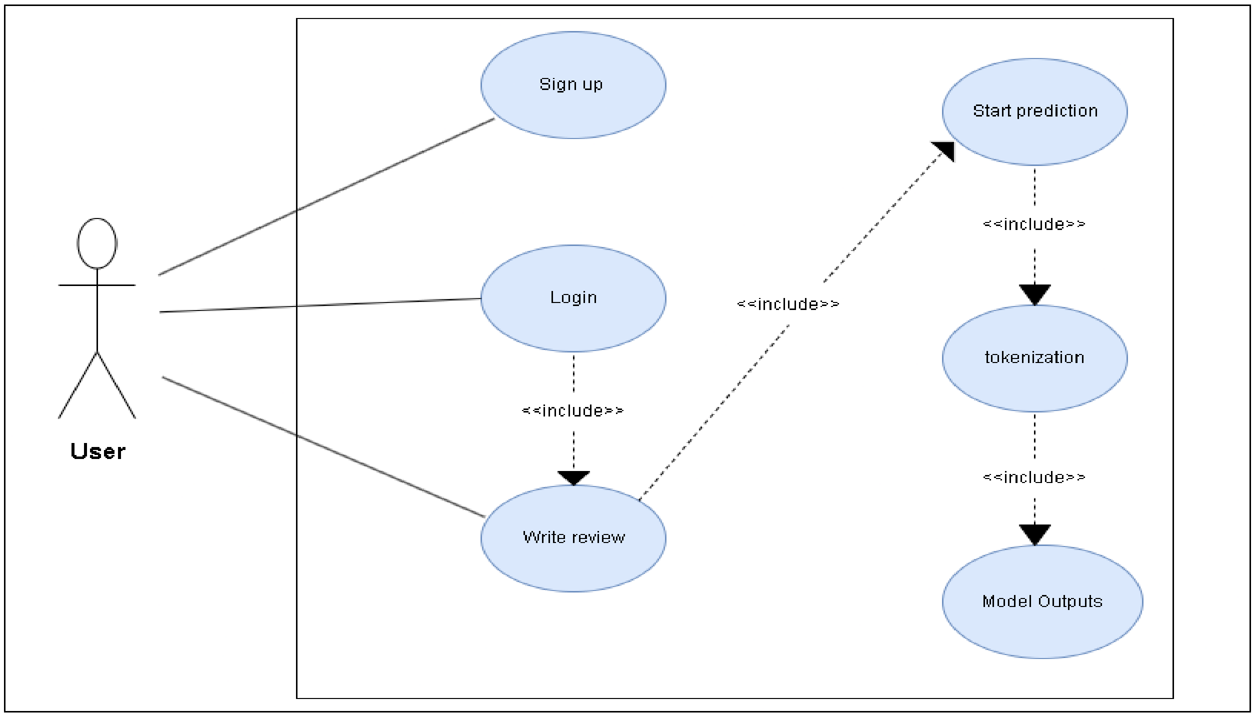


Figure 12 Use Case Diagram

This is the description of each use case in Figure 12

- **Sign up:** the user signs up with his information.
- **Login:** the user login with his credentials
- **Write review:** the user writes his review in the text area.
- **Start prediction:** after the user writes his review the start prediction phase starts.
- **Tokenization:** tokenization is done
- **Classification:** every model provides its output.

3.2.2 Class Diagram

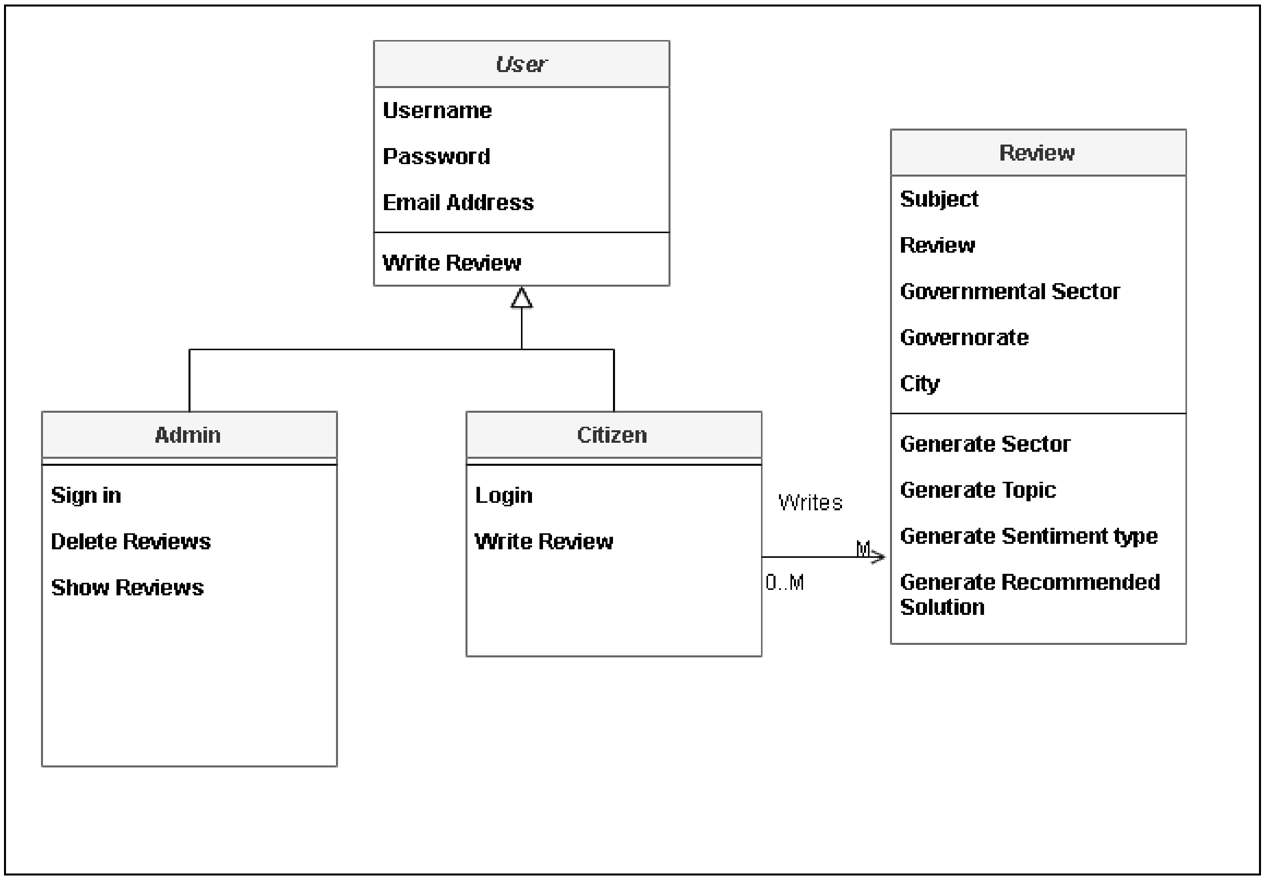


Figure 13 Class Diagram

This is the description of every class in Figure 13

1. **User Class:**

   Abstract class for citizen and admin classes.

2. **Citizen Class:**

   Inherits user class, citizen can login and write many reviews.

3. **Admin Class:**

   Inherits user class, admin can manage citizens' reviews, delete reviews, and show reviews.

4. **Review Class:**
   Every citizen can write many reviews, every review generates topic, sentiment type, recommended solution and sector.

3.2.3 Sequence Diagram

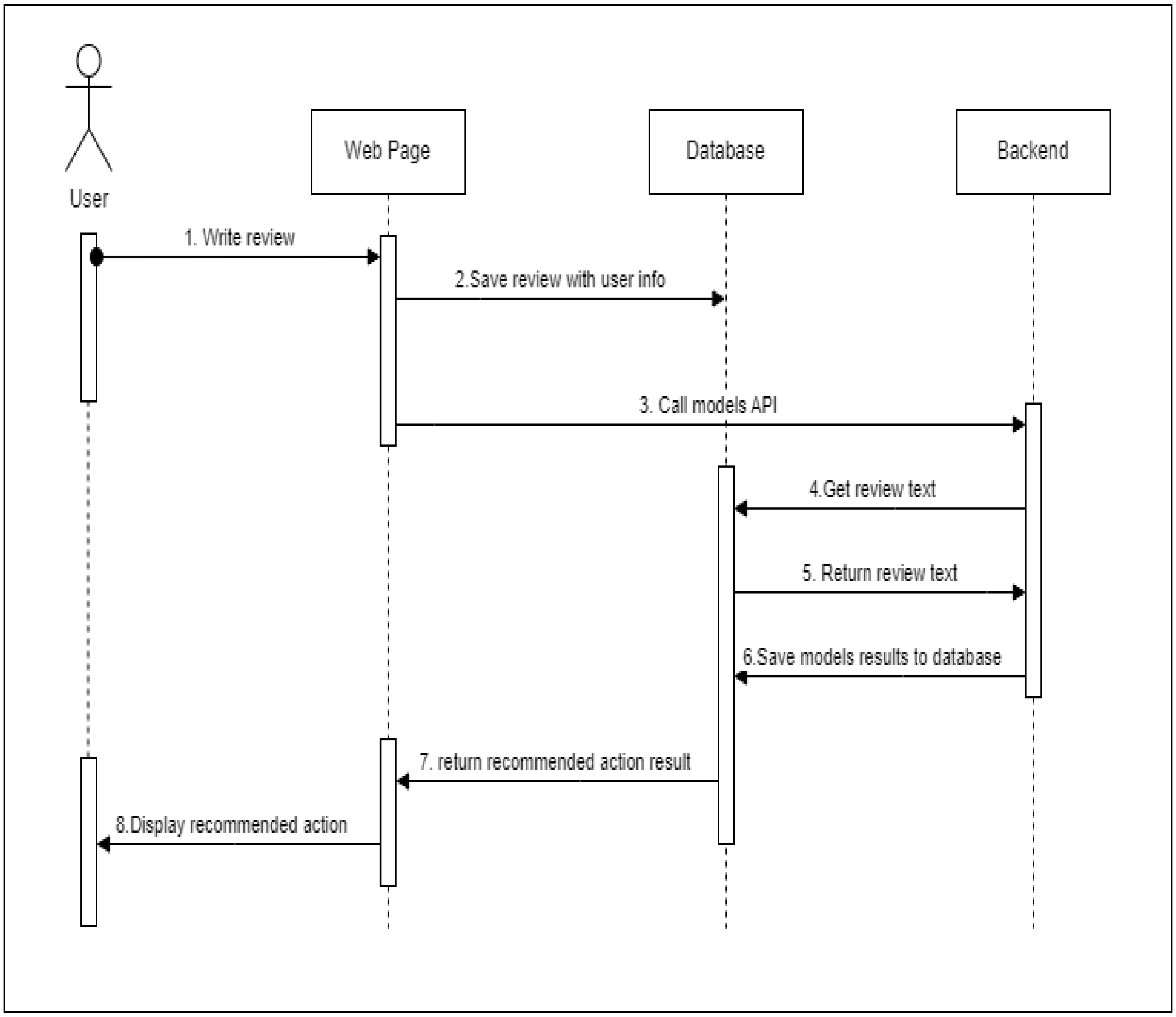
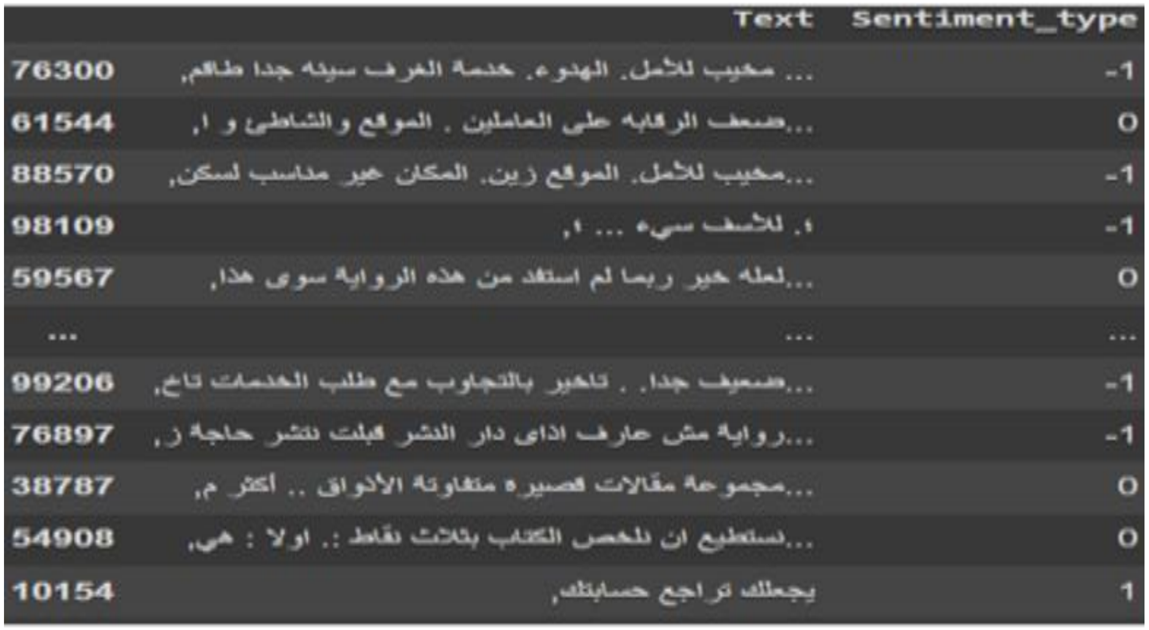

User
Web Page
Database
Backend
1. Write review
2.Save review with user info
3. Call models API
4.Get review text
5. Return review text
6.Save models results to database
7. return recommended action result
8.Display recommended action

Figure 14 Sequence Diagram

In Figure 14 we can see the full sequence of events in our project.

4- Datasets

4.1 100k Arabic Reviews Dataset



	Text	Sentiment_type
76300	... مخيب للأمل. الهدوء. خدمة الغرف سيئه جدا طاقم,	-1
61544	ضعف الرقابه على العاملين . الموقع والشاطئ و ا,...	0
88570	مخيب للأمل. الموقع زين. المكان غير مناسب لسكن,...	-1
98109	ا. للأسف سيء ... ا,	-1
59567	لعله خير ربما لم استفد من هذه الرواية سوى هذا,...	0
...	...	...
99206	ضعيف جدا. . تاخير بالتجاوب مع طلب الخدمات تاخ,...	-1
76897	رواية مش عارف اذاى دار النشر قبلت تنشر حاجة ز,...	-1
38787	مجموعة مقالات قصيره متفاوتة الأذواق .. اكثر م,...	0
54908	نستطيع ان نلخص الكتاب بثلاث نقاط :. اولا : هي,...	0
10154	يجعلك تراجع حساباتك,	1

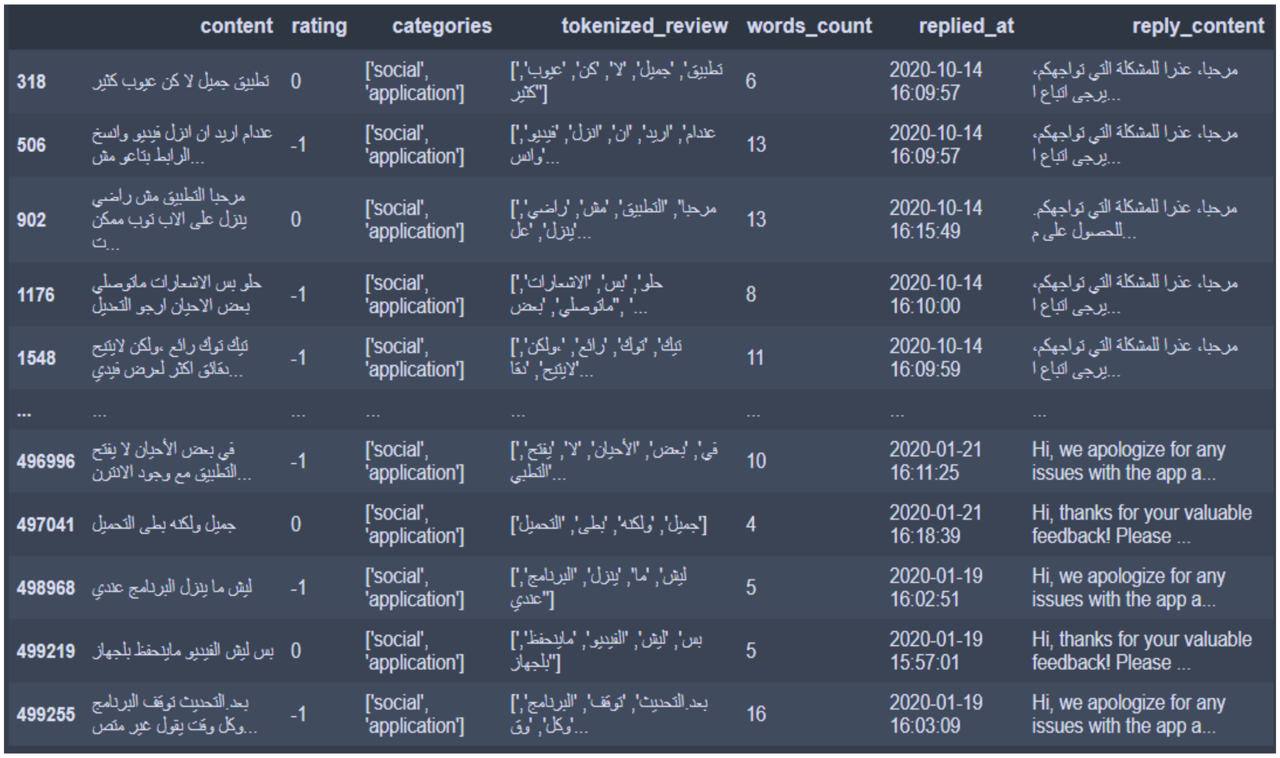
Figure 15 100k Arabic reviews dataset snippet

Company Arabic reviews collected from different domains labeled with sentiment type .

We can see a snippet from this dataset in Figure 15

Tasks: sentiment analysis
Categories: 1, 0, -1.
Number of records: 100k record
Language: Arabic

4.2 Glare Dataset



	content	rating	categories	tokenized_review	words_count	replied_at	reply_content
318	تطبيق جميل لا كن عيوب كثير	0	['social', 'application']	['تطبيق', 'جميل', 'لا', 'كن', 'عيوب', 'كثير']	6	2020-10-14 16:09:57	مرحبا، عذرا للمشكلة التي تواجهكم، يرجى اتباع ا...
506	عدام اريد ان انزل فيديو وانسخ الرابط بتاعو مش...	-1	['social', 'application']	['عدام', 'اريد', 'ان', 'انزل', 'فيديو', 'وانس...	13	2020-10-14 16:09:57	مرحبا، عذرا للمشكلة التي تواجهكم، يرجى اتباع ا...
902	مرحبا التطبيق مش راضي ينزل على الاب توب ممكن ت...	0	['social', 'application']	['مرحبا', 'التطبيق', 'مش', 'راضي', 'ينزل', 'عل...	13	2020-10-14 16:15:49	مرحبا، عذرا للمشكلة التي تواجهكم. للحصول على م...
1176	حلو بس الاشعارات ماتوصلي بعض الاحيان ارجو التعديل	-1	['social', 'application']	['حلو', 'بس', 'الاشعارات', 'ماتوصلي', 'بعض', '...	8	2020-10-14 16:10:00	مرحبا، عذرا للمشكلة التي تواجهكم، يرجى اتباع ا...
1548	تيك توك رائع ،ولكن لايتيح دقائق اكثر لعرض فيدي...	-1	['social', 'application']	['تيك', 'توك', 'رائع', '،ولكن', 'لايتيح', 'دقا...	11	2020-10-14 16:09:59	مرحبا، عذرا للمشكلة التي تواجهكم، يرجى اتباع ا...
...	...	...	...	...	...	...	...
496996	في بعض الأحيان لا يفتح التطبيق مع وجود الانترن...	-1	['social', 'application']	['في', 'بعض', 'الأحيان', 'لا', 'يفتح', 'التطبي...	10	2020-01-21 16:11:25	Hi, we apologize for any issues with the app a...
497041	جميل ولكنه بطى التحميل	0	['social', 'application']	['جميل', 'ولكنه', 'بطى', 'التحميل']	4	2020-01-21 16:18:39	Hi, thanks for your valuable feedback! Please ...
498968	ليش ما ينزل البرنامج عندي	-1	['social', 'application']	['ليش', 'ما', 'ينزل', 'البرنامج', 'عندي']	5	2020-01-19 16:02:51	Hi, we apologize for any issues with the app a...
499219	بس ليش الفيديو ماينحفظ بلجهاز	0	['social', 'application']	['بس', 'ليش', 'الفيديو', 'ماينحفظ', 'بلجهاز']	5	2020-01-19 15:57:01	Hi, thanks for your valuable feedback! Please ...
499255	بعد.التحديث توقف البرنامج وكل وقت يقول غير متص...	-1	['social', 'application']	['بعد.التحديث', 'توقف', 'البرنامج', 'وكل', 'وق...	16	2020-01-19 16:03:09	Hi, we apologize for any issues with the app a...

Figure 16 Glare Dataset Snippet

Android Applications Arabic reviews from Google Play [6]

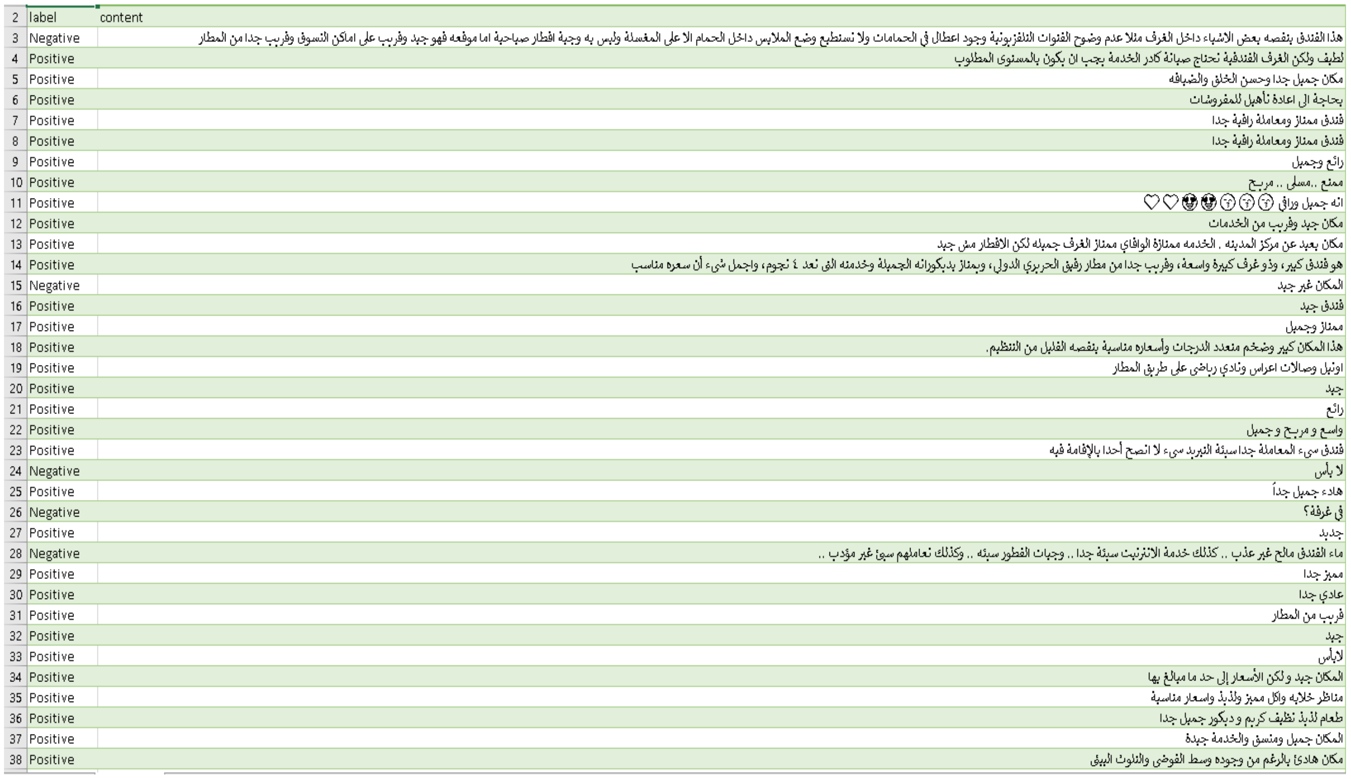
We can see a snippet for this dataset in Figure 16

Preprocessing

•filtering the data from English reviews or reply text
•restricting the sentiment type (rating) from 5 categories to only 2
•remove English words from Arabic reviews and reply text
•filtering the reviews to only who have more than 10 words

Tasks: sentiment analysis, topic classification and text generation.
Number of records: 79M + record.
Categories: for sentiment analysis 1 , -1 , 0 & for topic classification it has More than 64 different categories.
Languages: Arabic , English.

4.3 Oclar Dataset



2	label	content
3	Negative	هذا الفندق ينقصه بعض الاشياء داخل الغرف مثلا عدم وضوح القنوات التلفزيونية وجود اعطال في الحمامات ولا نستطيع وضع الملابس داخل الحمام الا على المغسلة وليس به وجبة افطار صباحية اما موقعه فهو جيد وقريب على اماكن التسوق وقريب جدا من المطار
4	Positive	لطيف ولكن الغرف الفندقية تحتاج صيانة كادر الخدمة يجب ان يكون بالمستوى المطلوب
5	Positive	مكان جميل جدا وحسن الخلق والضيافه
6	Positive	بحاجة الى اعادة تأهيل للمفروشات
7	Positive	فندق ممتاز ومعاملة راقية جدا
8	Positive	فندق ممتاز ومعاملة راقية جدا
9	Positive	رائع وجميل
10	Positive	ممتع ..مسلى .. مريح
11	Positive	انه جميل وراقي 😚😚😚😍😍♡♡
12	Positive	مكان جيد وقريب من الخدمات
13	Positive	مكان بعيد عن مركز المدينه . الخدمه ممتازة الوافاي ممتاز الغرف جميله لكن الافطار مش جيد
14	Positive	هو فندق كبير، وذو غرف كبيرة واسعة، وقريب جدا من مطار رفيق الحريري الدولي، ويمتاز بديكوراته الجميلة وخدمته التى تعد ٤ نجوم، واجمل شيء أن سعره مناسب
15	Negative	المكان غير جيد
16	Positive	فندق جيد
17	Positive	ممتاز وجميل
18	Positive	هذا المكان كبير وضخم متعدد الدرجات وأسعاره مناسبة ينقصه القليل من التنظيم.
19	Positive	اوتيل وصالات اعراس ونادي رياضي على طريق المطار
20	Positive	جيد
21	Positive	رائع
22	Positive	واسع و مريح و جميل
23	Positive	فندق سيء المعاملة جداسيئة التبريد سيء لا انصح أحدا بالإقامة فيه
24	Negative	لا بأس
25	Positive	هادء جميل جداً
26	Negative	في غرفة؟
27	Positive	جديد
28	Negative	ماء الفندق مالح غير عذب .. كذلك خدمة الانترنيت سيئة جدا .. وجبات الفطور سيئه .. وكذلك تعاملهم سيئ غير مؤدب ..
29	Positive	مميز جدا
30	Positive	عادي جدا
31	Positive	قريب من المطار
32	Positive	جيد
33	Positive	لابأس
34	Positive	المكان جيد و لكن الأسعار إلى حد ما مبالغ بها
35	Positive	مناظر خلابه واكل مميز ولذيذ واسعار مناسبة
36	Positive	طعام لذيذ نظيف كريم و ديكور جميل جدا
37	Positive	المكان جميل ومنسق والخدمة جيدة
38	Positive	مكان هادئ بالرغم من وجوده وسط الفوضى والتلوث البيئى

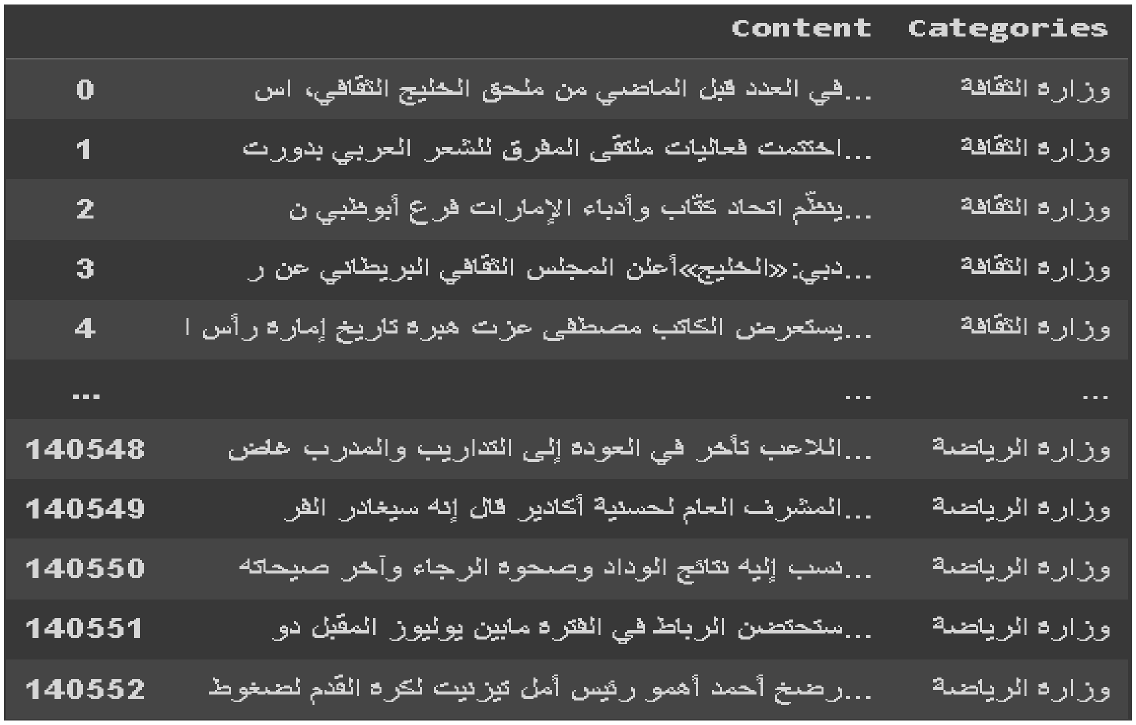
Figure 17 Oclar Dataset Snippet

Arabic Reviews from google maps including restaurants, hotels, hospitals, local shops [7].

We can see a snippet for this dataset in Figure 17

Tasks: sentiment analysis
Number of records: 4000+ record
Categories: positive , negative
Languages: Arabic

4.4 Merged Topics Dataset



	Content	Categories
0	في العدد قبل الماضي من ملحق الخليج الثقافي، اس...	وزاره الثقافة
1	اختتمت فعاليات ملتقى المفرق للشعر العربي بدورت...	وزاره الثقافة
2	ينظّم اتحاد كتّاب وأدباء الإمارات فرع أبوظبي ن...	وزاره الثقافة
3	دبي: «الخليج»أعلن المجلس الثقافي البريطاني عن ر...	وزاره الثقافة
4	يستعرض الكاتب مصطفى عزت هبره تاريخ إماره رأس ا...	وزاره الثقافة
...	...	...
140548	اللاعب تأخر في العوده إلى التداريب والمدرب غاض...	وزاره الرياضة
140549	المشرف العام لحسنية أكادير قال إنه سيغادر الفر...	وزاره الرياضة
140550	نسب إليه نتائج الوداد وصحوه الرجاء وآخر صيحاته...	وزاره الرياضة
140551	ستحتضن الرباط في الفتره مابين يوليوز المقبل دو...	وزاره الرياضة
140552	رضخ أحمد أهمو رئيس أمل تيزنيت لكره القدم لضغوط...	وزاره الرياضة

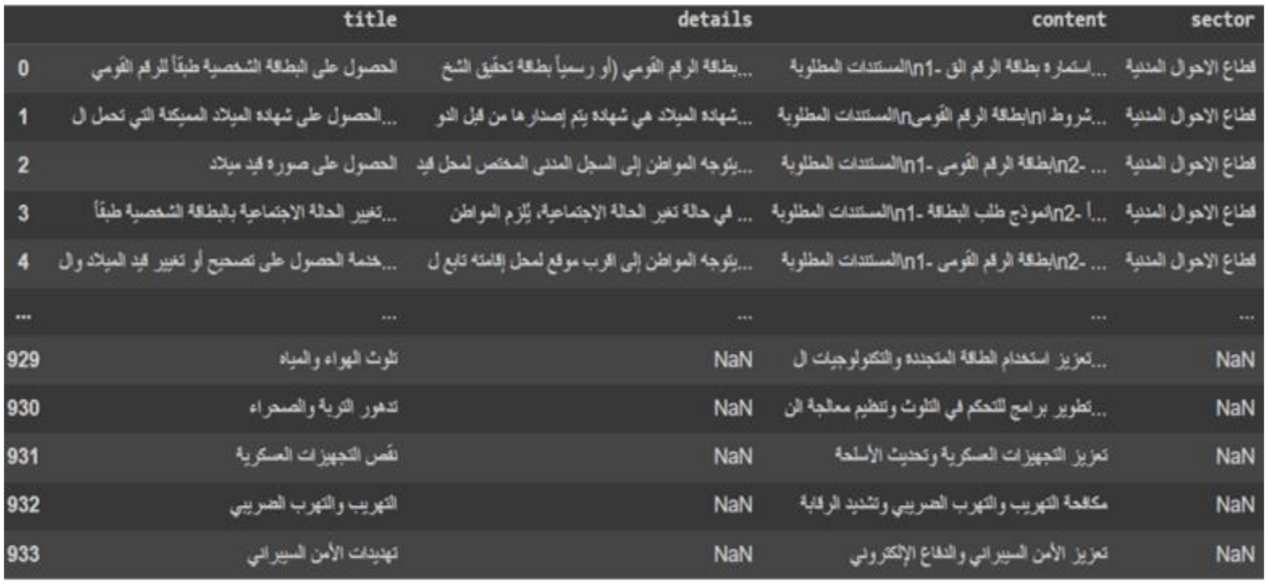
Figure 18 Merged Topics Dataset Snippet

Arabic reviews labeled with 7-different topics merge of several datasets, containing separated text files.

We can see a snippet for this dataset in Figure 18

Tasks: topic classification
Categories: 7 categories of different ministries' names
Number of records: 140K + record
Language: Arabic

4.5 Governmental Dataset



	title	details	content	sector
0	الحصول على البطاقة الشخصية طبقاً للرقم القومي	بطاقة الرقم القومي (أو رسمياً بطاقة تحقيق الشخ...	المستندات المطلوبة\n1- استماره بطاقة الرقم الق...	قطاع الاحوال المدنية
1	الحصول على شهاده الميلاد المميكنة التي تحمل ال...	شهاده الميلاد هي شهاده يتم إصدارها من قبل الدو...	المستندات المطلوبة\nبطاقة الرقم القومى\nشروط...	قطاع الاحوال المدنية
2	الحصول على صوره قيد ميلاد	يتوجه المواطن إلى السجل المدنى المختص لمحل قيد...	المستندات المطلوبة\n1- بطاقة الرقم القومى\n2- ...	قطاع الاحوال المدنية
3	تغيير الحالة الاجتماعية بالبطاقة الشخصية طبقاً...	في حالة تغير الحالة الاجتماعية، يُلزم المواطن ...	المستندات المطلوبة\n1- نموذج طلب البطاقة\n2- أ...	قطاع الاحوال المدنية
4	خدمة الحصول على تصحيح أو تغيير قيد الميلاد وال...	يتوجه المواطن إلى اقرب موقع لمحل إقامته تابع ل...	المستندات المطلوبة\n1- بطاقة الرقم القومى\n2- ...	قطاع الاحوال المدنية
...	...	...	...	...
929	تلوث الهواء والمياه	NaN	تعزيز استخدام الطاقة المتجدده والتكنولوجيات ال...	NaN
930	تدهور التربة والصحراء	NaN	تطوير برامج للتحكم في التلوث وتنظيم معالجة الن...	NaN
931	نقص التجهيزات العسكرية	NaN	تعزيز التجهيزات العسكرية وتحديث الأسلحة	NaN
932	التهريب والتهرب الضريبي	NaN	مكافحة التهريب والتهرب الضريبي وتشديد الرقابة	NaN
933	تهديدات الأمن السيبراني	NaN	تعزيز الأمن السيبراني والدفاع الإلكتروني	NaN

Figure 19 Governmental Dataset Snippet

Dataset using web scrapping on website (دليل الخدمات العامة) contains title (problem), content (solution), details about problem and sector of problem.
Dataset Augmented by getting samples from ChatGPT.

We can see a snippet for this dataset in Figure 19
Tasks: Information retrieval
Number of records: 934 records
Language: Arabic
Labels: sector for each review , steps for solving problem.

5- Implementation and Testing

We have created this project with many scripts, we used Python programming language as it was the simplest and most used language in the field of AI because it is open source, easy language and have enormous number of packages and libraries that are very useful in our project field.

We used Jupiter notebooks and visual studio code for our scripts; however, we used 3 environments to run those scripts due to the high processing needs some of these scripts required.

We used Google drive and GitHub for version control and for merging work together.

5.1 Environment setup & Tools

5.1.1 Environments

1. **Localhost**
    - While developing our scripts (preprocessing, training and testing), we tried running them locally, but the processing needs for these scripts were high, so localhost wasn't sufficient, it Tooks too much time and lots of memory.

2. **Google Colab**
    - Secondly, we tried Google colab, it worked for most of our scripts, however we faced lots of issues with it as the memory was very limited

3. **Google Colab Pro**
    - We tried Google colab pro as it was the best solution for the processing power needed as it provided V100 Tesla GPU with 40 Gb memory, 80 Gb ram and 150 Gb disk space.

4. **Kaggle Notebooks**
    - Same as Google colab with additional TPU.

5.1.2 Packages & Libraries

1. **Pandas**: a powerful, fast, flexible and easy to use open-source data analysis and manipulation tool, built on top of the Python programming language.

2. **Matplotlib**: a collection of command style functions that make matplotlib work like MATLAB, each pyplot function makes some change to figure.

3. **OS**: provides a way to interact with the operating system that Python is running on. This library offers a set of functions that allow you to perform various operations related to the file system, process management, environment variables, and other system-related tasks.

4. **NumPy:** Python library for performing numerical computations with multidimensional arrays and matrices.

5. **Py torch**: often referred to simply as Torch, is an open-source machine learning library for Python that provides a fast and flexible way to build and train neural networks.

6. **Transformers:** state-of-the-art natural language processing library in Python that provides a simple and efficient way to train and deploy pre-trained models for a wide range of language tasks, including text classification, question answering, and language translation.

7. **Sentence Piece:** language-agnostic sub word tokenization library in C++ and Python that uses unsupervised machine learning to segment sentences into sub word units, which can be used to improve the performance of natural language processing tasks such as machine translation and text classification.

8. **Tqdm:** Python library that provides a fast and extensible way to add progress bars to loops and other iterable objects, making it easier to track the progress of long-running tasks and providing a more user-friendly experience for users.

9. **Json:** provides a simple way to encode and decode data in the JSON (JavaScript Object Notation) format, which is widely used for data exchange between web applications and services.

10. **TensorFlow:** open-source machine learning library in Python that provides a flexible and efficient way to build and train a wide range of machine learning and deep learning models, using both CPUs and GPUs, and is widely used in industry and academia.

11. **Scikit-learn:** often abbreviated as sklearn, is a popular machine learning library in Python that provides a wide range of tools for data preprocessing, feature engineering, model selection, and performance evaluation, making it easier to build and deploy machine learning models in real-world applications.

12. **NLTK:** Python library that provides a suite of tools and resources for natural language processing (NLP), including tokenization, stemming, tagging, parsing, semantic reasoning, and corpus analysis, making it easier to process and analyze human language data.

13. **Re**: provides a set of functions for working with regular expressions, which are powerful and flexible patterns that can be used to match and manipulate text data, making it easier to process and extract relevant information from large amounts of unstructured text data.

14. Joblib: provides tools for easy and efficient parallel computing, including fast and efficient memory-mapped file storage, task scheduling, and caching, making it easier to speed up and optimize computationally intensive tasks in machine learning and other data-driven applications.

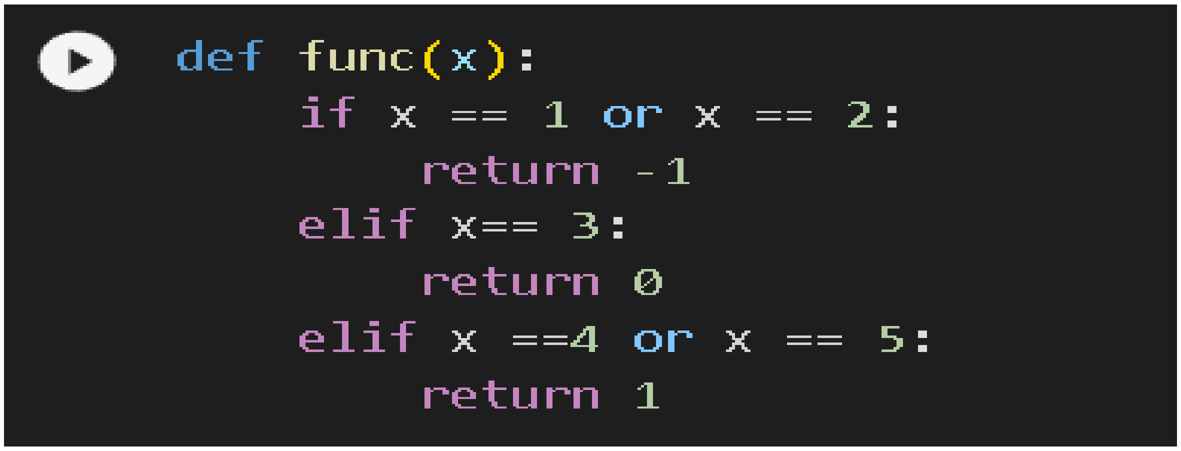
15. Selenium: widely used open-source library that provides a suite of tools and APIs for automating web browsers, making it easier for developers to test and interact with web applications

16. Beautiful Soup: popular Python library used for web scraping purposes, providing a convenient way to parse HTML and XML documents and extract useful information from them.

5.2 Data Preprocessing

5.2.1 Glare Dataset Preprocessing

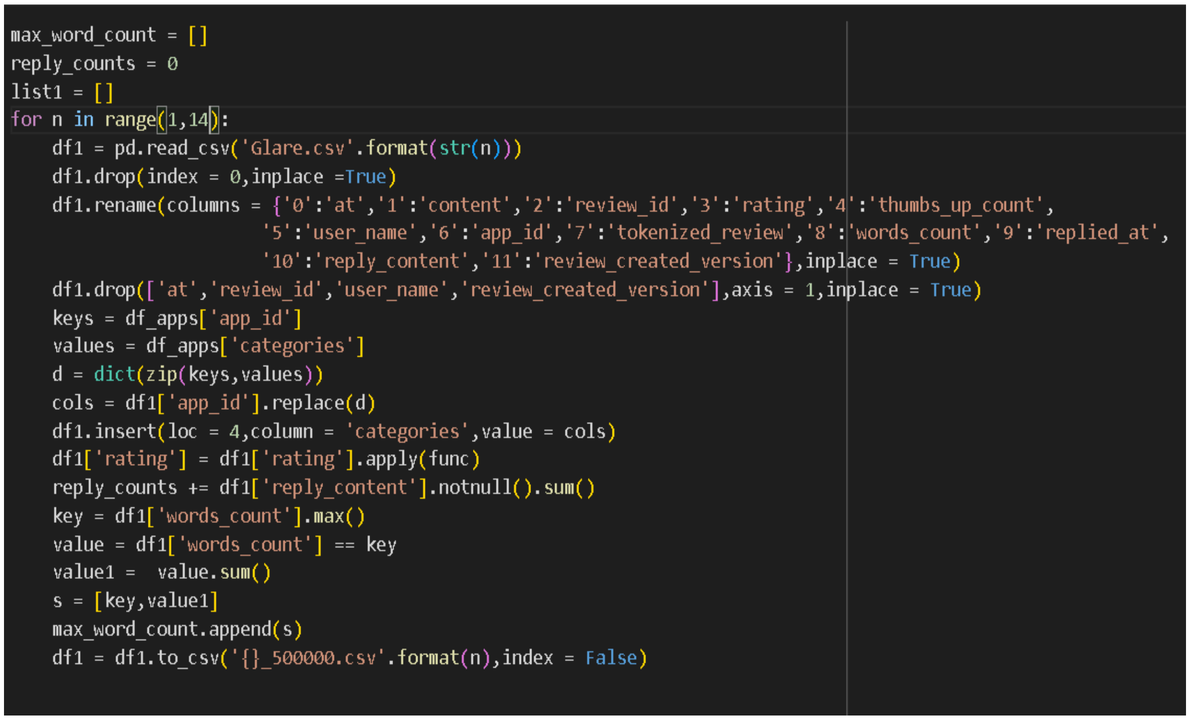
- We splitted the dataset into several small csv files to be able to work with it as it was exceeding 20 Gb, every small file was 5000000 rows.
- There was a separate file containing the category of each review linked with the dataset with unique id for each row, we applied a function to get the specific category for each review in the dataset.
- Then we worked on the rating column (sentiment analysis) to restrict its values from 5 to 3 categories only.
- Then we filtered the data set to only reviews which have reply content text (text generation).
- Then we got only the Arabic reviews and reply content text and remove any other rows containing any other language.
- We end up with dataset labeled with category, rating and recommended solution text.
- We can see the snippet of code of function dealing with rating review in Figure 20



```
def func(x):
    if x == 1 or x == 2:
        return -1
    elif x== 3:
        return 0
    elif x ==4 or x == 5:
        return 1
```

Figure 20 Code of function for processing rating column in glare dataset

We can see the snippet of code of function that preprocess glare dataset in Figure 21



```python
max_word_count = []
reply_counts = 0
list1 = []
for n in range(1,14):
    df1 = pd.read_csv('Glare.csv'.format(str(n)))
    df1.drop(index = 0,inplace =True)
    df1.rename(columns = {'0':'at','1':'content','2':'review_id','3':'rating','4':'thumbs_up_count',
                         '5':'user_name','6':'app_id','7':'tokenized_review','8':'words_count','9':'replied_at',
                         '10':'reply_content','11':'review_created_version'},inplace = True)
    df1.drop(['at','review_id','user_name','review_created_version'],axis = 1,inplace = True)
    keys = df_apps['app_id']
    values = df_apps['categories']
    d = dict(zip(keys,values))
    cols = df1['app_id'].replace(d)
    df1.insert(loc = 4,column = 'categories',value = cols)
    df1['rating'] = df1['rating'].apply(func)
    reply_counts += df1['reply_content'].notnull().sum()
    key = df1['words_count'].max()
    value = df1['words_count'] == key
    value1 =  value.sum()
    s = [key,value1]
    max_word_count.append(s)
    df1 = df1.to_csv('{}_500000.csv'.format(n),index = False)
```

Figure 21 Code of processing all glare dataset.

5.2.2 Merged Topics Data preprocessing

- We have searched for many datasets for topic classification task, they were in different domains, so we merged many datasets together to form one dataset that has 140k + rows to use in topic classification task.
- We have balanced the dataset categories to be in the same range.

5.2.3 Web Scrapping Module

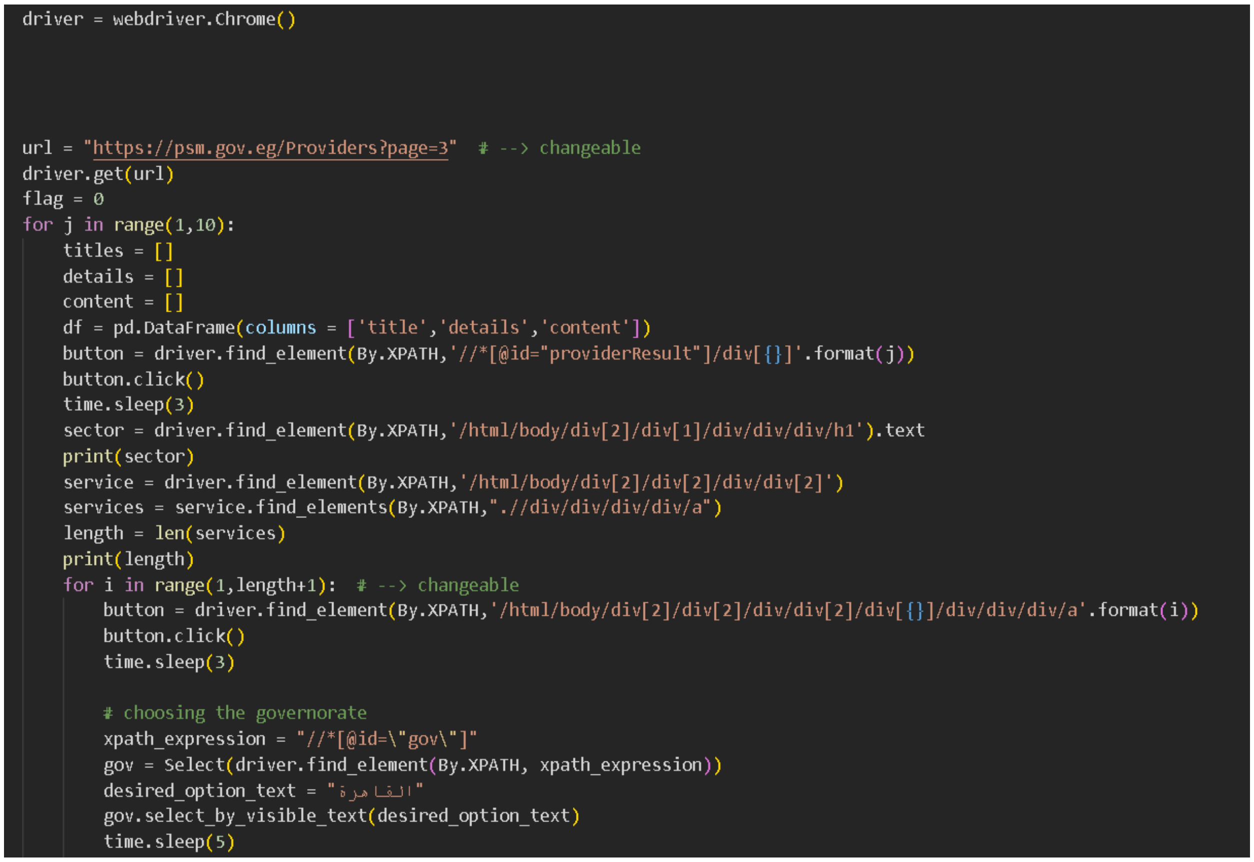
- Due to the lack of datasets in the task of recommending solution (text generation) in our domain we have decided to apply web scrapping techniques to get new data.

- We have scrapped open website (دليل الخدمات العامة) to get data in form of citizen problem (or service), solution & steps to get this service and governmental sector for each service.

- We used selenium and beautiful soup libraries to get these data.

- We ended up with 433 rows of data and augmented our data by using ChatGPT to produce the same form of data in our domain, lastly, we got 930+ rows of data.

- We can see a snippet of code for function that do web scrapping in Figure 22



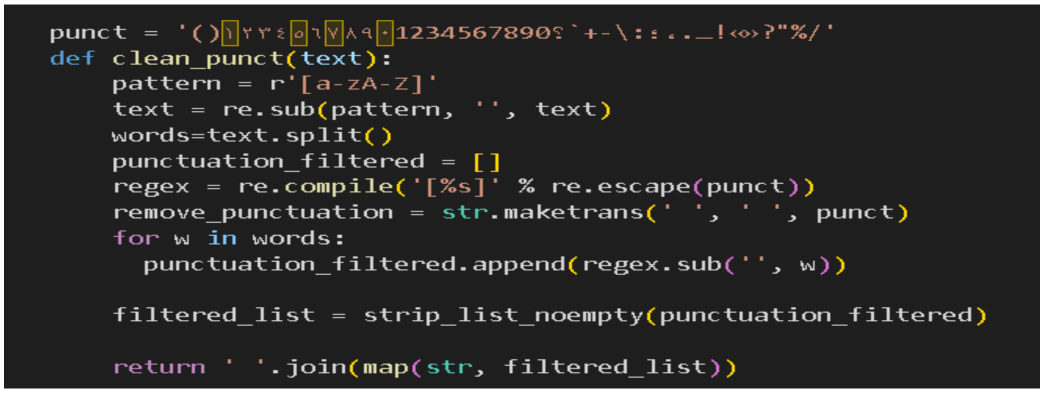
```
driver = webdriver.Chrome()



url = "https://psm.gov.eg/Providers?page=3"  # --> changeable
driver.get(url)
flag = 0
for j in range(1,10):
    titles = []
    details = []
    content = []
    df = pd.DataFrame(columns = ['title','details','content'])
    button = driver.find_element(By.XPATH,'//*[@id="providerResult"]/div[{}]'.format(j))
    button.click()
    time.sleep(3)
    sector = driver.find_element(By.XPATH,'/html/body/div[2]/div[1]/div/div/div/h1').text
    print(sector)
    service = driver.find_element(By.XPATH,'/html/body/div[2]/div[2]/div/div[2]')
    services = service.find_elements(By.XPATH,".//div/div/div/div/a")
    length = len(services)
    print(length)
    for i in range(1,length+1):  # --> changeable
        button = driver.find_element(By.XPATH,'/html/body/div[2]/div[2]/div/div[2]/div[{}]/div/div/div/a'.format(i))
        button.click()
        time.sleep(3)

        # choosing the governorate
        xpath_expression = "//*[@id=\"gov\"]"
        gov = Select(driver.find_element(By.XPATH, xpath_expression))
        desired_option_text = "القاهرة"
        gov.select_by_visible_text(desired_option_text)
        time.sleep(5)
```

Figure 22 Code for Web scrapping Governmental dataset.

5.2.4 Input Text Preprocessing Module

- Text preprocessing pipeline contains the following functions

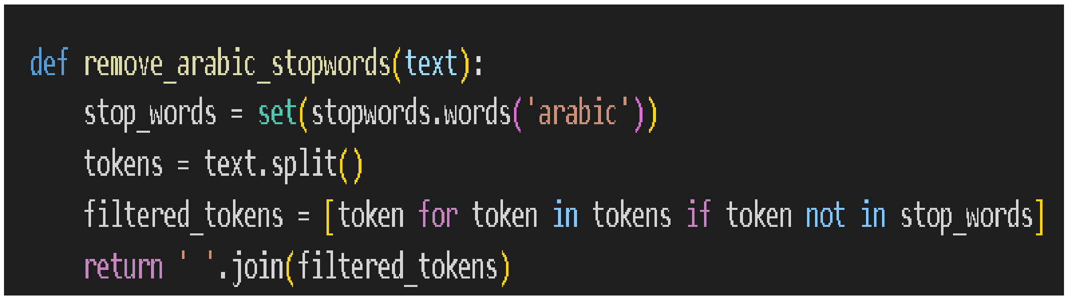
1. **Handling Punctuation Function**: this function removes all punctuation marks and numbers as shown in Figure 23



```
punct = '()١٢٣٤٥٦٧٨٩٠1234567890؟`+-\:؛،._!«»?"%/'
def clean_punct(text):
    pattern = r'[a-zA-Z]'
    text = re.sub(pattern, '', text)
    words=text.split()
    punctuation_filtered = []
    regex = re.compile('[%s]' % re.escape(punct))
    remove_punctuation = str.maketrans(' ', ' ', punct)
    for w in words:
      punctuation_filtered.append(regex.sub('', w))

    filtered_list = strip_list_noempty(punctuation_filtered)

    return ' '.join(map(str, filtered_list))
```

Figure 23 Code for cleaning punctuation function

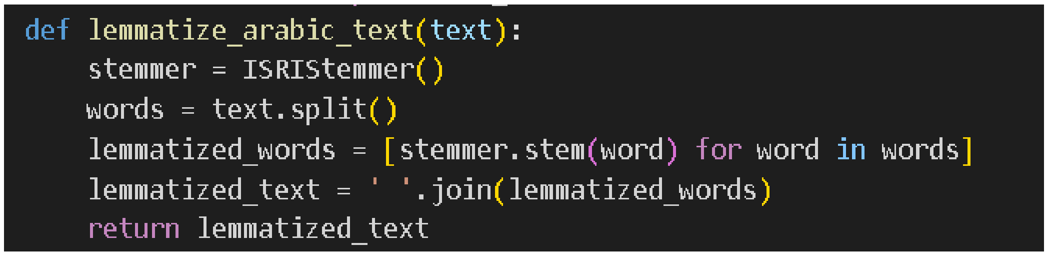
2. **Stop words Removal Function**: this function removes all Arabic stop words as shown in Figure 24



```
def remove_arabic_stopwords(text):
    stop_words = set(stopwords.words('arabic'))
    tokens = text.split()
    filtered_tokens = [token for token in tokens if token not in stop_words]
    return ' '.join(filtered_tokens)
```

Figure 24 Code for Stop words removal function.

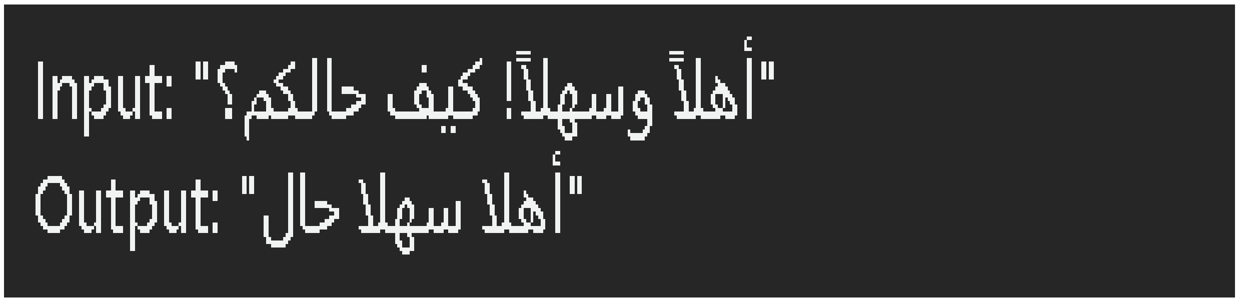
3. **Lemmatization Function**: this function applies Arabic lemmatization to text as shown in Figure 25



```
def lemmatize_arabic_text(text):
    stemmer = ISRIStemmer()
    words = text.split()
    lemmatized_words = [stemmer.stem(word) for word in words]
    lemmatized_text = ' '.join(lemmatized_words)
    return lemmatized_text
```

Figure 25 Code for lemmatization function

We can see a sample text input and output of this module in Figure 26



```
Input: "أهلاً وسهلاً! كيف حالكم؟"
Output: "أهلا سهلا حال"
```

Figure 26 Sample in\out text for input text preprocessing module

5.4 Topic classification Module

Models Used:

1. MARBERT [1]: is a large-scale pre-trained masked language model focused on both Dialectal Arabic (DA) and MSA provided by UBC\NLP.

2. Arabic-MARBERT-news-article-classification is a news article classification model that was built by fine-tuning the MARBERT model provided by Ammar-alhaj-ali.

Data Preparation

- **Corpus Collection:** The first step in data preparation for text classification is to collect a comprehensive and representative dataset. The dataset should cover a wide range of instances that are relevant to the classification task at hand.

- **Text Preprocessing:** Involves transforming raw text into a more structured and suitable format for classification.

- Key preprocessing techniques include:

  1. **Text Cleaning:** Removing noise, such as HTML tags, special characters, punctuation, and irrelevant content from the text data.

  2. **Tokenization:** Breaking down the text into smaller units called tokens, such as words, sub words, or characters. Tokenization allows for better analysis and feature extraction.

  3. **Stop Word Removal:** Eliminating common and insignificant words, known as stop words (e.g., "the," "and" "is"), which do not carry much semantic meaning.

  4. **Lemmatization and Stemming**: Reducing words to their base or root form (lemmatization) or stripping them down to their core (stemming). This helps in reducing vocabulary size and capturing the essence of words.

- **Feature Extraction:** involves transforming the preprocessed text data into numerical representations using the selected model’s embeddings.

- **Label Encoding:** Assigning unique integer values to each label, converting “Categories” column into numerical representations.

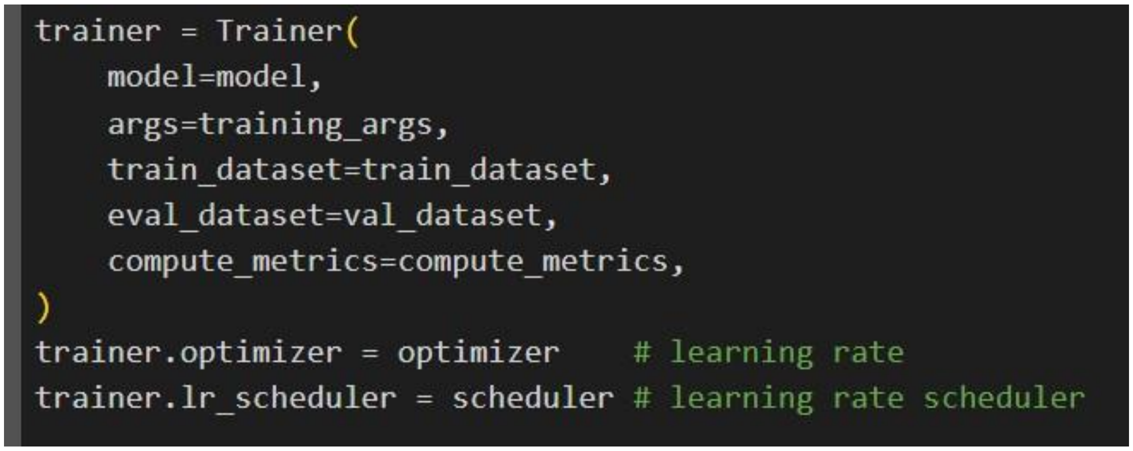
Training Procedure

- **Splitting the Dataset:** Divide the processed dataset into training, validation, and test sets.

- **Model Selection:** We choose for training procedure MARBERT model.

- **Model Training:** Train the MARBERT model on the preprocessed data using the Trainer library.

- We can see the code of using trainer function in Figure 27

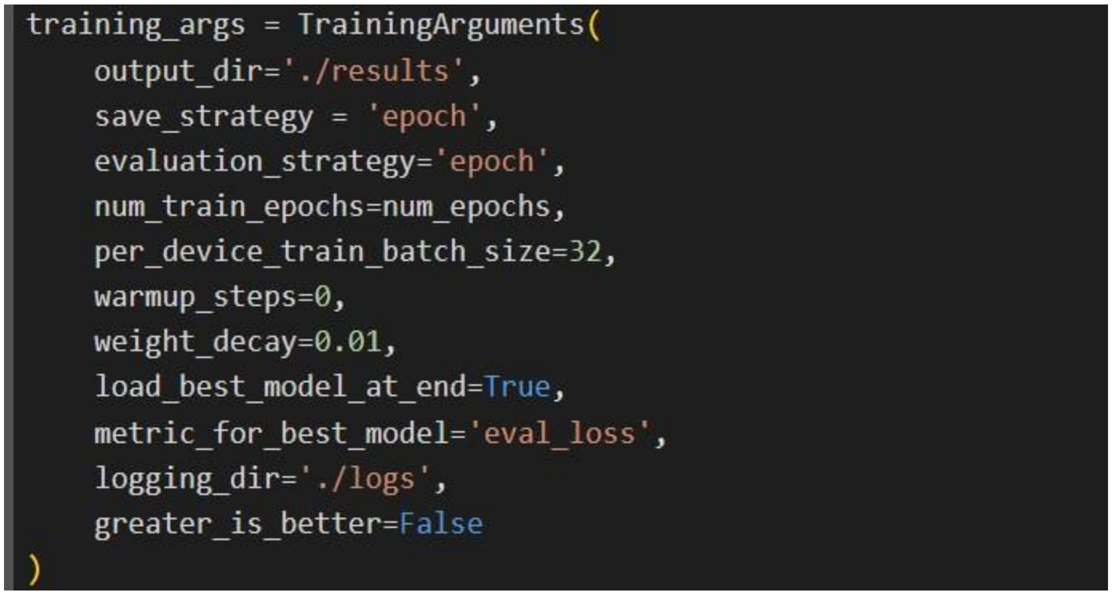


```
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,
    compute_metrics=compute_metrics,
)
trainer.optimizer = optimizer    # learning rate
trainer.lr_scheduler = scheduler # learning rate scheduler
```

Figure 27 Code for trainer function of topic classification

- **Hyperparameter Tuning:** We fine-tune the model's hyperparameters to achieve better performance. define the “AdamW” optimizer and learning rate scheduler starting with “LR=5e-5”, batch size “train_batch_size=32”, and the number of epochs “Num epochs = 5”. This process involves experimentation and validation using the validation set to find the optimal combination of hyperparameters that results in the best performance.

- **We can see the code of training arguments in** Figure 28

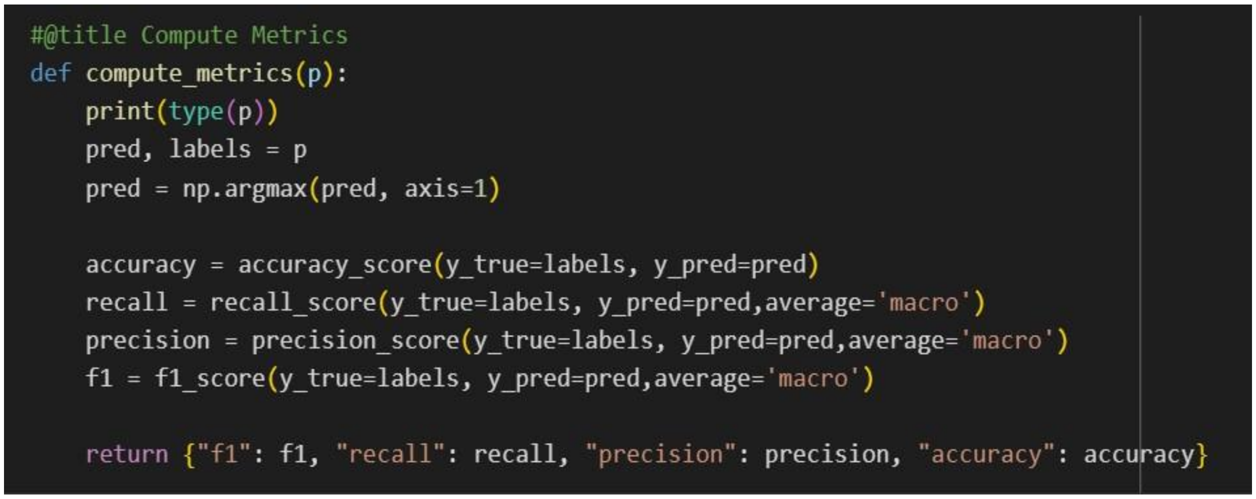


```
training_args = TrainingArguments(
    output_dir='./results',
    save_strategy = 'epoch',
    evaluation_strategy='epoch',
    num_train_epochs=num_epochs,
    per_device_train_batch_size=32,
    warmup_steps=0,
    weight_decay=0.01,
    load_best_model_at_end=True,
    metric_for_best_model='eval_loss',
    logging_dir='./logs',
    greater_is_better=False
)
```

Figure 28 Code for training arguments for topic classification

Evaluation Procedure

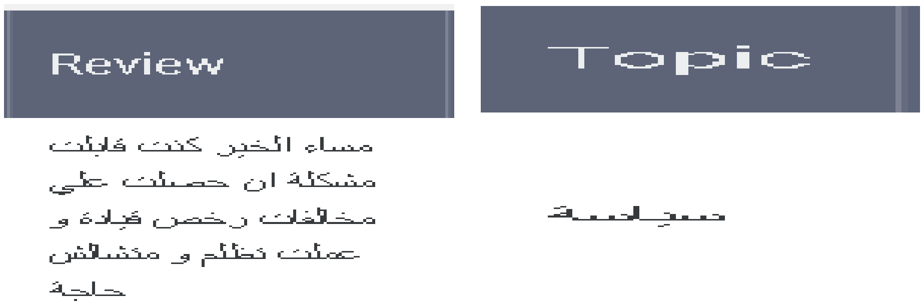
- **Evaluation Metrics**: We use metrics such as Macro F1-score, accuracy, precision, and recall can be used to assess how well the model generalizes to unseen data. We evaluate the trained model's performance using the validation set.
- We can see the code of evaluation function in Figure 29



```
#@title Compute Metrics
def compute_metrics(p):
    print(type(p))
    pred, labels = p
    pred = np.argmax(pred, axis=1)

    accuracy = accuracy_score(y_true=labels, y_pred=pred)
    recall = recall_score(y_true=labels, y_pred=pred,average='macro')
    precision = precision_score(y_true=labels, y_pred=pred,average='macro')
    f1 = f1_score(y_true=labels, y_pred=pred,average='macro')

    return {"f1": f1, "recall": recall, "precision": precision, "accuracy": accuracy}
```

Figure 29 Code for evaluation function for topic classification

We can find a sample input and output of topic classification module in Figure 30



Review	Topic
مساء الخير كنت قابلت مشكلة ان حصلت علي مخالفات رخص قيادة و عملت تظلم و متشالش حاجة	سياسة

Figure 30 Sample Input & output of topic classification module

5.5 Sentiment Analysis Module

Models Used:

1.Marbert model was used provided by UBC\NLP [1]

2.Arabert model provided by AUB\mind lab [2]

Both of which employ a multi-layered Transformer consisting of multiple encoder layers, each layer contains self-attention mechanisms and feed-forward neural networks.
And after extensive experiments with both models the best results was provided with Arabert model.

Arabert, like other language models, requires preprocessed data for training and fine-tuning. The preprocessing steps for Arabert involve tokenization, subword segmentation, and the addition of special tokens. Here's an overview of how Arabert preprocesses data:

- **Tokenization:** The first step in preprocessing Arabic data for Arabert is tokenization. Tokenization involves splitting the text into individual tokens, such as words, sub words, or characters. Tokenization is necessary to create input sequences that can be processed by the model. Arabert typically uses a tokenizer specifically designed for Arabic, which takes into account the unique characteristics of the language.

- **Sub word Segmentation:** Arabic words often exhibit complex morphology, with different forms and variations. To handle this, Arabert models often employ sub word segmentation. Sub word segmentation breaks down words into smaller meaningful units, such as root letters, prefixes, suffixes, and diacritics. This helps the model handle morphological variations and improves its ability to capture the meaning of Arabic words.

- **Special Tokens:** Arabert introduces special tokens to mark the boundaries of the input sequence and provide additional information to the model. These special tokens include:

    - **CLS:** The classification token, which is added at the beginning of the input sequence. It helps the model understand that the input is meant for a classification task.
    - **SEP**: The separator token, which is inserted between two consecutive sentences or segments of the input.

    - **PAD**: The padding token, used to pad the input sequences to a fixed length for efficient batch processing.
    - **MASK**: The mask token, which is used during pre-training to represent masked or missing tokens that the model tries to predict.

- **Input Formatting:** Arabert models expect a specific input format. The preprocessed input consists of tokenized and sub word segmented sequences, with the special tokens added. The input is often represented as token IDs or embedding indices that correspond to the vocabulary of the model.

These preprocessing steps help prepare the Arabic data for training or fine-tuning Arabert models. The preprocessed data is then fed into the model during the training or inference phase to learn contextualized representations or perform NLP tasks in Arabic. The specific implementation details may vary depending on the Arabert variant and the preprocessing tools used.

- When fine-tuning Arabert for sentiment analysis, several functions are commonly used to optimize the model's performance and adapt it to the specific task. Here is a description of the functions typically employed during the fine-tuning process for sentiment analysis with Arabert:

    - **Data Preparation:** This function involves preparing the sentiment analysis dataset for training. It includes tasks such as data cleaning, preprocessing, and tokenization. The text data is transformed into a format suitable for input to the model, with labels indicating the sentiment category (positive, negative, neutral, etc.) associated with each example.

    - **Model Architecture:** Arabert utilizes a transformer-based architecture, such as BERT, as its foundation. The model architecture function initializes the Arabert model, configuring its layers, attention mechanisms, and parameters. It sets up the basic structure and configuration of the model for sentiment analysis.

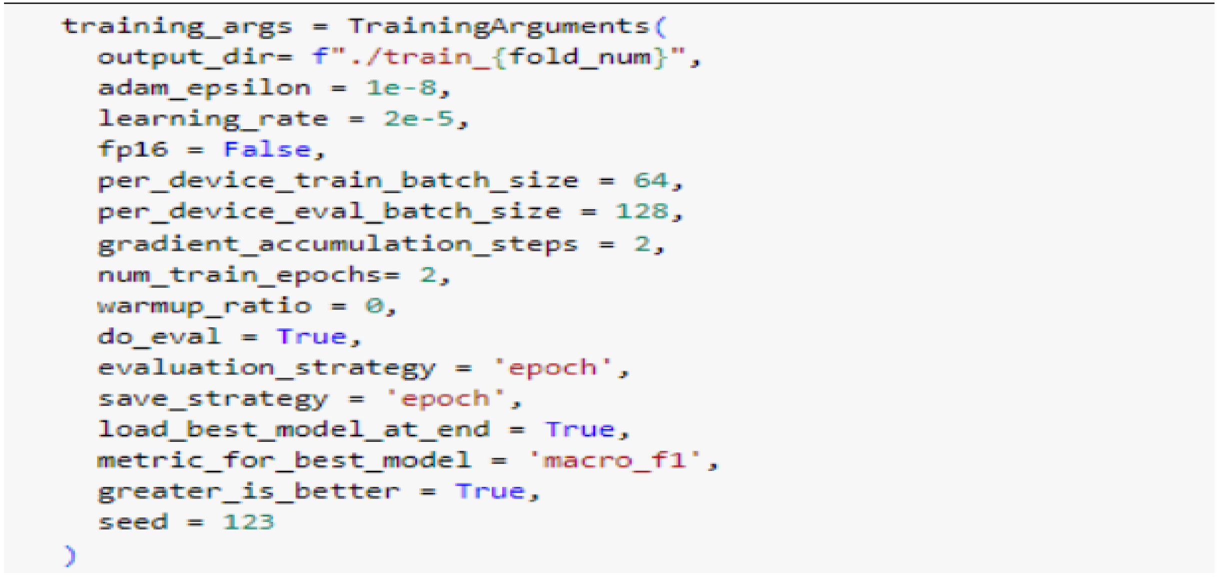
    - **Loss Function:** The loss function measures the discrepancy between the predicted sentiment and the true sentiment labels in the training data. we have used loss functions for sentiment analysis like categorical cross-entropy or binary cross-entropy. The loss function guides the model's learning process by providing feedback on its performance.

- **Optimization Algorithm:** we have used stochastic gradient descent (SGD) algorithm and its variants, such as Adam or AdaGrad. The optimization algorithm adjusts the model's parameters based on the gradients calculated from the loss function, aiming to minimize the loss and improve the model's performance.

- **Training Loop:** The training loop function iterates over the training data, feeding it to the model and optimizing the model's parameters. It consists of steps such as forward propagation, loss calculation, backward propagation (gradient computation), parameter updates, and repetition of these steps for a specified number of epochs. The training loop ensures that the model learns from the data and gradually improves its sentiment analysis capabilities.

- **Evaluation Metrics:** we have used evaluation metrics for sentiment analysis such as accuracy, precision, recall, F1 score.

- **Hyperparameter Tuning**: we have experimented with various Hyperparameters such as learning rate, batch size, number of layers, dropout rate, and others. These values significantly impact the model's training dynamics and overall performance.

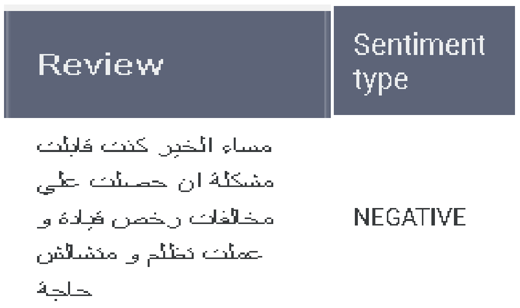


```
training_args = TrainingArguments(
    output_dir= f"./train_{fold_num}",
    adam_epsilon = 1e-8,
    learning_rate = 2e-5,
    fp16 = False,
    per_device_train_batch_size = 64,
    per_device_eval_batch_size = 128,
    gradient_accumulation_steps = 2,
    num_train_epochs= 2,
    warmup_ratio = 0,
    do_eval = True,
    evaluation_strategy = 'epoch',
    save_strategy = 'epoch',
    load_best_model_at_end = True,
    metric_for_best_model = 'macro_f1',
    greater_is_better = True,
    seed = 123
)
```

Figure 31 Training arguments for sentiment analysis model

Figure 31 represents the training arguments for best model in sentiment analysis.

We can find a sample input and output of sentiment analysis module in Figure 32



Review	Sentiment type
مساء الخير كنت قابلت مشكلة ان حصلت علي مخالفات رخص قيادة و عملت تظلم و متشالش حاجة	NEGATIVE

Figure 32 Sample input & output of sentiment analysis module

5.6 Recommended Solution generation Module

After the input review text has been through the text preprocessing pipeline, it is ready to be moved through this module to produce the governmental sector and recommended solution for the problem issued by the user.

In this module we have been through two NLP techniques which are:

5.6.1 Text Generation

Models Used

1. **Google\Mt5 [3]:** a multilingual sequence-to-sequence transformer that is pretrained on very big multilingual data from which is Arabic.

2. **UBC\NLP Arat5 [4]:** a t5 based transformer that is pretrained only on Arabic language using very large Arabic datasets.

3. **Aubmind\lab Aragpt2 [5] :** a gpt based transformer for sequence-to-sequence generation but pretrained on Arabic language.

Data Preparation

- The preprocessed input text is then prepared for training by getting tokenized using the model's tokenizer, then we extract the feature using the transformer model representation (using attention mask) to be able to be passed to the architecture of the model.

Training Procedure

- **Fine Tuning:** We have fine-tuned the sequence-to-sequence transformer models on our dataset to apply transfer learning from the pretrained model on our task and domain.

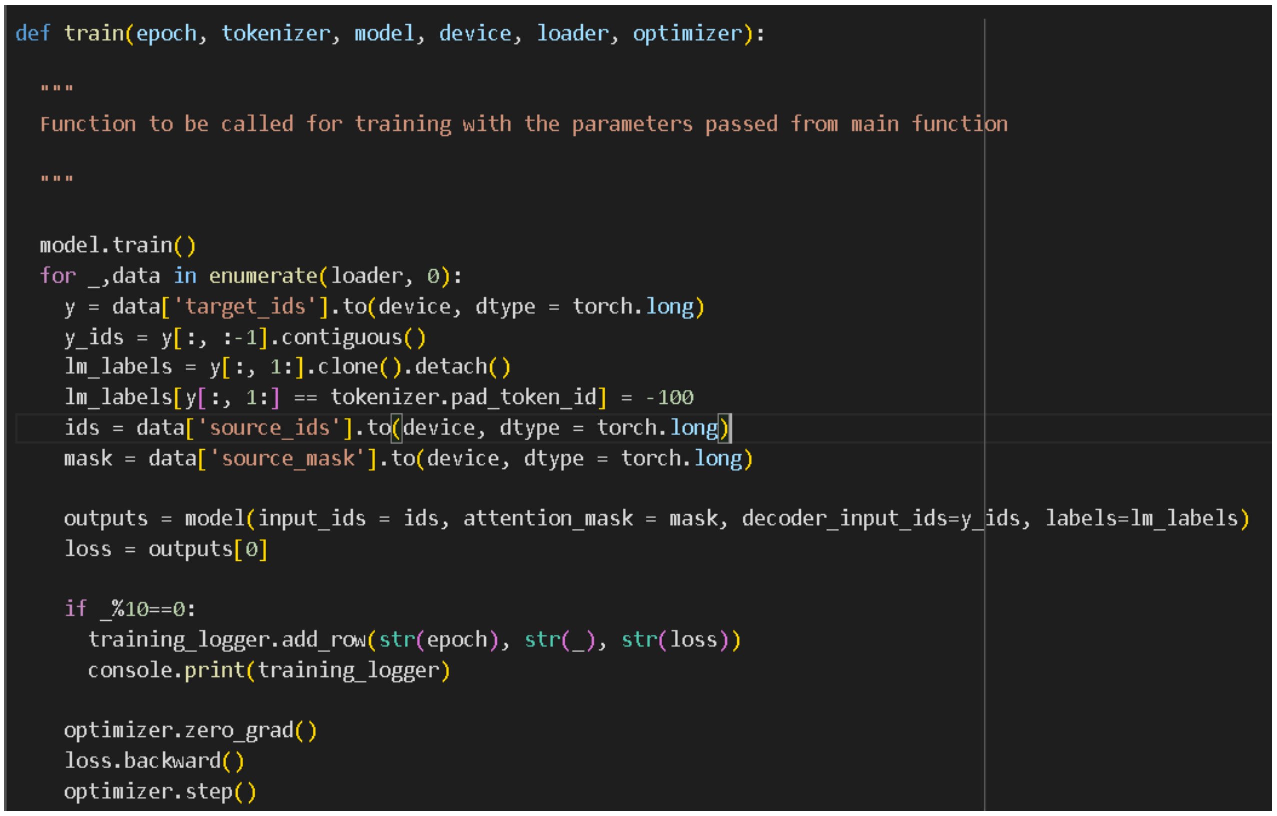
- **Training loop:** We used the transformer Trainer library and trained the model using PyTorch, we can find snippet of training function code in Figure 33

- **Loss Function:** Cross-entropy loss was used in our model as it is the most famous loss function in sequence-to-sequence generation as it proved the best results.

- **Optimization Algorithm:** we have used stochastic gradient descent (SGD) algorithm and its variants, such as Adam or AdaGrad. The optimization algorithm adjusts the model's parameters based on the gradients calculated from the loss function, aiming to minimize the loss and improve the model's performance.

Evaluation Procedure

- **Evaluation Metrics:** we used BLEU score for evaluation as it is a metric that measures the similarity between a machine-generated text and one or more human-generated reference texts, based on the number of overlapping n-grams (sequences of adjacent words) between the machine-generated text and the reference texts.



```
def train(epoch, tokenizer, model, device, loader, optimizer):

  """
  Function to be called for training with the parameters passed from main function

  """

  model.train()
  for _,data in enumerate(loader, 0):
    y = data['target_ids'].to(device, dtype = torch.long)
    y_ids = y[:, :-1].contiguous()
    lm_labels = y[:, 1:].clone().detach()
    lm_labels[y[:, 1:] == tokenizer.pad_token_id] = -100
    ids = data['source_ids'].to(device, dtype = torch.long)
    mask = data['source_mask'].to(device, dtype = torch.long)

    outputs = model(input_ids = ids, attention_mask = mask, decoder_input_ids=y_ids, labels=lm_labels)
    loss = outputs[0]

    if _%10==0:
      training_logger.add_row(str(epoch), str(_), str(loss))
      console.print(training_logger)

    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

Figure 33 Code for train function of text generation

5.6.2 Information Retrieval

After trying the text generation techniques, we found that it didn't get the desired result and it was outside the domain of our project, we just wanted to prove the concept, then we studied the Information Retrieval techniques, and it gets the best results.

Techniques used

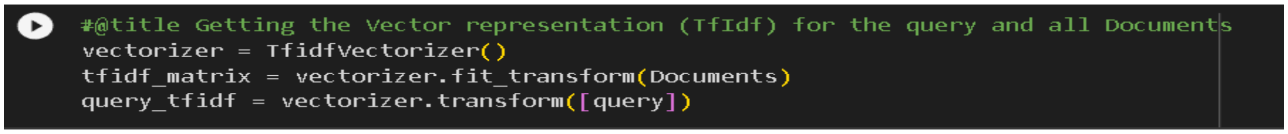
1. Tf-idf representation
2. Sentence cosine similarity

Algorithm

- **Feature Extraction:** we get the preprocessed text and apply feature extraction technique like tf-idf, bag of words, transformer models embeddings and others but the best was tf-idf due to the lack of data that prevent us from using deep learning, and the power of tf-idf in representing the frequency of important words in text, this procedure is done on the documents in database and the query of user, this process code is shown in Figure 34

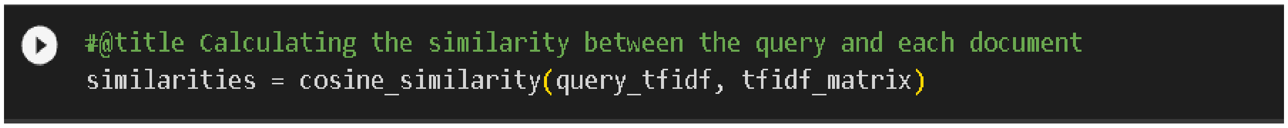
- **Similarity check:** here we check the similarity between the query and each document in database, we used cosine similarity from sklearn library as it is the most used metric in this task and proved the best results, this process code is shown in Figure 35

- **Ranking: then** we rank the documents retrieved by the query by their scores and get only the top one score document, this process code is shown in Figure 36



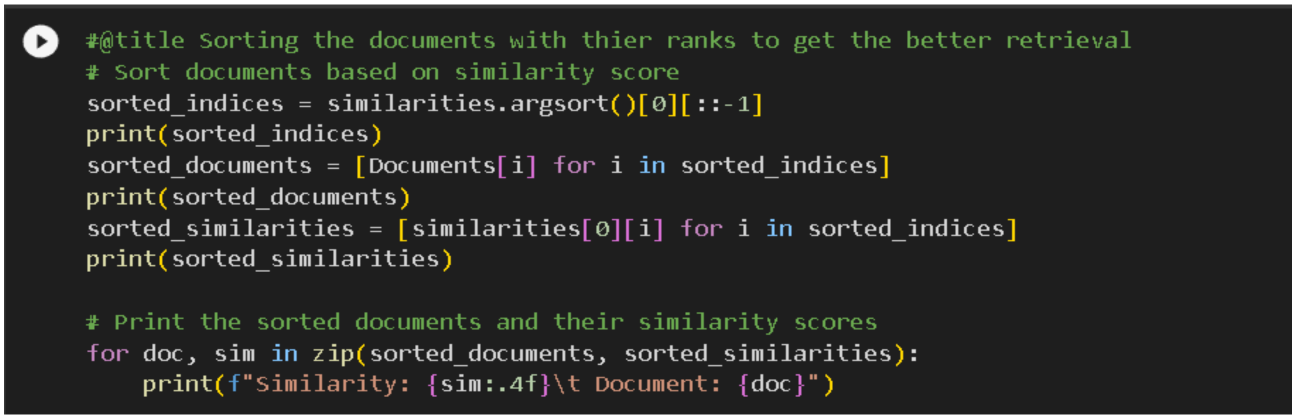
```
#@title Getting the Vector representation (TfIdf) for the query and all Documents
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(Documents)
query_tfidf = vectorizer.transform([query])
```

Figure 34 Code for tf-idf vectorization in information retrieval task



```
#@title Calculating the similarity between the query and each document
similarities = cosine_similarity(query_tfidf, tfidf_matrix)
```

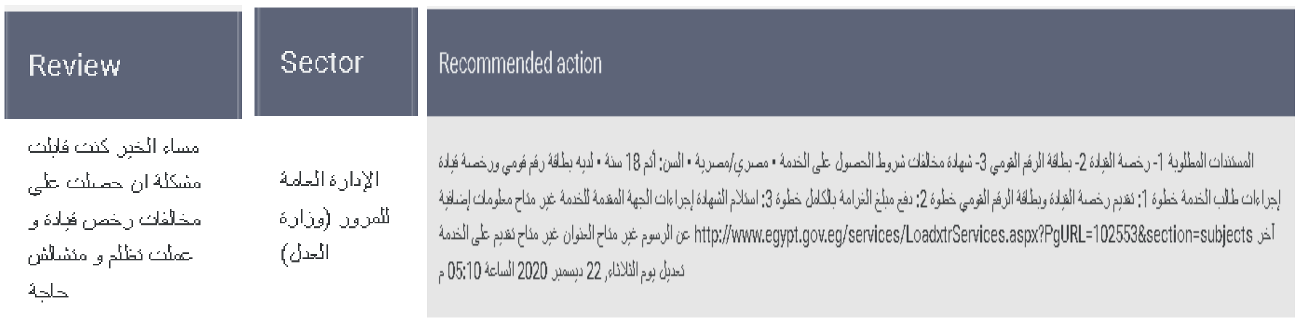
Figure 35 Code for Cosine similarity in information retrieval task



```
#@title Sorting the documents with thier ranks to get the better retrieval
# Sort documents based on similarity score
sorted_indices = similarities.argsort()[0][::-1]
print(sorted_indices)
sorted_documents = [Documents[i] for i in sorted_indices]
print(sorted_documents)
sorted_similarities = [similarities[0][i] for i in sorted_indices]
print(sorted_similarities)

# Print the sorted documents and their similarity scores
for doc, sim in zip(sorted_documents, sorted_similarities):
    print(f"Similarity: {sim:.4f}\t Document: {doc}")
```

Figure 36 Code for ranking documents in information retrieval task

We can see sample input and output of generating recommended solution module in



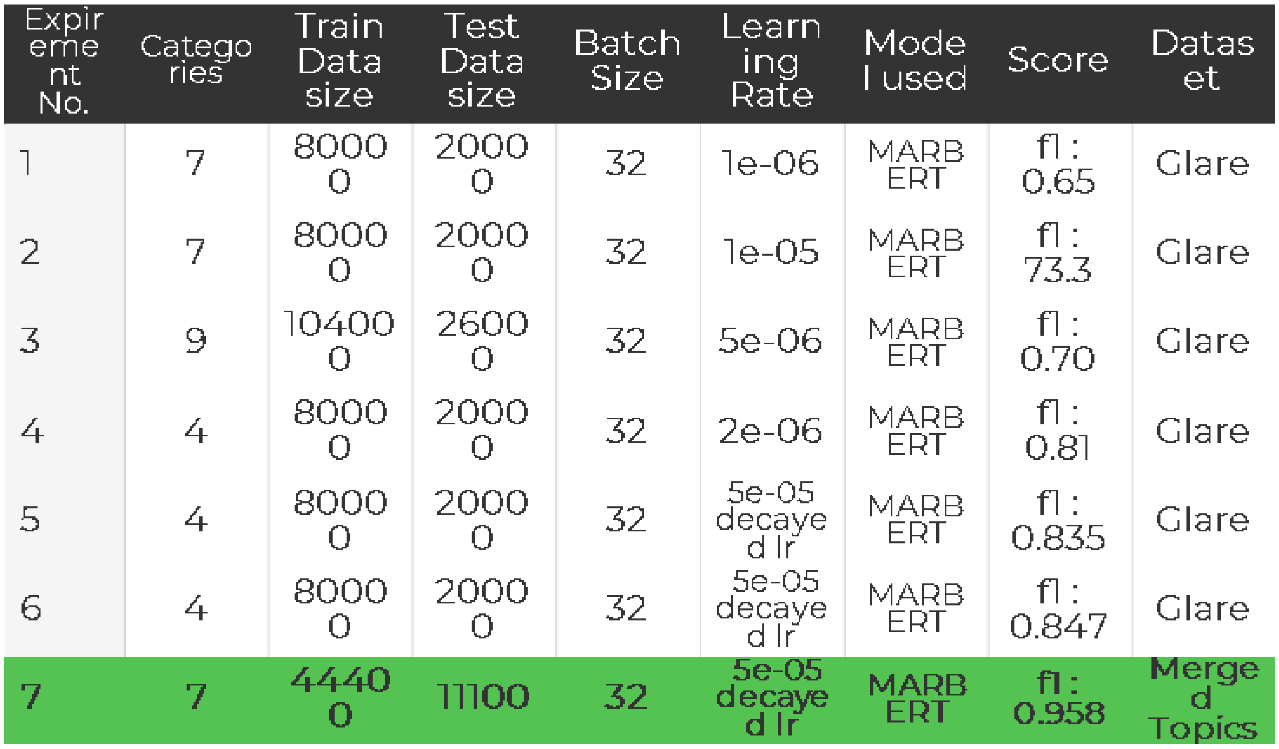
Review	Sector	Recommended action
مساء الخير كنت فاتلت مشكلة ان حصلت علي مخالفات رخص قيادة و عملت تظلم و متشالش حاجة	الإدارة العامة للمرور (وزارة العدل)	المستندات المطلوبة 1- رخصة القيادة 2- بطاقة الرقم القومي 3- شهادة مخالفات شروط الحصول على الخدمة • مصري/مصرية • السن: أتم 18 سنة • لديه بطاقة رقم قومي ورخصة قيادة إجراءات طالب الخدمة خطوة 1: تقديم رخصة القيادة وبطاقة الرقم القومي خطوة 2: دفع مبلغ الغرامة بالكامل خطوة 3: استلام الشهادة إجراءات الجهة المقدمة للخدمة غير متاح معلومات إضافية عن الرسوم غير متاح العنوان غير متاح تقديم على الخدمة http://www.egypt.gov.eg/services/LoadxtrServices.aspx?PgURL=102553&section=subjects آخر تعديل يوم الثلاثاء, 22 ديسمبر 2020 الساعة 05:10 م

Figure 37 Sample input & output of generating recommended solution module.

5.7 Experiments & Results

5.7.1 Topic classification

Table 4 Topic classification experiments & results

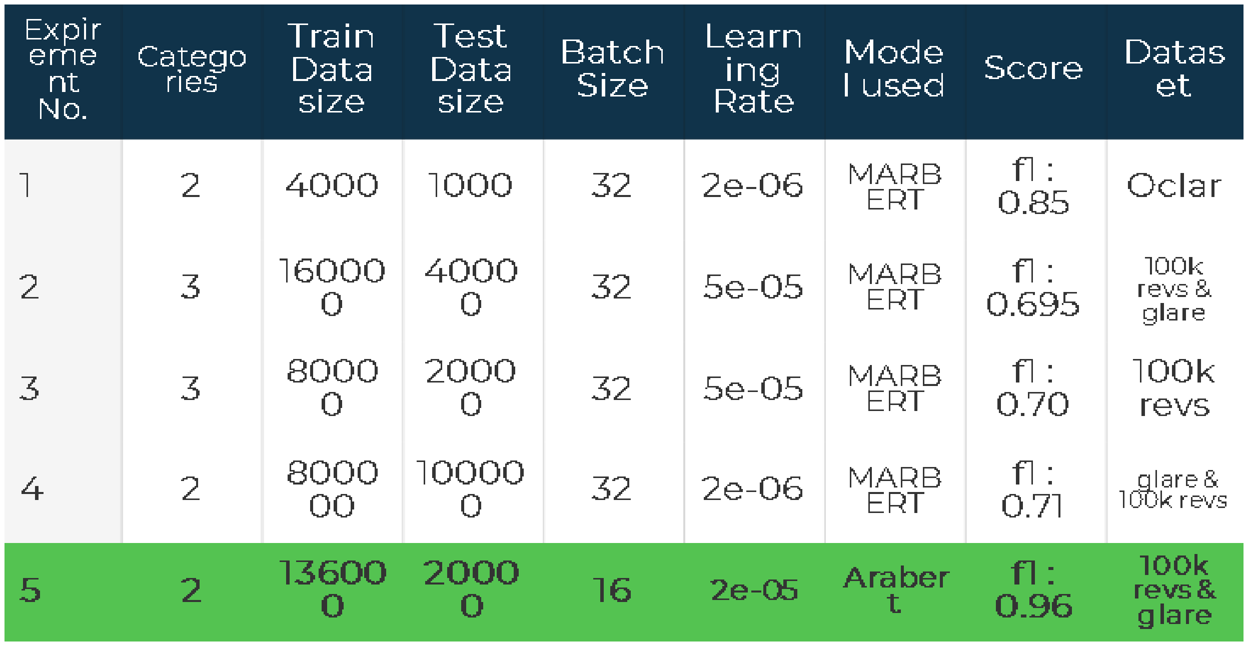


Expirement No.	Categories	Train Data size	Test Data size	Batch Size	Learning Rate	Model used	Score	Dataset
1	7	80000	20000	32	1e-06	MARBERT	f1 : 0.65	Glare
2	7	80000	20000	32	1e-05	MARBERT	f1 : 73.3	Glare
3	9	104000	26000	32	5e-06	MARBERT	f1 : 0.70	Glare
4	4	80000	20000	32	2e-06	MARBERT	f1 : 0.81	Glare
5	4	80000	20000	32	5e-05 decayed lr	MARBERT	f1 : 0.835	Glare
6	4	80000	20000	32	5e-05 decayed lr	MARBERT	f1 : 0.847	Glare
7	7	44400	11100	32	5e-05 decayed lr	MARBERT	f1 : 0.958	Merged Topics

After large number of experiments using different models, applying different text preprocessing, and applying hyperparameter tuning we concluded that experiment 7 using marbert model and preprocessed Merged topics dataset gives the best f1 score as shown in Table 4

5.7.2 Sentiment Analysis

Table 5 Sentiment analysis experiments & results

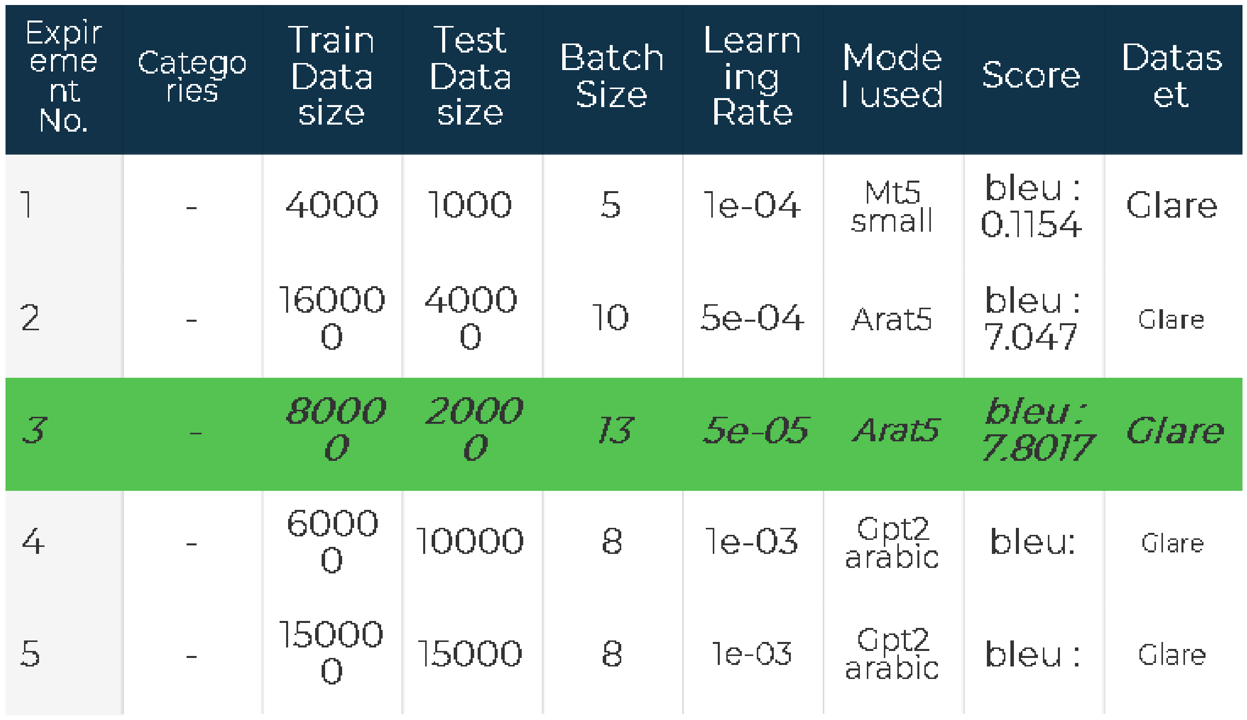


Expirement No.	Categories	Train Data size	Test Data size	Batch Size	Learning Rate	Model used	Score	Dataset
1	2	4000	1000	32	2e-06	MARBERT	f1 : 0.85	Oclar
2	3	160000	40000	32	5e-05	MARBERT	f1 : 0.695	100k revs & glare
3	3	80000	20000	32	5e-05	MARBERT	f1 : 0.70	100k revs
4	2	800000	100000	32	2e-06	MARBERT	f1 : 0.71	glare & 100k revs
5	2	136000	20000	16	2e-05	Arabert	f1 : 0.96	100k revs & glare

After large number of experiments using different models, applying different text preprocessing and applying hyperparameter tuning we concluded that experiment 5 using arabert model and preprocessed combination between 100k reviews dataset and glare dataset, gives the best f1 score as shown in Table 5

5.7.3 Text Generation

Table 6 Text generation experiments & results

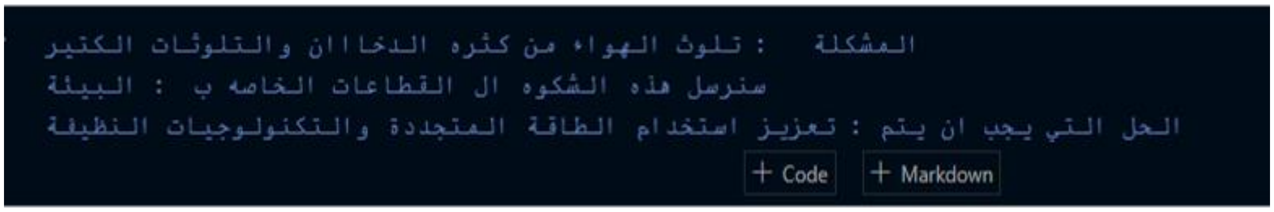


Expirement No.	Categories	Train Data size	Test Data size	Batch Size	Learning Rate	Model used	Score	Dataset
1	-	4000	1000	5	1e-04	Mt5 small	bleu : 0.1154	Glare
2	-	160000	40000	10	5e-04	Arat5	bleu : 7.047	Glare
3	*-*	*80000*	*20000*	*13*	*5e-05*	*Arat5*	*bleu : 7.8017*	*Glare*
4	-	60000	10000	8	1e-03	Gpt2 arabic	bleu:	Glare
5	-	150000	15000	8	1e-03	Gpt2 arabic	bleu :	Glare

After large number of experiments using different models, applying different text preprocessing and applying hyperparameter tuning we concluded that experiment 3 using arat5 model and preprocessed glare dataset, gives the best bleu score as shown in Table 6

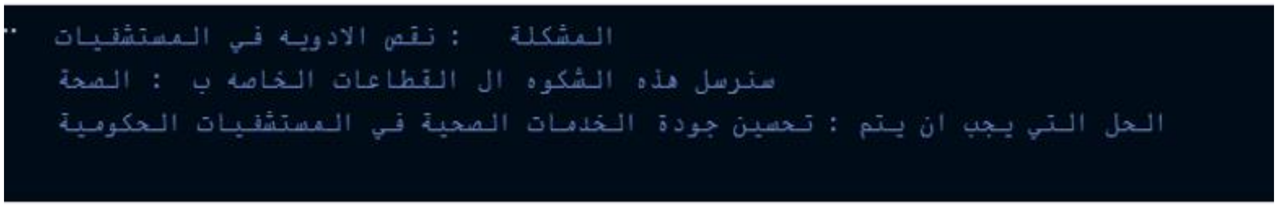
5.7.4 Information Retrieval

Here are some outputs from our algorithm shown in Figure 38 Figure 39 Figure 40 Figure 41



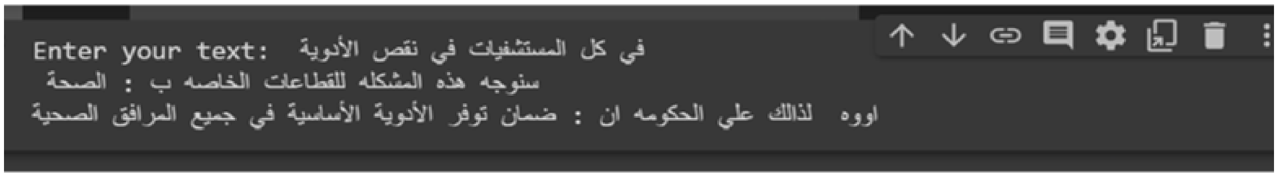
```
المشكلة   : تلوث الهواء من كثره الدخاا ان والتلوثات الكتير
سنرسل هذه الشكوه ال القطاعات الخاصه ب  : البيئة
الحل التي يجب ان يتم : تعزيز استخدام الطاقة المتجددة والتكنولوجيات النظيفة
```

Figure 38 IR example 1



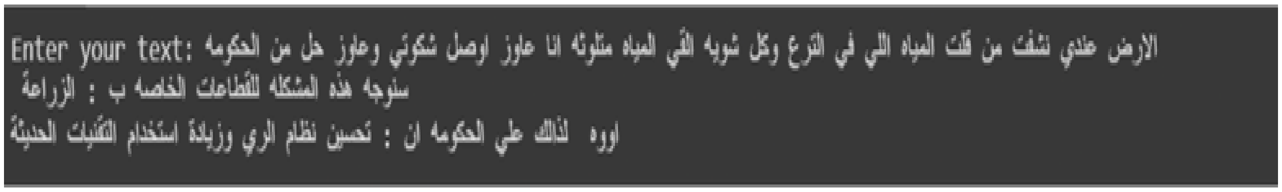
```
المشكلة   : نقص الادويه في المستشفيات
سنرسل هذه الشكوه ال القطاعات الخاصه ب  : الصحة
الحل التي يجب ان يتم : تحسين جودة الخدمات الصحية في المستشفيات الحكومية
```

Figure 39 IR example 2



```
Enter your text:  في كل المستشفيات في نقص الأدوية
 سنوجه هذه المشكله للقطاعات الخاصه ب : الصحة
اووه  لذالك على الحكومه ان : ضمان توفر الأدوية الأساسية في جميع المرافق الصحية
```

Figure 40 IR example 3



```
Enter your text: الارض عندي نشفت من قلت المياه اللي في الترع وكل شويه القي المياه متلوثه انا عاوز اوصل شكوتي وعاوز حل من الحكومه
 سنوجه هذه المشكله للقطاعات الخاصه ب : الزراعة
اووه  لذالك على الحكومه ان : تحسين نظام الري وزيادة استخدام التقنيات الحديثة
```

Figure 41 IR example 4

5.8 Interface & Deployment

Framework used

- **Django Framework:** Django is a high-level Python web framework that provides a set of tools and libraries for building fast, secure, and scalable web applications, using the Model-View-Controller (MVC) architectural pattern and a "batteries-included" philosophy that includes features such as an Object-Relational Mapping (ORM) system, form handling, authentication, and templating.

Programming Languages

- **Python:** Python is a high-level, interpreted programming language that emphasizes code readability, simplicity, and flexibility, and is widely used in data analysis, scientific computing, web development, artificial intelligence, and many other fields.

- **Html:** is a standard markup language used for creating web pages and web applications, consisting of a set of tags and attributes that define the structure, content, and appearance of a web document.
- **CSS:** is a stylesheet language used for describing the presentation of a web page or web application written in HTML, including layout, colors, fonts, and other visual aspects, and enabling the separation of content and design.

Deployment

- **We deployed our website and models locally.**

6- User Manual

6.1 Requirements

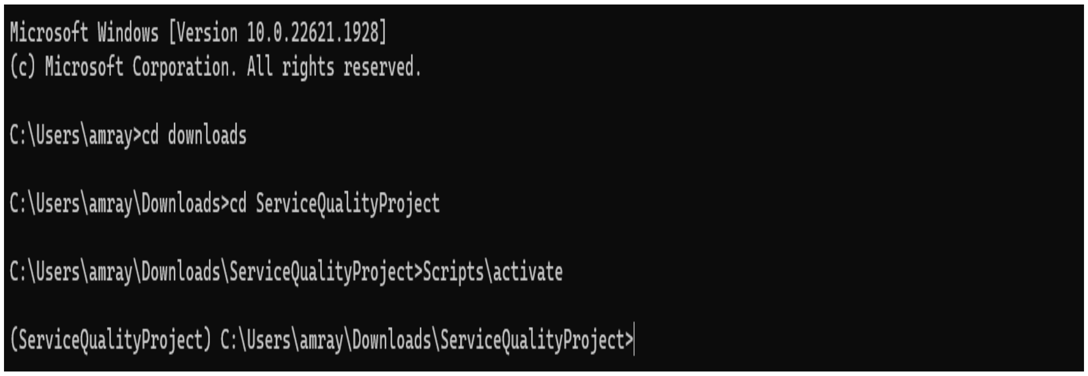
To be able to use our project user must preinstall the following dependencies:

- asgiref==3.7.2
- certifi==2023.5.7
- charset-normalizer==3.1.0
- click==8.1.3
- colorama==0.4.6
- Django==4.2.2
- django-suit @ https://github.com/darklow/django-suit/tarball/v2#sha256=b40c916de9528ca86bcccd42d1fd54e7773e5071d2bfd9efc576039c1a441c0b
- docopt==0.6.2
- filelock==3.12.2
- fsspec==2023.6.0
- huggingface-hub==0.15.1
- idna==3.4
- Jinja2==3.1.2
- joblib==1.2.0
- MarkupSafe==2.1.3
- mpmath==1.3.0
- networkx==3.1
- nltk==3.8.1
- numpy==1.25.0
- packaging==23.1
- pandas==2.0.2
- pipreqs==0.4.13
- python-dateutil==2.8.2
- pytz==2023.3
- PyYAML==6.0
- regex==2023.6.3
- requests==2.31.0
- safetensors==0.3.1

- scikit-learn==1.2.2
- scipy==1.10.1
- six==1.16.0
- sqlparse==0.4.4
- sympy==1.12
- threadpoolctl==3.1.0
- tokenizers==0.13.3
- torch==2.0.1
- tqdm==4.65.0
- transformers==4.30.2
- typing_extensions==4.6.3
- tzdata==2023.3
- urllib3==2.0.3
- yarg==0.1.9

6.2 Steps to use the project

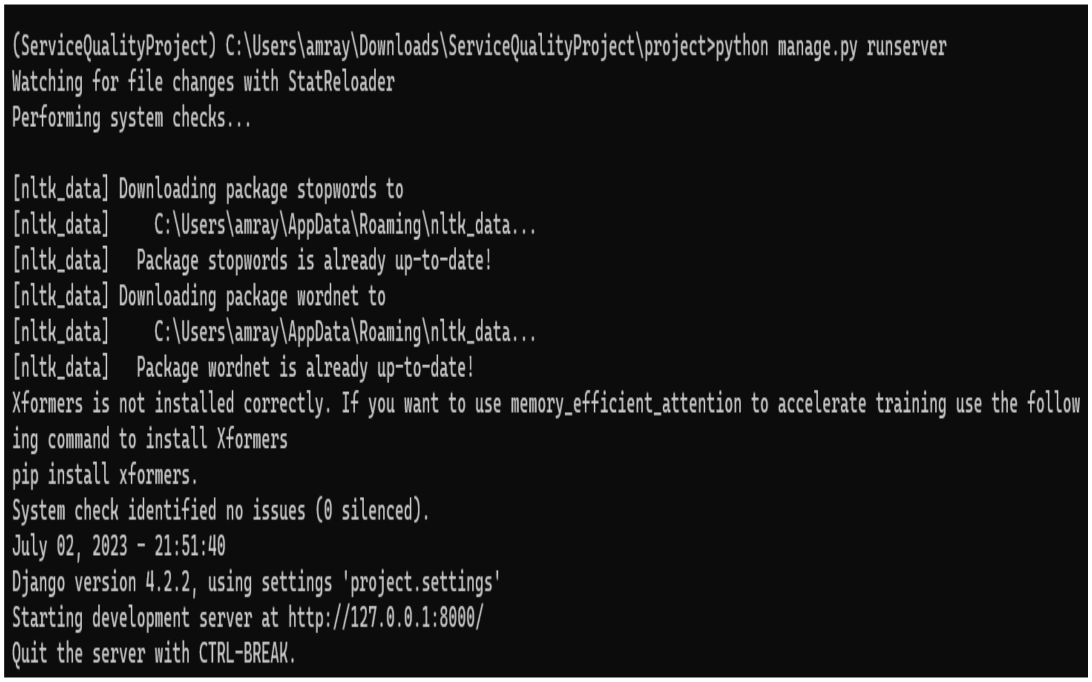
1. Install the required dependencies.
2. Activate the virtual environment as shown in Figure 42



```
Microsoft Windows [Version 10.0.22621.1928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\amray>cd downloads

C:\Users\amray\Downloads>cd ServiceQualityProject

C:\Users\amray\Downloads\ServiceQualityProject>Scripts\activate

(ServiceQualityProject) C:\Users\amray\Downloads\ServiceQualityProject>
```

Figure 42 Activating the virtual environment.

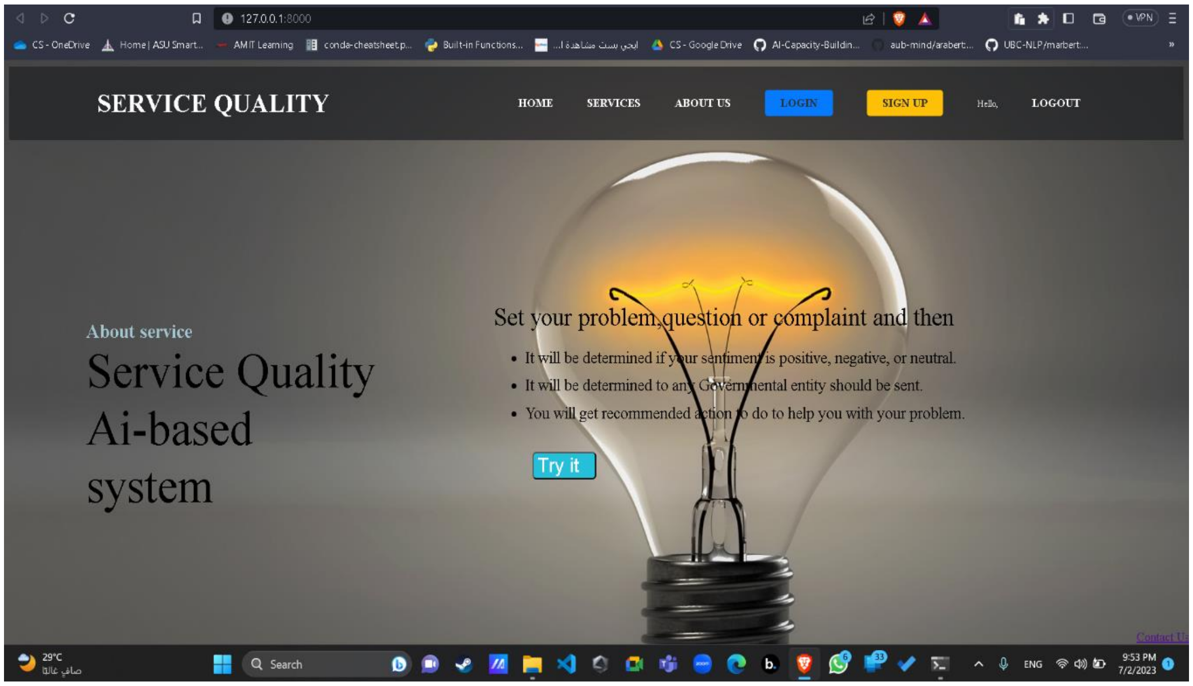
3. Run the server of the website as shown in Figure 43



```
(ServiceQualityProject) C:\Users\amray\Downloads\ServiceQualityProject\project>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\amray\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\amray\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
Xformers is not installed correctly. If you want to use memory_efficient_attention to accelerate training use the follow
ing command to install Xformers
pip install xformers.
System check identified no issues (0 silenced).
July 02, 2023 - 21:51:40
Django version 4.2.2, using settings 'project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Figure 43 Running the server.

4. Open the link of the server as shown in Figure 44



About service

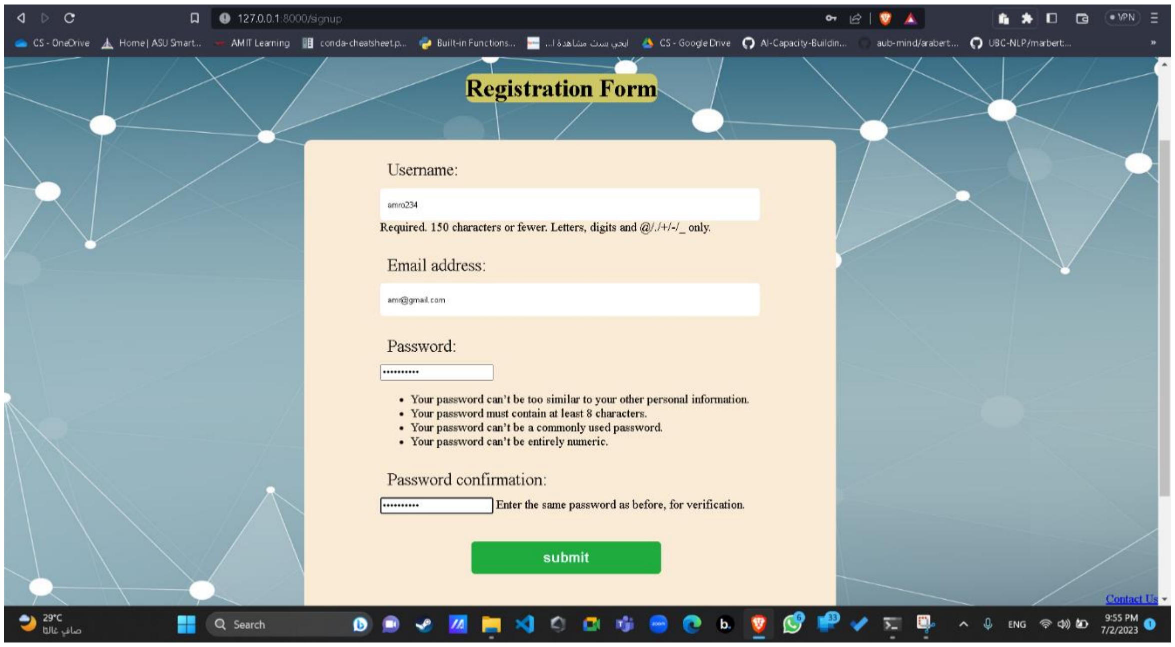
Service Quality Ai-based system

Set your problem,question or complaint and then

- It will be determined if your sentiment is positive, negative, or neutral.
- It will be determined to any Governmental entity should be sent.
- You will get recommended action to do to help you with your problem.

Figure 44 Website landing page.

5. The user must sign up with his information to be able to use the service as shown in Figure 45:



Registration Form

Username:

amro234

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email address:

amr@gmail.com

Password:

••••••••••

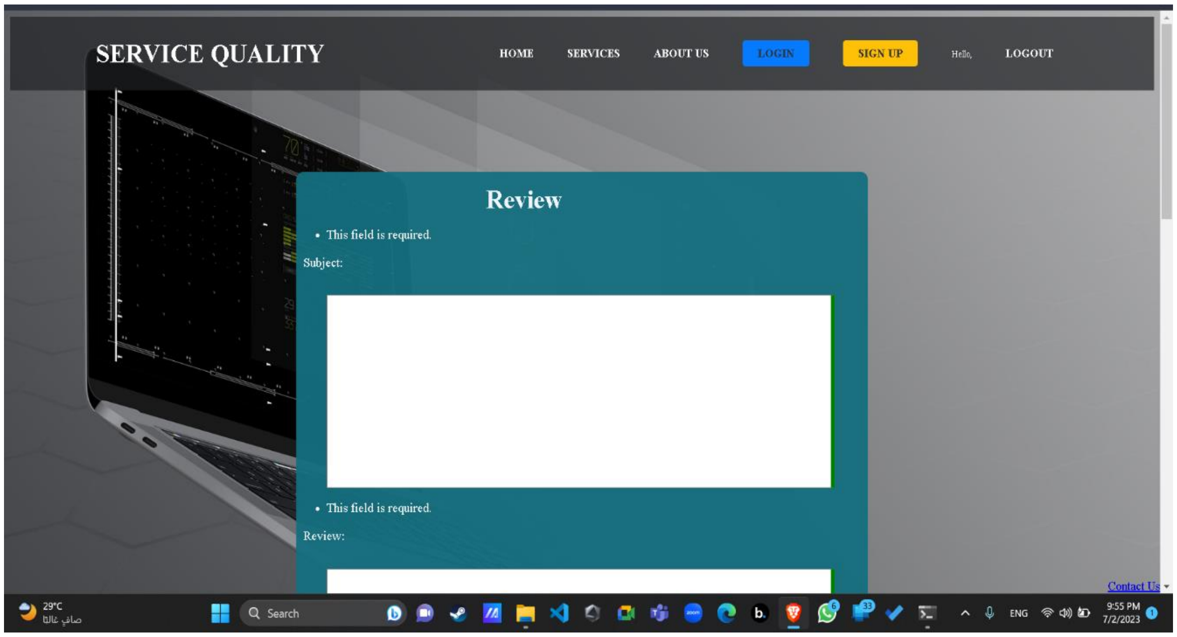
- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:

•••••••••• Enter the same password as before, for verification.

Figure 45 Sign Up page.

6. Then the user can go to the service page and write his metadata and his review as shown in Figure 46 Figure 47 Figure 49 Figure 48:



Review

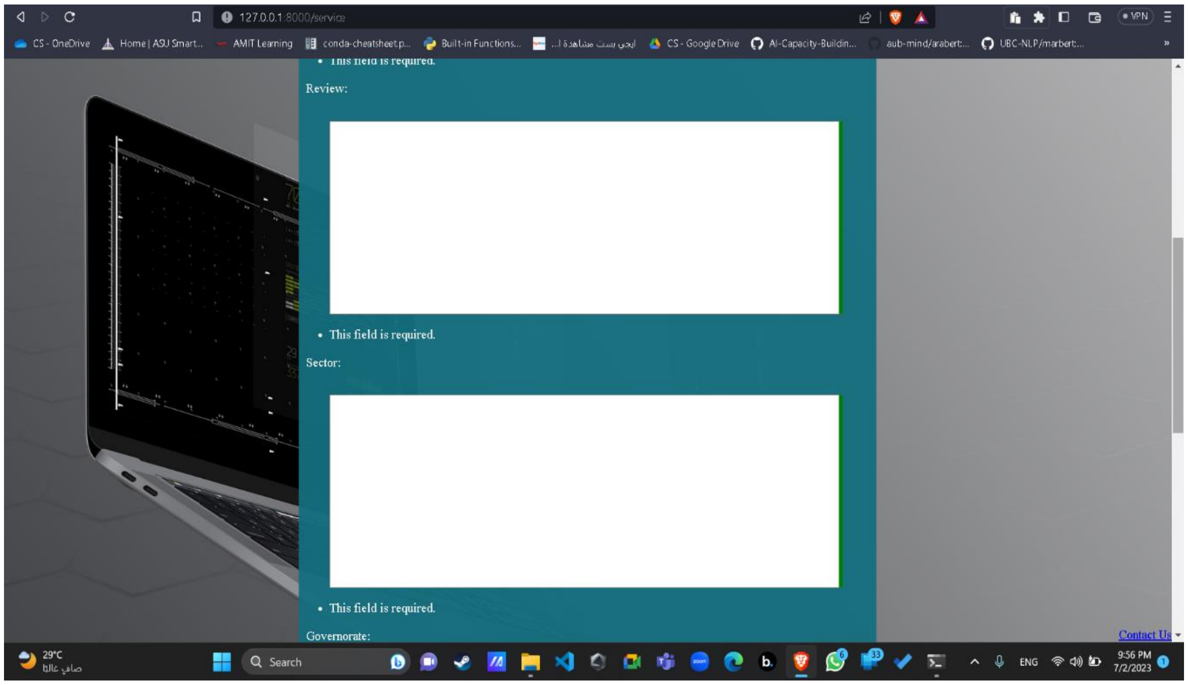
- This field is required.

Subject:

- This field is required.

Review:

Figure 46 Service page 1



- This field is required.

Review:

- This field is required.

Sector:

- This field is required.

Governorate:

Figure 47 Service page 2

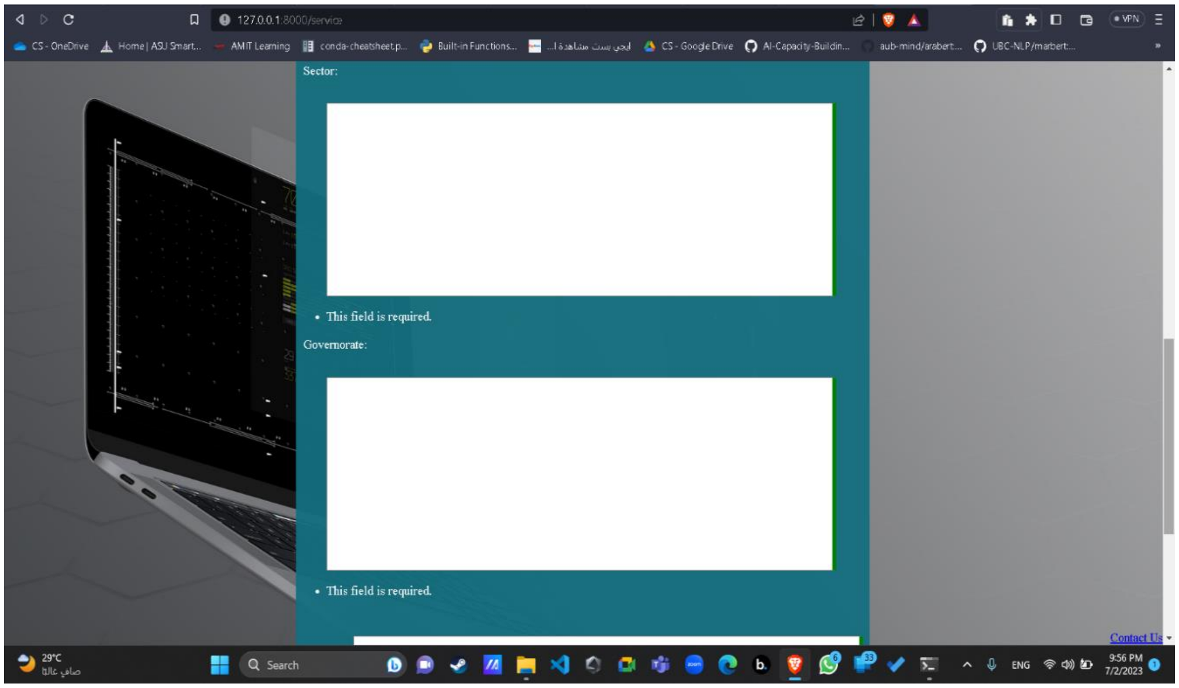


Figure 49 Service page 3

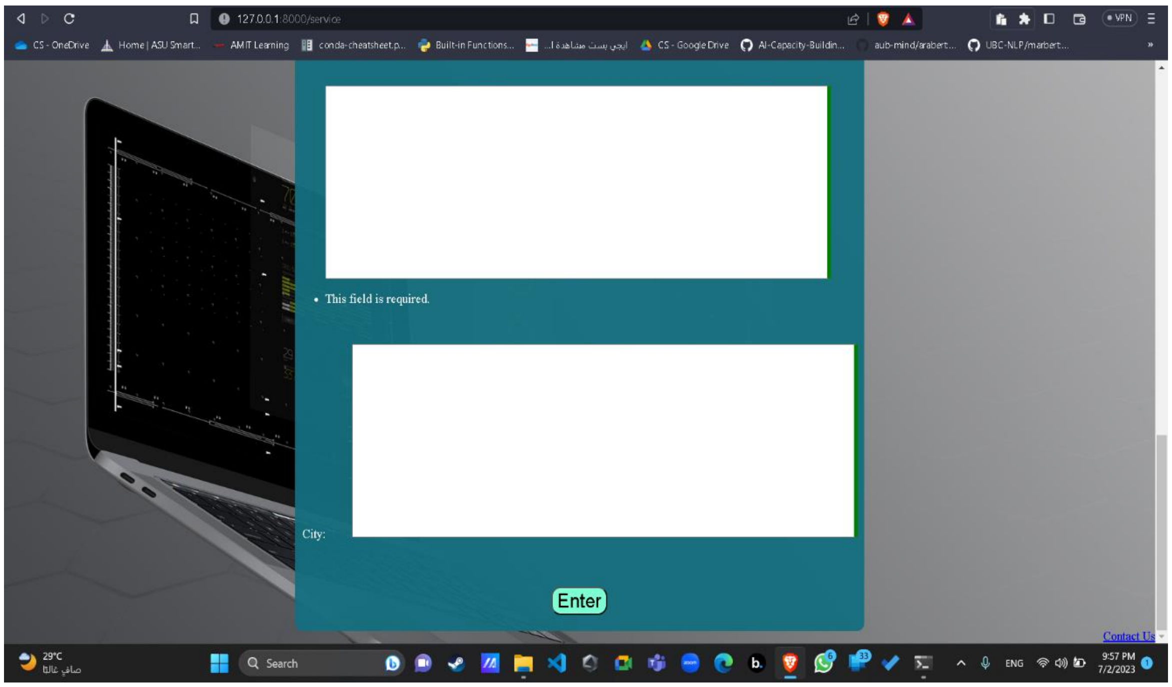
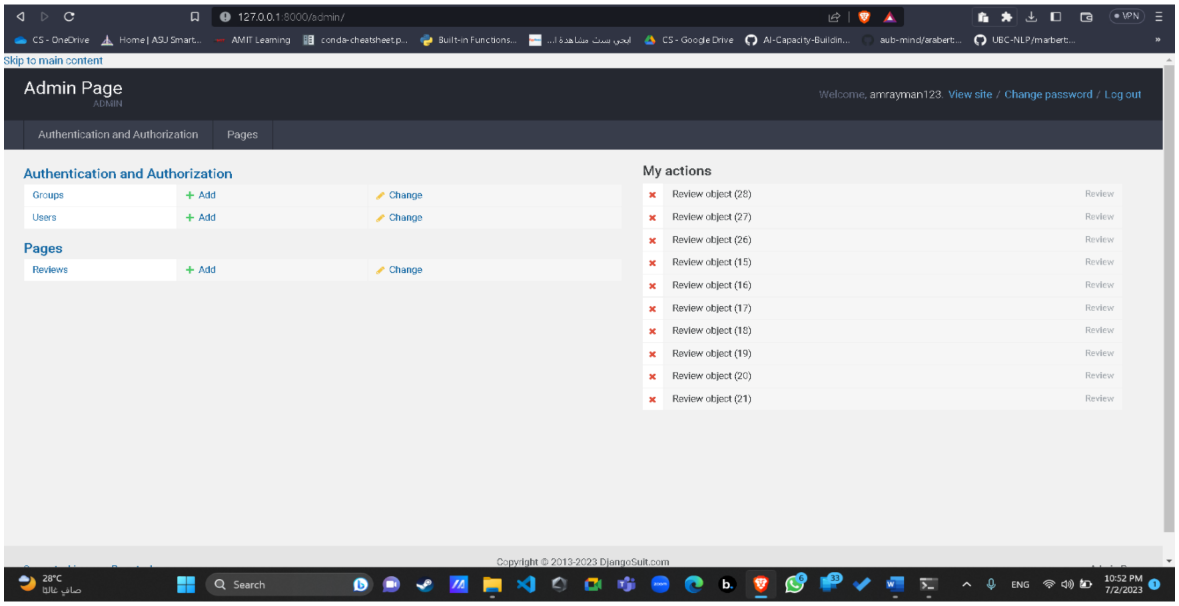


Figure 48 Service page 4

7. The review and metadata are gone through the pipelines and then stored with the output of the models in database, then the admin can view them and decide what to do , as shown in Figure 50 and Figure 51:



127.0.0.1:8000/admin/

Skip to main content

Admin Page
ADMIN

Welcome, amrayman123. View site / Change password / Log out

Authentication and Authorization | Pages

Authentication and Authorization

Groups	+ Add	Change
Users	+ Add	Change

Pages

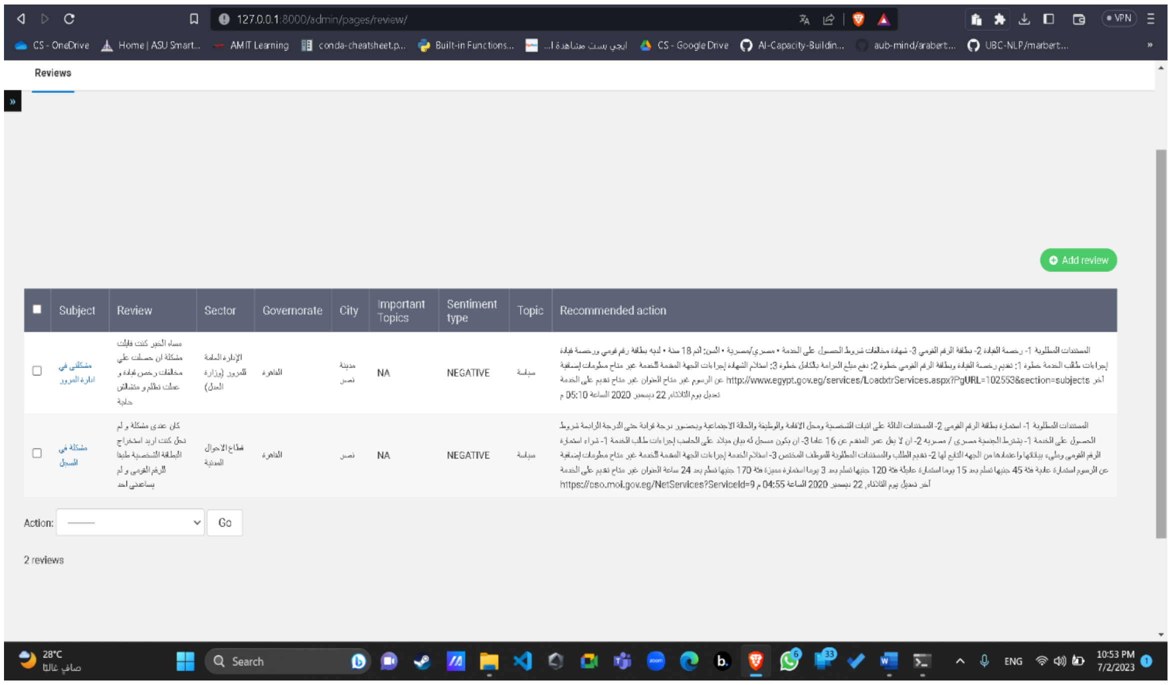
Reviews	+ Add	Change

My actions

✗	Review object (28)	Review
✗	Review object (27)	Review
✗	Review object (26)	Review
✗	Review object (15)	Review
✗	Review object (16)	Review
✗	Review object (17)	Review
✗	Review object (18)	Review
✗	Review object (19)	Review
✗	Review object (20)	Review
✗	Review object (21)	Review

Copyright © 2013-2023 DjangoSuit.com

Figure 50 Admin page 1



127.0.0.1:8000/admin/pages/review/

Reviews

Add review

	Subject	Review	Sector	Governorate	City	Important Topics	Sentiment type	Topic	Recommended action
☐	مشكلتي في ادارة المرور	مساء الخير كنت قابلت مشكلة ان حصلت علي مخالفات رخص قيادة و عملت تظلم و متشالش حاجة	الإدارة العامة للمرور (وزارة العدل)	القاهرة	مدينة نصر	NA	NEGATIVE	سياسة	المستندات المطلوبة 1- رخصة القيادة 2- بطاقة الرقم القومي 3- شهادة مخالفات شروط الحصول على الخدمة • مصري/مصرية • السن: أتم 18 سنة • لديه بطاقة رقم قومي ورخصة قيادة إجراءات طالب الخدمة خطوة 1: تقديم رخصة القيادة وبطاقة الرقم القومي خطوة 2: دفع مبلغ الغرامة بالكامل خطوة 3: استلام الشهادة إجراءات الجهة المقدمة للخدمة غير متاح معلومات إضافية عن الرسوم غير متاح العنوان غير متاح تقديم على الخدمة http://www.egypt.gov.eg/services/LoadxtrServices.aspx?PgURL=102553&section=subjects آخر تعديل يوم الثلاثاء, 22 ديسمبر 2020 الساعة 05:10 م
☐	مشكلة في السجل	كان عندى مشكلة و لم تحل كنت اريد استخراج البطاقة الشخصية طبقا للرقم القومى و لم يساعدني احد	قطاع الاحوال المدنية	القاهرة	نصر	NA	NEGATIVE	سياسة	المستندات المطلوبة 1- استمارة بطاقة الرقم القومى 2- المستندات الدالة على اثبات الشخصية ومحل الاقامة والوظيفة والحالة الاجتماعية وبحضور درجة قرابة حتى الدرجة الرابعة شروط الحصول على الخدمة 1- يشترط الجنسية مصرى / مصريه 2- ان لا يقل عمر المتقدم عن 16 عاما 3- ان يكون مسجل له بيان ميلاد على الحاسب إجراءات طالب الخدمة 1- شراء استمارة الرقم القومى وملىء بياناتها واعتمادها من الجهه التابع لها 2- تقديم الطلب والمستندات المطلوبة للموظف المختص 3- استلام الخدمة إجراءات الجهة المقدمة للخدمة غير متاح معلومات إضافية عن الرسوم استمارة عادية فئة 45 جنيها تسلم بعد 15 يوما استمارة عاجلة فئة 120 جنيها تسلم بعد 3 يوما استمارة مميزة فئة 170 جنيها تسلم بعد 24 ساعة العنوان غير متاح تقديم على الخدمة https://cso.moi.gov.eg/NetServices?ServiceId=9 آخر تعديل يوم الثلاثاء, 22 ديسمبر 2020 الساعة 04:55 م

Action: ——— Go

2 reviews

Figure 51Admin page 2

7- Conclusion and Future Work

7.1 Conclusion

The Smart Government Service Quality Project is an important initiative aimed at improving the quality of government services through the use of smart technologies. By leveraging data analytics, artificial intelligence, and other advanced technologies, this project can help governments to identify and address service delivery bottlenecks, reduce bureaucracy, enhance transparency, and increase citizen satisfaction. With its focus on continuous improvement and innovation, this project has the potential to transform the way governments deliver services, making them more efficient, effective, and responsive to the needs of citizens.

The project can be able to automatically analyze the text reviews of citizens and get the topic(ministry) of the review, the sentiment of citizen writing the review, the governmental sector responsible for the review content and recommend solutions for the problems in the review.

In this project we reached best scores in topic classification with **0.957** f1 score, in sentiment analysis with **0.96** f1 score, in generating the solution and sector for the review with reasonable solutions coming from the database of known services for the government.

7.2 Future Work

Our project can be improved by getting real time data from different sectors of government to be able to train the models well on the reliable data, the interface and models can be deployed in any of the cloud services to be more efficient and have high performance.

Our project can be improved also by providing suitable processing power to make the operation in less time, be more accurate and have more operations in specific time.

References

[1] Abdul-Mageed, M., Elmadany, A. and Nagoudi, E.M.B., 2020. ARBERT & MARBERT: deep bidirectional transformers for Arabic. arXiv preprint arXiv:2101.01785.

[2] Antoun, W., Baly, F. and Hajj, H., 2020. Arabert: Transformer-based model for arabic language understanding. arXiv preprint arXiv:2003.00104.

[3] Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A. and Raffel, C., 2020. mT5: A massively multilingual pre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934.

[4] Nagoudi, E.M.B., Elmadany, A. and Abdul-Mageed, M., 2021. AraT5: Text-to-text transformers for Arabic language generation. arXiv preprint arXiv:2109.12068.

[5] Antoun, W., Baly, F. and Hajj, H., 2020. AraGPT2: Pre-trained transformer for Arabic language generation. arXiv preprint arXiv:2012.15520.

[6] AlGhamdi, F., Mohammed, R. and Alowisheq13, H.A.K.A., GLARE: Google Apps Arabic Reviews Dataset.

[7] Omari, M.A., 2022. OCLAR: logistic regression optimization for Arabic customers' reviews. International Journal of Business Intelligence and Data Mining, 20(3), pp.251-273.

[8] Kobayashi, M. and Takeda, K., 2000. Information retrieval on the web. ACM computing surveys (CSUR), 32(2), pp.144-173.

[9] Al-Ayyoub, M., Khamaiseh, A.A., Jararweh, Y. and Al-Kabi, M.N., 2019. A comprehensive survey of arabic sentiment analysis. Information processing & management, 56(2), pp.320-342.

[10] Alruily, M., 2021. Classification of arabic tweets: A review. Electronics, 10(10), p.1143.

[11] Zakraoui, J., AlJa'am, J.M. and Salah, I., 2022, December. Domain-Specific Text Generation for Arabic Text Summarization. In 2022 International Conference on Computer and Applications (ICCA) (pp. 1-4). IEEE.

[12] Ameen, A., Alfalasi, K., Gazem, N.A. and Isaac, O., 2019, December. Impact of system quality, information quality, and service quality on actual usage of smart government. In 2019 first international conference of intelligent computing and engineering (ICOICE) (pp. 1-6). IEEE.

[13] Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[14] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V., 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.