Generating and Analyzing Chatbot Responses using Natural Language Processing

Marija Bezbradica

International Journal of Advanced Computer Science and Applications

Generating and Analyzing Chatbot Responses using Natural Language Processing

Moneerh Aleedy[1]
Information Technology Department
College of Computer and Information Sciences
Princess Nourah bint Abdulrahman University
Riyadh, Saudi Arabia

Hadil Shaiba[2]
Computer Sciences Department
College of Computer and Information Sciences
Princess Nourah bint Abdulrahman University
Riyadh, Saudi Arabia

Marija Bezbradica[3]
School of Computing, Dublin City University, Dublin, Ireland

***Abstract*—Customer support has become one of the most important communication tools used by companies to provide before and after-sale services to customers. This includes communicating through websites, phones, and social media platforms such as Twitter. The connection becomes much faster and easier with the support of today's technologies. In the field of customer service, companies use virtual agents (Chatbot) to provide customer assistance through desktop interfaces. In this research, the main focus will be on the automatic generation of conversation "Chat" between a computer and a human by developing an interactive artificial intelligent agent through the use of natural language processing and deep learning techniques such as Long Short-Term Memory, Gated Recurrent Units and Convolution Neural Network to predict a suitable and automatic response to customers' queries. Based on the nature of this project, we need to apply sequence-to-sequence learning, which means mapping a sequence of words representing the query to another sequence of words representing the response. Moreover, computational techniques for learning, understanding, and producing human language content are needed. In order to achieve this goal, this paper discusses efforts towards data preparation. Then, explain the model design, generate responses, and apply evaluation metrics such as Bilingual Evaluation Understudy and cosine similarity. The experimental results on the three models are very promising, especially with Long Short-Term Memory and Gated Recurrent Units. They are useful in responses to emotional queries and can provide general, meaningful responses suitable for customer query. LSTM has been chosen to be the final model because it gets the best results in all evaluation metrics.**

***Keywords*—Chatbot; deep learning; natural language processing; similarity**

I. Introduction

With the arrival of the information age, customer support has become one of the most influential tools companies use to communicate with customers. Modern companies opened up communication lines (conversations) with clients to support them regarding products before and after-sales through websites, telephones, and social media platforms such as Twitter. This communication becomes faster and much easier with the support of the technologies that are being used today.

Artificial intelligence (AI) improves digital marketing in a number of different areas from banking, retail, and travel to healthcare and education. While the idea of using human language to communicate with computers holds merit, AI scientists underestimate the complexity of human language, in both comprehension and generation. The challenge for computers is not just understanding the meanings of words, but understanding expression in how those words are collocated. Moreover, a chatbot is an example of a virtual conversational service robot that can provide human-computer interaction. Companies use robotic virtual agents (Chatbot) to assist customers through desktop interfaces [1, 2].

Natural language processing (NLP) is a subfield of computer science that employs computational techniques for learning, understanding and producing human language content. NLP can have multiple goals; it can aid human-human communication, such as in machine translation and aid human-machine communication, such as with conversational agents. Text mining and natural language processing are widely used in customer care applications to predict a suitable response to customers, which significantly reduces reliance on call center operations [3].

AI and NLP have emerged as a new front in IT customer service chatbots. The importance of these applications appears when no technicians manage the customer service office due to the end of working time or their presence outside the office [4].

In this project, the main focus will be on the automatic generation of conversation "Chat" between a computer and a human by developing an interactive artificial intelligent agent using deep learning. This will provide customers with the right information and response from a trusted source at the right time as fast as possible.

This project aims to build an automated response system (Chatbot) that responds to customer queries on social networking platforms (Twitter) to accelerate the performance of the service. Also, to keep the simplicity in mind while designing the system to enhance its efficiency.

This project centers around the study of deep learning models, natural language generation, and the evaluation of the generated results.

We believe that this contribution can add improvement by applying the right preprocessing steps which may organize sentences in a better way and help in generating proper responses. On the other hand, we start with the existing text generative models CNN and LSTM and then try to improve them as well as develop a new model such as GRU to compare results. We focus on evaluating the generated responses from two aspects: the number of words matches between the reference response and the generated response and their semantic similarity.

The rest of this paper is organized as follows. Section II provides reviews of the related works. The methodological approach is described in Section III. Moreover, dataset collection and analysis in details are provided in Section IV. The implementation strategy and results of this project are discussed in section V. Finally, the conclusion of the project and its future work are provided in Sections VI and VII respectively.

II. Literature Review

Developing computational conversational models (chatbots) took the attention of AI scientists, for a number of years. Modern intelligent conversational and dialogue systems draw principles from many disciplines, including philosophy, linguistics, computer science, and sociology [5]. This section will explore the previous work of chatbots and their implementations.

A. Chatbots Applications and Uses

Artificial dialogue systems are interactive talking machines called chatbots. Chatbot applications have been around for a long time; the first well-known chatbot is Joseph Weizenbaum's Eliza program developed in the early 1960s. Eliza facilitated the interaction between human and machine through a simple pattern matching and a template-based response mechanism to emulate the conversation [6, 7].

Chatbot became important in many life areas; one of the primary uses of chatbots is in education as a question answering system for a specific knowledge domain. In [8], the authors proposed a system that has been implemented as a personal agent to assist students in learning Java programming language. The developed prototype has been evaluated to analyze how users perceive the interaction with the system. Also, the student can get help in registering and dropping courses by using a chatbot spatialized in student administrative problems, as mentioned in [9]. The administrative student's chatbot helps the colleges to have 24*7 automated query resolution and helps students have the right information from a trusted source.

On another hand, information technology (IT) service management is an important application area for enterprise chatbots. In many originations and companies, IT services desk is one of the essential departments that helps to ensure the continuity of work and solving technical problems that employees and clients are facing. This variability demands manual intervention and supervision, which affects the speed and quality of processes execution. IT service providers are under competitive pressure to continually improve their service quality and reduce operating costs through automation. Hence, they need the adoption of chatbots in order to speed up the work and ensure its quality [10].

On the medical side, the field of healthcare has developed a lot, lately. This development appears with the use of information technology and AI in the field. In [11], the authors proposed a mobile healthcare application as a chatbot to give a fast treatment in response to accidents that may occur in everyday life, and also in response to the sudden health changes that can affect patients and threaten their lives.

Customer services agent is an application of applying chatbot technologies in businesses to solve customer problems and help the sales process. As companies become globalized in the new era of digital marketing and artificial intelligence, brands are moving to the online world to enhance the customer experience in purchasing and provide new technical support ways to solve after-sales problems. Moreover, fashion brands such as Burberry, Louis Vuitton, Tommy Hilfiger, Levi's, H&M, and eBay are increasing the popularity of e-service agents [1].

B. Natural Language Processing

NLP allows users to communicate with computers in a natural way. The process of understanding natural language can be decomposed into the syntactic and semantic analysis. Syntactic refers to the arrangement of words in a sentence such that they make grammatical sense. Moreover, syntactic analysis transforms sequences of words into structures that show how these words are related to each other. On the other hand, semantic refers to the meaning of each word and sentence. The semantic analysis of natural language content captures the real meaning; it processes the logical structure of sentences to find the similarities between words and understand the topic discussed in the sentences [12].

As part of the text mining process, the text needs many modification and cleaning before using it in the prediction models. As mentioned in [13], the text needs many preprocessing steps which include: removing URLs, punctuation marks and stop words such as a, most, and, is and so on in the text because those words do not contain any useful information. In addition, tokenizing, which is the process of breaking the text into single words. Moreover, text needs stemming, which means changing a word into its root, such as "happiness" to "happy". For features extraction, the authors use Bag of Words (BoW) to convert the text into a set of features vector in numerical format. BoW is the process of transforming all texts into a dictionary that consist of all words in the text paired with their word counts. Vectors are then formed based on the frequency of each word appearing in the text.

Before entering the data into a model or a classifier, it is necessary to make sure that the data are suitable, convenient, and free of outliers. In [14], the authors explain how to preprocess the text data. The main idea was to simplify the text for the classifier to learn the features quickly. For example, the

names can be replaced with one feature {{Name}} in the feature set, instead of having the classifier to learn 100 names from the text as features. This will help in grouping similar features together to build a better predicting classifier. On another hand, emoticons and punctuation's marks are converted to indicators (tags). Moreover, a list of emoticons is compiled from online sources and grouped into categories. Other punctuation marks that were not relevant to the coding scheme are removed.

Chat language contains many abbreviations and contractions in the form of short forms and acronyms that have to be expanded. Short forms are shorter representations of a word which are done by omitting or replacing few characters, e.g., grp → group and can't → cannot. The authors created a dictionary of these words from the Urban Dictionary to replace abbreviations by expansions. Spell checking is performed as the next step of the pre-processing pipeline on all word tokens, excluding the tagged ones from the previous steps [14].

Minimizing the words during the text pre-processing phase as much as possible is very important to group similar features and obtain a better prediction. As mentioned in [15], the authors suggest processing the text through stemming and lower casing of words to reduce inflectional forms and derivational affixes from the text. The Porter Stemming algorithm is used to map variations of words (e.g., run, running and runner) into a common root term (e.g., run).

Words can not be used directly as inputs in machine learning models; each word needs to be converted into a vector feature. In [4], the authors adopt the Word2vec word embedding method to learn word representations of customer service conversations. Word2vec's idea is that each dimension of inclusion is a possible feature of the word, which can capture useful grammatical and semantic properties. Moreover, they tokenize the data by building a vocabulary of the most frequent 100K words in the conversations.

C. Machine Learning Algorithm and Evaluation

A large number of researchers use the idea of artificial intelligence and deep learning techniques to develop chatbots with different algorithms and methods. As mentioned in [16], the authors use a repository of predefined responses and a model that ranks these responses to pick an appropriate response for a user's input. Besides, they proposed topic aware convolutional neural tensor network (TACNTN) model to classify whether or not a response is proper for a message. The matching model used to select a response for a user message. Specifically, it has three-stages that include: pre-processing the message, retrieving response candidates from the pre-defined message-response pair index, then ranking the response candidates with a pre-train matching model.

In [17], the authors train two word-based machine learning models, a convolutional neural network (CNN) and a bag of words SVM classifier. Resulting scores are measured by the Explanatory Power Index (EPI). EPI used to determine how much words contribute to the classification decision and filter relevant information without an explicit semantic information extraction step.

The customer service agent is an important chatbot that is used to map conversations from request to the response using the sequence to sequence model. Moreover, a sequence to sequence models has two networks one work as an encoder that maps a variable-length input sequence to a fixed-length vector, and the other work as a decoder that maps the vector to a variable-length output sequence. In [4], the authors generate word-embedding features and train word2vec models. They trained LSTMs jointly with five layers and 640 memory cells using stochastic gradient descent for optimization and gradient clipping. In order to evaluate the model, the system was compared with actual human agents responses and the similarity measured by human judgments and an automatic evaluation metric BLEU.

As a conclusion of reviewing works concerned with the conversational system, text generation in English language and the collaboration of social media in customer support service, this paper proposes a work that aims to fill the gap of limited works in the conversational system for customer support field, especially in the Twitter environment. The hypothesis of this project was aiming to improve the automated responses generated by different deep learning algorithms such as LSTM, CNN, and GRU to compare results and then evaluate them using BLEU and cosine similarity techniques. As a result, this project will help to improve the text generation process in general, and customer support field in particular.

III. Methodological Approach

This section discusses the background of the implemented methods, explain why these methods are appropriate and give an overview of the project methodology.

A. Text Generative Model

Based on the nature of this project, which is generating a proper response to every customer query in social media, applying sequence-to-sequence learning are needed. Moreover, sequence-to-sequence means mapping a sequence of words representing the query to another sequence of words representing the response, the length of queries and responses can be different. This can be applied by the use of NLP and deep learning techniques.

Sequence-to-sequence models are used in many fields, including chat generation, text translation, speech recognition, and video captioning. As shown in Fig. 1, a sequence-to-sequence model consists of two networks, encoder, and decoder. The input text enters the encoder network in reverse order, then it is converted into a sequence of fixed length context vector, which is then used by the decoder to generate the output sequence [18].

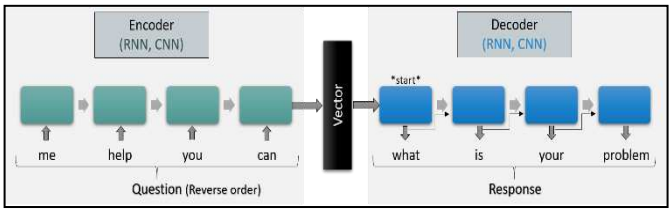

Fig. 1. Sequence to Sequence Model.

Before inserting the sequence of words into the encoder model, it needs to be converted into a numerical format; this can be applied by using NLP techniques. This project focused on Bag of Words, or BoW vector representations, which is the most commonly used traditional vector representation for text generating models. BoW is used to transforms all texts into a dictionary that consists of all words that appear in the document [13]. It then creates a set of features in real number inside a vector for each text.

B. Deep Learning Models

1) Convolutional Neural Network (CNN) Model: In this project, CNN is chosen mainly for its efficiency, since CNN is faster compared to other text representation and extraction methods [19]. The CNN consists of the convolution and pooling layers and provides a standard architecture that takes a variable-length sequence of words as an input and then passes it to a word embedding layer. The embedding layer maps each word into a fixed dimensional real-valued vector then passes it to the 1D convolutional layer. The output is then further down-sampled by a 1D max-pooling layer. Outputs from the pooling layers are then fed into the final output layer to produce a fixed-length feature vector [20]. CNN has been widely used in image and video recognition systems, and, lately, they have shown promising results in NLP applications [21]. Fig. 2 shows the standard architecture of the CNN model.

2) Recurrent Neural Network (RNN) Model: In a traditional neural network, all inputs and outputs are independent of each other, which is not useful when working with sequential information. Predicting the next word in a sentence requires knowing the sequence of the words in the sentence that come before the predicted word. Among all models for learning sentence representations, recurrent neural network (RNN) models, especially the Long Short Term Memory (LSTM) model, are the most appropriate models for processing sentences, as they have achieved substantial success in text categorization and machine translation [22]. Therefore, this project applies LSTM and Gated Recurrent Units (GRU) as a newer generation of Recurrent Neural Networks. Fig. 3 illustrates the basic architecture of RNN.

Hochreiter & Schmidhuber introduced Long Short Term Memory Networks in 1997. They solve the problem of vanishing and exploding gradient problem that is prevalent in a simple recurrent structure, as it allows some states to pass without activation. In 2014, Cho et al developed GRU networks in an effort to design recurrent encoder-decoder architecture [23]. They are relatively more straightforward than LSTM and retain a majority of its advantages.

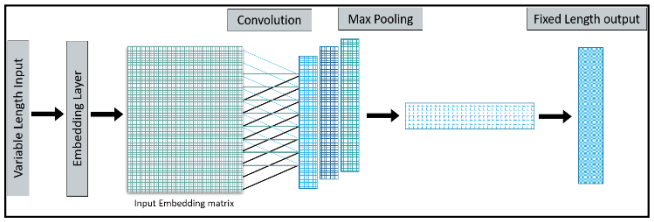

Fig. 2. The Architecture of CNN.

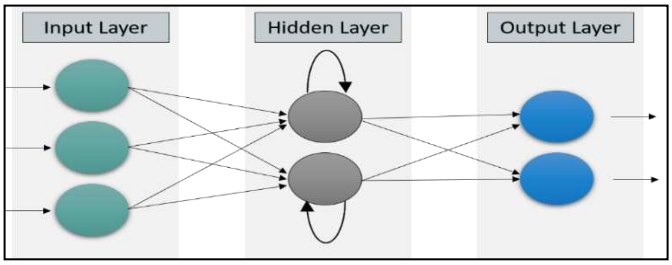

Fig. 3. The Architecture of RNN.

C. Project Methodology

In order to implement this project, several preprocessing and modeling steps are performed. First, split the original dataset into train and test sets. Then, prepare the dataset for modeling. The preparing process includes preprocessing steps and features extraction. After that, train models using train set with LSTM, GRU, and CNN. Finally, prepare the test set and use it for evaluating the models. Fig. 4 illustrates the methodology steps.

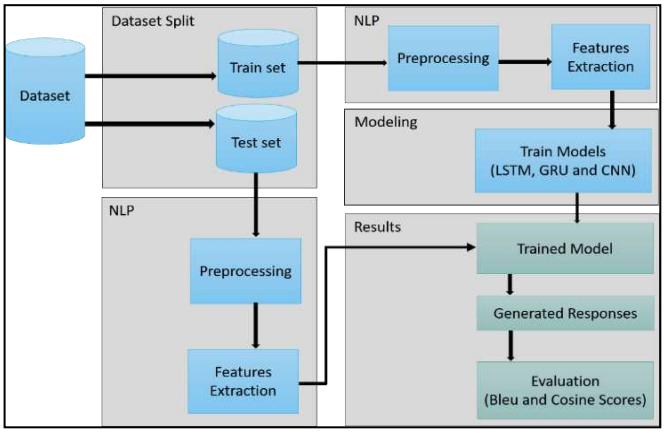

Fig. 4. The General Implementation Steps.

IV. Dataset Collection and Analysis

The dataset "Customer Support on Twitter" from Kaggle is used to develop and evaluate the models. The original dataset includes information such as: tweet_id, author_id, inbound, created_at, text, response_tweet_id and in_response_to_ tweet_id. The description of the original dataset is shown in Table I.

The original dataset contains 2,811,774 collections of tweets and replies from the biggest brands on Twitter as customer support (tweets and replies are in different rows). Moreover, the number of brands in the dataset is 108, and they responded to queries from 597075 users. Fig. 5 shows the top 10 customer support responses per brand.

While performing exploratory analysis on the dataset, it has been noticed, for instance, that Amazon customer support handles a lot of questions (around 84600 in seven months) which is a huge number to deal with if we consider the working hours and working days per week. Also, some of the questions have a delay in responding or had no responses at all. Fig. 6 shows the average delay in response to customers in hours per brand.

TABLE. I. DATASET FEATURES DESCRIPTION

Feature	Description	Datatypes
tweet_id	A unique, anonymized ID for the Tweet. Referenced by response_tweet_id and in_response_to_tweet_id.	int64
author_id	A unique, anonymized user ID. The real user_id in the dataset has been replaced with their associated anonymized user ID.	object
Inbound	Whether the tweet is "inbound" to a company doing customer support on Twitter. This feature is useful when reorganizing data for training conversational models.	bool
created_at	Date and time when the tweet was sent.	object
Text	Tweet content. Sensitive information like phone numbers and email addresses are replaced with mask values like __email__.	object
response_tweet_id	IDs of tweets that are responses to this tweet, comma-separated.	object
in_response_to_tweet_id	ID of the tweet this tweet is in response to, if any.	float64

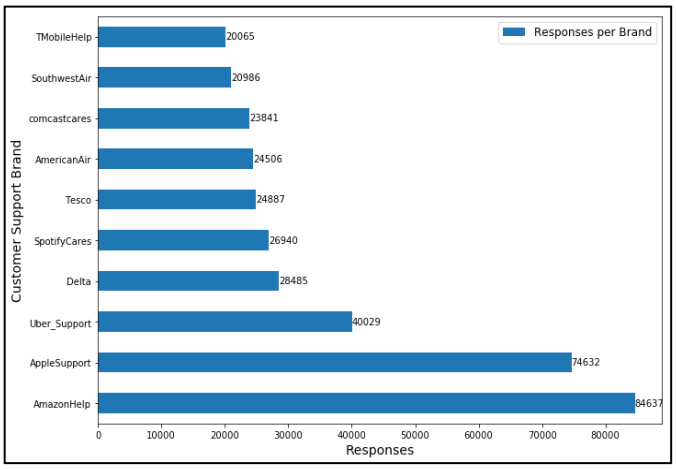

Fig. 5. Top 10 Customer Support Responses Per Brands.

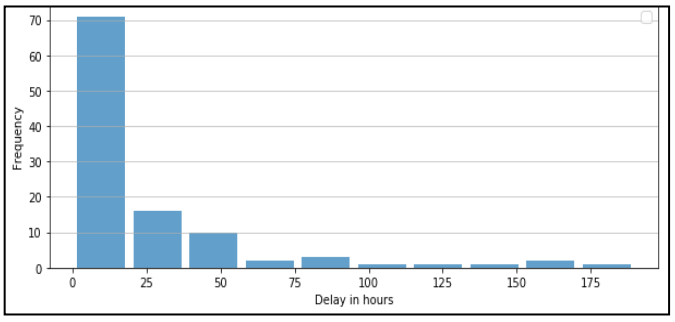

Fig. 6. The Average Delay in Response to Customers in Hours per Brand.

As shown in the above figure, around ten brands take more than two days (60 hours) to respond to customers queries, which may cause problems to customers, effect companies' reputation and the customers may start looking for other service providers.

A filtering process is used to convert the dataset records into a conversational dataset suitable for the experiments. The filtering is done as follows:

1) Pick only inbound tweets that are not in reply to any other tweet.

2) Organize each tweet with the corresponding reply by matching in_response_to_tweet_id with tweet_id features.

3) Filter out cases where reply tweets are not from a company based on the in inbound feature (if the inbound feature is False it means that the tweet is from a company; otherwise it is from a user).

However, when revising the dataset, it has been found that some of the tweets have no replies at all; they are from multiple languages, and some of them are just samples and emojis. For this type of tweets further preprocessing step is performed to remove non-English tweets by the use of the langdetect library which detects any non-English text [24]. Then, the non-responses English tweets are studied, as shown in the word cloud in Fig. 7, (which is a graph that illustrates the most words that appear in the text).

It can be observed that the words appear with no hint to a specific problem discussed, and most of the queries are thanking the customer support services for example:

- @AmazonHelp Thanks for the quick response
- @AppleSupport Awesome, thanks

Others asking for help in general:

- @Uber_Support Sent a DM Hope you could help soon.
- @O2 DM sent. Still no further forward!

The modified dataset contains 794,299 rows and 6 columns which are: author_id_x, created_at_x, text_x, author_id_y, created_at_y and text_y. X refers to the queries, and Y refers to the responses from customer support teams.

Fig. 7. Most Words used in the Queries without Responses Data.

V. Implementation Strategy

In this section, we are going to explain the methodology followed for this project. At first, prepare the dataset for modeling. The preparing process includes preprocessing step and features extraction then train the models using a training set and evaluate them with a test set.

A. Data Preprocessing

A data analyst cannot handle raw text directly to suit machine learning or deep learning methods. Therefore, it is necessary to work on texts' preprocessing from all existing impurities, for example, punctuation, expression code, and non-English words (Chinese, Spanish, French, and others). In order to do this, a number of python NLP libraries such as regular expression (RE), unicodedata, langdetect, and contractions are used.

In this project, the performed preprocessing steps include: remove links, images, Twitter ID, numbers, punctuation, emoji, non-English words and replace abbreviations with long forms. Table II illustrates the changes in the dataset before and after applying all the previous preprocessing steps.

The preprocessing steps are chosen carefully; not all preprocessing techniques are suitable for this kind of projects. For example, removing stopwords and text stemming cannot be applied because it will affect the sentences structures as well as the text generation process.

B. Feature Extraction

Before doing any complex modeling, the dataset needs to be transformed into a numerical format suitable for training. The Bag of Words (BOW) concept is applied to extract features from the text dataset. First, all of the texts in the dataset are split into an array of tokens (words). Then, a vocabulary dictionary is built with all of the words in the dataset and its corresponding index value. The array of words is then converted to an array of indexes. This process can be applied by the use of the sklearn' predefined method called CountVectorizer.

In order to handle variable length, the maximum sentence length needs to be decided. Moreover, all remaining vector positions should be filled with a value ('1' in this case) to make all sequences have the same length. On the other hand, words not in the vocabulary dictionary will be represented with UNK as a shortcut of unknown words. Moreover, each output text in the dataset will start with a start flag ('2' in this case) to help in training. Now the dataset is ready for training.

C. Modeling

The infrastructure used for experimentation involves google colaboratory and Crestle cloud services which are GPU-enabled Jupyter environments with powerful computing resources. All popular scientific computing and deep learning packages are pre-installed and configured to run on a GPU.

The experiments are applied using three different models LSTM, GRU, and CNN. The models use a training dataset of around 700k pairs of queries and responses and a testing dataset of 30k of unseen data. Training time is between 5 and 12 hours, depending on the model (see Table III).

TABLE. II. The Changes in Text before and after Applying Preprocessing Steps

Before preprocessing	**After preprocessing**
@115743 C91. Feel free to keep an eye on the PS Blog for news and updates: https://t.co/aLtfBAztyC	feel free to keep an eye on the ps blog for news and updates
@133100 We do our best to clear as many upgrades as we can, send us a DM with the reservation you're referring to and we'll take a look.	we do our best to clear as many upgrades as we can send us a dm with the reservation you are referring to and we will take a look
@129388 We'd like to look into this with you. To confirm, did you update to iOS 11.1? Please DM us here: https://t.co/GDrqU22YpT	we would like to look into this with you to confirm did you update to ios please dm us here

TABLE. III. Training Time in Hours

Model	**Training Time in Hours**
LSTM	12
GRU	8
CNN	5

In the experiments, multiple parameters are tested, and their effects are addressed. All models are tested with varying dimensionality of the word embeddings (100, 300 and 640), it was observed that models perform better and faster with 100-word embedding size.

The dataset is large, the number of vocabularies is 388,950 unique words, and our computers cannot handle it. So, only the frequent words appeared in the dataset should be used. The most frequent words are decided by the max_features parameter in the CountVectorizer function which sort words by its frequency then choose the most frequent words. The first vocabulary size in the experiments is 8000 and then it increases, taking into consideration memory limitation. A slight improvement has been recognized in all models and because of the memory limitation, only 10,000 of the vocabularies are used. Moreover, the GRU model was trained for eight epochs but without significant improvement. The three models are all trained under the same conditions. Table IV shows the common parameters used in all models.

TABLE. IV. The Common Parameters used in LSTM, GRU and CNN Models

Parameter	**Value**
Word embedding dimension size	100
Vocabulary size	10,000
Context dimension size	100
Learning rate	0.001
Optimization function	Adam
Batch size	1000 (the max that our computer can handle)
Max message length	30

The following are the common layers used in the models, starting from inserting the sequence of words into the model to generating the responses:

- Last Word Input Layer: Inputs the last word of the sequence.
- Encoder Input Layer: Inputs sequence data and pass it to the embedding layer.
- Embedding Layer: Used to create word vectors for incoming words.
- Encoder Layer (LSTM, GRU, CNN): Creates a temporary output vector from the input sequence.
- Repeated Vector Layer: Used like an adapter to fit the encoder and decoder parts of the network together. It can be configured to repeat the fixed-length vector one time for each time step in the output sequence.
- Concatenate Layer: Takes inputs and concatenates them along a specified dimension.
- Decoder Layer (LSTM, GRU, CNN)(Dense): Used as the output for the network.
- Next Word Dense Layer: Takes inputs from the previous layer and outputs a one vector representing the target word.
- Next Word softmax Layer: Applies a softmax function that turns the dense layer output into a probability distribution, from to pick the most likely next word.

D. Generating Responses

After training the models, the generating responses process is started using the 30k test set. The following are samples of the generated responses from all models (see Fig. 8 and 9).

E. Evaluation

The Bilingual Evaluation Understudy and cosine similarity evaluation metrics are used to compute the similarity between the generated response and the reference response.

1) Bilingual Evaluation Understudy (BLEU): BLEU was originally created to measure the quality of machine translation with respect to human translation. It calculates an n-gram precision (An n-gram is a sequence of n words that appear consecutively in the text) between the two sequences and also imposes a commensurate penalty for machine sequence being shorter than human one. A perfect match score is 1.0, whereas a perfect mismatch score is 0.0.

The computation of BLEU involves various components: n-gram precisions (Pn) and BLEU's brevity penalty. Those measures are calculated as shown in the following steps:

- Calculate n-gram precision (Pn): measures the frequency of the n-gram according to the number of times it appears in the generated response and reference response. Pn must be calculated for each value of n, which usually ranges from 1 to 4. Then the geometric average of Pn should be computed with a weighted sum of the logarithms of Pn.
- Calculate brevity penalty (equation 1): a penalization is applied to short answers, which might be incomplete.

$$BP=\begin{cases}1 & \text{if } c > r \\ e^{(1-(r/c))} & \text{if } c \le r\end{cases} \quad (1)$$

, where c is the length of generated response and r is the length of reference response.

- Then, calculate the BLEU score (equation 2) [23]:

$$BLEU= BP \cdot e^{\sum_{n=1}^{N}(Wn \cdot log\,(Pn))} \quad (2)$$

, where Wn = 1/N.

2) Cosine Similarity: On the other hand, cosine similarity also used to compute the similarity between the generated response and the reference response in vector representation. If there is more similarity between the two vectors, the cosine similarity value is near to one; otherwise, it is near to zero.

3) In order to implement the cosine similarity, the pre-trained model word2vec are used. The word2vec model is in gensim package, and it has been trained on part of Google News dataset (about 100 billion words) [25]. The model contains 300-dimensional vectors for 3 million words and phrases.

The word2vec model used to represent words in a vector space [26]. Words are represented in the form of vectors and placement is done in such a way that similar meaning words appear together, and different words are located far away.

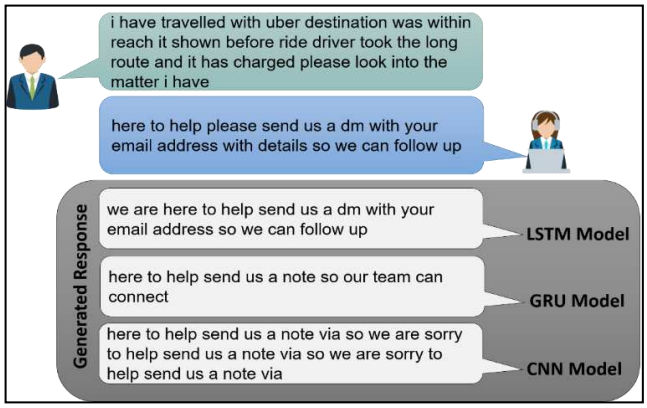

Fig. 8. Good Result Example.

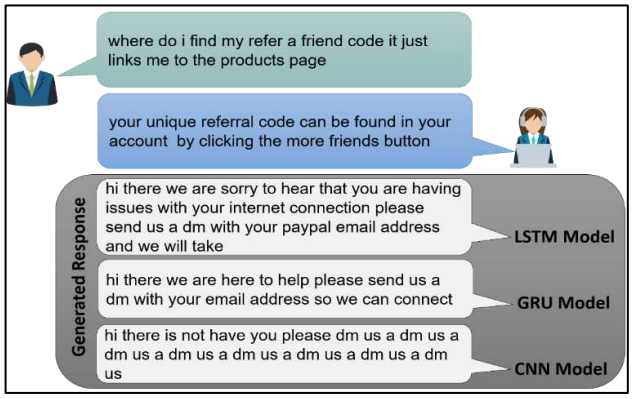

Fig. 9. Bad Result Example.

Gensim is a topic modeling toolkit which is implemented in python. Topic modeling is discovering the hidden structure in the text body. Word2vec model is imported from Gensim toolkit and uses a built-in function to calculate the similarity between the generated response and reference response.

F. Result and Discussion

Before discussing and reviewing the results, the most important features of the baseline model are discovered to have a rich discussion with clear comparisons. Table V shows the baseline model implementation.

In this project, the process of generating responses take around 6 hours for each model to be accomplished. Moreover, calculating BLEU and cosine similarity scores takes around 4 hours.

The models are evaluated automatically based on the words using BLEU score. The BLEU is applied for 1-gram, 2-gram, 3-gram, and 4-gram in order to explore the strength of the models. It can be seen that LSTM and GRU models outperform the official baseline LSTM model [4] with respect to the 4-gram BLEU score. Fig. 10, shows in details the performance of models in each n-gram.

Hence it can be seen that LSTM achieves the highest evaluation scores for all grams, but it takes a long time in training. Moreover, the GRU model has very close evaluation scores to LSTM. In the other hand, the CNN model has the lowest evaluation scores compared with all RNN models but achieves high-speed performance, which can be useful in application trained on large datasets.

TABLE. V. BASELINE MODEL IMPLEMENTATION

Preprocessing	Remove non-English queries, queries with images and @mentions.
Feature extraction	Word2vec
Model	LSTM with five layers.
Embedding size	640
Optimization Function	Stochastic gradient descent and gradient clipping.
Evaluation	BLEU with the best score achieved 0.36.

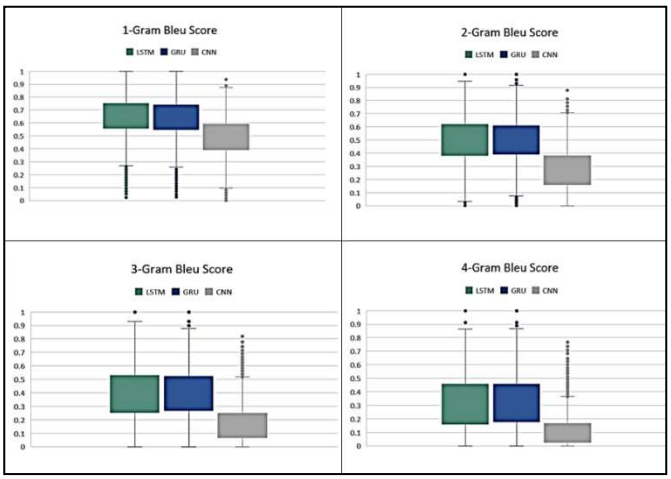

Fig. 10. The BLEU Scores for 1, 2, 3 and 4 Grams.

Furthermore, another evaluation metric cosine similarity are applied to captures the semantics beyond responses and gives similarity scores. It has been found that RNN models capture the semantics in the responses and they are more effective in improving the reply quality than the CNN model. Fig. 11 shows the similarity scores for each model.

After exploring the generated responses and get in-depth in the good and bad results, it has been found that RNN models, in general, are good in responses to emotional queries more than an informative one. The models can provide general, meaningful responses suitable for customer query. Table VI shows an example of an emotional query.

On the other hand, the queries that are more informative and ask about specific information are hard to generate, and the generated responses become less efficient. Table VII shows an example of an informative query.

By looking at the different responses from different models, it has been noticed that LSTM is generating better sentences that make sense and it is hard to say if the response is from a human or machine whereas GRU responses are not as good as LSTM.

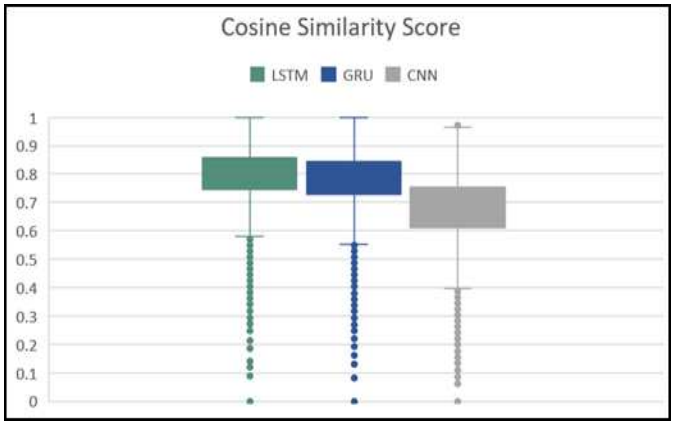

Fig. 11. The Cosine Similarity Scores.

TABLE. VI. EXAMPLE OF EMOTIONAL QUERY AND RESPONSES FROM ALL MODELS

Customer Query	my package is days late and i am leaving tomorrow on holidays could you please help it is extremely
Customer Support Response	sorry to hear this please dm us your tracking and phone number
LSTM Generated Response	i am sorry for the trouble with your order please report this to our support team here and we will check this
GRU Generated Response	i am sorry for the trouble with your order please reach out to us here and we will look into this for you please do not provide your order details
CNN Generated Response	hi there is not provide your order number and we can you please dm us a dm us a dm us a dm us a dm us a dm us

TABLE. VII. EXAMPLE OF INFORMATIVE QUERY AND RESPONSES FROM ALL MODELS

Customer Query	guys when are you going to open your services in middle east
Customer Support Response	hulu store is only available in the us at this time but we will share the interest in bringing our service to the middle east
LSTM Generated Response	hi there we are sorry to hear about this please dm us with your email address so we can connect
GRU Generated Response	hi there i am sorry to hear about this please dm me the details of the issue you are having with your services
CNN Generated Response	hi there is not have you are you

VI. CONCLUSION

In this project, we build customer support chatbot that helps companies to have 24 hours of automated responses. After analyzing the dataset and understanding the importance to have automated responses to customers and companies, we start exploring existing techniques used for generating responses in the customer service field. Then, we attempt to try three different models LSTM, GRU, and CNN. The experimental results show that LSTM and GRU models(with modified parameters) tend to generate more informative and valuable responses compared to CNN model and the baseline model LSTM. Besides, we used a BLEU score and cosine similarity as evaluation measures to support the final decision.

VII. FUTURE WORK

In future work, we plan to incorporate other similarity measures such as soft cosine similarity. Also, we plan to improve the experiments by increase the vocabulary size and try to increase the epoch parameters to reach 100 after providing proper infrastructure. We further can add more data for the training by taking benefits from the queries without responses and translate non-English queries.

ACKNOWLEDGMENT

This research was funded by the Deanship of Scientific Research at Princess Nourah bint Abdulrahman University through the Fast-track Research Funding Program.

REFERENCES

[1] M. Chung, E. Ko, H. Joung, and S. J. Kim, “Chatbot e-service and customer satisfaction regarding luxury brands,” J. Bus. Res., Nov. 2018.

[2] J. Hill, W. Ford, I. F.-C. in H. Behavior, and undefined 2015, “Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations,” Elsevier.

[3] J. Hirschberg and C. D. Manning, “Advances in natural language processing,” Science (80-.)., vol. 349, no. 6245, pp. 261–266, Jul. 2015.

[4] A. Xu, Z. Liu, Y. Guo, V. Sinha, and R. Akkiraju, “A New Chatbot for Customer Service on Social Media,” in Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI ’17, 2017, pp. 3506–3510.

[5] S. Oraby, P. Gundecha, J. Mahmud, M. Bhuiyan, and R. Akkiraju, “Modeling Twitter Customer ServiceConversations Using Fine-Grained Dialogue Acts,” in Proceedings of the 22nd International Conference on Intelligent User Interfaces - IUI ’17, 2017, pp. 343–355.

[6] H. Shah, K. Warwick, J. Vallverdú, and D. Wu, “Can machines talk? Comparison of Eliza with modern dialogue systems,” Comput. Human Behav., vol. 58, pp. 278–295, May 2016.

[7] R. DALE, “The return of the chatbots,” Nat. Lang. Eng., vol. 22, no. 05, pp. 811–817, Sep. 2016.

[8] M. Coronado, C. A. Iglesias, Á. Carrera, and A. Mardomingo, “A cognitive assistant for learning java featuring social dialogue,” Int. J. Hum. Comput. Stud., vol. 117, pp. 55–67, Sep. 2018.

[9] S. Jha, S. Bagaria, L. Karthikey, U. Satsangi, and S. Thota, “STUDENT INFORMATION AI CHATBOT,” in International Journal of Advanced Research in Computer Science, 2018, vol. 9, no. 3.

[10] P. R. Telang, A. K. Kalia, M. Vukovic, R. Pandita, and M. P. Singh, “A Conceptual Framework for Engineering Chatbots,” IEEE Internet Comput., vol. 22, no. 6, pp. 54–59, Nov. 2018.

[11] K. Chung and R. C. Park, “Chatbot-based heathcare service with a knowledge base for cloud computing,” Cluster Comput., pp. 1–13, Mar. 2018.

[12] J. Savage et al., “Semantic reasoning in service robots using expert systems,” Rob. Auton. Syst., vol. 114, pp. 77–92, Apr. 2019.

[13] S. T. Indra, L. Wikarsa, and R. Turang, “Using logistic regression method to classify tweets into the selected topics,” in 2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS), 2016, pp. 385–390.

[14] A. Shibani, E. Koh, V. Lai, and K. J. Shim, “Assessing the Language of Chat for Teamwork Dialogue,” 2017.

[15] A. Singh and C. S. Tucker, “A machine learning approach to product review disambiguation based on function, form and behavior classification,” Decis. Support Syst., vol. 97, pp. 81–91, May 2017.

[16] Y. Wu, Z. Li, W. Wu, and M. Zhou, “Response selection with topic clues for retrieval-based chatbots,” Neurocomputing, vol. 316, pp. 251–261, Nov. 2018.

[17] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek, “"What is relevant in a text document?": An interpretable machine learning approach,” PLoS One, vol. 12, no. 8, p. e0181142, Aug. 2017.

[18] S. Sen and A. Raghunathan, “Approximate Computing for Long Short Term Memory (LSTM) Neural Networks,” IEEE Trans. Comput. Des. Integr. Circuits Syst., vol. 37, no. 11, pp. 2266–2276, Nov. 2018.

[19] Z. Wang, Z. Wang, Y. Long, J. Wang, Z. Xu, and B. Wang, “Enhancing generative conversational service agents with dialog history and external knowledge I,” 2019.

[20] J. Zhang and C. Zong, “Deep Neural Networks in Machine Translation: An Overview,” IEEE Intell. Syst., vol. 30, no. 5, pp. 16–25, Sep. 2015.

[21] R. C. Gunasekara, D. Nahamoo, L. C. Polymenakos, D. E. Ciaurri, J. Ganhotra, and K. P. Fadnis, “Quantized Dialog – A general approach for conversational systems,” Comput. Speech Lang., vol. 54, pp. 17–30, Mar. 2019.

[22] G. Aalipour, P. Kumar, S. Aditham, T. Nguyen, and A. Sood, “Applications of Sequence to Sequence Models for Technical Support Automation,” in 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 4861–4869.

[23] J. Singh and Y. Sharma, “Encoder-Decoder Architectures for Generating Questions,” Procedia Comput. Sci., vol. 132, pp. 1041–1048, 2018.

[24] N. Shuyo, “Language Detection Library for Java.” 2010.

[25] R. Rehurek and P. Sojka, “Software Framework for Topic Modelling with Large Corpora,” in Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, 2010, pp. 45–50.

[26] Y. Zhu, E. Yan, and F. Wang, “Semantic relatedness and similarity of biomedical terms: examining the effects of recency, size, and section of biomedical publications on the performance of word2vec.,” BMC Med. Inform. Decis. Mak., vol. 17, no. 1, p. 95, Jul. 2017.