Ain-Shams University
Faculty of Computer Science and Information Sciences
Scientific Computing Department

Diagnosing Diseases using Deep Learning Techniques

This documentation submitted as required for the degree of bachelor's in computer and Information Sciences.

By

Fady Makram Megar	[Scientific Computing Department]
Manar Alaa Azouz	[Scientific Computing Department]
Malak Ismail Mohamed	[Scientific Computing Department]
Manar Mohamed Mahdy	[Scientific Computing Department]
Mennattallh Ibrahim Kamal	[Scientific Computing Department]
Manar Ibrahim Hadhoud	[Scientific Computing Department]

Under Supervision of

Dr. Dina El-Sayad
Lecturer of Scientific Computing Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

TA. Heba Gamal
Teaching Assistant of Scientific Computing Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Cairo
July 2024

Acknowledgements

This project is the result of the cooperation of multiple individuals, without whom this system would not exist today.

All praise and thanks to ALLAH, who provided me with the ability to complete this work. I hope to accept this work from me.

I am grateful to my parents and my family who are always providing help and support throughout the whole years of study. I hope I can give that back to them.

I also offer my sincerest gratitude to my supervisors, Prof. Dr. Dina El-Sayad and T.A. Radwa El-Hussieny who have supported me throughout my thesis with their patience, knowledge, and experience.

Finally, I would like to thank my friends and all the people who gave me support and encouragement.

Abstract

The COVID-19 pandemic has limited daily activities and even contact between patients and primary care providers. This makes it more difficult to provide adequate primary care services, which include connecting patients to an appropriate healthcare service, particularly remote and automated healthcare consultations, have gained increased attention, and medical bots, which provide medical advice, are becoming increasingly popular. They offer many benefits, including 24/7 access to medical counseling, reduced appointment waits times by providing quick answers to common questions or concerns, and cost savings or tests required for diagnosis and treatment plans. The success of medical bots depends on the quality of their learning, which in turn depends on the appropriate corpus within the domain of interest.

Arabic is one of the most commonly used languages for sharing users' internet content. However, implementing medical bots in Arabic faces several challenges, including the language's morphological composition, the diversity of dialects, and the need for an appropriate and large enough corpus in the medical domain. To address this gap, this document introduces the largest Arabic Healthcare Q &A dataset, called MAQA, consisting of over 430,000 questions distributed across 20 medical specializations. We applied preprocessing techniques like drop duplicate, remove advertisements or inappropriate links, remove any diacritics, remove unnecessary repeated characters, and Standardizing letters. We split data to 80%, 20% for training and testing. Furthermore, this document adopts three deep learning models (LSTM, Bi-LSTM, and Transformer. The experimental results demonstrate that the recent Transformer model outperforms the traditional deep learning models, achieving an average cosine similarity of 83%.

أدت جائحة كوفيد-19 الى الحد من الأنشطة اليومية والتواصل بين المرضى ومقدمي الرعاية الصحية الأولية، مما جعل من الصعب توفير خدمات الرعاية الصحية الأولية الكافية. وقد تزايد الاهتمام بالاستشارات الصحية عن بُعد والآلية، بما في ذلك الروبوتات الطبية التي تقدم المشورة الطبية والتي باتت أكثر شيوعًا. وتقدم هذه الروبوتات مزايا عديدة مثل إمكانية الوصول إليها على مدار الساعة، وتقليل أوقات انتظار المواعيد، والوفورات في التكاليف.

يواجه تطبيق الروبوتات الطبية باللغة العربية تحديات عدة، بما في ذلك التركيب الصرفي للغة والتنوع اللهجي، إضافة إلى الحاجة إلى قاعدة بيانات طبية كبيرة وملائمة. ولمعالجة هذه الفجوة، يُقدِّم هذا المستند أكبر مجموعة بيانات أسئلة وأجوبة طبية باللغة العربية، MAQAوالتي تتكون من أكثر من 430,000 سؤال موزعة على 20 تخصصًا طبيًا. وقد تم تطبيق تقنيات معالجة مسبقة على البيانات مثل إزالة النسخ المكررة والإعلانات والروابط غير المناسبة والشكل الموحد للحروف. كما تم تقسيم البيانات إلى 80% للتدريب و20% للاختبار. وأخيرًا، تم اختبار ثلاثة نماذج تعلم عميق (LSTM وBi-LSTMو Transformer) على مجموعة بيانات MAQA، وأظهرت النتائج التجريبية أن نموذج Transformer تفوق على نماذج التعلم العميق التقليدية، حيث حقق متوسط تشابه جيب قدره 83%

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI:	Artificial intelligence
AIML:	Artificial intelligence markup language
AUC:	Area Under the Receiver
BERT:	Bidirectional encoder representations from transformers
BI-LSTM:	Bidirectional long short-term memory
BLEU:	Bilingual evaluation understudy
CBOW:	Continuous bag of words
CNTK:	Computational network toolkit
GPT:	Generative pre-trained transformer
GRU:	Gated recurrent units
GUI:	Graphical User Interface
HBAM:	Health-Based Allocation Model
HTTP	Hypertext Transfer Protocol
KNN:	K-nearest neighbors
LSTM:	Long short-term memory
MAQA:	Medical Arabic Question Answer
MNB:	Multinomial Naive Bayes
NLP:	Natural language processing
RELU:	Rectified linear unit
RNN:	Recurrent neural network
RoBERT:	Robustly optimized BERT approach
Seq2Seq:	Sequence Two Sequence
SGD:	Stochastic gradient descent
SVM:	Support Vector Machine
TFIDF:	Term Frequency, Inverse Document Frequency
UI:	User interface

Chapter 1
Introduction

In this chapter, we outline the motivation and objectives behind developing a medical app to address healthcare disparities. It defines the problem of limited access to healthcare facilities, highlights the consequences of misdiagnosis due to physician inexperience, and discusses the impact of the COVID-19 pandemic on healthcare systems. The chapter also provides a detailed time plan for the project.

1.1 Problem Definition:

In some regions, access to healthcare facilities is severely limited or entirely absent. This scarcity leaves communities vulnerable, as individuals lack essential medical services and resources to address their health needs. Without nearby healthcare facilities, people may delay seeking treatment for illnesses or injuries, leading to worsened health outcomes or even preventable deaths. The absence of healthcare infrastructure also exacerbates disparities in healthcare access, disproportionately affecting marginalized populations who may already face numerous barriers to receiving adequate medical care.

Misdiagnosis can have serious consequences, especially when it occurs due to a lack of experience or information on the part of the attending physician. In some cases, patients' conditions may be overlooked or incorrectly identified, leading to delays in appropriate treatment or even unnecessary interventions. When doctors lack sufficient expertise or access to comprehensive medical resources, they may struggle to accurately diagnose complex or rare conditions. This not only impacts individual patients but also erodes trust in the healthcare system.

The emergence of the COVID-19 pandemic in 2019 profoundly disrupted human interaction, leading to a significant decrease in face-to-face interactions among individuals. This shift had far-reaching consequences, particularly in healthcare settings, where hospitals became inundated with patients infected with the virus. The fear of contracting COVID-19 deterred some individuals from seeking medical attention, even for unrelated illnesses, resulting in the exacerbation of their health conditions. Tragically, this reluctance to seek timely medical care contributed to the deaths of some individuals who could have otherwise received life-saving treatment.

1.2 Motivation:

The motivation behind creating a medical app to address these challenges stems from the recognition of the immense suffering experienced by patients, particularly in cases where misdiagnosis by less experienced medical professionals exacerbates their condition. It's evident that a solution is urgently needed to not only alleviate the suffering of these individuals but also to streamline healthcare delivery and improve patient outcomes. By developing a medical app, we aim to extend a helping hand to those living in areas where healthcare is scarce or inaccessible, as well as to individuals who face difficulty accessing traditional healthcare services due to mobility issues or other constraints. Furthermore, the app can play a crucial role in minimizing person-to-person interactions, thus reducing the risk of infectious disease transmission. By leveraging technology to overcome barriers to healthcare access and enhance patient care.

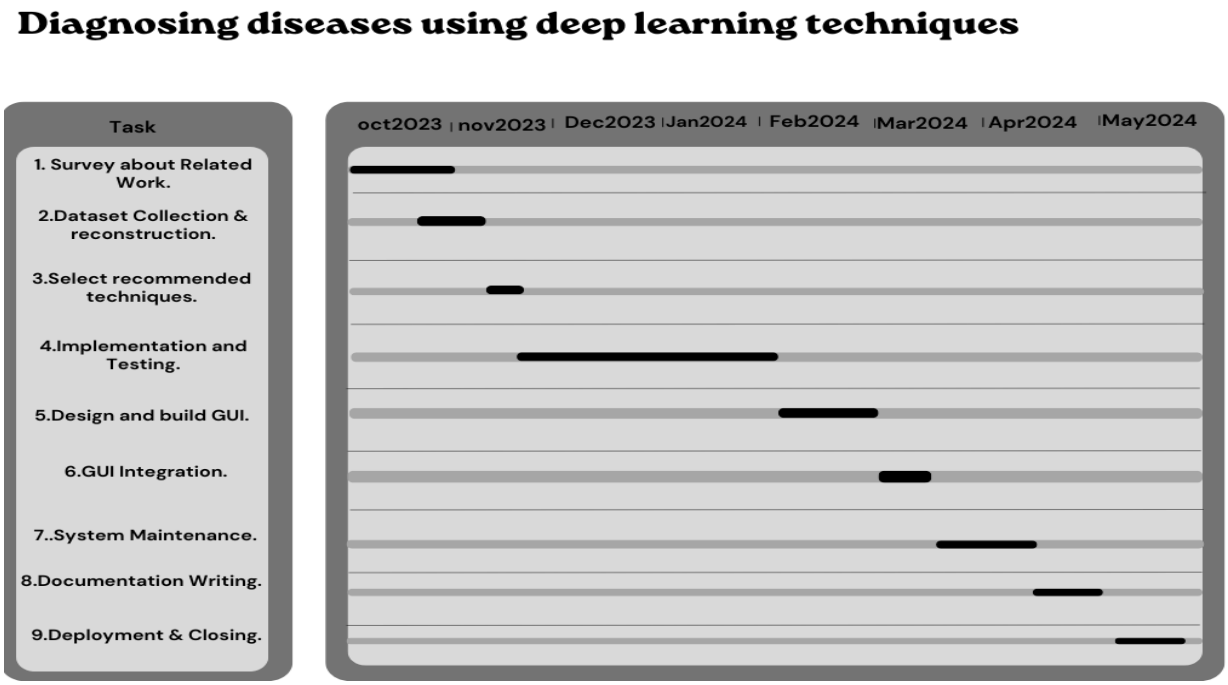
1.3 Objectives:

The primary aim of this project is to revolutionize healthcare by enhancing disease diagnosis and access to accurate medical information. By leveraging advanced technologies, the project streamlines the process of obtaining correct medical insights, thereby aiding in both disease prevention and initial diagnosis. Through reducing time and costs associated with traditional medical procedures, it aims to make healthcare more efficient and accessible to all, ultimately improving overall public health outcomes.

1.4 Time Plan

1. Survey about Related Work.
2. Dataset Collection & reconstruction.
3. Select recommended techniques.
4. Implementation and Testing.
5. Design and build GUI.
6. GUI Integration
7. System Maintenance.
8. Documentation Writing.
9. Deployment & Closing.

Figure 1.1: Time Plan



Developing an application that is contain diagnosis disease using chat bot requires careful planning and execution. Most of the following processes were made in parallel with each other to move in sync with each other. For example, each new idea in the implementation or system functionality was added to the documentation as soon as possible. Here is a time plan paragraph description that outlines the key steps and estimated timelines for building the application:

I. Reading Papers & Related Works (21days):

Conduct an extensive literature review and study existing research papers and related works on Arabic data, chatbot and medical papers. Gain insights into successful case studies, approaches, potential challenges faced in similar projects.

II. Dataset Collection & reconstruction (11days):

Collect data by search, scraping from websites and translate data of diagnosis disease, make some statistic to data to know more about data construct data in excel sheet and in same format.

III. Select recommended techniques(6days):

Define the application's architecture, data models, and system requirements based on our understanding and research. Develop a detailed experimental plan to implement chatbot with best result

IV. Implementation and Testing(55days):

Utilize the insights gained from the experimental design stage to implement the application. Initialize model for do tasks efficient. And use methods such as similarity and bleu to test it. implement new techniques to get accurate answers.

V. Design and build GUI (16days):

The development team built the GUI using appropriate tools and frameworks, ensuring it is responsive and compatible with various devices and screen sizes. This phase also includes integrating the GUI with backend services and databases to ensure seamless functionality.

VI. GUI Integration (16days):

This stage begins with a thorough review of both the GUI and backend components to ensure compatibility and readiness for integration. The development team worked on connecting the front-end interface with backend services, APIs, and databases, facilitating smooth data flow and interaction.

VII. System Maintenance(16days):

Ensuring the newly integrated system operates smoothly and efficiently over time. This phase involves a series of proactive and reactive measures designed to maintain system stability and performance. The team addressed any bugs or issues reported by users or detected through automated monitoring tools.

VIII. Documentation Writing(13days):

This phase involves creating comprehensive and clear documentation that serves various stakeholders, including developers, end-users, and maintenance teams. The team worked closely with developers and other key personnel to ensure accuracy and completeness.

IX. Deployment & Closing(5days):

The final stage of the project focused on successfully launching the system and ensuring a smooth transition to operational status. This phase involves meticulous planning, coordination, and execution to ensure all elements are in place for a successful deployment.

1.5 Thesis Outline:

- **Chapter 1 "Introduction"**

  This chapter discusses the motivation and objectives for creating a medical app to address healthcare disparities, focusing on limited access to healthcare facilities, the consequences of misdiagnosis due to physician inexperience, and the impact of the COVID-19 pandemic on healthcare systems. It also includes a detailed project timeline.

- **Chapter 2 "Literature Review"**

  Firstly, we talked about the background of the field of the project. Secondly, we presented some related work and similar systems that existed on research papers and on the web.

- **Chapter 3 "System Architecture"**

  In this chapter, we will examine the various technologies utilized in healthcare system architecture and analyze the components of those systems to construct our own healthcare system.

- **Chapter 4 "System Implementation"**

  In this chapter we included a detailed description of the code. It explains the functions in the system and what each one does specifically.

- **Chapter 5 "System Testing"**

  In this chapter we will display all experiments we made and its result, we begin by outlining the experimental setup, including the datasets, model architectures, and hyperparameter configurations,

  For each experiment, we provide detailed results.

- **Chapter 6 “Mobile Application”**

  Along with screenshots of the GUI, we illustrated how the project operates. It contains all the steps that someone must be aware of to use the system.

- **Chapter 7 “Conclusions and Future Work”**

  This chapter is about two aspects. The first aspect is the conclusions. The conclusion is a complete summary of the whole project along with the results obtained. The second aspect is the future work. This is about what can be done in the future to improve the performance of the project and what additional functions can be added.

These chapter descriptions provide a general understanding of the content covered in each chapter.

Chapter 2
Background

There have been numerous works and papers related to the application of Diagnosing diseases using deep learning techniques.

2.1 Overview

Most of papers we read during the last period used machine learning models, the most used of which were **SVM**, **KNN**, **Random Forst**, **Decision Tree**, and **MNB**, and used deep learning models, the most widely used of which were **LSTM**, **Bi-LSTM**, **Transformer**, **LSTM and GRU**, and **Rasa Framework**.

These papers differed in their results depending on the dataset used, different size, languages, and methods of preprocessing of this data, methods of implementing the model, and methods of measurement criteria.

2.2 Related Works

There have been numerous works and papers related to the application of Diagnosing diseases using deep learning techniques.

2.2.1 Traditional Papers

- "Health Consultant Bot: Primary Health Care Monitoring Chatbot for Disease Prediction" by Asad Ur Rahman et al [5]: This research paper presents a disease prediction chatbot. This system allows the user to describe their medical health condition in natural language, and by processing their natural language-based statement, system detects the symptoms, predicts the disease, and provides basic precautions as well as a brief introduction about the disease. They have used IBM Watson Assistant to build this system. Watson assistant provides several machine learning algorithms (SVM) to

process user statements and symptoms extraction. In an experimental evaluation, they carried out a study having 156 subjects, who interact with the system in a daily use scenario. Results show the effectiveness and accuracy of the system to support the patient in taking good care of their health. The final response contains the predicted disease with a brief introduction and precaution.

- "Multilingual Healthcare Chatbot Using Machine learning" by Sagar Badlani et al [6]: The proposed solution describes a multilingual healthcare chatbot application that can perform disease diagnosis based on user symptoms. The chatbot application converses with the user using concepts of Natural Language Processing and supports speech to text and text to speech conversion. Five different Machine Learning algorithms have been analyzed for disease prediction. The Random Forest Classifier produces the best results and gives an accuracy of 98.43%. The user can either enter symptoms which he is experiencing or enter some health-related queries. Depending on the user input, the chatbot will predict the disease or provide relevant information about his queries. Initially, the user must select their preferred language of communication. Currently, the system supports three languages., English, Hindi and Gujarati. The next step for the user is to select their preferred mode of communication, i.e., voice or text. The input entered by the user will first be converted into text by the system if the user is communicating via speech. The input text will be converted into English if it is in some other language. The translated input is passed on to the NLP module. The system checks if the obtained keywords correspond to a health-related user query or if they correspond to symptoms that the user is

experiencing. If the keywords are user symptoms, the system performs disease diagnosis. The trained Machine Learning model is used for this purpose. For user symptoms, there is a threshold of four symptoms for better disease prediction. If a user enters less than four symptoms, the accuracy of the prediction will be less as many diseases have common symptoms. The system then applies the TF-IDF and Cosine Similarity techniques to find the most appropriate response to the user query from the knowledge database that has been provided to it. The final response is the output is presented to the user based on their preferred mode of communication.

2.2.2 Deep-Learning Papers

- "Deep learning for Arabic healthcare: MedicalBot" by Mohammed Abdelhay et al [1]: this paper introduces medical chatbot with the largest Arabic Healthcare Q &A dataset, called MAQA, consisting of over 430,000 questions distributed across 20 medical specializations. This paper adopts three deep learning models, namely LSTM, Bi-LSTM, and Transformers, for experimenting and benchmarking the proposed corpus MAQA. The experimental results demonstrate that the recent Transformer model outperforms the traditional deep learning models, achieving an average cosine similarity of 80.81% and a Bleu score of 58%. Besides, the utilized embedding models are the PyTorch embedding, and the Aravec-WWW-CBOW at dimension 300 [9].

- "Medical Specialty Recommendations by an Artificial Intelligence Chatbot on a Smartphone: Development and Deployment" by Hyeonhoon Lee et al [7]: (AI) chatbot that classifies patients' symptoms and recommends the appropriate medical specialty could

provide a valuable solution. They collected 118,008 sentences containing information on symptoms with labels (medical specialty), conducted data cleansing, and finally constructed a pipeline of 51,134 sentences for this study. including 4 different long short-term memory (LSTM) models with or without attention and with or without a pretrained FastText embedding layer, as well as bidirectional encoder representations from transformers for NLP, were trained and validated using a randomly selected test data set. The BERT model yielded the best performance, with an AUC of 0.964 and F1 -score of 0.768, followed by LSTM model with embedding vectors, with an AUC of 0.965 and F1 -score of 0.739.

The architecture of the chatbot, Users send a sentence through the chatbot's input box. The Containerized Node js chat application includes responding logic without classifying a medical department. The GCP computing engine, which has a natural language processing model, which classifies the medical department from the sentence input. This dockerized NodeJS-based chat app is deployed by continuous git push steps.

- "DZchatbot: A Medical Assistant Chatbot in the Algerian Arabic Dialect using Seq2Seq Model" by Abdennour Boulesnane et al [2]: this paper build an automated chatbot system that interacts with people in the Arabic Algerian dialect and helps patients ask general medical questions. they have collected medical data of 2150 pairs of medical questions and answers written in the Algerian Arabic dialect. They created their chatbot based on the sequence-to-sequence model

(seq2seq) with RNN encoder and decoder. They use LSTM and GRU and get accuracy 78% and BiLSTM and achieved 71%

- "Empathy-driven Arabic Conversational Chatbot" by Tarek Naous et al [13]: A challenging aspect in developing human-like conversational models is enabling the sense of empathy in bots, making them infer emotions from the person they are interacting with. By learning to develop empathy, chatbot models can provide human-like, empathetic responses, thus making the human-machine interaction close to human-human interaction. They created an Arabic conversational dataset that comprises empathetic responses. However, the dataset is not large enough to develop very complex encoder-decoder models. A sample input in this dataset would be a statement of a speaker describing personal experience in which they felt a specific emotion. The corresponding output would be the empathetic response of a listener, which infers the emotional state of the speaker and provides an appropriate reply. Since no such dataset is available in the Arabic language, they translated the Empathetic Dialogues dataset [14], which is the only available dataset in English for building empathetic chatbots. Empathetic Dialogues consists of 24,850 English conversations obtained via crowdsourcing. They make use of the Googletrans1 API to perform the translations from English to Arabic. Therefore, we considered the dataset to be of high quality for the purpose of training the proposed empathetic conversational model. The code and dataset are publicly available at [15]

- "A Conditional Generative Chatbot using Transformer Model" by Nura Esfandiari et al [16]: NLP & Sequential Models are used to build a generative Chatbot. in this paper, a novel architecture is proposed using conditional Wasserstein Generative Adversarial Networks and a transformer model for answer generation in Chatbots. While the generator of the proposed model consists of a full transformer model to generate an answer, the discriminator includes only the encoder part of a transformer model followed by a classifier. To the best of their knowledge, this is the first time that a generative Chatbot is proposed using the embedded transformer in both generator and discriminator models. Relying on the parallel computing of the transformer model, the results of the proposed model on the Cornell Movie-Dialog corpus and the Chit-Chat datasets confirm the superiority of the proposed model compared to state-of-the-art alternatives using different evaluation metrics. Chatbot development can be classified into two categories: open and closed domain. Chatbots with the ability to answer on more than one domain, are called open domains. In contrast, closed domain Chatbots can answer only questions concerning a particular domain. The main contributions of this paper can be listed as follows:1) Model: they propose a novel model using the transformer model, which is used in both generator and discriminator of a cWGAN. To the best of their knowledge, this is the first time that such a model has been proposed in Chatbot.

  2) To tackle less accurate results in previous seq2seq model, they are using the cWGAN and a transformer model for answer generation in Chatbots.

3) Extensive experiments conducted on two challenging datasets show that our architecture outperforms state-of- the-art methods in the field.

- "Chatbot in Arabic language using seq to seq model" by Mohamed Boussakssou et al [17]: In this paper, they present midoBot: a deep learning Arabic chatbot based on the seq2seq model.They built the model and tested it in the Tensorfow2 deep learning framework using the most seq 2 seq Model architectures. they use a dataset of ~81,659 pairs of conversations created manually and without any handcrafted rules. The results obtained are significant, in most questions the chatbot was able to reproduce good answers. They applied these preprocessing techniques: Filtering Non-Arabic Content, Split the text into lines and pairs of sentences, Normalization, and Filter text by length and content.

- "HHH: An Online Medical Chatbot System based on Knowledge Graph and Hierarchical Bi-Directional Attention" by Qiming Bao et al [3]: They build HHH, an online question-and-answer (QA) Healthcare Helper system for answering complex medical questions. HHH also implements a novel text representation and similarity deep learning model, Hierarchical BiLSTM Attention Model (HBAM), to find the most similar question from a large QA dataset. They compare HBAM with other state-of-the-art language models such as bidirectional encoder representation from transformers (BERT) and Manhattan LSTM Model (MaLSTM). They train and test the models with a subset of the Quora duplicate questions dataset in the medical area. They first designed a framework to implement a generic chatbot

system. Their chatbot framework contains two main modules: - The first module is the user interface, which contains a web-based chatbot front-end, a local GUI, and a back end to handle database management. The second module serves to respond to user's queries based on our hybrid QA model, which contains a knowledge graph and the hierarchical BiLSTM attention model (HBAM). The knowledge graph stores more than 600 different kinds of disease records and can answer six different types of questions, while the HBAM can query from a big dataset containing 29287 medical questions-and-answer pairs (171 from ehealthforumQAs, 5679 from questionDoctorQAs and 23437 from webmdQAs). l. A user's question will first be queried from the knowledge graph. If it cannot find any result, a text similarity model will be used to find the answers from a large medical QA dataset. The GitHub link includes code and data [18]. The number of disease and symptom keywords in the dictionary is 668 and 2367, respectively. With the dictionary, they collect nearly 70,000 medical-related records from the Quora dataset. For training the models faster and easier, they randomly select 10,000 records (positive: negative = 1:1) as the experiment data. The Quora duplicate questions dataset is an open domain sentence pair dataset. It has more than 400,000 tagged sentence pairs formatted like "text1 text2 is_duplicate" means whether the two sentences are semantically similar. If they are semantically equal, the tag will be "1", otherwise "0". The hyperparameters of HBAM include the batch_size is 1024, the n_epoch is 9, the n_hidden is 100, the embedding_dim is 300 and the max_seq_length is 10, GoogleNews-vectors-negative300.bin.gz from Word2Vec14, the activation function is tanh. 6.2.3 Comparison. To evaluate the performance of our system, we compare it with two state-of-the-art

sentence pair similarity algorithms, namely BERT and MALSTM [14]. BERT was proposed by Google in 2018 and has refreshed records in 11 NLP tasks, including Q&A (SQuAD v1.1), reasoning (MNLI), and more. MALSTM was proposed by the MIT team in 2016 and has achieved excel lent results in calculating the similarity of sentences. It is better than a few well-known sentence similarity comparison algorithms include Dependency Tree-LSTM, ConvNet, and more. Superior performance over these two benchmarks means that their system would have achieved a level that is higher than or on par with the current state-of-the-art methods. They divide the dataset by 6:2:2 for training, validation, and testing in the BERT baseline model for fine-tuning. In other models, they use 9:1 for training and testing. It can be clearly seen that their HBAM has the best performance to check the duplication of two text sentences. Experiment 2: - they also perform a second experiment on the remaining more than 50,000 medical sentence pairs as well. We separately select thousands of tags from the three kinds of datasets: ehealthforumQAs, questionDoctorQAs, and webmdQAs, respectively. The tags of each dataset are extracted as the keywords to take the intersection with the disease symptom keyword dictionary. Then the intersection results are used to search for matching sentence pairs in the remaining 50,000 medical sentence pairs. Finally, the first 1000 matched sentence pairs are taken out for each dataset, and 10 times evaluation results are obtained, respectively.

- Prateek Mishra et al. have developed a healthcare chatbot using the Rasa Open-Source architecture for disease detection and medical advice. The model achieved an F1-score of 77.3% and an

accuracy of 78.7%, with an accuracy of 56.50% in human evaluation. The chatbot uses natural language understanding (NLU) and dialogue management to handle intent classification, entity extraction, and response retrieval. The chatbot uses policies to decide which actions to perform at each step in a conversation.

The memorization policy matches the current conversation stories in stories.yml file, predicting the next action with a confidence of 1.0. The TED Policy is a multitask architecture for next action prediction and entity recognition, involving several transformer encoders. A sequence of entity labels is predicted through a Conditional Random Field (CRF) tagging layer on top of the user sequence transformer encoder output corresponding to the input sequence of tokens.

The primary aim of this work is to create a healthcare dataset covering almost 35 diseases. The NLU pipeline defines the processing steps that convert unstructured user messages into intents and entities. The dataset is publicly available, and the preprocessing techniques used in the study are also discussed.

2.3 Comprehension Analysis

Table 2.1: Comprehension Analysis

Paper	Dataset	Models	Measurements
Deep learning for Arabic healthcare: MedicalBot [1] [2023]	MAQA Size = 434544 records	LSTM Bi-LSTM Transformers Model	Cos Similarity :56% Bleu: 31% Cos Similarity :72% Bleu: 39% Cos Similarity :80% Bleu: 58%
DZchatbot: A Medical Assistant Chatbot in the Algerian Arabic Dialect using Seq2Seq Model [2] [2022].	DZMedicaldata, Size = 2150 records	LSTM, GRU Bi-LSTM	Accuracy: 78%. Accuracy: 71%.
HHH: An Online Medical Chatbot System based on Knowledge Graph and Hierarchical Bi-Directional Attention [3] [2020]	Collected Data from medical websites Size = 29287 records	BERT MaLSTM HBAM	Accuracy: 78.2%. Accuracy: 78.4%. Accuracy: 81.2%.
Personalized Healthcare Chatbot: Dataset and Prototype System [4] [2022]	Collected Data from medical websites Size = 189 records	Rasa Architecture	Accuracy: 78.7%.

Chapter 3
System Architecture

In this chapter, we will explore the different technologies used in healthcare system architecture and analyze their components to build our own healthcare system

3.1 Preprocessing

3.1.1 Data cleaning and exploration

The complexities and characteristics of the Arabic language have made processing Arabic text, a major challenge, wherein researchers must deal with several difficulties, such as ambiguity, diglossia, and the difficulties in reading and understanding the Arabic script: the Arabic letter changes according to its position in the word, no capitalization or dedicated letter, and the complex structure of the word and morphology. Another challenge is the normalization of inconsistency in the use of certain letters, dialect words, or diacritical marks. Tackling the difficulties in NLP tools such as tokenization, stemming, and morphology analysis is yet another challenge faced by researchers and developers.

In the first step of the data cleansing process, duplicate and missing data were eliminated. Second, ambiguous sentences that were not sufficient for accurate decision, including sentences with ≤ 2 words or those not related to medical consultations, were manually excluded. Third, very few instances of mislabeled data were appropriately relabeled by a well-trained physician. Fourth, data normalization techniques were applied to standardize the text, such as converting all text to lowercase, removing punctuation, and correcting common misspellings and typographical errors.

Before selecting and working with data, it is always preferable to become acquainted with and comprehend its contents. An initial investigation of the dataset is required before employing it in research or additional analysis. It can aid in detecting trends and patterns in data, identifying outliers, and discovering valuable relationships between variables. Prior to developing the deep learning-based NLP models, exploratory data analysis (EDA) was used to extract interpretable features from the data. First, we counted the phrases linked to symptoms in each class to determine the data distribution. We also visualized the most used terms to generate word lists, such as stop words (unsuitable for classification) and keywords (suitable for decision making) for word representation. The sentence lengths (in terms of both word count and character count) were calculated to estimate the maximum length of the input sequence for each model.

3.1.2 Data Augmentation

We described Data Augmentation as a strategy to prevent overfitting via regularization. This regularization is enabled through an intuitive interface. As we study a task or dataset, we learn more about what kind of priors or what kind of additional data we need to collect to improve the system. For example, we might discover characteristics about our question answering dataset such as that it fails with symmetric consistency on comparison questions. The following list of augmentations describes the mechanisms we currently have available to inject these priors into our datasets.

1) Symbolic augmentation

We categorize these augmentations as “Symbolic Augmentations” in contrast to “Neural Augmentations”. As stated earlier, the key difference is the use of auxiliary neural networks, or other types of statistical models, to generate data compared to using symbolic rules to augment data. A key benefit of symbolic augmentation is the interpretability for the human designer. Symbolic augmentations also work better with short transformations, such as replacing words or phrases to form augmented examples. However, some information-heavy applications rely on longer inputs such as question answering or summarization. Symbolic rules are limited in applying global transformations such as augmenting entire sentences or paragraphs.

2)Rule-based augmentation

Rule-based Augmentations construct rules to form augmented examples. This entails if-else programs for augmentation and symbolic templates to insert and re-arrange existing data. Easy Data Augmentation presents four augmentations. One of the main reasons to be excited about Easy Data Augmentation is that it is relatively easy to use off-the-shelf. Many of the Augmentations mentioned later in this survey are still in the research phase, waiting for large-scale testing and adoption. Easy Data Augmentation includes random swapping, random deletion, random insertion, and random synonym replacement.

3)Back-translation augmentation

Back-translation describes translating text from one language to another and then back from the translation to the original language. An example could be taking 1,000 IMDB movie reviews in English and translating them to French and back, Chinese, and back, or Arabian and back. There has been an enormous interest in machine translation. This

has resulted in the curation of large, labeled datasets of parallel sentences. We can also imagine the use of other text datasets such as translations between programming languages or writing styles as we describe in more detail under Style Augmentation.

4) Style augmentation

Finally, we present another augmentation strategy utilizing Deep Networks to augment data for the training of other Deep Nets. In our previous survey of Image Data Augmentation, we explored works that use Neural Style Transfer for augmentation. Artistic style transfers such as a Picasso-themed dog image, may be useful as an OOD augmentation in a Negative Data Augmentation framework, which we will present later. However, we are more interested in styles within the dataset. This is an interesting strategy to prevent overfitting to high-frequency features or blurring out the form of language such as to focus on meaning. In the text data domain, this could describe transferring the writing-style of one author to another for applications such as abstractive summarization or context for extractive question answering.

5) Generative data augmentation

Generative Data Augmentation is one of the most exciting emerging ideas in Deep Learning. This includes generating photorealistic facial images or indistinguishable text passages. These models have been very useful for Transfer Learning, but the question remains: What is the killer application of the generative task? These generations are certainly interesting for artistic applications, but more importantly is their use for representation learning and Data Augmentation.

3.1.3 Tokenization

Out of sentence and word tokenization, this model explicitly uses only word tokenization in the execution part of the chatbot, where inputs or user query is broken down into a list of only words. It comes under the data processing or clean-up phase to eliminate any irregularities or noise from the text.

3.1.4 Lemmatization

Lemmatization is a better version of the stemming process, which involves shortening words into their most basic form. This process further helps normalize the tokenized data so that they can be learned in a more enhanced way by the model. In this model, lemmatization is done before the training as well as the execution phase to ensure that the matched words or tokens are in similar form.

3.1.5 Bag of Words

It helps in extracting the most important features of the text data by converting them into vector values, which are the forms capable of training the created model. If a user specifies the words likely related to Diabetes, the Bow for the Diabetes group will have to be matched by the user's text after tokenization and lemmatization. In this model, a hard-coded function is used to create the bag of words for each class and later converted to NumPy vectors before training the model.

3.1.6 Pretrained models for preprocessing

Pretrained models play a crucial role in preprocessing text data, leveraging vast amounts of previously acquired knowledge to enhance the accuracy and efficiency of text analysis tasks. These models, such as BERT, GPT, and RoBERTa, are trained on extensive datasets encompassing diverse linguistic patterns, contexts, and semantic relationships. By employing a pretrained model, one can perform sophisticated text preprocessing tasks, including tokenization, lemmatization, and context-aware word embeddings, with significantly reduced computational effort and time. This approach not only ensures consistency and reliability in text processing but also enhances the model's ability to understand and interpret nuanced language structures, ultimately leading to improved performance in downstream natural language processing tasks.

3.2 Classification System

Medical specialty identification is an important procedure in global healthcare, and patients benefit from early diagnosis. Currently, a wide range of symptoms, behavioral traits, genetic mutations, environmental factors, psychosocial issues, and comorbidities have been observed or mentioned, and medical doctors require assistance in effectively diagnosing diseases using computer-aided systems and intelligent diagnosis scoring. For machine learning, it is critical to effectively capture doctors' diagnostic thinking and combine techniques such as pattern recognition, feature selection, and fast training algorithms, allowing them to understand patterns and distinguish distinct diseases.

In this system, we will categorize unstructured text data (doctor's notes and patients' medical records) into medical specialties to help automate administrative processes in healthcare systems that require doctor prescriptions. Using various deep learning-based methodologies, patterns, and other types of evidence inherent in these free-text documents can assist forecast and identify potential areas of specialty, as well as which medical specialists should be in charge of the patients' treatment pathway.

3.3 Assist System

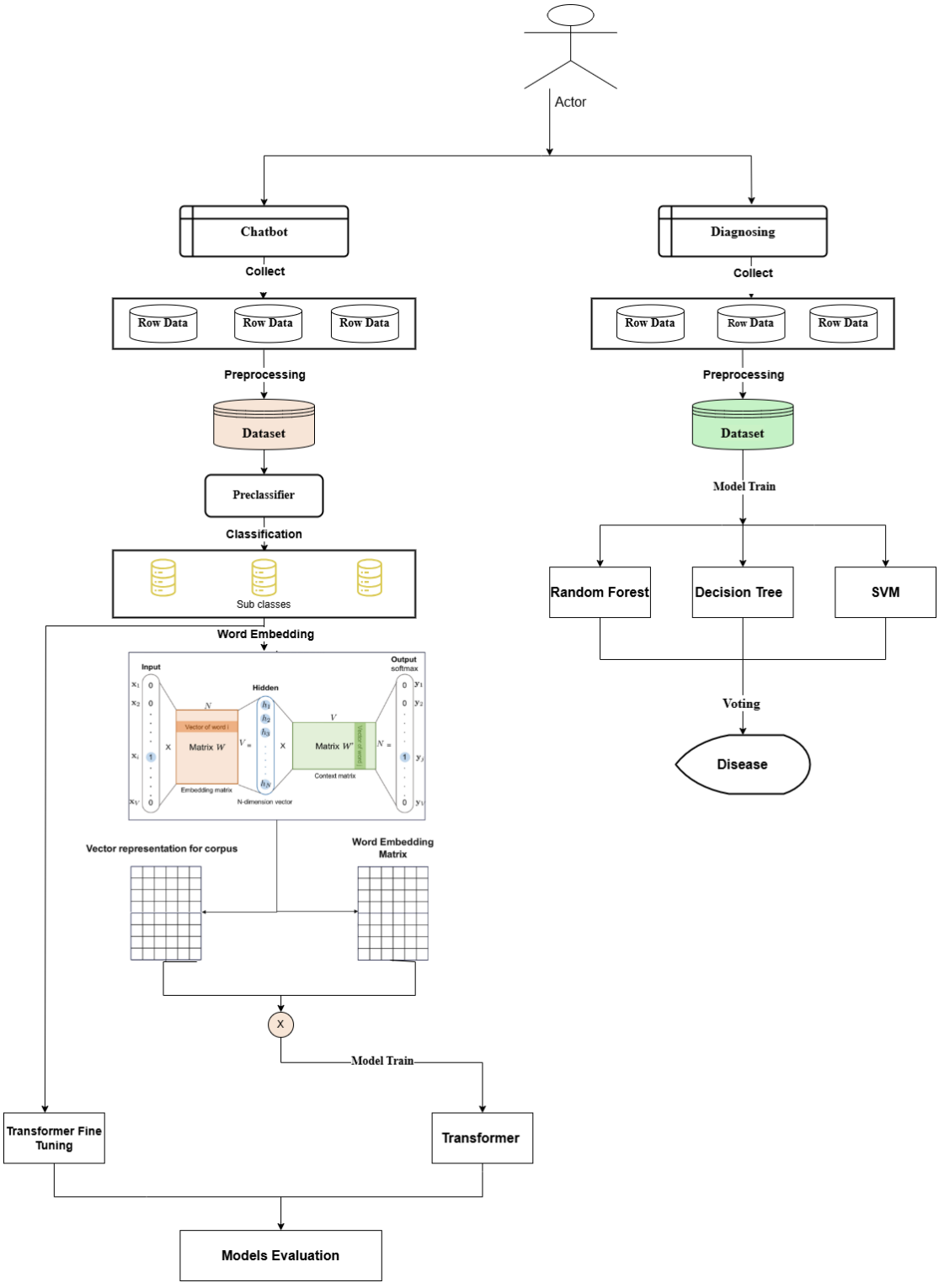
AI should not replace human critical thinking, but it can help improve scientific practice by serving as an inventive tool for advancing human knowledge and developing cost-effective resources and systems. This study investigates the usage of specialized apps, such as chatbots, in the healthcare industry to acquire a better understanding of several relevant theories.

The AI community has a significant effect on medicine and health care, and the growing number of chatbots designed to administer a timely and adequate response to inquiries from users indicates a changing landscape. Chatbot services are everywhere, largely due to recent advancements in AI that have enhanced NLP capabilities and enabled them to automate conversation. They can be used in many different sectors of the health care system. In order to meet new modeling requirements or to re-engineer chatbots, many innovations have been proposed for the development of models like the development of a user simulation plan. In previous studies, chatbots in the healthcare sector have had a great deal of scientific interest, specifically in the case of mobile and mental

Chapter 4
System Implementation

In this chapter, we provide a detailed description of the code, explaining the system's functions and the specific role each one plays.

Figure4.1: system Architecture



4.1 Chatbot

4.1.1 Preprocessing:

- **Data Balancing:**

a. Translation-based Augmentation:

The initial step is to translate the Arabic text into English using the google trans library. The augmentation techniques employed in this code focus on word-level modifications. Each technique introduces variations in the English text:

I. Technique randomly selects two words in the English text and swaps their positions. By swapping words, the sentence structure is altered, potentially generating new sentence variations while preserving the original context.

II. Technique replaces a word in the English text with one of its synonyms. Synonyms have similar meanings but different expressions.

After applying each augmentation technique to the English text, the augmented text is translated back to Arabic using the google trans library.

b. Scraping Technique:

Second technique we scrap data from altibbi.com website for small classes and drop records from big classes we made all classes have 6800 train record and 1700 test record.

- **Classification preprocessing:**

1. Drop for unnecessary columns such as (a_body, a_bodycount)
2. Get max length of question
3. Label encoder for category
4. Tokenize question body
5. Padding question body

➢ generation preprocessing:

1. Get average length for question and use it as total length

2. Duplicate Removal and Missing Value Handling:
   - Remove duplicate entries in the dataset to ensure data integrity.
   - Handle missing values by replacing them with appropriate indicators (e.g., Nan).

3. Advertisement Removal:
   - Identify and remove any advertisements or promotional content present in the text data.
   - This step helps in focusing on the actual content and removing irrelevant information.

4. Diacritics Removal:
   - Remove diacritical marks (such as vowels and other accent marks) from Arabic text.
   - Diacritics are often omitted in natural language processing tasks to simplify the text representation.

5. Repeating Character Removal:
   - Identify and remove repeated characters within words or sequences in the text.
   - Repeating characters may not contribute significantly to the semantic meaning and can be eliminated for text normalization.

6. Standardizing Arabic Letters:
   Arabic letters may have different forms depending on their position within a word, and standardization helps ensure uniformity.

7. Replace numbers with words:
   Replace numerical digits with text data.

8. English Text Detection and Handling:
   - Detect the presence of English text within the Arabic text data.
   - Remove or replace English words or phrases to maintain the coherence and consistency of the Arabic text.

9. Data filtration:

   Set a maximum length limit for the questions and answers then maintain only the questions and answers that are within the length limit.

10. Answer Trimming:
    - Limit the number of words in each answer to specific chosen number by splitting and joining the words.
    - Update the answers list with the trimmed answers.

11. Word Counting:
    - Create a dictionary word2count to count the occurrences of each word in both questions and answers.
    - Iterate over the questions and answers, split them into words, and update the word counts in word2count.

12. Vocabulary Creation:
    - Define a threshold value (thresh) to determine the minimum frequency required for a word to be included in the vocabulary.
    - Create a vocabulary dictionary vocab to store the words with frequencies above the threshold, along with their corresponding indices.
    - Update the vocabulary with additional tokens like <PAD>, <EOS>, <OUT>, and <SOS>.

13. Encoding and Padding:
    - Iterate over the cleaned questions and convert each word to its corresponding index from the vocabulary.
    - Pad the sequences with zeros at the end (post padding) and truncate any sequences longer than the maximum length.

4.1.2 models:

4.1.2.1 Classification model:

Machine learning models

- The Random Forest model:

  is an ensemble learning method used for classification and regression. It operates by constructing multiple decision trees during training and outputting the mode of the classes (classification) or mean prediction (regression) of the individual trees.

- Multi (MultinomialNB):

  is a probabilistic learning algorithm based on applying Bayes' theorem with the strong independence assumption between the features. It is particularly well-suited for classification with discrete features, such as word counts or term frequencies in text classification.

- SVM:

  is a powerful supervised learning algorithm used for both classification and regression tasks. SVM works by finding the hyperplane that best separates the data into different classes, with the goal of maximizing the margin between the classes.

- A Voting Classifier:

  is an ensemble learning method that combines multiple machine learning models to improve predictive performance. The final prediction is made by aggregating the predictions of each individual model. This example uses several classifiers: Logistic Regression, XGBoost, Gaussian Naive Bayes, Multinomial Naive Bayes, and (SVM).

Deep learning models

- LSTM model:

LSTM networks are a type of (RNN) capable of learning long-term dependencies, making them ideal for sequence prediction problems, such as text classification. **LSTM Layer** Captures sequential dependencies in the data.

- Transformer Model with GRU for Text Classification:

Text classification model using a Transformer architecture enhanced with a GRU (Gated Recurrent Unit) layer. Transformers are well-suited for capturing global dependencies in sequences, while GRU helps in capturing temporal dependencies. The model also utilizes early stopping to prevent overfitting.

- Fined tuning for aubmindlab/bert-base-arabertv2 model:

utilizes a pre-trained BERT model for sequence classification. BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art model for natural language understanding tasks. Here, we use the AutoModelForSequenceClassification and AutoTokenizer from the transformers library by Hugging Face to fine-tune a pre-trained BERT model on a specific text classification task.

4.1.2.2 Generation model:

Seq2Seq models with Long-Range dependencies

- LSTM model:

LSTM is a type of recurrent neural network (RNN) that is well-suited for processing sequential data. It overcomes the limitations of traditional RNNs in capturing long-term dependencies by introducing memory cells and gating mechanisms.

Memory cells:

The key component of an LSTM is its memory cells. These cells can store information over long periods, allowing the model to capture dependencies between distant words or tokens in a sequence. Each memory cell has an internal state that is updated and modified as the model processes the input sequence.

Gating mechanisms:

LSTMs use gating mechanisms to control the flow of information through the memory cells. The three main gates are:

1- Forget gate: Determines which information to discard from the previous memory state.

2- Input gate: Determines which new information to add to the memory state.

3- Output gate: Determines which information to output from the current memory state.

Training:

Like other neural network models, an LSTM model is trained using gradient descent and backpropagation. It learns to adjust the weights and biases of its connections to minimize the difference between the predicted output and the ground truth.

Text generation:

Once an LSTM model is trained, it can be used for text generation. Given an initial input or seed, the model predicts the next word or character based on the context it has learned from the training data. This process is repeated iteratively, where the generated output becomes part of the input for the next prediction step, allowing the model to generate a sequence of text.

- BILSTM:

A Bidirectional LSTM (BiLSTM) is an extension of the LSTM model that processes input sequences in both forward and backward directions. It consists of two LSTM layers: one processes the input sequence from left to right (forward direction), and the other processes it from right to left (backward direction).

Forward and backward processing:

The forward LSTM layer processes the input sequence in a standard sequential manner, starting from the first token and moving to the last. Simultaneously, the backward LSTM layer processes the sequence in the reverse order, starting from the last token and moving to the first. This bidirectional processing allows the model to capture both past and future dependencies of each token.

Hidden states:

As the forward and backward LSTMs process the input sequence, they maintain hidden states that encode the information learned from previous tokens in each direction. The hidden states capture context and dependencies, allowing the model to make informed predictions for each token based on its surrounding context.

Concatenation:

After processing the input sequence in both directions, the hidden states from the forward and backward LSTMs are concatenated. This combination of information from both directions provides a more comprehensive representation of the input sequence and helps capture a wider range of dependencies.

Training:

The BiLSTM model is trained using backpropagation and gradient descent, like other neural network models. During training, the model learns to adjust its weights and biases to minimize the difference between predicted outputs and the ground truth.

Text generation:

The model takes a seed or initial input and iteratively predicts the next word or character based on the combined context from both directions.

- Transformers:

Transformers are a type of deep learning model introduced in the paper "Attention Is All You Need" by Vaswani et al. Transformers have revolutionized various natural language processing tasks, including text generation, due to their ability to capture long-range dependencies and effectively process sequential data.

Training:

Transformers are trained using a variant of the backpropagation algorithm called the "transformer algorithm." During training, the model learns to adjust its weights and biases by minimizing a loss function, typically using techniques like stochastic gradient descent (SGD) or its variants.

A transformer model consists of multiple layers, typically composed of self-attention mechanisms and feed-forward neural

networks. Here's a detailed explanation of the flow of data through each layer of a transformer model:

Self-Attention Layer:

Input: A sequence of embedded tokens, each represented by an embedding vector including semantic information and positional encoding.

Computation: Each token is transformed into three vectors: query, key, and value, derived from linear projections of the input embeddings.

Attention Scores: Computed by taking dot products between query and key vectors to measure relevance or similarity between tokens.

Attention Weights: Attention scores are scaled and passed through a SoftMax function to obtain weights, determining the importance of each token's contribution.

Weighted Sum: Attention weights are applied to value vectors, resulting in a weighted sum representing attended information for each token.

Output: The attended information, combined with the original input embeddings via a residual connection, allows retention of important information.

Feed-Forward Neural Network Layer:

Input: The output from the self-attention layer.

Linear Transformation: The input is transformed using a fully connected layer, projecting it into a higher-dimensional space.

Activation Function: A rectified linear unit (ReLU) introduces non-linearity.

Linear Projection: The output is projected back to the original dimensionality.

Output: Passed through a residual connection and normalized using layer normalization.

Normalization:

- Layer Normalization: Normalizes values across the hidden dimension of the input, ensuring a mean of zero and a standard deviation of one.
- Output: Added to the input through another residual connection.

Advantages of transformers compared to previous models:

- Capturing long-range dependencies: Transformers excel at capturing long-range dependencies in sequences, which was a challenge for earlier models like LSTMs and RNNs. Self-attention allows the model to access any token in the sequence when making predictions, enabling the capture of long-distance relationships.

- Parallel processing: Unlike recurrent models that process sequences sequentially, transformers can process tokens in parallel. This parallelism allows for faster training and inference, making transformers more computationally efficient.

- Scalability: Transformers can scale to handle large datasets and capture complex patterns. The self-attention mechanism enables the model to process sequences of arbitrary lengths, making transformers more adaptable to different tasks and domains.

- Reduced context fragmentation: Unlike traditional models that process input sequentially, transformers process the entire sequence simultaneously. This reduces the fragmentation of context, allowing the model to have a better global understanding of the input.

- Interpretability: Transformers' attention mechanism provides interpretability, as it indicates which tokens in the input sequence are most important for generating each output token. This attention-based interpretability allows users to gain insights into the model's decision-making process.

4.2 disease-symptom-prediction

4.2.1 preprocessing

- Translate data from English to Arabic by using google translate
- Shuffle data
- Replace _ with space
- Fill nan by 0
- Replace symptom with its weight

4.2.2 models

- We use machine learning model such as (random forest)
- Second Decision tree classifier:
    - is a simple yet powerful algorithm used for classification tasks. It works by recursively splitting the data into subsets based on the most significant feature at each node. This implementation demonstrates how to use a Decision Tree for text classification by first converting text data into numerical features and then training a Decision Tree on those features
- voting models contain SVM, Random Forest and decision tree

Chapter 5
System Testing

This chapter presents our experiments and results, beginning with the experimental setup, including datasets, model architectures, and hyperparameters. We detail results with metrics such as cosine similarity and BLEU scores, and analyze these to compare models and configurations, highlighting factors influencing their performance.

5.1 Datasets

5.1.1 MAQA Dataset

We introduce the largest Arabic Healthcare Q &A dataset as we know. (MAQA) was collected from various websites which are listed in Table 5.1. The dataset consists of more than 430k questions distributed into 20 medical specializations that we contribute to the research community on Arabic computational linguistics.

The distribution of questions per category is summarized in Table 5.2. All questions are unique. The data is kept in raw format, cleaned but not stemmed, or any other preprocessing has applied after scraping. The questions and answers contain some English symbols and digits, and almost no Arabic diacritics or punctuation. Table 5.4 shows an example of a question from the "امراض نسائية","امراض الجهاز الهضمي " categories.

The data contains the following columns q_body contains the question content, a_body contains the answer content, q_body_count contains the question content word count, a_body_count Contains the answer content word count, category Contains category name and category_id Contains category number from table categories.

Table 5.1: website

Website	Percent
altibbi.com	70
tbeeb.net	20
cura.healthcare	10

Table 5.2: statistics of data

Command	Number	Appear
Max number of token per Q	100	2
Max number of token per A	30	1
Min number of token per Q	1	5
Min number of token per A	1	63
Mode number of token per Q	30	18293
Mode number of token per A	9	6812
Mean number of tokens per Q	19	4496
Mean number of tokens per A	17	4750

Table 5.3: Category

Label	Count
امراض نسائية	103683
امراض المسالك البولية والتناسلية	33847
امراض العضلات والعظام و المفاصل	33050
الامراض الجلدية	29262
الطب العام	26870
امراض باطنية	23722
امراض الجهاز الهضمي	22373
الامراض الجنسية	21773
طب الاسنان	20207
امراض الاطفال	18636
امراض نفسية وعصبية	18295
امراض القلب و الشرايين	15368
جراحة عامة	15185
امراض العيون	14439
انف اذن وحنجرة	13933
الاورام الخبيثة والحميدة	11210
امراض الغدد الصماء	5186
امراض الجهاز التنفسي	4567
جراحة تجميل	1596
امراض الدم	1341

Table 5.4: Example from MAQA dataset

Question	Answer	q_body_count	a_body_count	category	category_id
اذا كانت مده الدوره عندي 32 يوم فماهي ايام الاباضه عندي لان بدي احمل	18 17 16 15 14	14	5	امراض نسائية	15
السلام عليكم جاي يصير عده تهيج في تحدث عده خربطه في المعده والقالون العصبي تستخدم علاج مرفق في الفايل ومستقره على العلاج فقط تحدث مشكله في الاكل ماذا تكل وماذا...	This is not colon treatment it is stomach treatment	30	9	امراض الجهاز الهضمي	7

the challenges of data are there is some answers contain English words or numbers

5.1.2 Diagnosing Dataset

Disease symptom prediction is English dataset which helps to create a disease prediction or healthcare system. The dataset contains diseases, their symptoms, precautions to be taken, and their weights. The dataset distributed into 41 diseases and 17 symptoms. We use different files from the diagnose dataset.

- First file for disease prediction classifier
- Second file for disease description retrieve.
- Third file for disease precaution.

Table 5.5: Example from Diagnosing Dataset

Symptom	Disease	Description	precaution
الم المفاصل وجع البطن فقدان الشهية القئ	إلتهاب الكبد أ	التهاب الكبد (أ) هو عدوى الكبد شديدة العدوى الناجمة عن فيروس التهاب الكبد (أ). يعد الفيروس واحدًا من عدة أنواع من فيروسات التهاب الكبد التي تسبب الالتهاب وتؤثر على قدرة الكبد على أداء وظائفه.	استشارة أقرب مستشفى غسل اليدين تجنب الأطعمة الغنية بالتوابل الدهنية

5.2 Evaluation Metrices

Deep learning model uses trained word vectors In Word2vec CBOW, especially pre-trained CBOW model called Aravec (Mohammad et al. 2017). This model was trained on 132,750,000 Arabic documents with 3,300,000,000 words. A word embedding matrix is then generated based on the pre-trained CBOW model. The word embedding sentences of the corpus generated by the word embedding matrix are then fed to each network as input features.

Test the model on a held-out test set and evaluate its performance using appropriate metrics, such as cosine similarity and BLeU score as following:

- Cosine Similarity

To evaluate our generated answer against the actual answer, we start by getting the embedding vector for each word in the sentence, then get the average for all words' vectors as in equation A1, where A is the sentence vector, Vi is the word vector and N the words count in the sentence. Then, we calculate the vectors product in equation A2, where A and B are two nonzero vectors can be derived by using the Euclidean dot product formula. After that we calculate the cosine similarity between both average vectors as in equation A3, where Ai and Bi are its components of vectors A and B, respectively. Finally, we calculate the Cosine Distance as in equation A4. The greater Cosine Distance, the greater the model efficiency and accuracy (Hendy et al. 2023) [27]

Figure 5.1: Cosine Similarity Equation

$$A = \sum_{i=1}^{N} \frac{Vi}{N} \quad \text{(A1)}$$

$$A.B = \|A\| \ \|B\| \cos\theta \quad \text{(A2)}$$

$$\text{cosine similarity} = \frac{\sum_{i=1}^{n} AiBi}{\sqrt{\sum_{i=1}^{n} Ai^2}\sqrt{\sum_{i=1}^{n} Bi^2}} \quad \text{(A3)}$$

$$\text{Cosine Distance} = 1 - \text{Cosine Similarity} \quad \text{(A4)}$$

- BLeU Score

The BLeU score is an algorithm for evaluating the quality of text generated using deep learning algorithms (Papineni et al. 2002) [11]; accuracy is considered the correspondence between a machine's output and that of a human. The base stone of the BLeU score is the familiar precision measure, which is calculated by counting the number of candidate translation words (unigrams) that occur in any reference translation and then divided by the total number of words in the candidate translation as shown in equation A5 (Hendy et al. 2023) [27]. However, as in our bot task, the modified n-gram can be generalized as in equation A6 to the case: one candidate sentence and one reference sentence, where ŷ is candidate sentence and y is one reference sentence. Then, we start with the n-gram count summation as in equation A7 (Hendy et al. 2023) [27]. This count summation cannot be used to compare sentences since it

is not normalized. If both the reference and the candidate sentences are long, the count could be huge, even if the candidate is of poor quality (Hendy et al. 2023) [27]. So, we normalize it as in equation A8, and equation A9 shows the final definition of BLEU, where w := (w1, w2, ⋯) is the weighting vector, and Ŝ := (ŷ (1), ⋯, ŷ(M)) is candidate corpus, and S = (S1, ⋯, SM) is reference candidate corpus

Figure 5.2: BLeU Score Equation

$$p_n(\hat{S};S) := \frac{\sum_{i=1}^{M} \sum_{s \in G_n(\hat{y}^{(i)})} \min(C(s,\hat{y}^{(i)}), \max_{y \in S_i} C(s,y))}{\sum_{i=1}^{M} \sum_{s \in G_n(\hat{y}^{(i)})} C(s,\hat{y}^{(i)})} \quad \text{(A5)}$$

$$\sum_{s \in G_n(\hat{y})} \min(C(s,\hat{y}), C(s,y)) \quad \text{(A6)}$$

$$\sum_{s \in G_n(\hat{y})} C(s,y) = \text{number of n-substrings in } \hat{y} \text{ that appear in } y \quad \text{(A7)}$$

$$p_n(\hat{y};y) = \frac{\sum_{s \in G_n(\hat{y})} \min(C(s,\hat{y}), C(s,y))}{\sum_{s \in G_n(\hat{y})} C(s,\hat{y})} \quad \text{(A8)}$$

$$BLEU_w(\hat{S};S) := BP(\hat{S};S) \cdot \exp\left(\sum_{n=1}^{\infty} w_n \ln p_n(\hat{S};S)\right) \quad \text{(A9)}$$

5.3 Chatbot Experimental Result

5.3.1 Unbalanced data

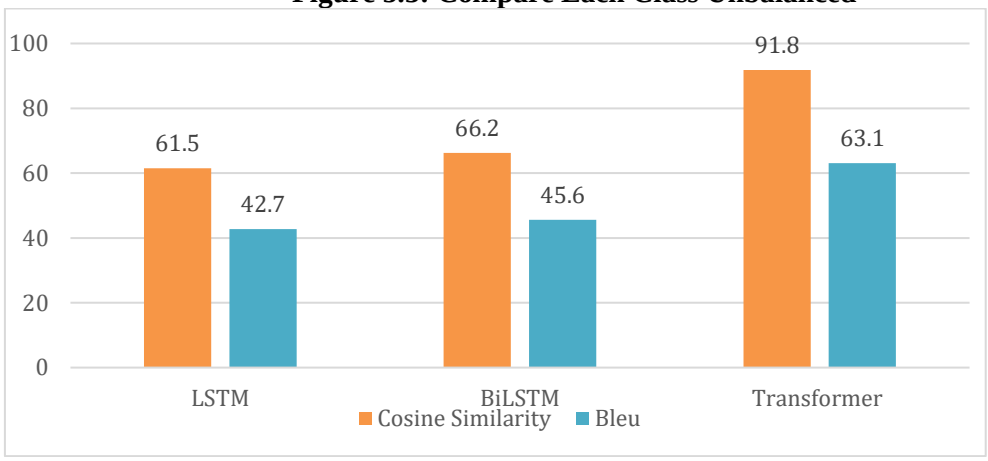
a. For MAQA this unbalanced data we run each class on LSTM, BiLSTM and Transformer.

We get max average similarity in transformer is 91.8% and average bleu63.1%

Table 5.6: result for each class unbalanced

	LSTM	BiLSTM	Transformer
Cosine similarity	61.5	66.2	91.8
Bleu	42.7	45.6	63.1

Figure 5.3: Compare Each Class Unbalanced



In this data the best classes are "امراض الجهاز التنفسي" that has 8764 record its similarity is 96.65% but for this result we used special parameters as (number of layers=2, number of heads=10, units=1024, d_model=300, dropout0.2 and target vocab size =2^13)

In this data the worst classes are "امراض نسائية" that has 70160 record its similarity is 90.2 % but for this result we used special parameters as (number of layers=6, number of heads=10, units=1024, d_model=300, dropout0.2 and target vocab size =2^14)

One of the most important parameters is length of embedding and padding there is a huge number between maxlen and minlen, so we used the average length of data of each class

b. For MAQA run in all data we use transformer because it has max average similarity and bleu and achieve similarity 83.8% and bleu 59.58.

To get these values we split data to 80 and 20 for train and test. we use the maximum between the mean length of question and the mean length of answer that is 18 words as a maximum length and parameters that shown in table 9

Table 5.7: parameters of transformer

parameters	**value**
Number of layers	2
Number of heads	10
units	1024
D-model	300
dropout	0.2
Target vocab size	32,768
Epochs	100

5.3.2 Balanced data

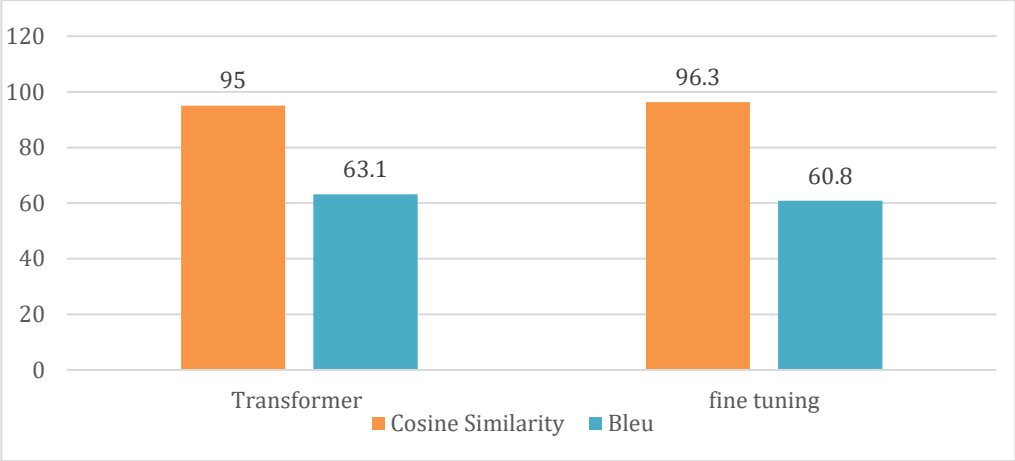
After data augmentation to create balanced dataset, we apply the following experiments:

a. For MAQA this balanced data we run each class on transformer and fine tuning. We get max average similarity in transformer is 95% and average bleu 60.5%

Table 5.8: result for each class Balanced

	transformer	fine tuning
Cosine similarity	95%	96.3%
Bleu	60.5%	60.8

Figure 5.4: Compare Each Class balanced



b. For MAQA run in all data we use transformer because it has max average similarity and bleu and achieve similarity 96.2% and bleu 63%.

Table 5.9: result for all data

Data	Models	
Unbalanced Data	**Transformer** Similarity :83% Bleu: 59.9%	
Balanced Data	**Transformer** Similarity :95.7% Bleu: 62%	**Fine Tuning** Similarity :96.2% Bleu: 63%

Pre-classification model:

Classification models play crucial roles in enhancing model performance and accuracy. Preclassification involves the initial sorting and categorization of data into predefined classes, which helps in reducing the complexity of the data and focusing the model's learning process on relevant subsets. This step is particularly important when dealing with large and diverse datasets, as it streamlines the subsequent training phases. Once the data is pre-classified, a pre-trained model can be fine-tuned to specialize in specific tasks or domains. Fine-tuning involves further training a pre-existing model on a new dataset, adjusting its weights and biases to improve its performance on the specific task at hand. This process leverages the general knowledge the model has already acquired and refines it, resulting in more accurate and efficient outcomes. Together, pre-classification and fine-tuning enable the development of robust and highly specialized machine learning models. For this we applied classification model and then we run each class on transformer and pretrained model to generate text and classify it with two different types of models (machine learning model and deep learning model)

- First machine learning:

We vectorize question to vectors 1D by tfidf_vectorizer and label encode category to id because machine learning can't encode text data.

We use soft voting by (Logistic Regression, Random Forst, multinomialNB) and get class with max probability as target class and inverse id of class to get class name. the best result is 71.6% accuracy

- Deep learning model:

We fine tune pretrained model that named "aubmindlab/bert-base-arabertv2" first we change our data frame to transformer dataset, we use many batch size but that give me max accuracy we use 32batch size and we make epoch by epoch and the best number of epochs is 4 epoch and this give us accuracy 73.6%.

The best parameters give us Average

AUC: 97%

Accuracy: 73.6%

F1 Score (micro): 73.6

F1 Score (macro): 73.5

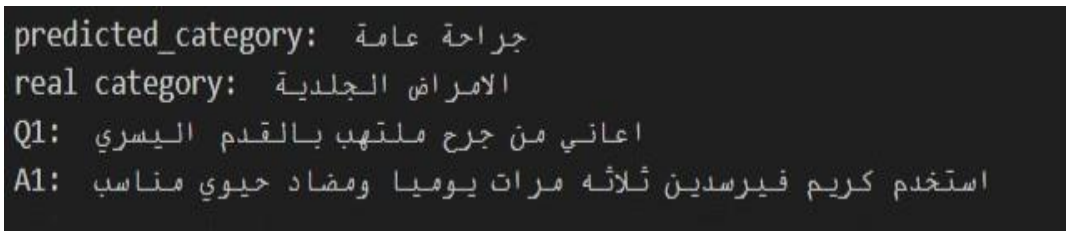
F1 Score (weighted): 73.5

Precision:73.5

Recall: 73.6

The accuracy is small because data have some record with wrong category as Figure5

Figure 5.5: Classification example



Merge generation with classification:

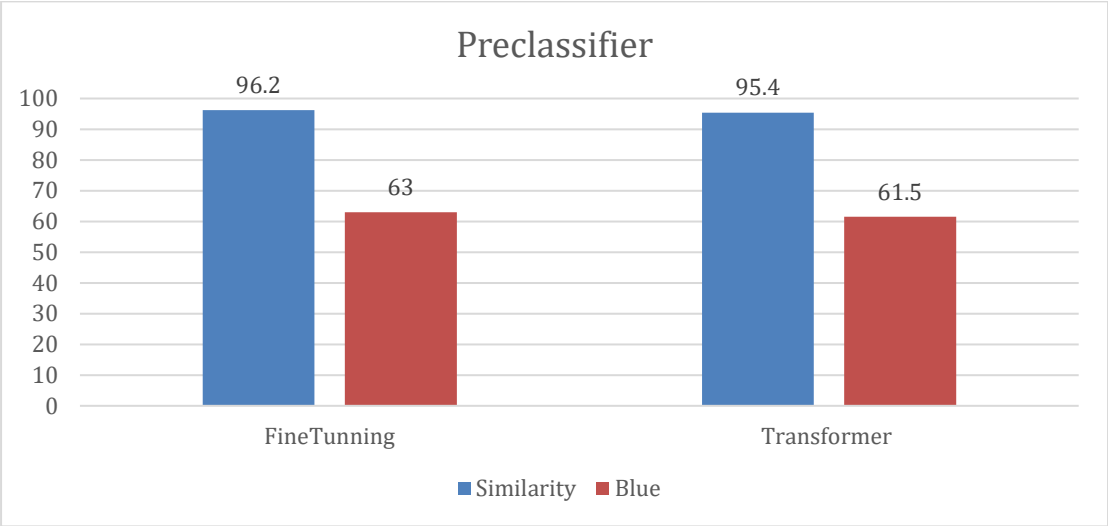
We run class by class for each class parameter for his data. The parameters changed based on length of data in file and mean length. Then we run classification to get model file this is suitable file model of generation models. Then we load model and enter question that have predicted category equal to class that model trained on it.
This technique we used it to get result in scope of question, we use two generation model in it (fine tune and transformer).

We get in fine tuning model 96.6% similarity and 63% in bleu.
We use transformer although and achieve 95.4.% similarity and 61.5% bleu.

Table 5.10: Result for pre-classifier

	Transformer	Fine Tuning
Balanced With pre-classifier	Similarity 95.4.% Bleu: 61.5%	Similarity :96.2% Bleu: 63%

Figure 5.6: Pre-classifier Result



5.3 diagnosing Experimental Result

Read file of disease symptoms if there are symptoms that have null value, we replace null values with 0 then we are embedding symptoms by second file Symptom-severity that have symptom and his weight.

We split data in 80%train and 20%test.
We use machine learning models and train it on embedding data.
We use random forest model that give us accuracy 98.75%.
We try decision tree and achieve 94.31% accuracy.
Final model hard voting in data using
(random forest, decision tree, SVM)
We achieved 99.95% and this is the best data.

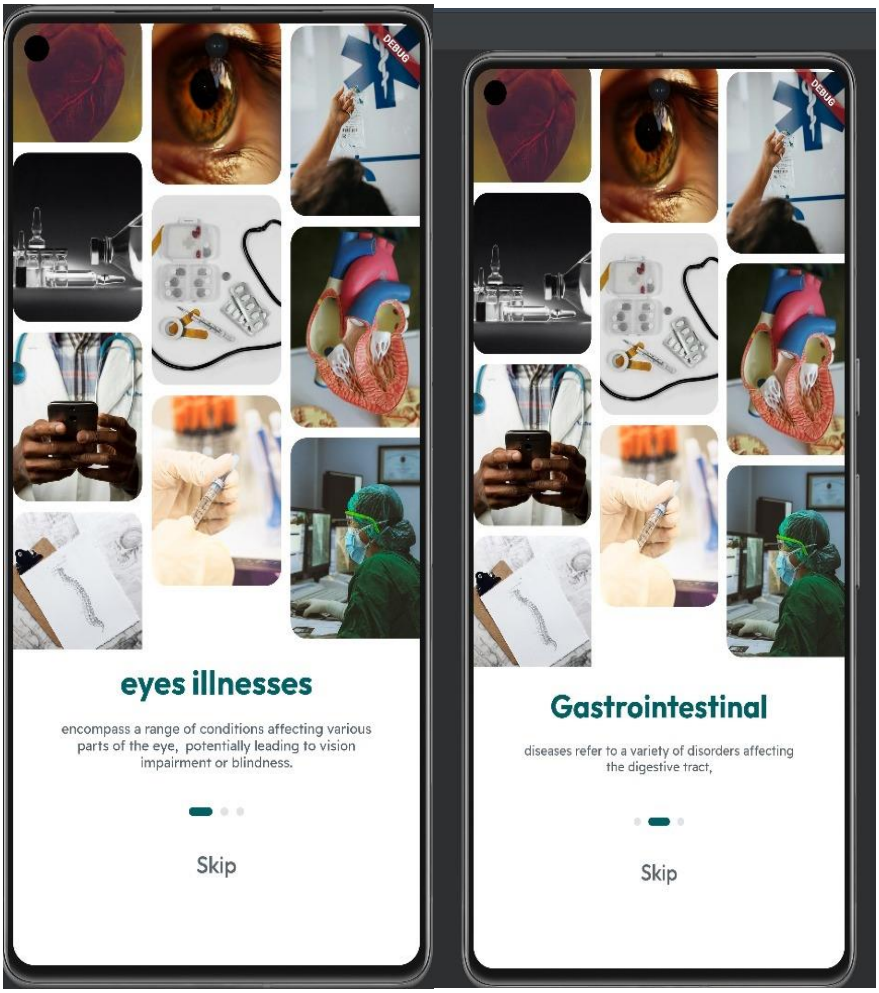
Then we take predicted disease and search on description file to get description of this disease. And in finally we search in file of symptom precaution to get some advice to this disease user need to enter at at least 4 symptoms to get high accuracy of prediction

Chapter 6
Mobile Application

We provided screenshots of the GUI to demonstrate the project's operation, detailing all the steps necessary for users to navigate and use the system effectively.

Welcome Pages

Figure 6.1: Mobile Application Welcome Pages

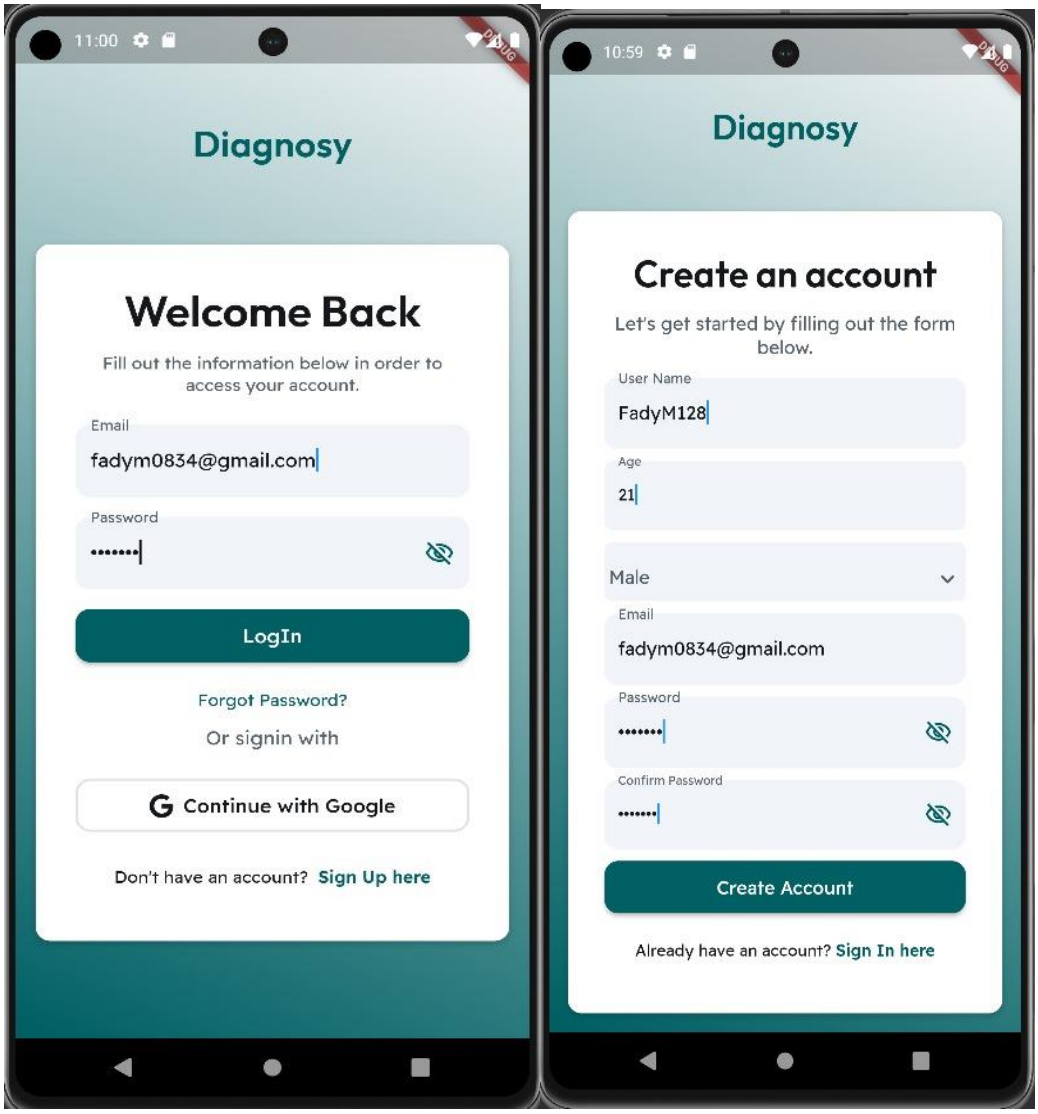


In welcome page we are raising awareness about diseases through scrolling in list of disease.

Login or Registration

Figure 6.2: Mobile App login Page

Figure 6.3: Mobile Application Registration Page

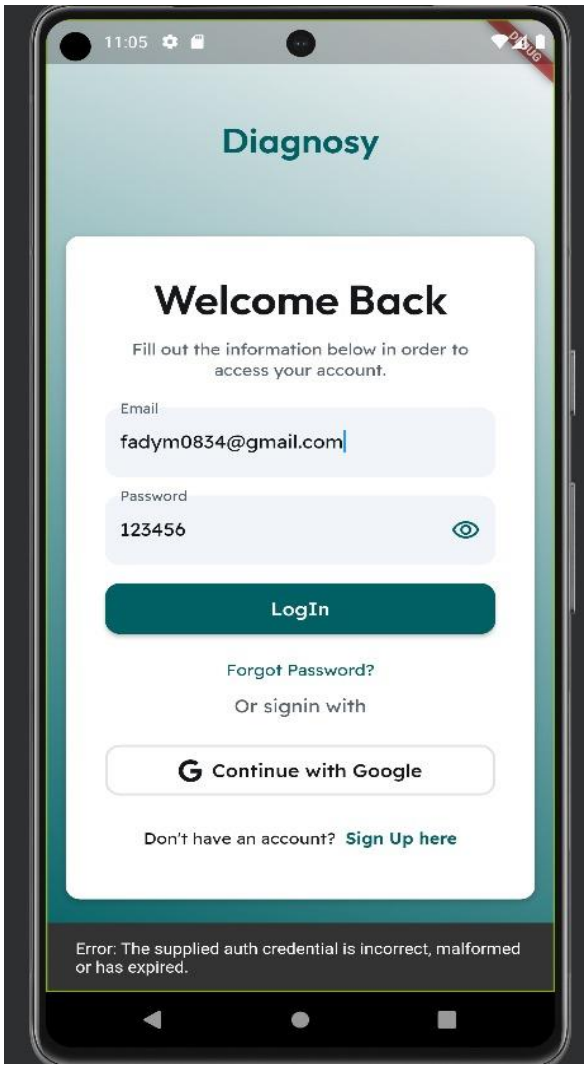


For Signed Users (Already Registered): To access the app's features, signed users must authenticate by entering their registered email address and password.

For Unsigned Users (New Users): New users must create an account to gain authorization and access the app's functionalities. This involves registering with their email and choosing a password to securely enjoy all app features.

Wrong password case

Figure 6.4: wrong password detection



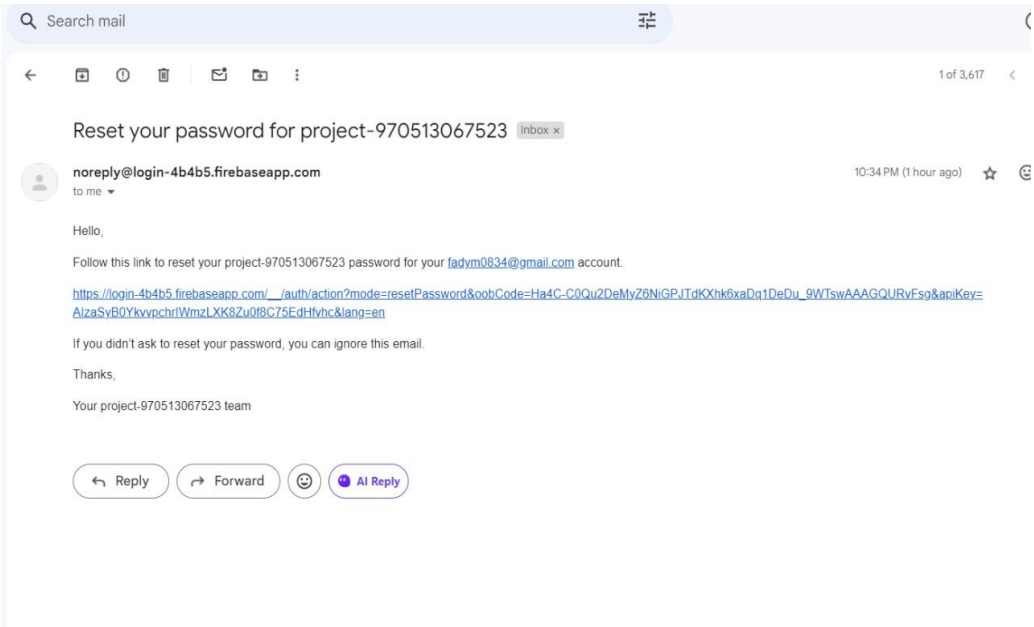
If a user enters an incorrect password, an error message will prompt them. They can continue using our app by entering the correct password or choose to reset it by clicking on "Forget password.

Forget password Button

Figure6.5: enter email to reset password

a user provides his email then clicks on "reset password"

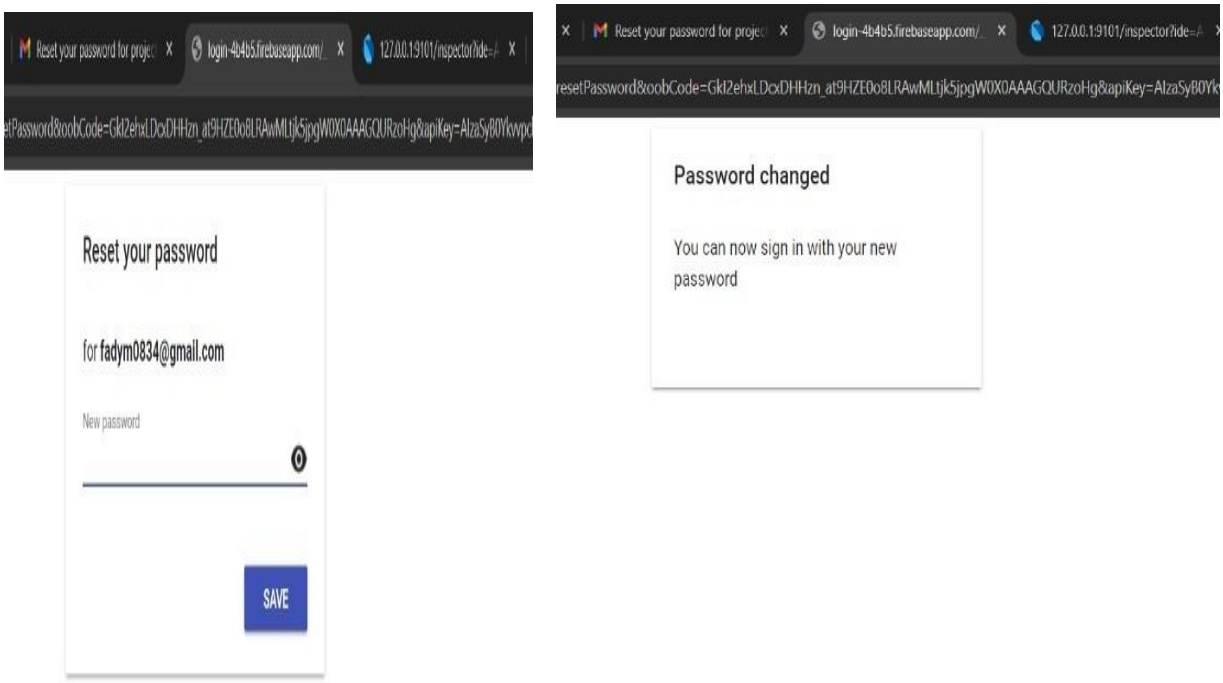
Figure6.6: receive email to reset password



When a user clicks on "Forget password," the system sends an authentication email to the registered email address. This email verifies the user's identity and provides a link to a secure page where they can reset their password.

Reset forget password

Figure6.7: reset password



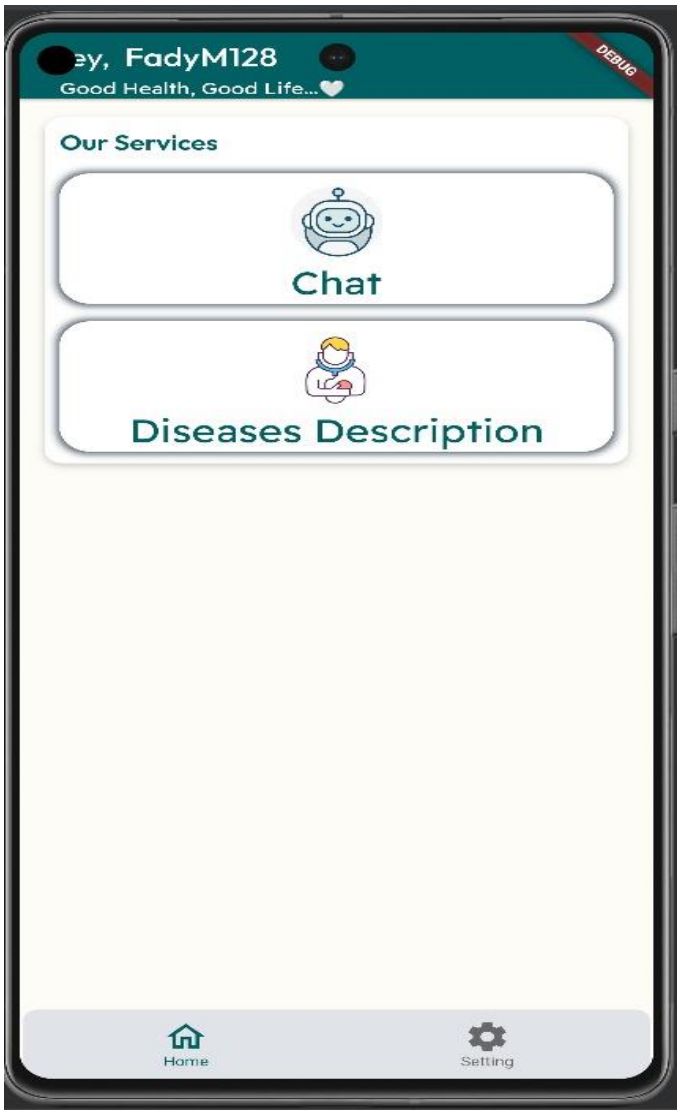
Initiating Password Reset: When a user clicks on the password reset link in the authentication email, they are directed to a secure page where they can enter and submit a new password.

Updating Password in Firebase: The new password is updated securely in Firebase.

Confirmation Message: Upon successful password reset, the user receives a confirmation message indicating that their password has been changed.

Signed user to home page

Figure 6.8: Homepage



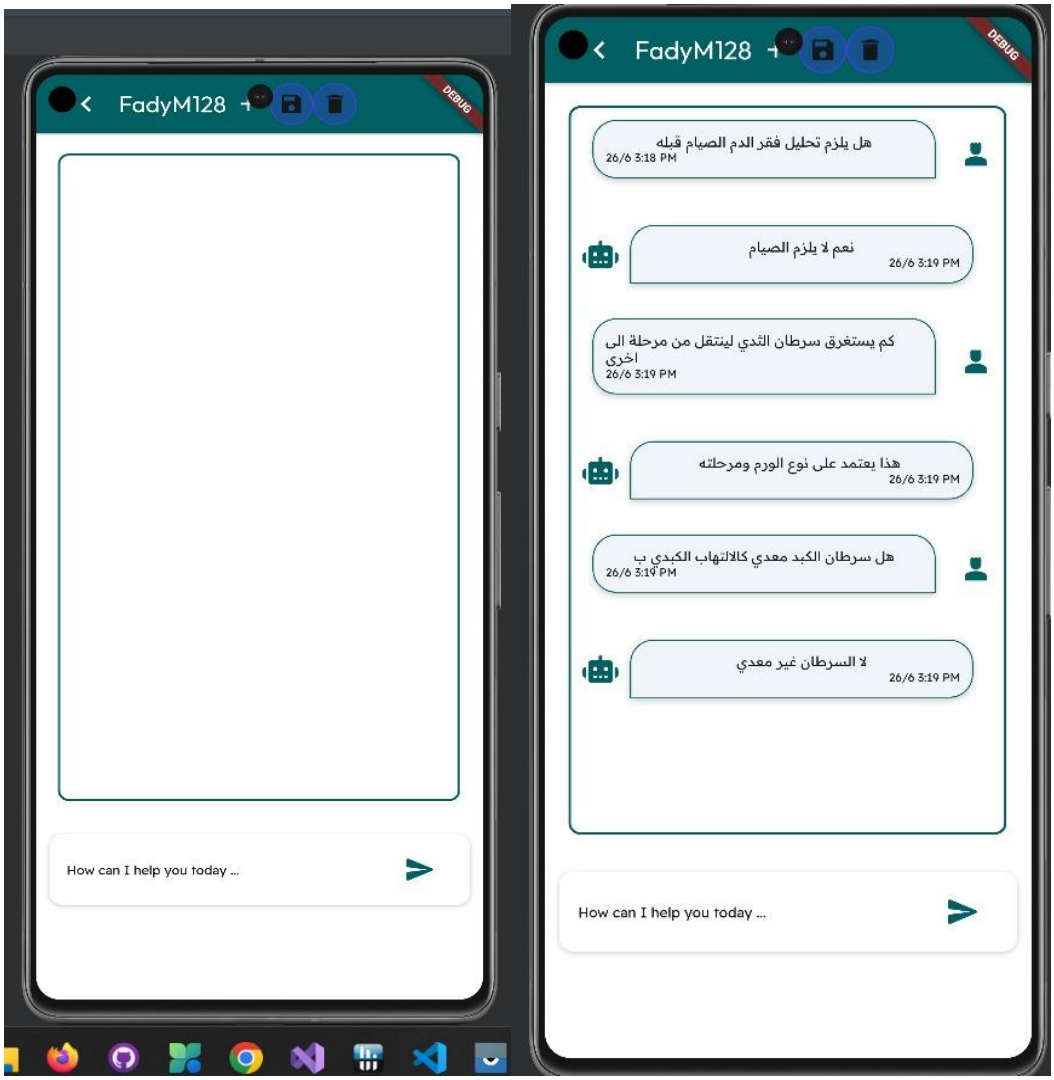
Home Page has two Options:

1. **Medical Bot Chat:** Users can interact with a medical bot to receive accurate answers to their health-related queries.
2. **Disease Description:** Users can input at least four symptoms. The system will then use a model to identify and display details about the disease associated with those symptoms, including disease details and precautions.

Chatbot Page

Figure 6.9: Mobile App Med Chat start

Figure 6.10: Mobile App Test Results

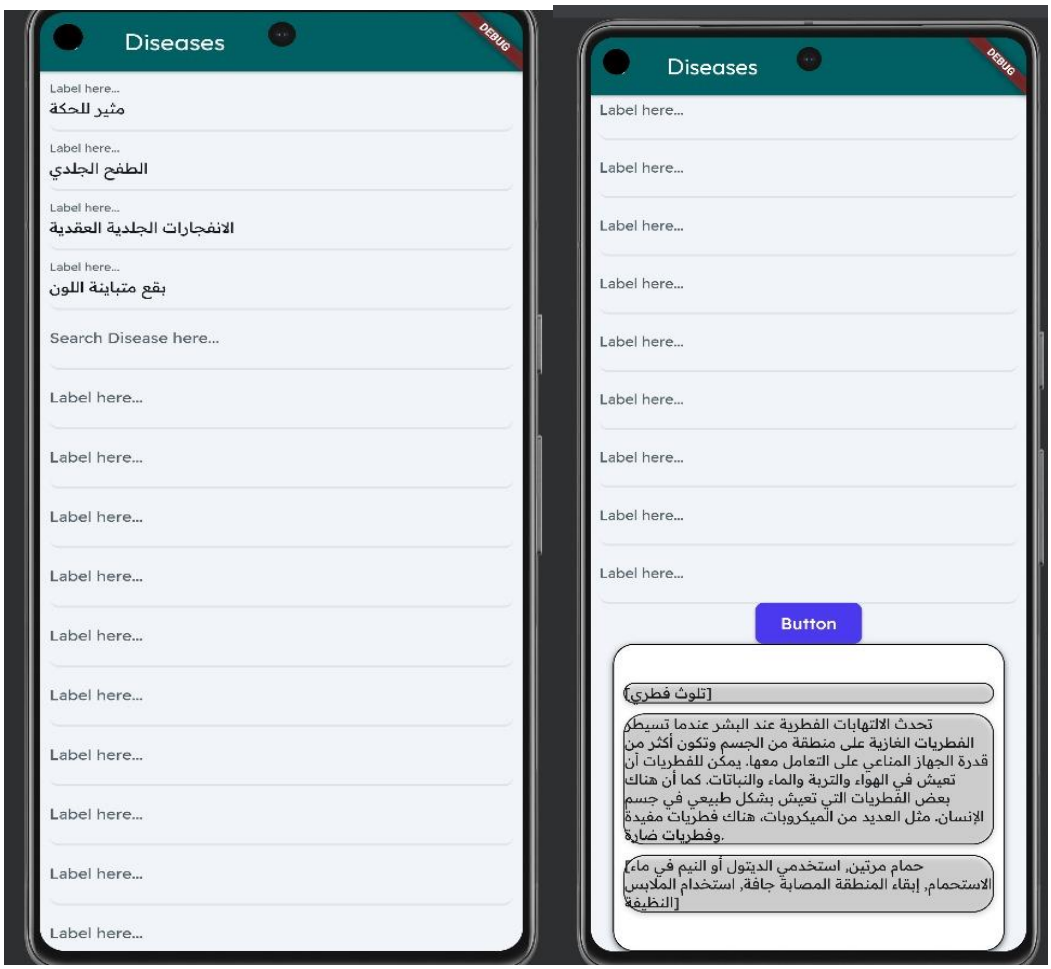


- Our app features a medical chatbot designed to provide users with preliminary medical assistance.
- These are examples of some medical questions asked by the user with their responses provided on UI generated by trained models.
- These interactions demonstrate the capability of our medical chatbot to provide accurate and relevant information to users, enhancing their access to preliminary medical guidance.

Diseases Discerption

Figure 6.11: symptoms entry

Figure 6.12: disease prediction results

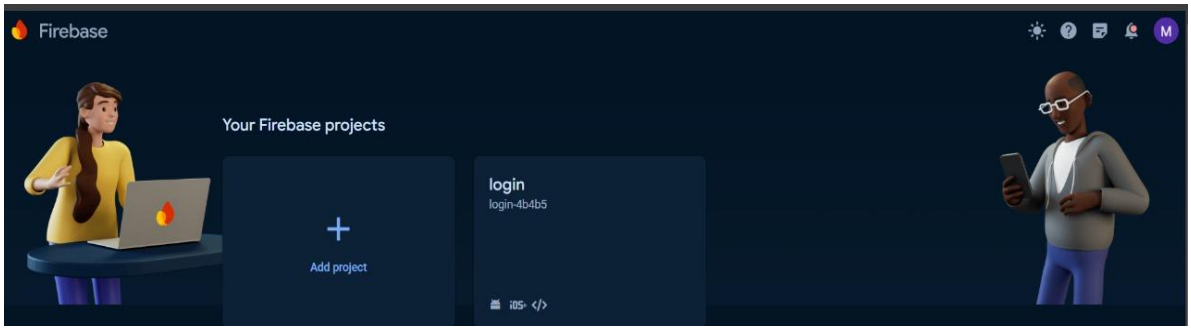


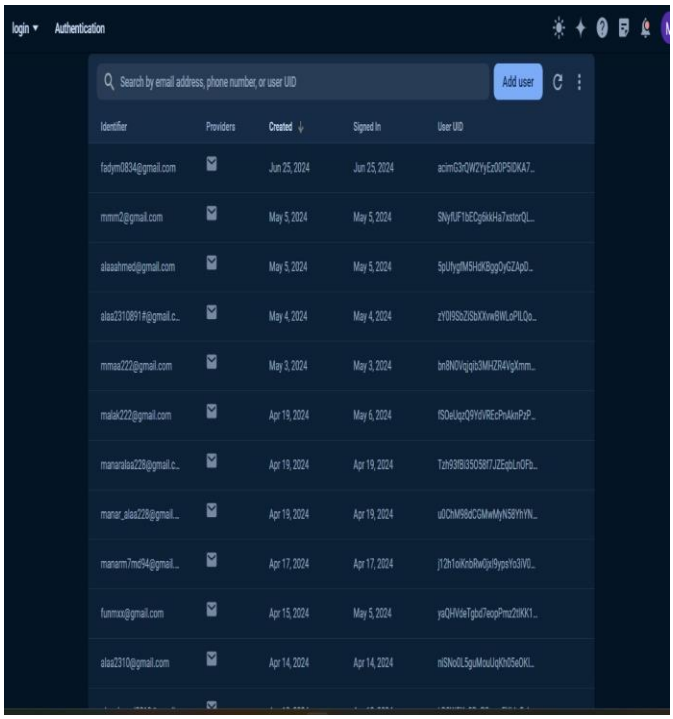
Users can enter a minimum of four and a maximum of seventeen symptoms to receive a response identifying the disease most closely associated with those symptoms. The system will then provide a comprehensive description of the identified disease.

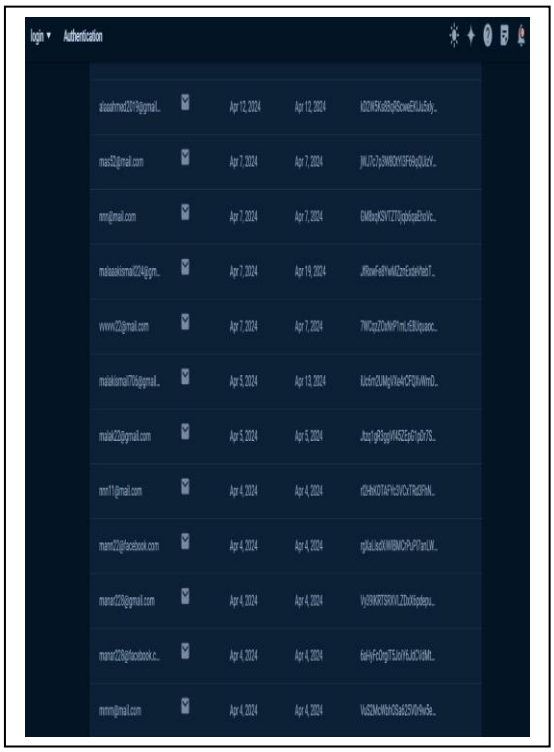
Application Accounts on Firebase

User authentication

Figure 6.13: Firebase Authenticated Accounts







Chapter 7
Conclusion and future work

This chapter covers two aspects: conclusions, summarizing the entire project and its results, and future work, discussing potential improvements and additional functionalities.

7.1 Conclusion

The Arabic language and its dialects are very complex. This poses a new challenge for artificial intelligence. Chat bots are more important now than ever because they can save you a lot of different services in many fields. So, creating The Chabot will be versatile in Arabic, a wonderful addition to enriching the Arabic language and its uses.

Furthermore, the healthcare field is essential in our lives. Given its importance, it will be useful to use chatbots to support this area and provide health assistance to patients. In this paper, we propose a new Chabot called" DIGNOSY" to help patients and improve healthcare delivery. The proposed Chabot adopts the Arabic dialect to communicate with a larger number of patients. Technically Speaking, the created" DIGNOSY" is based on the sequence-to-sequence model with LSTM, BiLSTM, and Transformer. Furthermore, the experimental results showed good efficiency of the proposed Chabot despite the number of data used in this study (num of data). Moreover, the analysis also showed the extent to which Chabot's effectiveness was related to the length of the sentences used, whether of questions or answers.

7.2 Future Work

Future work should focus on optimizing the model's algorithm similarity, feature extraction of dual-lead information. We also will collect more data to enrich the Arabic medical resources, on the one hand, and to improve Chabot's results and similarity on the other hand.

We also aim to implement this project on a web or mobile application. This will help it to go viral even more and so be used by more users. We also can add location functionality which will help the patient to get immediately to the nearest health care organization. So instead of just diagnosing the case, we will also suggest the nearest medical place the patient can visit.

Tools

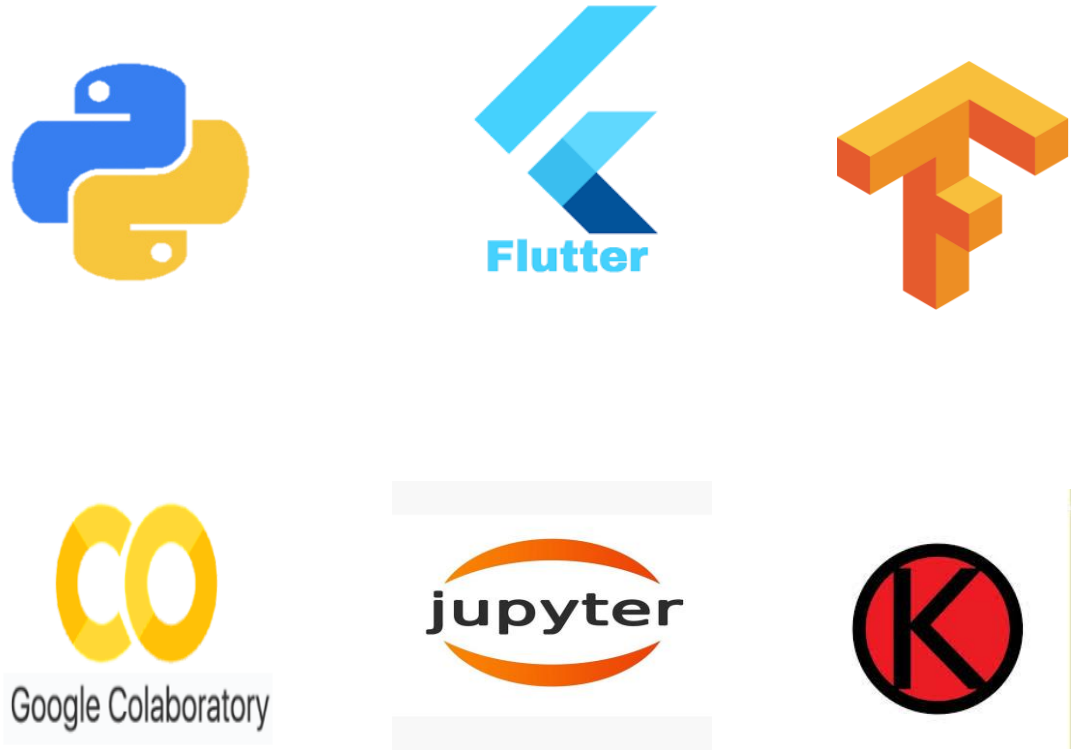


Figure 7.1: Software Tools Used

Python Language:

Python is a high-level programming language known for its simplicity and readability. It has a large standard library and supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is widely used for various purposes, such as web development, data analysis, machine learning, and automation.

Flutter:

Flutter is an open-source UI framework developed by Google for building cross-platform mobile applications. It uses the Dart programming language and provides a rich set of pre-designed widgets and tools for creating beautiful and responsive user interfaces. Flutter allows developers to write code once and deploy it on both iOS and Android platforms, reducing development time and effort.

Dart:

Dart is a programming language developed by Google, primarily used for building mobile, web, and desktop applications. It is the language used in the Flutter framework for developing cross-platform mobile apps. Dart is designed to be easy to learn, with syntax similar to languages like C, Java, and JavaScript.

Keras:

Keras is a high-level neural networks API written in Python. It provides a user-friendly interface for building and training deep learning models. Keras supports various deep learning frameworks as backend engines, including TensorFlow, Theano, and CNTK. It simplifies the process of building neural networks by providing abstractions for defining layers, compiling models, and performing training and evaluation.

TensorFlow:

TensorFlow is an open-source machine learning framework developed by Google. It provides a comprehensive ecosystem of tools, libraries, and resources for building and deploying machine learning models.
TensorFlow supports various types of models, including deep neural networks, and offers flexibility for both research and production deployments. It provides high-level APIs like Keras for ease of use, as well as lower-level APIs for advanced customization.

Google Colab:

Google Colab is a cloud based Jupyter notebook environment provided by Google.
It allows users to write and execute Python code in a browser-based interface without requiring any local setup or installation.
Colab provides free access to computational resources, including CPUs, GPUs, and TPUs, enabling users to train machine learning models efficiently.
It integrates with Google Drive for easy storage and sharing of notebooks.

Jupyter Notebook:

Jupyter Notebook is an open-source web application that allows creating and sharing documents containing live code, visualizations, and explanatory text. It supports various programming languages, including Python, R, and Julia. Jupyter Notebook provides an interactive environment for data analysis, exploration, and collaboration. It allows executing code in cells, which can be rearranged, modified, and executed independently, making it a popular choice for data scientists and researchers. These software tools play key roles in different aspects of software development, data analysis, and machine learning, providing developers and researchers with powerful resources and capabilities.

API:

An API (Application Programming Interface) using JSON (JavaScript Object Notation) and Flask enables efficient communication between different software components, facilitating data exchange and functionality access over the web. Flask, a lightweight and flexible web framework for Python, simplifies the creation of web APIs by providing tools and libraries for routing, request handling, and response formatting. JSON, a popular data interchange format, is used for its simplicity and readability, allowing easy serialization and deserialization of data structures. When building an API with Flask and JSON, developers define endpoints that clients can interact with through HTTP methods such as GET, POST, PUT, and DELETE. Each endpoint processes incoming requests, performs necessary operations, and returns responses in JSON format, ensuring consistent and structured data exchange. This combination is powerful for developing RESTful APIs, enabling integration with various front-end applications, mobile apps, and other services, promoting interoperability and scalability in modern software development.

Firebase:

Firebase is a comprehensive platform developed by Google for building web and mobile applications. It provides a suite of cloud-based services, including real-time databases, authentication, cloud storage, hosting, and machine learning capabilities. Firebase's real-time database allows

developers to store and sync data across all clients in real-time, making it ideal for collaborative applications and real-time data updates. The authentication service simplifies user management by supporting various authentication methods, including email and password, Google, Facebook, and other third-party providers. Firebase's cloud storage is designed to store and serve user-generated content, such as photos and videos, securely and efficiently. Additionally, Firebase hosting offers fast and secure static web hosting with custom domain support. The platform's integration with Google Cloud and its robust set of APIs, SDKs, and intuitive interface makes it a powerful tool for developers, enabling them to focus on building great user experiences without worrying about infrastructure management. Firebase's scalability, reliability, and extensive features have made it a popular choice for startups and established companies alike, fostering rapid development and deployment of high-quality applications.

REFERENCES

[1] Abdelhay, Mohammed, Ammar Mohammed, and Hesham A. Hefny. "Deep learning for Arabic healthcare: MedicalBot." *Social Network Analysis and Mining* 13.1 (2023): 71.

[2] Boulesnane, Abdennour, et al. "Dzchatbot: a medical assistant chatbot in the algerian arabic dialect using seq2seq model." *2022 4th international conference on pattern analysis and intelligent systems (PAIS)*. IEEE, 2022.

[3] Bao, Qiming, Lin Ni, and Jiamou Liu. "HHH: an online medical chatbot system based on knowledge graph and hierarchical bi-directional attention." *Proceedings of the Australasian computer science week multiconference*. 2020.

[4] Mishra, Prateek, et al. "Personalized Healthcare Chatbot: Dataset and Prototype System." *International Conference on Computational Intelligence in Communications and Business Analytics*. Cham: Springer International Publishing, 2022.

[5] Rahman, Asad & Jour, Ijist. (2022). Health Consultant Bot: Primary Health Care Monitoring Chatbot for Disease Prediction. 4. 201-212. 10.33411/IJIST/2022040115.

[6] Badlani, Sagar & Aditya, Tanvi & Dave, Meet & Chaudhari, Sheetal. (2021). Multilingual Healthcare Chatbot Using Machine Learning. 1-6. 10.1109/INCET51464.2021.9456304.

[7] Lee, Hyeonhoon & Kang, Jaehyun & Yeo, Jonghyeon. (2021). Medical Specialty Recommendations by an Artificial Intelligence Chatbot on a Smartphone: Development and Deployment (Preprint). Journal of Medical Internet Research. 23. 10.2196/27460.

[8] Kolla, Bhanu & Kanagachidambaresan, G.. (2021). Programming with TensorFlow Solution for Edge Computing Applications: Solution for Edge Computing Applications. 10.1007/978-3-030-57077-4.

[9] Mohammad, Abu Bakr & Eissa, Kareem & El-Beltagy, Samhaa. (2017). AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP. Procedia Computer Science. 117. 256-265. 10.1016/j.procs.2017.10.117.

[10] Zhou K, Ethayarajh K, Card D, et al (2022) Problems with cosine as a measure of embedding similarity for high frequency words. arXiv preprint arXiv:2205.05092

[11] Papineni K, Roukos S, Ward T, et al (2002) Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting of the association for computational linguistics. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp 311–318, https://doi.org/10.3115/10730 83.1073135

[12] Meister C, Cotterell R (2021) Language model evaluation beyond perplexity. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: Long Papers). Association for Computational Linguistics, Online, pp 5328–5339, https://doi.org/10.18653/v1/2021.acl-long.414

[13] Naous, Tarek & Hokayem, Christian & Hajj, Hazem. (2020). Empathy-driven Arabic Conversational Chatbot.

[14] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. arXiv preprint arXiv:1811.00207.

[15] The code and dataset are publicly available at: - https://github.com/aub-mind/Arabic-Empathetic-Chatbot

[16] Esfandiari, Nura & Kiani, Kourosh & Rastgoo, Razieh. (2023). A Conditional Generative Chatbot using Transformer Model.

[17] Boussakssou, Mohamed & Ezzikouri, Hanane & Erritali, Mohammed. (2022). Chatbot in Arabic language using seq to seq model. Multimedia Tools and Applications. 81. 1-13. 10.1007/s11042-021-11709-y.

[18] The GitHub link includes code and data: - https://github.com/14H034160212/HHH-An-Online-Question-Answering-System-for-Medical-Questions

[19] https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs

[20] https://www.nhsinform.scot/symptoms-and-self-help/a-to-z

[21] https://www.healthnavigator.org.nz/health-a-z/

[22] https://colab.research.google.com/notebooks/welcome.ipynb

[23] Mishra, Prateek & Dadure, Pankaj & Pranav, K. & SriHarsha, Medisetti & Upadhyay, Devi & Biswas, Nirmita & Pakray, Dr. Partha. (2022). Personalized Healthcare Chatbot: Dataset and Prototype System. 10.1007/978-3-031-10766-5_30.

[24] The dataset is publicly available at https://github.com/Medic-Bot-India/rasaModel

[25] https://rasa.com/docs/rasa/policies#memoization-policy

[26] https://rasa.com/docs/rasa/policies#ted-policy

[27] Hendy, Amr & Abdelrehim, Mohamed & Sharaf, Amr & Raunak, Vikas & Gabr, Mohamed & Matsushita, Hitokazu & Kim, Young Jin & Afify, Mohamed & Awadalla, Hany. (2023). How Good Are GPT Models at Machine Translation? A Comprehensive Evaluation. 10.48550/arXiv.2302.09210