The University of Jordan

Authorization Form

I, Boshra Taha Al-Sadder, authorize the University of Jordan to supply copies of my Thesis/ Dissertation to libraries or establishments or individuals on request, according to the University of Jordan regulations.

Signature:

Date: 15 Dec 2021

USING MACHINE LEARNING TO BUILD RECOGNITION ENGINE FOR ARABIC BOOKING CHATBOT

By

Boshra Taha Rida Al-Sadder

Supervisor

Dr. Gheith Ali Abandah, Prof

This Thesis was submitted in Partial Fulfillment of the Requirements for the Master's Degree of Computer and Networks Engineering

School of Graduate Studies

The University of Jordan

December 21st, 2021

COMMITTEE DECISION

This thesis/dissertation (Using Machine Learning to Build Recognition Engine for Arabic Booking Chatbot) was successfully defended and approved on ----------.

Examination Committee Signature

Dr. Gheith Abandah, (Supervisor) ----------
Prof. of Computer Engineering

Dr. Iyad Jafar, (Member) ----------
Prof. of Computer Engineering

Dr. Ramzi Saifan, (Member) ----------
Associate Prof. of Computer Engineering

Dr. Ali Al-Haj, (Member) ----------
Prof. of Computer Engineering
(Princess Sumaya University for Technology)

ACKNOWLEDGEMENT

I would like to express my gratitude to my thesis supervisor, Prof. Gheith Abandah, who guided me throughout this thesis and helped me finalize it. And I would like to appreciate Prof. Iyad Jafar for his continuous support and guidance throughout my thesis. And I would like to thank my sister, Rahma Al-Sadder, for her cooperation with me, as our theses are related to each other. Also, I would like to thank my family who supported me throughout the study period.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION AND SYMBOLS

Abbreviation	Clarification
ABC Dataset	Arabic Booking Chatbot Dataset
AI	Artificial intelligence
ANERcorp	Arabic Named Entity Recognition Corpus
AraBERT	BERT Model for the Arabic Language
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
BLEU	Bilingual Evaluation Understudy
CA	Classical Arabic
CLS	The Classification Token
CRF	Conditional Random Field Layer
DS	Dialogue systems
IOB	(In/Out/Begin) Format
mBERT	Multilingual BERT Model
MSA	Modern Standard Arabic
NER	Named Entity Recognition
NLP	Natural Language Processing
NLU	Natural Language Understanding
SEP	The Separation Token

USING MACHINE LEARNING TO BUILD RECOGNITION ENGINE FOR ARABIC BOOKING CHATBOT

By
Boshra Taha Al-Sadder

Supervisor
Dr. Gheith Ali Abandah, Prof

ABSTRACT

Chatbots have recently become essential in various fields ranging from customer service and information acquisition to entertainment, as the use of chatbots in customer service reduces operational costs, reduces human errors, and saves time by providing an immediate response at any time.

In this thesis, we propose a new solution for the Arabic booking chatbot model that is dedicated to booking tickets and appointments for seven different domains, which are flights, hotels, cinemas, football matches, cars, restaurants, and clinics. We also build an Arabic booking chatbot dataset (ABC dataset) that contains 76,117 samples and 26 entity classes. We have also collected reservation samples from volunteers to evaluate our model on various real samples. The adopted work that we based on is the state-of-the-art model, AraBERT, a large language model pretrained specifically for the Arabic language. The AraBERT model was fine-tuned to solve the named entity recognition (NER) task on the Arabic Named Entity Recognition Corpus (ANERcorp).

We propose a recognition engine model to extract the reservation entities in the built dataset. We adopt a deep machine learning and transfer learning approaches to develop an accurate and efficient model. We used and fine-tuned the AraBERTv0.2 base model to solve this task. The inputs of our proposed model are text sequences of user reservation

requests, and the outputs are sequences of tags that represent the entities of the input sequences. The proposed model achieved 100% and 95.5% accuracy scores on the ABC test set and the real dataset, respectively. The ABC dataset, the Real dataset, and the source code of our model are publicly available on https://github.com/Boshra-sadder/Arabic-Booking-Chatbot to support chatbot research for the Arabic language.

Chapter 1: Introduction

This chapter introduces the background of this thesis, the research problem, the contribution, the importance, and gives the outline of the thesis.

1.1 Background

Human-computer interaction is a technology that allows interaction between humans and computers using natural languages such as English and Arabic (Almansor and Hussain, 2019). The idea of using natural languages (human language) to communicate with computers can take advantage of artificial intelligence (AI) (Al Humoud *et al*., 2018). However, human language complexity has led to the need for a technology that can understand human language (Almansor and Hussain, 2019). In the past decades, AI and deep learning have shown great development in many areas such as computer vision, natural languages processing (NLP), and speech processing (Al-Ayyoub *et al*., 2018).

The NLP is a branch of AI research that explores the ability of computers to understand human language (Almansor and Hussain, 2019). A chatbot is an example of an AI application that uses natural language (Al-Ghadhban and Al-Twairesh, 2020). A chatbot is a program that enables human-like conversations to be conducted with a computer via auditory or textual methods using a natural language (Shaikh *et al*.,2016).

Chatbots are also referred to as conversational agents or dialogue systems (DS), and these technologies have become very common in our daily life to improve customer service. This includes, but not limited to, instant response to users, providing suggestions and recommendations, and low-cost around the clock customer service (Mnasri, 2019). Hence, many researchers interested in human-computer conversation have shown great interest in this area to make the conversation more rational (Al Humoud *et al*.,2018).

Over the years, many chatbots have been developed for various languages to serve many fields. Unfortunately, there is extremely limited research on Arabic chatbots due to the complexity of the Arabic language, lack of data sources, and the wide spectrum of dialects. Additionally, the processing of Arabic language texts suffers from some challenges such as the rich linguistic structure, high degree of ambiguity, and orthographic variations (Al Humoud *et al*., 2018), The Arabic language is a Semitic language with rich derivational and inflectional features (Al-Ajmi and Al-Twairesh, 2021).

Moreover, the written Arabic text can be classified into three categories. First, Classical Arabic (CA) which is the language of the Qur'an, secondly, the Modern Standard Arabic (MSA) which is the official language used in the Arab world, and the third, dialectal Arabic, which varies according to the region or country (Al Humoud *et al*., 2018).

Conversational agents are classified into two main categories, namely, task oriented and non-task oriented. Task-oriented systems are usually designed based on hand-crafted rules to help users achieve their goals such as making a booking, shopping, or ordering food.

There are two methods to implement the task-oriented systems which are the supervised approach and the unsupervised approach. The supervised approach has several restrictions, such as the process of annotating data and extracting handcrafted features, which are expensive processes and scale poorly. On the other hand, the unsupervised approach learns features automatically from unlabeled datasets (Almansor and Hussain, 2019).

The non-task-oriented chatbots are systems for making an open conversation between a person and a computer without accomplishing any specific task. For example,

it may only be for entertainment. A data-driven approach has been proposed to enable the non-task-oriented dialogue system to learn from the massive amount of conversations that are available publicly, in order to enhance the human-computer interaction (Almansor and Hussain, 2019).

The knowledge source of the chatbot system is the dataset. There are two dataset models that have been developed for this purpose: retrieval-based and generation-based. In the retrieval-based model, the chatbot is trained to provide the most appropriate response from predefined responses. This model relies on a set of rules and keywords that are to be detected in the user's input. This kind of model uses a set of common techniques to build the conversational agent like pattern matching, artificial intelligence markup language, and parsing (Al Humoud *et al*., 2018). The problem with this type occurs when the dataset does not have the required response.

On the other hand, the generation-based models, which require an enormous amount of training data, show the most promising results. The technologies used in these models are neural networks and deep learning techniques (Al Humoud *et al*., 2018). Numerous works in the literature confirm that most of the available work in Arabic task-oriented chatbots is retrieval-based, due to the lack of training data available for task-oriented Arabic chatbots (Al-Ajmi and Al-Twairesh, 2021).

1.2 Research Problem

One of the chatbot applications we are interested in is the booking applications. The booking chatbot is an important and essential application for many companies. There are many chatbots that are specialized to book tickets for flights, hotels, and cinema, *etc*. But most of the current work on modeling booking chatbots is specialized in one domain and a few of them support two domains. According to our knowledge, there is no chatbot that supports multiple booking domains for Arabic.

Also, when the Arabic language is considered, booking chatbots are not only a few but also inefficient. In general, the published studies on Arabic chatbots are rare and there is a gap in knowledge. So, in this research, we highlight the area that needs more study and research.

In this work, I develop an accurate and efficient Arabic booking chatbot, that can analyze and extract booking information for various domains from the user input. This work is done by building a recognition engine using state-of-the-art techniques depending on deep machine learning and transfer learning approaches to obtain accurate and satisfactory results.

This thesis was done in collaboration with another related thesis, Rahma Al-Sadder's thesis, with the title "Developing Domain-Configurable Arabic Booking Chatbot Based on Machine Learning", which looks at how to refine the output of our proposed model given the booking domain type as an additional input to improve the recognition accuracy. In addition to our collaboration to build the ABC dataset and collect samples for the real dataset.

1.3 Research Contributions

As most of the available work in Arabic task-oriented chatbots is retrieval-based due to the lack of training data available for task-oriented Arabic chatbots, this research builds an Arabic dataset for booking chatbots, and makes it available on GitHub to encourage researchers and academics to develop different machine learning models that help expand and develop Arabic language applications in different fields.

Also, we build a recognition engine for Arabic booking chatbots that can recognize reservation entities with high accuracy. This model can be used by companies to build an integrated chatbot for making reservations.

1.4 Research Importance

The primary goal of developing chatbots is to enable human-computer interaction and entertain users by using a natural language. Over the years, numerous chatbots have been developed for many languages to serve many fields such as medicine, education, entertainment, and commerce (Al-Ghadhban and Al-Twairesh, 2020).

Chatbots have become recently essential in our lives. They are increasingly being presented as an alternative method of customer service (Følstad *et al*., 2018). A variety of online chatbots are available for service in various areas ranging from customer service and information acquisition to entertainment (Al Humoud *et al*., 2018).

However, booking chatbots have the potential to offer a much more flexible experience than booking sites. Where customers can save their precious time and delegate the reservation task for the chatbots by typing a free text containing the information necessary to make the reservation.

Many organizations are looking to automate their customer services using AI technologies such as chatbots to provide faster and better assistance to their customers (Almansor and Hussain, 2019). Chatbots can interact with customers at any time by instantly responding to their common queries. Businesses can leverage chatbots to automate bookings of tickets, orders, and appointments so that they reduce operational costs, reduce human errors, and save time (REVE Chat, 2021).

The chatbots provide support and many services to customers. For example, there is a chatbot that specializes in health care that assists specialists in providing health care and following up with the patients through a conversation medium. In the medical field, there is a chatbot that can provide advice to patients, and it can also be used to schedule doctor's appointments. In the business field, commercial chatbots can provide suggestions and help customers to choose the right product. There are also chatbots that

are designed to answer frequently asked questions in specific fields (Fadhil, 2019), and these are just some examples of services that can be provided by chatbots.

The importance of this research lies in building a suitable Arabic dataset for our research problem and choosing the best model to build a recognition engine to recognize the booking entities for several domains that are written in Arabic. This provides companies with an accurate and efficient recognition engine model for the development an Arabic booking chatbots that have the potential to save manpower, money, and, possibly, increase customer satisfaction.

1.5 Thesis Outline

This thesis is organized as follows. We introduce the related work of our research problem in Chapter 2. Chapter 3 includes theoretical background and definitions for the technologies that we use in our research. The methodology for building the Arabic booking chatbot dataset is presented in Chapter 4. The experimental work and results are presented in Chapter 5. Finally, the conclusions and future work for our research are presented in Chapter 6.

Chapter 2: Related Work

This chapter is organized as follows: in the first section, we present papers that include examples of applications of Arabic chatbots and other languages, and in the second section, we present the main papers on which this research is based.

2.1 Chatbots Applications

Chatbot systems are basically software applications used to conduct a conversation between humans and computers using natural languages. These systems have recently become essential in our lives, as there are many organizations that use them to provide customer service.

Dialogue System Development Framework is a set of services, tools, and software development kits that provide a rich foundation or framework for AI chatbot developers easier (Class Central, 2021). For example, it provides web interfaces to create a knowledge base (Ali, 2020). With the advent of these development frameworks, the implementation of the dialogue system has become easier and many DSs have been introduced to serve different tasks. Examples of such frameworks are Wit.ai, Rasa, and IBM Watson Assistant.

In the past few years, many chatbots have been developed for various languages to serve several domains. Unfortunately, there is limited research on Arabic chatbots due to the difficulties of the Arabic language. Al-Hammoud *et al* (2018) surveyed the available research conducted in the field of Arabic chatbots and concluded that there is a scarceness of researches available on Arabic chatbots and that all available works are retrieval-based.

Al-Ajmi and Al-Twairesh (2021) proposed an Arabic dialogue system for flight booking tasks using a hybrid rule-based and data-driven approach utilizing the Wit.ai framework since it has a set of predefined trained entities that support Arabic. This system

is supposed to process the text provided by the user to understand and extract a set of entities such as location, route type, date, time, and ticket class. In addition, the system must be able to recognize whether the user's input contains an intention to book a flight to proceed with the reservation process.

The developed system was evaluated as a whole regardless of what happens within its components, as the evaluation was done by human evaluators. There were 21 participants, and 90.48% of them have previous experience in flight ticket booking. The DS was evaluated in two stages to test the ease of use and the effectiveness of the system. In the first stage, participants evaluated the system to improve the system based on their feedback. Then, the developed models were improved to be tested at the next stage. The overall experience of booking airline tickets using the developed flight booking DS was positive.

Fadhil (2019) introduced OlloBot, an Arabic conversational agent that helps physicians follow up with their patients and provide 24/7 support to patients. This chatbot used the IBM Watson Conversation platform to handle the dialogue flow and to provide AI support for capturing various user intents and entities. After building the chatbot system, it was integrated with the Telegram Bot Platform.

A user experiment with 43 Arabic-speaking users was conducted to evaluate the developed system. The system was tested via a questionnaire consisting of 30 questions to test four main points namely usefulness, ease of use, ease of learning, and user satisfaction with OlloBot reliability. The questionnaires are organized on a scale of 1-5, with 1 “strongly disagree” and 5 “strongly agree”. The results showed good indicators, and most users have also shown interest in the social intelligence of the bot.

Alshareef and Siddiqui (2020) presented an open-domain CA in the Gulf dialect. They used a sequence-to-sequence architecture model that utilized an attention technique

to handle the long dependencies input. The conversational problem was defined as a machine translation problem, and therefore they collected their corpus from tweets in the post–reply format. The proposed model was trained with and without the use of pre-trained word embedding, fastText.

The evaluation of the system was carried out in two ways, the first is by using the Bilingual Evaluation Understudy (BLEU) score and the second is the human evaluation. The results show that the use of embedding gives outputs that are more understandable and more related to the input than those without embedding, as the results were 25.1 and 11.4 BLEU scores respectively with and without the use of pre-trained word embedding.

Ali (2020) proposed a simple design for an English chatbot that as presented in Figure 1. The system architecture contains a very important service for the chatbot system, which is the natural language understanding (NLU) service that is composed of the named entity recognition (NER) model and the Intent classification model. This system is useful for simple tasks that do not require much memory and processing. This Chatbot was used for a transportation network company, and it can be used for other business domains by training the system on a knowledge base of the domain we want.

The proposed system was evaluated using the CoNLL-2003 dataset, and it achieved an accuracy rate of 75.96%. Although this system does not outperform the state-of-the-art NER models, its benefit is that it does not depend on a word sequence (sentence) as input and works well in situations where the data set is business-specific.

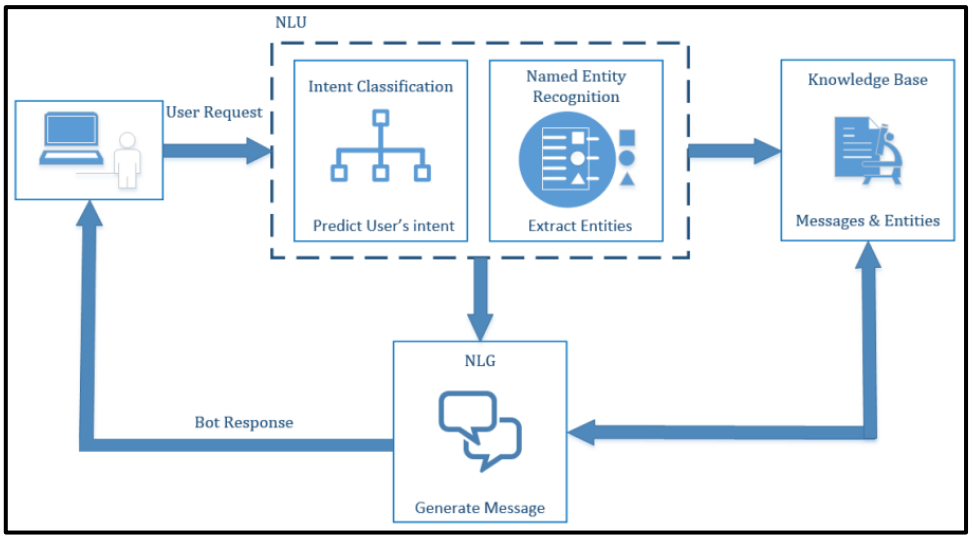

Figure 1. The Architecture of Proposed System (Ali, 2020).

2.2 Related Work

One major task of goal-oriented human-machine conversational understanding systems is to fill a set of slots embedded in a semantic frame or automatically extract important information such as intents and entities to achieve the objective of the dialogue (Mesnil *et al*., 2014). Solving this task is usually done using the NER technique, which seeks to locate and classify named entities mentioned in the user's input into predefined categories such as person names, organizations, and locations.

In the NER task, each token in the sentence is labeled with IOB (in/out/begin) format, where the "B" tag is associated with the first word for the entity, the "I" tag is associated with the rest of the words for the same entity, and the "O" tag denotes that the tagged word is not a desired named entity. Since the task is considered a multi-class classification process, some methods of text classification are used to name the tokens (Antoun *et al*., 2020).

Devlin *et al*. (2018) introduced a deep bidirectional language model called BERT, which stands for bidirectional encoder representations from transformers. Language-specific BERT models are very effective in understanding language, provided they are pre-trained on a very large corpus. The pre-trained BERT model can be fine-tuned with

just one additional output layer to efficiently solve many NLP tasks such as question answering, named entity recognition (NER), *etc*.

Antoun *et al*. (2020) developed the AraBERT model by pre-training BERT on a large-scale Arabic corpus. AraBERT was evaluated on three different natural language processing tasks namely sentiment analysis, question answering, and NER using eight different datasets. The results of AraBERT's experiments are compared with Google's multilingual BERT (mBERT) and other state-of-the-art approaches, where AraBERT outperformed all approaches on most of the tested datasets.

Youssef *et al*. (2020) proposed an end-to-end deep learning NER model for Arabic. This model consists of stacked embeddings that combine the pooled contextual embedding, pre-trained word embeddings (FastText), and BERT embeddings (pre-trained AraBERT model), that are fed into bidirectional long short-term memory with a conditional random field layer (BiLSTM-CRF) as presented in Figure 2. The results showed that this model outperformed all previously published results for deep learning and non-deep learning models on the AQMAR dataset with an F1 score of 77.62%.

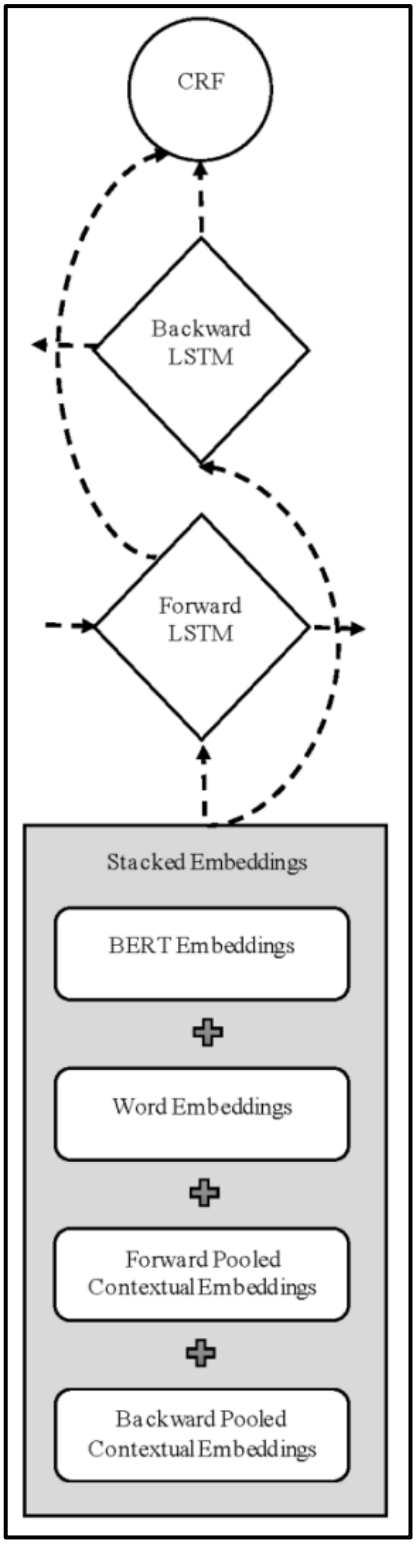

Figure 2. The Proposed NER Model (Youssef *et al*., 2020).

Taher *et al.* (2020) proposed a model for NER in Persian. In the architecture of their model, they used BERT pre-trained model, fully connected layer, and conditional random field (CRF) layer as presented in Figure 3. These two layers were trained on the

representation of tokens that are extracted from BERT. In the NSURL-2019 competition for the NER task for the Persian language, this model achieved second place and the results were 83.5 and 88.4 F1 scores, respectively, in phrase and word level evaluation.

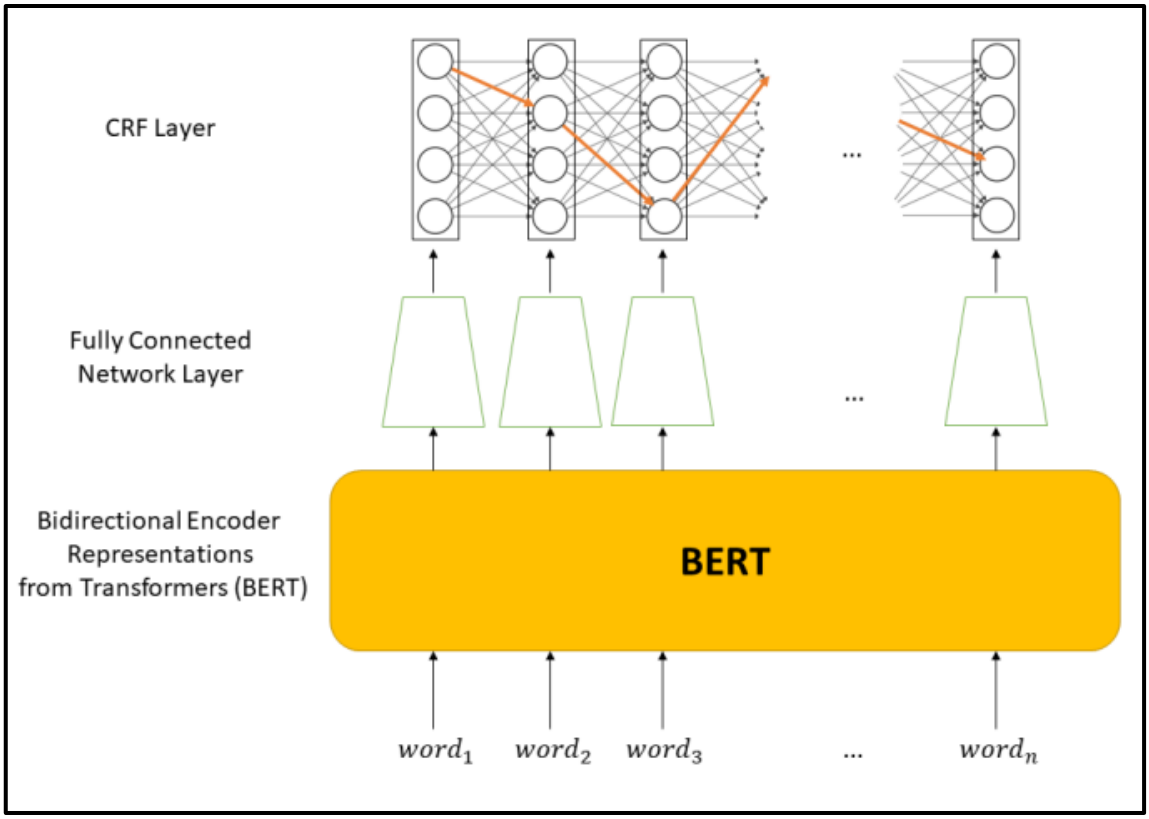

Figure 3. The Proposed NER Model for Persian (Taher *et al.*, 2020).

The main function of the booking chatbots is to extract the reservation information from user input. And to solve this task we use the named entity recognition (NER) technique. So, in this research, we develop a recognition engine model for Arabic booking chatbots based on the state-of-the-art NER solution (Antoun *et al.,* 2020).

Chapter 3: Technologies Used

This chapter explains the main techniques used in this thesis.

3.1 Transfer learning

It is a machine learning technique that takes the relevant parts of a pre-trained machine learning model and applies them to solve a different but related problem. This technique is often used when developing a model to solve a task that requires a huge amount of resources and requires a large amount of labeled data (Trotter, 2021).

3.2 Transformers library

It is a Python library that provides several pre-trained models that efficiently solve NLP tasks such as information extraction, text summarization, and classification in more than 100 languages. The BERT model is an example of a large pre-trained language model (Python Package Index, 2021).

3.3 BERT

The BERT, bidirectional encoder representations from transformers, is a deep bidirectional language model that is very effective in understanding language. BERT is a multi-layer bidirectional transformer encoder, the BERT-Base model consists of 12 layers, 768-hidden size, 12 self-attention heads, and 110M parameters, while the BERT-Large model consists of 24 layers, 1024-hidden size, 16 self-attention heads, and 340M parameters (Devlin *et al*., 2018).

The BERT model is used for many NLP tasks such as question answering, NER, and language inference (Devlin *et al*., 2018). This bidirectional model can read the text input once unlike the unidirectional models, which read the text input sequentially from one direction, this feature allows the BERT model to learn the context of the word (Horev, 2018) in both directions.

In the pre-training stage of the BERT transformer, the BERT model is trained with an unsupervised approach on a very large corpus of data over two tasks, the masked language models (MLM) task, and the next sentence prediction (NSP) task.

This transformer has a very large number of parameters that can be fine-tuned for a certain NLP task, by adding an additional layer or more on top of BERT that fits the shape of the output and then training the entire model together. In this stage, the model parameters are first initialized with the pre-training parameters and then are fine-tuned using the supervised approach, by training the model using a labeled dataset associated with the NLP task (Devlin *et al.,* 2018).

This means that most NLP tasks have the same model architectures except for the output layer. Figure 4 shows two different NLP tasks which are the single sentence tagging task and the sentence pair classification task, for each task type, the input and the output of the model are different.

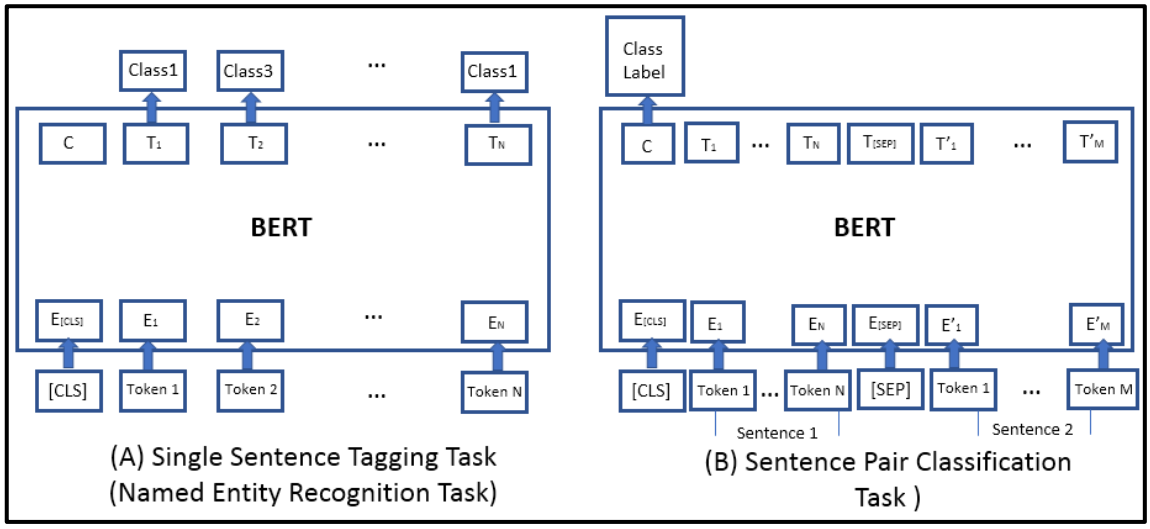

Figure 4. Using BERT Model with Different NLP Tasks. Modified after (Devlin *et al.,* 2018).

As we mentioned that BERT is used for NLP tasks, meaning that the BERT input is text, but before it is entered into BERT, the text is broken down into words (tokens). In addition, some special tokens are entered with the input tokens such as the classification

token (CLS) that is used at the beginning of BERT inputs and the separation token (SEP) which is used to separate the parts of the input. The output of BERT is a sequence of vectors (Shulga, 2019).

If the NLP task is a classification task such as the single sentence classification task or the sentence pair classification task, then we use the output of the CLS token (the first token), see Figure 4B. And for other tasks with complex output like single sentence tagging tasks and question answering tasks, we can use the output of the other tokens, see Figure 4A (Shulga, 2019).

3.4 AraBERT

The AraBERT was produced by pre-training BERT on a large-scale Arabic corpus. This model was evaluated for three NLP tasks namely sentiment analysis, question answering, and NER using eight different datasets. The results of the AraBERT experiments showed that the AraBERT model outperforms all state-of-the-art approaches on most of the tested datasets (Antoun *et al*., 2020).

In our work, we use the AraBERTv0.2-Base model from the arabert repository on GitHub, because the recent versions (v0.2/ v2) of AraBERT are trained on a larger dataset and have better-preprocessing functionality than the previous versions (v0.1/ v1). Table 1 shows the different AraBERT versions (Antoun, *et al.,* 2021).

The difference between versions (v0.1/ v0.2) and versions (v1/ v2) is that the versions (v1/ v2) require pre-segmentation of the input before it is fed into the model. Antoun *et al*. (2020) presented their experiments showing that using AraBERTv0.1 with the NER task gives better results than using AraBERTv1, which needs pre-segmentation of the inputs.

Table 1. Six Different Versions of AraBERT (Antoun *et al*., 2021).

Model	HuggingFace Model Name	Pre-Segmentation	DataSet (Sentences/Size/nWords)
AraBERTv0.2-base	bert-base-arabertv02	No	200M / 77GB / 8.6B
AraBERTv0.2-large	bert-large-arabertv02	No	200M / 77GB / 8.6B
AraBERTv2-base	bert-base-arabertv2	Yes	200M / 77GB / 8.6B
AraBERTv2-large	bert-large-arabertv2	Yes	200M / 77GB / 8.6B
AraBERTv0.1-base	bert-base-arabertv01	No	77M / 23GB / 2.7B
AraBERTv1-base	bert-base-arabert	Yes	77M / 23GB / 2.7B

3.5 Tokenization

The tokenization process means splitting the input text into smaller parts. And there are three types of tokenization which are word tokenization, character tokenization, and sub-word tokenization (Hugging Face, 2020).

Word tokenization results in a very large vocabulary that leads to problems for a huge text corpus. Whereas character tokenization is simpler and does not need a complex memory and time, but this leads to the problem that the model is difficult to learn meaningful input representations. So, transformer models use the sub-word tokenization that uses a hybrid method between the two previous types (Hugging Face, 2020).

The Sub-word tokenization algorithm only breaks the words that are rarely used, while the frequently used words are not broken down, and this allows for a reasonable vocabulary size with the ability to better represent complicated words. In addition, it can deal with out-of-vocabulary words by breaking them down into sub-words that are known (Hugging Face, 2020). Figure 5 shows an example of a sentence that has been tokenized and one of its words is not in the vocabulary, so this word has been broken down.

```
import transformers
BASE_MODEL_PATH = 'aubmindlab/bert-base-arabertv02'
TOKENIZER = transformers.BertTokenizer.from_pretrained(BASE_MODEL_PATH)

tokens =TOKENIZER.tokenize("  مرحبا اريد حجز تذكرة لحضور مبارة كرة القدم ")
print(tokens)
# ['مرحبا' , 'اريد' , 'حجز' , 'تذكرة' , 'لحضور' , 'مبار' , '##ة' , 'كرة' , 'القدم']
```

Figure 5. The Code Showing the Output of Using the Tokenizer from the Arabertv0.2-Base Model to Tokenize a Sentence.

Note that all the words in the previous sentence existed in the vocabulary of this BERT tokenizer except for the ("مبارة") word, and thus it was broken down into smaller sub-words: ['##ة' ,'مبار']. This symbol ("##") is used in the detokenization process, and this means that the token with this symbol must be concatenated to the previous token, without a space (Hugging Face, 2020).

3.6 AraBERT Pre-processor Class

The preprocessor class can be imported from the preprocess package in the arabert repository. This class is used for cleaning and preprocessing input text for all models in the arabert repository. Some of the preprocessing functions are inserting white space between words and numbers and *vice versa*, replacing slashes with a dash, removing diacritics like (FATHATAN and SHADDA), applying segmentation, *etc*. (Antoun *et al.,* 2021). Figure 6 shows the AraBERT pre-processor class.

Also, we can use the "unprepocess" function to reverse some preprocessing changes, by removing the added space around non-alphabetical characters such as (? ' -) also, de-segmentation is used when the selected model needs pre-segmentation (Antoun *et al.,* 2021).

```
ArabertPreprocessor(
  model_name= "",
  keep_emojis = False,
  remove_html_markup = True,
  replace_urls_emails_mentions = True,
  strip_tashkeel = True,
  strip_tatweel = True,
  insert_white_spaces = True,
  remove_non_digit_repetition = True,
  replace_slash_with_dash = None,
  map_hindi_numbers_to_arabic = None,
  apply_farasa_segmentation = None
)
```

Figure 6. The AraBERT Pre-processor Class (Antoun *et al.*, 2021).

3.7PyTorch Library

PyTorch is an open-source deep-learning library developed by Facebook that performs computations efficiently. The new PyTorch API provides classes that make developing deep learning models easy. The easy and flexible use of PyTorch has made PyTorch a favorite library for many academics and researchers (Brownlee, 2020). And we developed our models using PyTorch deep learning library.

3.8 Stratified Train-Test Splits

It is a way to split data into train-test sets by returning the indices of the samples that maintain the same percentage of samples for each category as observed in the original dataset (Brownlee, 2020).

3.9 Evaluation Metrics

Evaluation metrics are used to determine the performance of the machine learning model. This is usually done by training the model on the training dataset, then making predictions using the trained model on the test dataset, and then comparing the resulting values with the expected ones. In our experiments, we used evaluation metrics from the Sklearn library (Scikit Learn, 2021). These metrics are:

1) **Accuracy Score:** It is the ratio of the number of correctly predicted cases to the total number of data samples.
2) **Precision Score:** It is a performance measure of the classifier's ability to not label a negative sample as positive. And it is calculated with the following ratio:

$$Precision = True\ Positive\ /\ (True\ Positive\ +\ False\ Positive) \quad (1)$$

3) **Recall Score:** It is a performance measure of the classifier's ability to find all positive samples. And it is calculated with the following ratio:

$$Recall\ =\ True\ Positive\ /\ (True\ Positive\ +\ False\ Negative) \quad (2)$$

4) **F1 Score:** It is the harmonic mean of the precision and recall.

$$F1\ =\ 2\ \times\ (precision\ \times\ recall)\ /\ (precision\ +\ recall) \quad (3)$$

5) **F1 Score (macro avg):** Finds the precision and recall metrics for each class, then calculates the arithmetic mean for all the precision (P) values and for all the recall (R) values, finally calculates F1 score for these averages. This type does not consider the imbalance in labeling.
6) **Classification Report:** It is a report showing the main classification metrics like accuracy and F1 Score.
7) **Confusion Matrix:** A technique used to evaluate the performance of a classification model by showing the number of correct and incorrect predictions and broken down by each category (Brownlee, 2016).

Chapter 4: Building the Arabic Booking Chatbot Dataset

In this chapter, we explain in detail how we built the Arabic booking chatbot dataset (ABC Dataset). And how we split it into training, validation, and test sets for training the recognition engine model. Also, we collected a set of booking templates from volunteer people and named it Real Dataset.

To train a machine learning model, we need several thousand samples to get excellent and satisfactory results. But writing a huge number of booking samples for different domains and tagging each word manually will be very difficult.

Therefore, we proposed a solution to build a large dataset, by thinking of different templates through which reservations are made for several domains. Each template contains reservation entities related to a specific reservation domain. We wrote a program that produces many samples for each template so that the reservation entities are changed per each sample.

4.1 The Methodology Used to Build the ABC Dataset

In this section, we explain in detail the methodology used in building the dataset. Initially, we identified seven domains for ticket or appointment reservations. Then we defined the reservation entities needed for these domains and wrote a set of templates for each domain. These templates contain a set of reservation entities related to the adopted domains. Table 2 shows these domains and the number of templates we have written for each domain.

Table 2. The Seven Domains for Reservations and The Number of Templates for each Domain.

ID	Domain	Number of Templates
D1	Flights Booking	12
D2	Hotel Booking	5
D3	Cinema Booking	5
D4	Football Match Booking	7
D5	Car Booking	6
D6	Restaurant Booking	7
D7	Clinic Appointment Booking	10
Total	7 Domains	52 Templates

Then we gave each entity a specific number as its tag (label). In addition, we defined the “Others” entity with a “0” tag. This tag is used for each word in the reservation sentence that does not belong to any of the predefined entities. Table 3 shows the names of the reservation entities that we defined for the seven domains and the tag assigned for each entity.

Table 3. Twenty-six Entities for the Seven Reservation Domains, with a Tag Assigned to each Entity.

Tag	Entity Name
0	Others
1	City 1 (Departure City)
2	City 2 (Arrival City)
3	Date and Day 1 (Start Date)
4	Time 1 (Start Time)
5	Date and Day 2 (End Date)
6	Time 2 (End Time)
7	Ticket Class
8	Round Trip
9	Number of Tickets
10	Name (Customer Name)
11	Doctor Name
12	Phone (Phone Number)
13	First Team
14	Second Team
15	League (League Name)
16	Hotel (Hotel Name)
17	Room (Number of Rooms)
18	Restaurant (Restaurant Name)
19	Movie (Movie Name)
20	Cinema (Cinema Name)
21	Car Name
22	Car Model
23	Period of Time
24	Stadium Name
25	Clinic Name

After that, we prepared the templates in a way that enables us to enter them into the program to produce many samples for each template that differ in the values of the reservation entities. For this purpose, we changed each template from plain text to string with variables, where each variable corresponds to an entity in the template. Figure 7 shows that.

```
"اريد حجز رحلة ذهاب من  الرياض اللى الامارات لطفلين و ٣ بالغين يوم الخميس ٢٠٢١/٦/٢٦ الساعة ٩ مساء على الدرجة الاولى "

"اريد حجز رحلة ذهاب من "+ الرياض+ " الى " +الامارات +"" +"لطفلين و ٣ بالغين  +"" +"  يوم الخميس ٢٠٢١/٦/٢٦  "الساعة"+٩ مساء + " على " + الدرجة الاولى

"اريد حجز رحلة ذهاب من "+City1+ " الى " + City2  + Numbers +"" "+  DateAndDay1  +""+ الساعة " +Time1+ " على " + TicketClass
```

Figure 7. An example of a Reservation Template for the Flights Domain and Converting it into a String-Variable Format.

To change the values of the entity variables in each sample produced, we created a dictionary containing all pre-defined reservation entities, and a different set of values was set for each entity. Table 4 shows the number of values given to each entity.

Table 4. The Entities for the Seven Reservation Domains, with the Number of Values Used for each Entity.

Entity	Number of Values
Time 1	49
Time 2	49
Date and Day 1	48
Date and Day 2	48
Name	39
Number of Tickets (Flights, Hotels)	11
Number of Tickets (Cinemas, Football Matches, Restaurants)	21
City 1	47
City 2	47
Round Trip	3
Ticket Class (Flights)	5
Car Name	31
Car Model	10
Movie	20
Cinema	4
Hotel	12
Room	5
Ticket Class (Cinemas, Football Matches)	4
League	5
First Team	27
Second Team	27
Restaurant	15
Doctor Name	39
Phone	30
Period of Time	13
Stadium Name	11
Clinic Name	9

After that, we converted this dictionary into a dictionary with tags instead of values. The left column of Table 5 shows the “Flight Ticket Class” entity values, and the right column shows the “Flight Ticket Class” entity tags. Then we converted each string in the templates into tags, as shown in Figure 8.

Table 5. The Values and Tags for the “Flight Ticket Class” Entity as Defined in the Values Dictionary and the Tags Dictionary.

Ticket Class (Flights)	Ticket Class (Flights)
درجة اقتصادية	7 7
الدرجة الاولى	7 7
درجة اولى	7 7
درجة رجال الاعمال	7 7 7
درجة اعمال	7 7

```
"اريد حجز رحلة ذهاب من "+City1 + " الى " + City2 + " "+ Numbers +" "+ DateAndDay1
الساعة " +Time1+ " على " + TicketClass

"0 8 0 0 0"+" "+ NumbersTag +" "+ CityTag2 + " 0 " +CityTag1 +" "+ DateAndDayTag1
0 " + TimeTag1+ " 0 " + TicketClassTag
```

Figure 8. Example of Template Conversion from String-Variable Format to Tags-Variable Format.

We built a program that generates samples for each template using the “Nested For Loop” method. So that the number of loops is equal to the number of reservation entities in the template. In each loop, the possible values of the entity are read, where the program generates for each sample a sentence and a sequence of tags that represent the entities of the sentence.

The total number of samples that were generated for all domains is 76,117 samples. Table 6 shows the distribution of these samples by domain type and template type, where the samples for a specific template are labeled with its domain ID and template ID. The last column of the table shows that.

Table 6. Number of Samples per Template, Total Number of Samples in Each Domain, Total Number of Samples in The ABC Dataset, and Label of Samples per Template.

Domain	Template	Samples per Template	Domain-Template
Flights Booking D1	T1	11,760	D1T1
	T2	6,072	D1T2
	T3	1,500	D1T3
	T4	1,100	D1T4
	T5	1,950	D1T5
	T6	500	D1T6
	T7	5,500	D1T7
	T8	770	D1T8
	T9	500	D1T9
	T10	45	D1T10
	T11	500	D1T11
	T12	240	D1T12
Samples per Domain		**30,437**	
Hotel Booking D2	T1	1,584	D2T1
	T2	564	D2T2
	T3	132	D2T3
	T4	1,584	D2T4
	T5	18,612	D2T5
Samples per Domain		**22,476**	
Cinema Booking D3	T1	2,000	D3T1
	T2	2,500	D3T2
	T3	3,000	D3T3
	T4	780	D3T4
	T5	80	D3T5
Samples per Domain		**8,360**	
Football Match Booking D4	T1	1,000	D4T1
	T2	480	D4T2
	T3	660	D4T3
	T4	1,600	D4T4
	T5	500	D4T5
	T6	624	D4T6
	T7	735	D4T7
Samples per Domain		**5,599**	
Car Booking D5	T1	403	D5T1
	T2	403	D5T2
	T3	1,300	D5T3
	T4	310	D5T4
	T5	400	D5T5
	T6	900	D5T6
Samples per Domain		**3,716**	
	T1	315	D6T1

Restaurant Booking D6	T2	720	D6T2
	T3	175	D6T3
	T4	192	D6T4
	T5	315	D6T5
	T6	550	D6T6
	T7	624	D6T7
Samples per Domain		**2,891**	
Clinic Appointment Booking D7	T1	400	D7T1
	T2	686	D7T2
	T3	400	D7T3
	T4	45	D7T4
	T5	240	D7T5
	T6	380	D7T6
	T7	39	D7T7
	T8	100	D7T8
	T9	300	D7T9
	T10	48	D7T10
Samples per Domain		**2,638**	
Samples per Dataset		**76,117**	

Table 7 shows the structure of the ABC dataset file. Each record in the dataset has four columns. The first column contains the sentence through which the reservation is made, the second column contains a sequence of tags that represent the entities of the sentence, the third column contains the domain id, the fourth column contains the label of the sample (domain ID-template ID).

Table 7. The Structure of The ABC Dataset File.

Sentences	Tags	Domain	Domain-Template
مرحبا انا علي عوده الله بدي اسافر من البرازيل الى الصين بيوم 6 الشهر بتذكرة طيران من درجة اقتصادية	0 0 10 10 10 0 0 0 1 0 2 3 3 3 9 0 0 7 7	1	D1T5
بدي انزل بفندق لو رويال في الإسكندرية واحجز 4 غرف نوم ل 3 اشخاص كبار	0 0 0 16 16 0 1 0 17 17 0 0 9 9 9	2	D2T5
لو سمحت احجزلي تذاكر طيران ذهاب وعودة من الدمام الى القاهرة لشخصين بالغين و 3 اطفال على درجة اولى	0 0 0 9 0 8 8 0 1 0 2 9 9 0 9 9 0 7 7	1	D1T7

4.2 Splitting the ABC Dataset

We divided the ABC dataset into a training set, a validation set, and a test set. So that the machine learning model is trained using the training set. After each epoch, the model is validated using the validation set. Finally, the performance of the model is checked using the test set. The split was done using the stratified train-test split based on domain and template type. Table 8 shows the percentage and the number of samples for each set.

Table 8. The Number of Samples for Training Set, Validation Set, and Test Set in the ABC Dataset.

Set Name	Training Set	Validation Set	Test Set
percentage	90%		10%
	90%	10%	
Number of Samples	61,654	6,851	7,612
Total Samples	76,117		

4.3 Real Dataset

After we evaluated our proposed model, the testing accuracy was 100%. This is a reflection on the fact that AraBERT which is an Arabic trained transformer is used, in addition to the fact that there is a high similarity between the training set and the test set. Hence, we decided to build an additional testing dataset from volunteers.

This dataset is referred to as the Real Dataset and it contains reservation queries in different domains. Participants were asked to freely write reservation queries. We collected this dataset from 38 volunteers, and it contains 200 reservation requests spread across the seven domains and we labeled them manually. Table 9 shows the distribution of these samples for the different domains.

Table 9. The Number of Samples for Each Domain in the Real Dataset.

Domain	D1	D2	D3	D4	D5	D6	D7
Number of Samples	30	24	29	29	27	30	31
Total	200						

Table 10 shows the structure of the Real dataset file. Each record in the dataset has five columns. The first column contains the sentence written by the volunteer to make a reservation in a specific domain. the second column contains the label of samples (domain ID). the third column contains the volunteer name. the fourth column contains a sequence of tags that represent the entities of the sentence. the fifth column contains the domain id.

Table 10. The Structure of The Real Dataset File.

Sentences	Domain ID	Volunteer Name	Tags	Domain
مطلوب حجز موعد لعيادة الاسنان عند الدكتور نزار غنام الساعة الخامسة مساءا	D1	نزار جهاد هاشم عبد الواحد	0 0 0 0 25 0 0 11 11 0 4 4	1
مرحباً بدي حجز طاولة في مطعم جبري ع التراس لشخصين بدون ارجيلة.	D6	ايناس علي عبد الحليم الحياصات	0 0 0 0 0 0 18 0 0 9 0 0	6
لو سمحت اريد حجز سيارة ميتسوبيشي باجيرو لتلات ايام وانكم تحضروها لموقعي في جبل الحسين شارع المجدل عمارة 30 ، و يتم تسليمها من قبلي في المكتب عندكم	D5	هدى طه رضا الصدر	0 0 0 0 0 21 21 23 23 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	5

Chapter 5: Experimental Work and Results

This chapter includes an explanation of the platform used to perform our experiments, an explanation of the state-of-the-art model used, the presentation of the experimental work, and the results presentation and discussion.

5.1 Experiments Platform and Setup

This section includes a description of the platform used to build the dataset and to conduct the experiments, as well as the necessary setup.

5.1.1 Experiments Environment

In our experiments, we used Kaggle notebooks to build the ABC dataset and build and test the models. The Kaggle platform is a free-of-charge environment that runs inside the browser and it allows users to find and publish datasets, write, save and execute code on Jupyter notebooks using powerful computing resources like CPUs, GPUs, and TPUs (Yufeng, 2017).

This saves time and effort rather than setting up a local environment. In addition to that, the processing power comes from the servers in the cloud and not from a local machine. Also, accessing the notebook can be from anywhere that the Internet is available (Yufeng, 2017).

5.1.2 Experiments Setup

With the Jupyter notebook environment, we don’t need to make a complicated setup. Where easily we can obtain the needed libraries by using the pip installation statements and the import statements. Figure 9 shows the libraries that we used in our code and their versions.

```
!git clone https://github.com/aub-mind/arabert
!pip install openpyxl
!pip install xlsxwriter
from arabert.preprocess import ArabertPreprocessor

import transformers
from tqdm import tqdm

import torch
import torch.nn as nn
import torchvision.models as models

import copy
import numpy as np
import pandas as pd
import xlsxwriter
from datetime import datetime
from collections import Counter
import matplotlib.pyplot as plt

from sklearn.model_selection import StratifiedShuffleSplit
from sklearn.metrics import accuracy_score, f1_score,
    precision_score,recall_score, classification_report,confusion_matrix
```

```
transformers==4.5.1
tqdm==4.62.3
torchvision==0.10.1
torch==1.9.1
scikit-learn==0.23.2
pandas==1.3.3
numpy==1.19.5
matplotlib==3.4.3
```

Figure 9. The Libraries and their Versions that we used for the Proposed Work.

5.2 Base Models

This section explains the state-of-the-art model that we used to build our model. Where we build a recognition engine for an Arabic booking chatbot that can book tickets for various domains in an efficient way.

5.2.1 AraBERTv0.1 Model_ ANERcorp

The AraBERTv0.1 model is considered the state-of-the-art solution to solve the NER task in the Arabic named entity recognition corpus (ANERcorp), where this model achieves an F1 score of 84.2. ANERcorp contains 16.5 k entities divided into 4 categories of entities, person, organization, location, and miscellaneous (Antoun *et al*., 2020). The IOB2 (in/out/ begin) format was used to label this corpus. Table 11 shows all label types and the frequency of each label in both Train-ANER Corp and Test-ANER Corp (Antoun *et al*., 2021).

Table 11. The Categories and their Frequencies in the Training Set and Test Set of ANERCorp (Antoun *et al*., 2021).

ANERcorp	O	B-PERS	I-PERS	B-LOC	I-LOC	B-ORG	I-ORG	B-MISC	I-MISC
Train Set	111,921	2,721	2,205	3,776	525	1,576	1,115	888	375
Test Set	21,633	858	641	668	83	450	275	235	165

This model was developed using Python programming language, PyTorch library, and AraBERTv0.1 transformer. They fine-tuned the AraBERTv0.1 model by adding two new layers, a dropout layer of 0.3 and a linear output layer with 9 outputs that correspond to the number of classes at the top of this model, and then trained the entire model together on Train-ANER Corp. Adam optimizer with a learning rate scheduler was used to compile the model and the cross-entropy function was used as a loss function (Antoun *et al*., 2021).

For training, they used the cross-validation iterator with 5 folds. In each fold, the model was trained for 5 epochs, in each epoch, the model was trained using 4 out of 5 folds of the training set with a batch size of 32. The model predicted the tags of the validation set, the remaining fold, with a batch size of 8. At the end of each epoch, they checked the F1 score value with the previous epoch in order to save the model with the greatest F1 score.

Where the best F1 result on the training set was an 84.9 F1 score in the third epoch of the fourth fold (Antoun *et al*., 2021). The results of the evaluation on the test set are not available on GitHub, but as we mentioned earlier, the result announced in the papers is an 84.2 F1 score (Antoun *et* al., 2020).

5.3 Experimental Works

To develop an effective model and use it to build an Arabic chatbot that can book tickets for various domains, we need to develop a recognition engine that can extract reservation entities with high accuracy. Since the AraBERT is considered the most recent solution used to solve the NER task at ANERcorp (Antoun *et al.,* 2020), we develop our work based on it. We built a recognition engine using the AraBERTv0.2_base model because this version is trained on a larger dataset and has better preprocessing functionality than the previous version (AraBERTv0.1). Our proposed model solves the

NER task on the ABC dataset. This dataset is for ticketing and appointments, it also contains more categories than those at ANERcorp. and we start by loading and preparing this dataset.

5.3.1 Load and Prepare the ABC Dataset

The ABC dataset was stored in an Excel file named "ABC_Dataset". In this part, we explain how to load and prepare this dataset for use in training and validating our machine learning model.

5.3.1.1 Load the Data

We load the ABC dataset by reading the Excel file (ABC_Dataset) as a DataFrame structure using Python Pandas. Figure 10 shows some rows of the dataset. We converted the first and second columns of the DataFrame into lists, one for sentences and one for labels, each with a length of 76,117.

	Sentences	Tags	Domain	Domain-Template
0	مرحبا انا علي عوده الله بدي اسافر من البرازيل...	0 0 10 10 10 0 0 0 1 0 2 3 3 3 9 0 0 7 7	1	D1T5
1	... بدي انزل بفندق لو رويال في الإسكندرية واحجز	0 0 0 16 16 0 1 0 17 17 0 0 9 9 9	2	D2T5
2	لو سمحت احجزلي تذاكر طيران ذهاب وعودة من الد...	0 0 0 9 0 8 8 0 1 0 2 9 9 0 9 9 0 7 7	1	D1T7
3	اريد حجز رحلة ذهاب من الهند الى برج العرب لشخ...	0 0 0 8 0 1 0 2 2 9 9 9 3 0 4 4 0 7 7	1	D1T1
4	بدي انزل بفندق الانتركونتيننتال في هولندا وا...	0 0 0 16 0 1 0 17 17 0 0 9 9 9	2	D2T5
...	...	...	...	...
76112	لو سمحت احجزلي تذاكر طيران ذهاب وعودة من الم...	0 0 0 9 0 8 8 0 1 1 0 2 2 9 9 0 7 7	1	D1T7
76113	بدي انزل بفندق ماريوت في عمان واحجز 3 غرف ن...	0 0 0 16 0 1 0 17 17 0 9 9	2	D2T5
76114	اريد حجز رحلة ذهاب من روسيا الى شرم الشيخ لشخص...	0 0 0 8 0 1 0 2 2 9 9 0 9 9 3 0 4 4 0 7 7	1	D1T1
76115	... ارغب بحجز 7 تذاكر لفيلم في المنزل بمفردي يوم	0 0 9 9 0 19 19 19 3 3 0 4 4 20 0 0 20 20 0 0...	3	D3T2
76116	اريد حجز رحلة ذهاب من عدن الى ديار بكر ل 3 اشخ...	0 0 0 8 0 1 0 2 2 0 9 9 9 0 3 3 0 4 4 4 0 7 7	1	D1T1

76117 rows × 4 columns

Figure 10. The ABC Dataset is Represented as a DataFrame.

As this file contains the three sets of data as we mentioned in Chapter 4 (the first 7,612 samples for test data, the next 6,851 samples for validation data, and the last 61,654 samples for training data), We divided the sentences list into three lists (Sentence-Test,

Sentence-Val, Sentence-Train). As well as for the labels list. Figure 11 shows all these lists and their names in detail.

```
columns = pd.read_excel (r'../input/dataset-file/ABCD.xlsx', sheet_name='ABC_Dataset',engine = 'openpyxl')
train_samples=61654
test_samples=7612
val_samples=6851
samples = test_samples+val_samples

Sentences =columns.values.T[0].tolist()
labels =columns.values.T[1].tolist()

#Test 7612
Sentences_Test =Sentences[:test_samples]
labels_Test =labels[:test_samples]
data_test= preprossingFun(Sentences_Test, labels_Test)

#Val   6851
Sentences_Val =Sentences[test_samples: samples]
labels_Val =labels[test_samples: samples]
data_val= preprossingFun(Sentences_Val, labels_Val)

#Train   61654
Sentences_Train = Sentences[samples:]
labels_Train =labels[samples:]
data_train= preprossingFun(Sentences_Train, labels_Train)
```

Figure 11. A Piece of Code Representing How to Read the Dataset and Divide it into 3 Sets (data_test, data_val, data_train).

5.3.1.1.2 Prepare the Data

When using the pre-process function from the ArabertPreprocessor class. A space is added around the special characters and this leads to a problem when dividing a sentence into words based on the space. Since some words contain a special character, this splits that word into more than one token. Where each word in the sentence corresponds to only one tag as our dataset built. And thus, we created a new preprocess function that uses the preprocess/unpreprocess functions from the ArabertPreprocessor class as well as making some modifications.

In this new preprocess function, we first remove some unimportant special characters, such as the plus symbol. But some other symbols are necessary such as the colon, it is mostly used to express time like (4:30), and if there is a space around the colon, this token will be split into two parts. So, we delete the space around the colon symbol

after applying the preprocess/unpreprocess functions from the ArabertPreprocessor Class. Figure 12 shows how the processing function adds space around the special symbol.

```
!git clone https://github.com/aub-mind/arabert

from arabert.preprocess import ArabertPreprocessor
ArabertPrep = ArabertPreprocessor(model_name=" aubmindlab/bert-base-arabertv02")

string=' ممكن انه احجز موعدي مع الدكتور عبد الرحمن العزة اليوم على الساعة 3:30 '
preprocess_text = ArabertPrep.preprocess(string)
newString = ArabertPrep.unpreprocess(preprocess_text)
print(preprocess_text) #30 : 3 ممكن انه احجز موعدي مع الدكتور عبد الرحمن العزة اليوم على الساعة
print(newString)       #30 :3 ممكن انه احجز موعدي مع الدكتور عبد الرحمن العزة اليوم على الساعة
```

Figure 12. Appling the preprocess/unpreprocess Functions from The ArabertPreprocessor Class to a Sentence Containing the Special Colon Character.

At the end of the new preprocess function, we split each pre-processed sentence based on space and convert them to a list. As well as for its corresponding sequence of tags. Finally, they are grouped into a tuple, so each instance in the Train data, Test data, or Val data is about a tuple of 2 lists. One for tokens and one for tags, where the two lists must be equal in length for all samples. Figure 13 shows one instance from the data train list.

```
print(data_train[18133])
```

```
(['بدي', 'انزل', 'بفندق', 'لو', 'رويال', 'في', 'الرياض', 'واحجز', 'ل
'4', 'غرف', 'نوم', 'لشخص', 'مسن', 'وشخصين', 'بالغين', 'وطفل',
واحد''], ['17', '0', '1', '0', '16', '16', '0', '0', '0
9', '9', '9', '9', '9', '9', '0', '17''])
```

Figure 13. A Sample of The Data Train List is a Tuple Containing a List of Tokens and A List of Tags.

Before training a PyTorch neural network we need to create a custom dataset class that matches our problem. This class is used to convert the data to PyTorch tensors, and then serve up the data in batches using data loader object from DataLoader interfaces. Where each sentence will be represented by 4 torch tensors which are (Input Ids, Input Mask, Segment Ids, and Label Ids).

5.3.2 AraBERTv0.2-Base Model ABC Dataset

This experiment aims to build an accurate and efficient recognition engine capable of extracting and classifying reservation entities into pre-defined categories such as people names, time expressions, day, and date expressions, *etc*. Where the user enters text to book an appointment or ticket for several domains including flights, hotels, cinemas, football matches, cars, restaurants, and clinics.

In this experiment, we solve the NER task on the ABC dataset, which contains 26 entities categories, 76,117 sentences with their corresponding sequence of tags. Table 12 shows all label types and the frequency of each label in the training set, validation set, and test set for the ABC dataset.

Table 12. The Categories and their Frequencies in the Training Set, Validation Set, and Test Set of ABC Dataset.

ABC Dataset				
Class	**Label**	**Train Set**	**Val Set**	**Test Set**
Others	0	542,049	60,291	66,881
City1	1	50,671	5,610	6,277
Name	10	13,551	1,506	1,719
Doctor Name	11	5,626	645	691
Phone	12	1,105	123	137
First Team	13	5,662	621	699
Second Team	14	4,652	514	576
League	15	4,908	562	617
Hotel	16	29,131	3,269	3,589
Room	17	36,519	4,060	4,505
Restaurant	18	3,592	400	432
Movie	19	13,393	1,458	1,626
City2	2	31,341	3,470	3,864
Cinema	20	11,000	1,222	1,358
Car Name	21	6,606	754	825
Car Model	22	2,357	262	291
Period of Time	23	4,051	455	508
Stadium Name	24	3,734	397	441
Clinic Name	25	1,433	160	178
Date and Day 1	3	63,176	6,965	7,800
Time1	4	60,296	6,722	7,471
Date and Day 2	5	3,302	375	407
Time2	6	3,247	369	393
Ticket Class	7	57,200	6,376	7,106
Round Trip	8	23,734	2,636	2,930
Number of Tickets	9	152,259	16,860	18,651

To develop our model, we also used the Python programming language and the PyTorch library, but a later version of AraBERT transformer was used which is “AraBERTv0.2 base”. And we fine-tuned it in the same way as the previous model by adding two new layers, a 0.3 dropout layer and a linear output layer with 26 outputs at the top of this model.

Compilation of the proposed model was performed using the same configuration of the based model. Adam optimizer was used with a learning rate scheduler and the cross-entropy loss function, and then we trained the entire model together using ABC Dataset (Antoun *et al.,* 2021). The accuracy score metric is used to evaluate this model.

Figure 14 shows the architecture of our model. The inputs to the AraBERT transformer are presented as 3 torch tensors (Input Ids, Input Mask, Segment Ids) that translates them to a torch tensor with the size of (Number of Samples, Max Sequence Length = 60, and 768). The 768 is the hidden size dimensions of the encoder layer inside the AraBERT model.

The output from AraBERT is then fed as input to the next layer which is a dropout layer with a probability of 0.3 to avoid overfitting. The output layer for this model is a linear layer with 768 inputs and with 26 outputs. The final output from this model is a torch tensor with the size of (Number of Samples, Max Sequence Length = 60, and 26), where 26 represents the number of different entity classes present in the ABC dataset. Then we defined a new function which is the “align predictions”. We pass the output of the recognition engine model to this function in order to reduce its dimension by selecting the classes that have the maximum probabilities to be (Number of Samples, Max Sequence Length = 60), and removing the padding from the sequence to be (Number of Samples, the sequence length according to each sample).

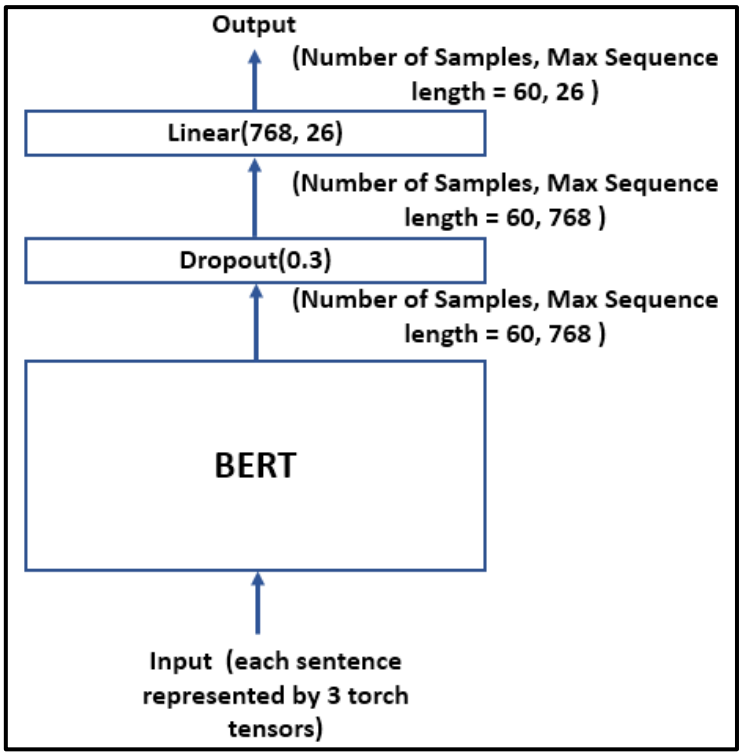

Figure 14. The Architecture of the Recognition Engine Model.

For training, we don't use the k-folds cross-validation iterator, because there are many samples in our dataset. We used a maximum of 100 epochs for the training process. In each epoch, the model trains using the training set with a batch size of 32 and then predicts the tags of the validation set with a batch size of 8. We used the validation loss value to stop training early if the validation loss does not improve for five consecutive epochs.

Figure 15 shows the training loss and the validation loss during training, as well as the value of the lowest validation loss which is the early stopping point. The model with the least validation loss is adopted and used for testing.

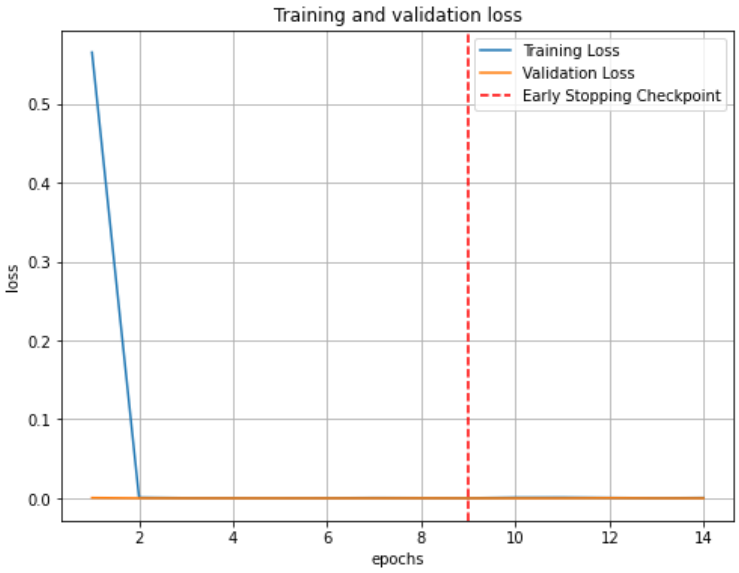

Figure 15. Training Loss and Validation Loss *vs*. the Number of Epochs.

This model was trained using GPU for 14 epochs with approximately 10 minutes per epoch and then stopped due to the use of early stopping with the patience value set to 5. The ninth epoch has the lowest validation loss of $1.31e^{-06}$.

5.4 Results and Discussion

This section includes the presentation of the results achieved by this work on the test set of the ABC dataset and the real dataset, where the results are presented using the classification report and the confusion matrix from the Scikit-Learn library, then error analysis of some samples of the real dataset are presented. At the end of this section, we summarize the results of this work and the previous work.

5.4.1 The Results of The Recognition Engine Model

Table 13 shows the evaluation of the recognition engine model on the ABC dataset and the real dataset in terms of accuracy and number of errors in the entities. Also, this table shows the number of samples from the ABC dataset and the real dataset that was tested on.

Table 13. The Accuracy Score and The Number of Error for The Recognition Engine Model on The ABC Dataset and The Real Dataset.

Model Name	ABC Dataset (7,612 Test Samples) (139,972 Entities)	Real Dataset (200 Samples) (2,936 Entities)	
	Accuracy	Accuracy	Errors
Recognition Engine Model	100%	95.5%	134

In our experiment, we use the accuracy metric for evaluation. The recognition engine model achieves 100% classification accuracy on the test dataset of the ABC dataset. High accuracy is expected since the model is a transformer for the Arabic language. But this accuracy score may indicate that the test dataset is very similar to the training dataset, as we described in Chapter 4 that the dataset is split into Train, Val, and Test sets using the stratified train-test split based on domain and template type.

Therefore, we evaluated this model on the real dataset that we collected from more than 30 volunteers for high diversity, and this dataset contains 200 booking tickets spread across the seven domains.

Table 14 shows the classification report for evaluating the recognition engine model on the ABC test set. This report contains the precision, recall, and F1 score metrics for the 26 classes. The last column shows the frequency of each class in the test set. At the end of this report, the accuracy score of this model is presented.

Table 14. The Classification Report of the Recognition Engine Model for The ABC Test Set.

Label	Class	Precision	Recall	F1 Score	Support
0	Others	1.000	1.000	1.000	66,881
9	Number of Tickets	1.000	1.000	1.000	18,651
3	Date and Day 1	1.000	1.000	1.000	7,800
4	Time 1	1.000	1.000	1.000	7,471
7	Ticket Class	1.000	1.000	1.000	7,106
1	City 1	1.000	1.000	1.000	6,277
17	Room	1.000	1.000	1.000	4,505
2	City 2	1.000	1.000	1.000	3,864
16	Hotel	1.000	1.000	1.000	3,589
8	Round Trip	1.000	1.000	1.000	2,930
10	Name	1.000	1.000	1.000	1,719
19	Movie	1.000	1.000	1.000	1,626
20	Cinema	1.000	1.000	1.000	1,358
21	Car Name	1.000	1.000	1.000	825
13	First Team	1.000	1.000	1.000	699
11	Doctor Name	1.000	1.000	1.000	691
15	League	1.000	1.000	1.000	617
14	Second Team	1.000	1.000	1.000	576
23	Period of Time	1.000	1.000	1.000	508
24	Stadium Name	1.000	1.000	1.000	441
18	Restaurant	1.000	1.000	1.000	432
5	Date and Day 2	1.000	1.000	1.000	407
6	Time 2	1.000	1.000	1.000	393
22	Car Model	1.000	1.000	1.000	291
25	Clinic Name	1.000	1.000	1.000	178
12	Phone	1.000	1.000	1.000	137
Accuracy		1.000			
Macro Avg		1.000	1.000	1.000	**Total: 139,972**

Table 15 shows the classification report for evaluating the recognition engine model on the real dataset, and this model achieves 95.5% classification accuracy on this dataset.

Table 15. The Classification Report of the Recognition Engine Model for the Real Dataset.

Label	Class	Precision	Recall	F1score	Support
0	Others	0.985	0.964	0.9744	1,798
3	Date and Day1	0.984	0.966	0.975	262
9	Number of Tickets	0.927	0.953	0.940	214
4	Time1	0.894	0.988	0.939	163
7	Ticket Class	0.916	0.950	0.933	80
23	Period of Time	0.927	0.900	0.913	70
1	City1	0.756	0.919	0.829	37
19	Movie	0.941	0.889	0.914	36
21	Car Name	1.000	0.853	0.921	34
16	Hotel	0.952	0.769	0.851	26
17	Room	0.677	0.920	0.780	25
25	Clinic Name	0.846	0.880	0.863	25
11	Doctor Name	0.950	1.000	0.974	19
5	Date and Day2	0.864	1.000	0.927	19
13	First Team	1.000	1.000	1.000	17
14	Second Team	0.944	1.000	0.971	17
2	City2	0.923	0.800	0.857	15
18	Restaurant	0.684	0.929	0.788	14
10	Name	0.619	1.000	0.765	13
20	Cinema	0.818	0.692	0.750	13
6	Time2	1.000	1.000	1.000	13
24	Stadium Name	1.000	1.000	1.000	11
8	Round Trip	1.000	0.750	0.857	8
15	League	1.000	0.667	0.800	6
22	Car Model	0.333	1.000	0.500	1
Accuracy		**0.955**			
Macro Avg		**0.878**	**0.912**	**0.881**	**Total: 2,936**

In the classification report of the real dataset, there are only 25 classes, as this dataset does not contain the “Phone” class of label “12”. There are 3 classes that have an F1 score value of 1.000, and many classes have a high value of F1 score regardless of how many entities are in the dataset for each class. For example, the “Stadium Name” class has 11 entities with an F1 score of 1.000 and the “City 1” Class has 37 entities with an F1 score of 0.829.

5.4.2 Confusion Matrix

Figure 16 shows the confusion matrix for the real dataset of the recognition engine model. Each line of this matrix shows the number of correct and incorrect predictions for each class.

Since the value of the F1 score depends on the precision and recall values, according to Table 15, the classes that have low precision values. For example, the “Restaurant” class and the “Room” class, or low recall values. For example, the “League” class and the “Cinema” class, have lower values of F1 score than other classes.

	Label		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	Others	1734	5	6	1	0	0	0	1	0	2	2	1	2	0	2	5	0	4	2	13	0	0	7	0	11
2	1	City1	3	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	10	Name	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	11	Doctor Name	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	13	First Team	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	14	Second Team	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	15	League	2	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	16	Hotel	0	1	0	0	0	0	0	20	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	17	Room	1	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
10	18	Restaurant	1	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	19	Movie	3	0	1	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	2	City2	0	2	0	0	0	1	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0
13	20	Cinema	0	2	0	0	0	0	0	0	0	2	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0
14	21	Car Name	3	0	0	0	0	0	0	0	2	0	0	0	0	29	0	0	0	0	0	0	0	0	0	0	0
15	22	Car Model	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
16	23	Period of Time	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0	63	0	0	2	0	0	0	0	0	2
17	24	Stadium Name	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0
18	25	Clinic Name	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	22	0	0	0	0	0	0	1
19	3	Date And Day1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	253	6	3	0	0	0	0
20	4	Time1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	161	0	0	0	0	0
21	5	Date and Day2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0
22	6	Time2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	0	0	0
23	7	Ticket Class	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	76	0	0
24	8	RoundTrip	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	1
25	9	Number Of Tickets	6	1	1	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	204

Predictions

Actuals

Figure 16. Confusion Matrix for the Recognition Engine Model Predictions on The Real Dataset Containing 25 Classes.

According to Figure 16, the “Name” class has a very low precision value of 0.619 because the model classifies the other classes as the “Name” class (see Column 3 in the confusion matrix), while this class has a recall value of 1 (see Row 3 in the confusion matrix). On the contrary, for the “League” class, the recall value is 0.667 and the precision value is 1.000, see Row 7 and Column 7 in the confusion matrix, respectively.

5.4.3 Error Analysis in The Real Dataset

Table 16 shows 4 samples from the real dataset, for each sample, the prediction labels of the recognition engine model are compared with its true labels, where the error

is colored red. The sentence is read from right to left (Arabic), while the labels are read from left to right. For example, "بدي أحجز " from the third sample take the first two tags from the left "0 0", while the last word in the sentence is "الظهر " takes the last tag "4".

Table 16. Four Samples from The Real Dataset (Sentence, True Label, Prediction Label of The Recognition Engine Models).

مسا الخير، بأدر احجز الجي كلاس لتلات ايام بس ضروري كتير تكون لبلاك لأنها طلب المدام، ممنونكم.	Sentence
0 0 0 0 21 21 23 23 0 0 0 0 0 0 0 0 0 0	True Label
4 0 0 0 17 17 23 23 0 0 0 0 10 0 0 10 0	Recognition Engine Model
اريد حجز تذكرة لحضور فيلم المستقبل في سينما رغدان بتاريخ 27 مارس الساعة السادسة مساءا	Sentence
0 0 9 0 0 19 0 0 20 0 3 3 0 4 4	True Label
0 0 9 0 0 19 0 0 18 0 3 3 0 4 4	Recognition Engine Model
بدي أحجز موعد في عيادة سما للأطفال مع الدكتورة مها بتاريخ 20- 9 الساعة 1 الظهر.	Sentence
0 0 0 0 0 25 25 0 0 11 0 3 0 4 4	True Label
0 0 0 0 0 18 25 0 0 11 0 3 0 4 4	Recognition Engine Model
بدي احجز بمطعم جبران ع ترس بوفيه فطور لثلاث اشخاص	Sentence
0 0 0 18 0 0 0 0 9 9	True Label
0 0 0 18 0 1 0 0 9 9	Recognition Engine Model

5.4.5 Experiments Comparison

It is difficult to compare this work with the previous work for several reasons, the most important of which is that the dataset used to evaluate each model differs in its size, number of classes, and content. Also, the adopted metric for the previous work is the F1 score while the metric that we adopted in our work is the accuracy. Table 17 summarizes the results of this work and the previous work.

Table 17. A Summary of The Results of our Work and The Previous Work with The Identification of The Dataset Used for Evaluation.

Model	Dataset	Size	Metric/ Score
AraBERTv0.1 (The previous work)	ANERcorp	16.5 k Entities (without the "other" entity)	F1 score/ 84.2%
Recognition Engine Model	ABC Dataset	76,117 Samples	Accuracy/ 100%
Recognition Engine Model	The Real Dataset	200 Samples	Accuracy/ 95.5%

Chapter 6: Conclusion and Future Works

This chapter includes the conclusion of our work for developing a recognition engine model. We also suggest future works to improve the efficiency and accuracy of this model.

6.1 Conclusion

In this thesis, we have developed a recognition engine model that can recognize booking information for various domains with high accuracy from unstructured text written by a user trying to book a ticket through a chatbot. The previous work that we based on is the state-of-the-art model by Antoun *et al*. (2020). They have fine-tuned the AraBERT model to solve the NER task on the ANERcorp to achieve an 84.2 F1score.

In our work, we have built an Arabic booking chatbot dataset (ABC Dataset) that contains 76,117 reservation sequences for seven different domains, where the number of entity classes in this dataset is 26 classes. Also, we have built a real dataset by collecting reservation samples from volunteers. For building our recognition engine models we have used the AraBERTv0.2-Base model.

We conducted an experiment to build a recognition engine model that solves the NER task of tickets and appointments for the seven different domains. This experiment achieves 100% and 95.50% accuracies on the ABC dataset and the real dataset, respectively. So, developing one recognition engine model that can be used and adapted to perform different types of booking services instead of several models, each one is dedicated to each possible booking service, is more beneficial in terms of development efforts and cost.

6.2 Future Work

For future work, we suggest the following directions: We propose to improve the predictions from the proposed recognition engine model on the real dataset, by taking advantage of the domain type. We also would like to check which approach gives better result: using the recognition engine model trained on multiple domains or using multiple one-domain models. Another area is to improve the recognition engine model capability by enabling the model to learn online to increase its ability to recognize new entities of the reservation text. Also, we are looking forward to integrating our model with an available relevant library to build a chatbot reservation system and then enable users to book tickets in Arabic text, and set some conditions and restrictions for booking tickets, to avoid misunderstanding when using the booking chatbot.

REFERENCES

Al-Ajmi, A. H., & Al-Twairesh, N. **(2021)**. Building an Arabic Flight Booking Dialogue System Using a Hybrid Rule-Based and Data Driven Approach. **IEEE Access, 9, 7043-7053.**

Al-Ayyoub, M., Nuseir, A., Alsmearat, K., Jararweh, Y., & Gupta, B. **(2018)**. Deep learning for Arabic NLP: A survey. **Journal of computational science, 26, 522-531.**

Al-Ghadhban, D., & Al-Twairesh, N. **(2020)**. Nabiha: An Arabic dialect chatbot. **Int. J. Adv. Comput. Sci. Appl, 11(3), 1-8.**

Al Humoud, S., Al Wazrah, A., & Aldamegh, W. **(2018)**. Arabic chatbots: a survey. **Int. J. Adv. Comp. Sci. Appl., 535-541.**

Ali, N. **(2020)**. Chatbot: A Conversational Agent employed with Named Entity Recognition Model using Artificial Neural Network. **arXiv preprint arXiv:2007.04248.**

Almansor, E. H., & Hussain, F. K. **(2019)**. Survey on intelligent chatbots: State-of-the-art and future research directions. **In Conference on Complex, Intelligent, and Software Intensive Systems (pp. 534-543). Springer, Cham.**

Alshareef, T., & Siddiqui, M. A. **(2020)**. A seq2seq Neural Network based Conversational Agent for Gulf Arabic Dialect. **In 2020 21st International Arab Conference on Information Technology (ACIT) (pp. 1-7). IEEE.**

Antoun, W., Baly, F., & Hajj, H. **(2020).** Arabert: Transformer-based model for arabic language understanding. **arXiv preprint arXiv:2003.00104.**

Antoun, W., Salti, M., Baly, F., & Alaziz, A. **(2021)**. aub-mind/arabert: Pre-trained Transformers for the Arabic Language Understanding and Generation (Arabic BERT, Arabic GPT2, Arabic Electra). **https://github.com/aub-mind/arabert.**

Brownlee, J. **(2016)**. What is a Confusion Matrix in Machine Learning. **https://machinelearningmastery.com/confusion-matrix-machine-learning/**

Brownlee, J. **(2020)**. PyTorch Tutorial: How to Develop Deep Learning Models with Python. **https://machinelearningmastery.com/pytorch-tutorial-develop-deep-learning-models/**

Brownlee, J. **(2020)**. Train-Test Split for Evaluating Machine Learning Algorithms. **https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/**

Class Central **(2021)**. Microsoft Bot Framework and Conversation as a Platform. **https://www.classcentral.com/course/edx-microsoft-bot-framework-and-conversation-as-a-platform-11325**

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. **(2018).** Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805.**

Fadhil, A. **(2019, September).** OlloBot-Towards A Text-Based Arabic Health Conversational Agent: Evaluation and Results. **In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019) (pp. 295-303).**

Følstad, A., Nordheim, C. B., & Bjørkli, C. A. **(2018, October).** What makes users trust a chatbot for customer service? An exploratory interview study**. In International conference on internet science (pp. 194-208). Springer, Cham.**

Horev, R. **(2018)**. BERT Explained: State of the art language model for NLP. **https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.**

Hugging Face. **(2020)**. Summary of the tokenizers. **https://huggingface.co/transformers/tokenizer_summary.html**

Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., ... & Zweig, G**. (2014).** Using recurrent neural networks for slot filling in spoken language understanding. **IEEE/ACM Transactions on Audio, Speech, and Language Processing, 23(3), 530-539.**

Mnasri, M. **(2019)**. Recent advances in conversational NLP: Towards the standardization of Chatbot building. **arXiv preprint arXiv:1903.09025**.

Open Source Libraries. **(2020)**. Seqeval. **https://opensourcelibs.com/lib/seqeval.**

Python Package Index **(2021)**. Transformers. https://pypi.org/project/transformers/

REVE Chat **(2021)**. 10 Excellent Benefits of Using Chatbots in Your Business. **https://www.revechat.com/blog/chatbot-business-benefits/**

Scikit Learn. **(2021)**. sklearn.metrics. **https://scikit-learn.org/stable/index.html.**

Shaikh, A., Phalke, G., Patil, P., Bhosale, S., & Raghatwan, J. **(2016).** A Survey on Chatbot Conversational Systems. **International Journal of Engineering Science, 3117**.

Shulga, D. **(2019)**. BERT to the rescue. **https://towardsdatascience.com/bert-to-the-rescue-17671379687f.**

Taher, E., Hoseini, S. A., & Shamsfard, M. **(2020)**. Beheshti-NER: Persian named entity recognition Using BERT**. arXiv preprint arXiv:2003.08875.**

Trotter, M. **(2021)**. Transfer Learning for Machine Learning. **https://www.seldon.io/transfer-learning/**

Youssef, A., Elattar, M., & El-Beltagy, S. R. **(2020, October)**. A Multi-Embeddings Approach Coupled with Deep Learning for Arabic Named Entity Recognition. **In 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES) (pp. 456-460). IEEE.**

Yufeng, G. **(2017)**. Introduction to Kaggle Kernels**.** **https://towardsdatascience.com/introduction-to-kaggle-kernels-2ad754ebf77**

استخدام التعلم الآلي لبناء محرك التعرف لروبوت المحادثة باللغة العربية

إعداد
بشرى طه رضا الصدر

المشرف
الأستاذ الدكتور غيث علي عبنده

ملخـــــص

أصبحت روبوتات الدردشة مؤخرًا ضرورية في مجالات مختلفة تتراوح من خدمة العملاء والحصول على المعلومات إلى الترفيه، حيث إن استخدام روبوتات المحادثة في خدمة العملاء يقلل من التكاليف التشغيلية ويقلل من الأخطاء البشرية ويوفر الوقت من خلال توفير استجابة فورية في أي وقت.

في هذه الرسالة، نقترح حلاً جديدًا لبناء نموذج لروبوت المحادثة باللغة العربية المخصص لحجز التذاكر والمواعيد في سبعة مجالات مختلفة، وهي الرحلات الجوية، والفنادق، ودور السينما، ومباريات كرة القدم، والسيارات، والمطاعم، والعيادات. نقوم أيضًا ببناء مجموعة بيانات روبوت المحادثة باللغة العربية (ABC dataset) التي تحتوي على 76.117 عينة و26 فئة. وقمنا بجمع عينات حجز من متطوعين من أجل تقييم نموذجنا على عينات حقيقية مختلفة. العمل الذي استندنا إليه هو نموذج AraBERT الحديث، وهو نموذج لغوي كبير تم إعداده وتدريبه مسبقًا للغة العربية. قاموا بضبط نموذج AraBERT لحل مهمة التعرف على الكيانات المسماة (NER) في مجموعة التعرف على الكيانات المسماة باللغة العربية (ANERcorp).

نقترح نموذج محرك التعرف لاستخراج كيانات الحجوزات في مجموعة البيانات الخاصة بنا. نعتمد نهج التعلم الآلي العميق ونقل التعلم لتطوير نموذج دقيق وفعال. استخدمنا وضبطنا النموذج الأساسي AraBERTv0.2 لحل هذه المهمة. مدخلات نموذجنا المقترح هي نص استعلامات حجز

المستخدم والمخرجات عبارة عن سلسلة من العلامات التي تمثل كيانات تسلسل الإدخال. حقق النموذج المقترح درجات دقة بنسبة 100٪ و95.5٪ في مجموعة اختبار ABC ومجموعة البيانات الحقيقية، على التوالي. وفرنا مجموعة بيانات ABC ومجموعة البيانات الحقيقية والبرمجيات الخاصة بنموذجنا على https://github.com/Boshra-sadder/Arabic-Booking-Chatbot من أجل دعم أبحاث روبوتات الدردشة للغة العربية.