****

**AI Chat Bot for Banking**



**Mayank Mamgain**

# Executive Summary

1. The rampant Covid-19 stimulated the wide use of chatbots in many different fields to reduce human-to-human contact. Many of the big-name companies like Amazon, Apple, Google have deployed such artificial intelligent chatbots in their working sectors.
2. For the Banking sector the situation is no different, and a AI enabled chatbot could bring benefits such as reduction in customer waiting time as well as the reduction in workload in a call center (and associated costs) especially during the COVID-induced remote surge.
3. Banking sectors can deploy such AI enabled chatbots in popular third-party platforms like WhatsApp and Telegram which are the most common mode of communication. We explored this possibility using an AI model, some banking dataset to train it, NLP techniques to feed the data to the model and then some free-to-use API to deploy the chatbot in WhatsApp.
4. Our training data for the model has 8623 commonly asked questions by customers of a bank which are divided into 77 categories. Each category has a pre-defined response. We try to predict the category from the question asked.
5. The model architectures we used for predicting response labels on banking dataset were LSTM and Bi-Directional LSTM.
6. The text-processing steps of removing punctuations and lemmatization were carried out before tokenizing. Lemmatization alone led to 6% boost in the test-accuracy. The steps of tokenization, sequence encoding, and sequence padding were carried out using Keras preprocessing tools: the Tokenizer Class and pad\_sequence module.
7. Models performed well with both LSTM and Bi-directional LSTM achieving test-set accuracy of ~79%. We were able to slightly overcome the overfitting issue with the use of Bi-directional wrapper class by reducing the gap between train and test-set by ~10% as compared to unidirectional LSTM.
8. The chatbot is deployed on the popular WhatsApp platform using a free-to-use Twilio account, an API which can interact with WhatsApp.
9. By converting the chatbot into a Flask App, using applications like ‘ngrok’ to enable communication via internet between Twilio and our Flask chatbot App, anyone can talk to our chatbot via WhatsApp.
10. The chatbot can identify 77 types of categories of questions with 79% accuracy, and provide an appropriate response.

Contents

[1. Introduction & Objective 3](#_Toc27779)

[2. Literature Overview 4](#_Toc6110)

[3. Business Problem 7](#_Toc13546)

[4. Data Overview 8](#_Toc21469)

[5. Data Preparation 1](#_Toc23153)2

[6. Model and Configuration 1](#_Toc7657)3

Model Architecture 1- LSTM

Model Architecture 2- Bidirectional LSTM

[7. Business Solution & Recommendation 1](#_Toc23764)6

[8. Conclusion and Future Study 2](#_Toc9200)1

[9. References 2](#_Toc10016)2

# Introduction & Objective

The chatbot market was valued about $17.17 billion in 2020, and it is predicted to reach by $102.29 billion in 2026, with a growth rate of 34.75% over the forecast period, 2021 - 2026. Virtual assistants are increasingly used in many industries because of deep neural networks, machine learning, and other advancements in AI technologies.

The simplest AI chatbots were originally designed to serve up answers to Frequently Asked Questions, based on pre-configured knowledge systems tied to a limited set of specific keywords. Banking answers for each question needed to be programmed in, and the bot followed a fixed decision path to generate an answer. More recently, Natural Language Understanding emerged to create intelligent chatbots. Here the AI chatbot ‘learns’ to understand a question the same way a human does. After millions of repetitions and broad coverage of questions, the bot has learned enough to be useful. This process is called ‘AI Training.’ The challenge with this approach is that the bot starts with no knowledge. The generic AI systems used by Google, Amazon and others have tremendous learning capacity, but they start out with no banking knowledge. It all has to be trained, at great time and expense.

* Huge datasets of questions are required to train pre-launch.
* Training continues with live customer conversations.
* Bank-specific training is required for a chatbot to know that a “branch” means a building, not a tree part.

Virtual assistants must be taught to understand the context of a question. A banking-specific proprietary dataset starts with thousands of questions and answers related to the products and services offered by the institution and with those questions customers actually ask. The key to success is starting with a chatbot pre-trained on understanding the banking-specific questions, not trying to build the knowledge from a general purpose AI platform.

A wide range of banking services can be automated with the help of bots, substantially slashing costs while boosting operational efficiencies. Applied to customer service, it can proactively surface the right content and resolve customer issues 24/7. BI estimates that while the average cost of enabling a customer transaction via phone is around $2.50, for a digital transaction (online or on mobile) it comes down to around $0.17. With constant improvement in technologies such as such as natural language processing (NLP) and machine learning (ML), these bots will become as effective (if not better) in customer engagement and query handling as humans - making them a feasible alternative to call centers in the long run.

Although, there are many different applications of a chatbot in the banking sector for example - chatbots can assist in lead generation, serve as an aid to cross-sell products, serve as an assistant to manage money etc, our goal is to build a chatbot capable of reducing the dependence on human customer service representatives. The AI bot enables a bank to provide 24×7 support, give prompt answers and enhance customer interactions

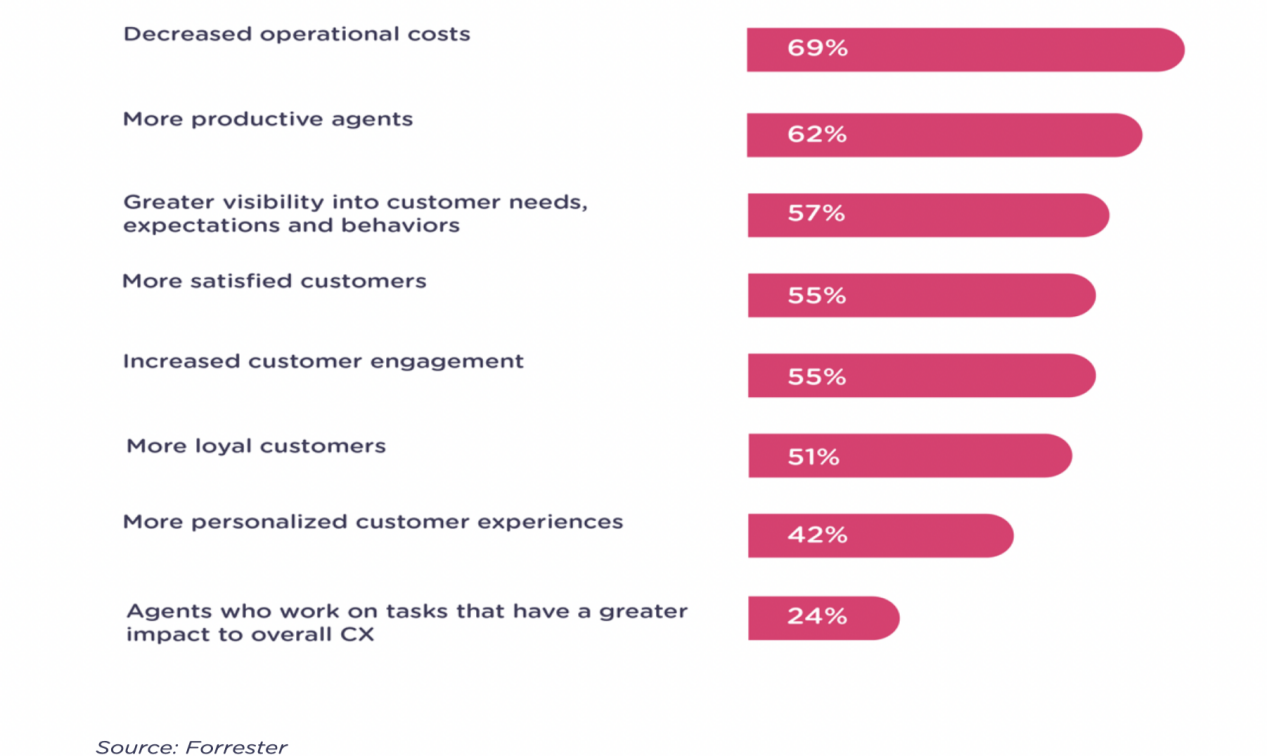
In this project, we built a chatbot which can provide answers to simple and most common customer queries received by a bank. Through the use of Artificial Intelligence - Recurrent Neural Networks and LSTMs, the chatbot can recognize the intent of the user, assign a particular category to it, and then provide an appropriate response. Furthermore we explore the ways to deploy the chatbot in the WhatsApp platform for easy access and provide immediate responses.

# 2. Literature Overview

**1. Business Literature: advanced chatbots applications**

Ample evidence exists to support the view/hypothesis that embedding chat bots in the systems seems to have so many advantages. According to Forrester’ statistic, for example, chat bots can decrease operational cost by 69%, it can better fulfill customers experiences and so on.

In addition, an Accenture survey showed that a staggering 91% of consumers are more likely to choose brands that provide tailor-made offers and recommendations. There are many statistics like this. According to a study by Epsilon, 80% of consumers are more likely to purchase products that provide a personalized experience.



1. **Technical Literature: Recurrent neural network**

**Why not Vanilla RNN?**

**The Problem, Short-term Memory**

Recurrent Neural Networks suffer from short-term memory. If a sequence is long enough, they’ll have a hard time carrying information from earlier time steps to later ones. So if you are trying to process a paragraph of text to do predictions, RNN’s may leave out important information from the beginning. During back propagation, recurrent neural networks suffer from the vanishing gradient problem. Gradients are values used to update a neural networks weight. The vanishing gradient problem is when the gradient shrinks as it back propagates through time. If a gradient value becomes extremely small, it doesn’t contribute too much learning. So in recurrent neural networks, layers that get a small gradient update stops learning. Those are usually the earlier layers. So because these layers don’t learn, RNN’s can forget what it seen in longer sequences, thus having a short-term memory

**Why LSTMS?**

LSTMs were created as the solution to short-term memory. An LSTM has a similar control flow as a recurrent neural network. The differences are the operations within the LSTM’s cells Below is the schematics representation of LSTM unit.

Diagram, schematic

Description automatically generated

The core concept of LSTM’s are the cell state, and it’s various gates. The cell state act as a transport highway that transfers relative information all the way down the sequence chain. You can think of it as the “memory” of the network. The cell state, in theory, can carry relevant information throughout the processing of the sequence. So even information from the earlier time steps can make it’s way to later time steps, reducing the effects of short-term memory.

For our project, we have used LSTM as well as Bidirectional LSTM architecture to compare our model performance on the banking dataset.

**Bidirectional LSTMs**

Bidirectional recurrent neural networks(RNN) are really just putting two independent RNNs together. This structure allows the networks to have both backward and forward information about the sequence at every time step. Using bidirectional will run your inputs in two ways, one from past to future and one from future to past and what differs this approach from unidirectional is that in the LSTM that runs backward you preserve information from the future and using the two hidden states combined you are able in any point in time to preserve information from both past and future.

A picture containing clock, watch

Description automatically generated

As in NLP, sometimes to understand a word we need not just to the previous word , but also to the coming word  . In Bi-LSTM, both activations(forward , backward) would be considered to calculate the output y^ at time t.

Diagram

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Ref: [https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-ste p-explanation-44e9eb85bf21](https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21)

Ref:

https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0

For our banking dataset chatbot implementation, we have implemented both the LSTM and Bi-directional LSTM. While both the models performed equally well, the issue of over-fitting was slightly resolved using the Bi-directional wrapper class by Keras. The model configuration and architecture explanation can be found in the subsequent section “Model and Configuration”

# Business Problem

Today’s financial institutions and the people they serve waste too much time on routine queries. When COVID-19 hit, customers waited on hold, sometimes for hours, trying to speak with an agent. This should not have to be the case. Most customer service interactions queries are basic and transactional, making them ripe to be answered quickly by a chatbot.

* 75% of customer service requests are routine requests for information and action.
* The other 25% of requests are complex. These are customer-specific problems or special cases.

The complex requests are where your team should be spending their time.

This benefits the bank in two ways: Customer wait times are reduced, and call center human workload (and associated costs) are reduced, especially during the COVID-induced remote banking surge.

During pandemic in banking fields, customers and staffs are reluctant to walk in the branch with risk of being infected. Business conducted offline seems to be less efficient and more restricted, which restricts the experience of people and leads them into inconvenience. However, operating services online seems to be safer but more operational costs. Though handling business online will spike to some extent, since customers do not need to walking in branches and it’s more convenient for consulting online even for some mundane questions, human labor is a non- neglect cost in banking system.

The top 4 issues faced by banks today due to the pandemic are:

* From online banking to account access, customers want to access products and services at any time through online processing.
* Develop a personalized strategy: leading customers in to a right business sector by learning users’ intent.
* Choose the right technology: Chatbots are excellent in helping to improve banking sector
* Provide 24 x 7 support - chatbots never sleep, unlike human customer representatives chatbots can operate 24 x 7 saving costs and enhancing customer experience

We try to explore the last usage of a chatbot in the banking sector. How can we build a chatbot which can recognize the intent of the customer, and based on that provide and appropriate response?

# Data Overview

The training data to design a chatbot for banking customer service was obtained from Github: <https://github.com/jianguoz/Few-Shot-Intent-Detection> -(BANKING77)

The data contains 77 categories of inputs or queries that the user may ask the chatbot. There are a total of 8622 queries which are assigned to 77 categories as shown below:

|  |  |  |
| --- | --- | --- |
| Sno | category | No. of inputs |
| 1 | activate\_my\_card | 141 |
| 2 | age\_limit | 92 |
| 3 | apple\_pay\_or\_google\_pay | 108 |
| 4 | atm\_support | 69 |
| 5 | automatic\_top\_up | 108 |
| 6 | balance\_not\_updated\_after\_bank\_transfer | 152 |
| 7 | balance\_not\_updated\_after\_cheque\_or\_cash\_deposit | 162 |
| 8 | beneficiary\_not\_allowed | 137 |
| 9 | cancel\_transfer | 138 |
| 10 | card\_about\_to\_expire | 110 |
| 11 | card\_acceptance | 47 |
| 12 | card\_arrival | 135 |
| 13 | card\_delivery\_estimate | 93 |
| 14 | card\_linking | 120 |
| 15 | card\_not\_working | 93 |
| 16 | card\_payment\_fee\_charged | 167 |
| 17 | card\_payment\_not\_recognised | 151 |
| 18 | card\_payment\_wrong\_exchange\_rate | 147 |
| 19 | card\_swallowed | 45 |
| 20 | cash\_withdrawal\_charge | 157 |
| 21 | cash\_withdrawal\_not\_recognised | 142 |
| 22 | change\_pin | 104 |
| 23 | compromised\_card | 72 |
| 24 | contactless\_not\_working | 30 |
| 25 | country\_support | 112 |
| 26 | declined\_card\_payment | 135 |
| 27 | declined\_cash\_withdrawal | 153 |
| 28 | declined\_transfer | 117 |
| 29 | direct\_debit\_payment\_not\_recognised | 162 |
| 30 | disposable\_card\_limits | 102 |
| 31 | edit\_personal\_details | 103 |
| 32 | exchange\_charge | 101 |
| 33 | exchange\_rate | 95 |
| 34 | exchange\_via\_app | 98 |
| 35 | extra\_charge\_on\_statement | 146 |
| 36 | failed\_transfer | 119 |
| 37 | fiat\_currency\_support | 106 |
| 38 | get\_disposable\_virtual\_card | 78 |
| 39 | get\_physical\_card | 88 |
| 40 | getting\_spare\_card | 111 |
| 41 | getting\_virtual\_card | 82 |
| 42 | lost\_or\_stolen\_card | 66 |
| 43 | lost\_or\_stolen\_phone | 104 |
| 44 | order\_physical\_card | 100 |
| 45 | passcode\_forgotten | 87 |
| 46 | pending\_card\_payment | 140 |
| 47 | pending\_cash\_withdrawal | 123 |
| 48 | pending\_top\_up | 129 |
| 49 | pending\_transfer | 130 |
| 50 | pin\_blocked | 96 |
| 51 | receiving\_money | 80 |
| 52 | Refund\_not\_showing\_up | 143 |
| 53 | request\_refund | 149 |
| 54 | reverted\_card\_payment? | 142 |
| 55 | supported\_cards\_and\_currencies | 109 |
| 56 | terminate\_account | 91 |
| 57 | top\_up\_by\_bank\_transfer\_charge | 93 |
| 58 | top\_up\_by\_card\_charge | 96 |
| 59 | top\_up\_by\_cash\_or\_cheque | 95 |
| 60 | top\_up\_failed | 127 |
| 61 | top\_up\_limits | 79 |
| 62 | top\_up\_reverted | 128 |
| 63 | topping\_up\_by\_card | 87 |
| 64 | transaction\_charged\_twice | 158 |
| 65 | transfer\_fee\_charged | 153 |
| 66 | transfer\_into\_account | 94 |
| 67 | transfer\_not\_received\_by\_recipient | 152 |
| 68 | transfer\_timing | 112 |
| 69 | unable\_to\_verify\_identity | 86 |
| 70 | verify\_my\_identity | 87 |
| 71 | verify\_source\_of\_funds | 97 |
| 72 | verify\_top\_up | 107 |
| 73 | virtual\_card\_not\_working | 32 |
| 74 | visa\_or\_mastercard | 115 |
| 75 | why\_verify\_identity | 102 |
| 76 | wrong\_amount\_of\_cash\_received | 160 |
| 77 | wrong\_exchange\_rate\_for\_cash\_withdrawal | 145 |

We have kept split 50% data as trainset and applied the remaining data as testset for our model. Both train and test have at least one input per 77 categories.

Example head of the data:

|  |  |  |
| --- | --- | --- |
| sno | sequence | label |
| 1 | i am still waiting on my card? | card\_arrival |
| 2 | what can i do if my card still hasn't arrived after 2 weeks? | card\_arrival |
| 3 | i have been waiting over a week. is the card still coming? | card\_arrival |
| 4 | can i track my card while it is in the process of delivery? | card\_arrival |
| 5 | how do i know if i will get my card, or if it is lost? | card\_arrival |
| 6 | when did you send me my new card? | card\_arrival |
| 7 | do you have info about the card on delivery? | card\_arrival |
| 8 | what do i do if i still have not received my new card? | card\_arrival |
| 9 | does the package with my card have tracking? | card\_arrival |

# Data Preparation

**Data Wrangling: Text- Preprocessing of input to chatbot**

1. **Text- Preprocessing of input to chatbot**

**1) Cleaning**:

1. **Remove non-alpha characters**: Regular expression to remove the non-alpha characters
2. **Remove punctuations**: string.punctuation from the string library to remove punctuations
3. **Lemmatization**: Lemmatizing maps common words into one base thus will help us reduce the vocab\_size to the model. We have used WordNetLemmatizer from nltk.stem. Lemmatizing alone was able to boost our test accuracy by 6%.

**2) Tokenize data**:

1.   Tokenize the data using Keras tokenizer class specify oov\_token='<unk>' for unknown  words in the test set.

2.   Encode training data sentence into sequences using Keras. Tokenizer.texts\_to\_sequences

3.   Padding sequences to equal length using keras.preprocessing.sequence.pad\_sequences. For our project, we have used the s etting as padding='pre'

# Model and Configuration

**Implementation of Model Architecture 1:**

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

In the model architecture 1, We used the Keras library to build a neural network classifier. It is a high-level framework based on tensorflow, theano or cntk backends.

1. **Embedding Layer**: The model begins with an embedding layer which turns the input integer indices into the corresponding word vectors. Word embeddings allow the value of the vector’s element to be trained. After training, words with similar meanings often have the similar vectors. For the purpose of our project,

we are using pre-built embedding layer class from Keras.

1. **SpatialDropout1D**: performs variational dropout in NLP models.
2. **LSTM layer:**  layer with memory units assigned as embed\_dim=128 for our dataset
3. **Output Dense Layer**: Contains values equal to the number of target labels

which is 77 for our banking dataset.

1. **Activation function**: softmax for multi-class classification.
2. **Loss\_function**: Because it is a multi-class classification problem, categorical\_crossentropy is used as the loss function

The output from our model essentially predicts a label for the input amongst the 77 targets it is trained on. It is a many to one operation.Once the label is classified, the corresponding response is returned as the output of the query entered by the user.

We basically use one type of data in the process, “bank”. For “bank” , they are the most important datasets we mainly focus on, which are highly correlated to the questions that customers may ask when they are trying to talk real officers. In order to precisely catch the intention behind questions. “Bank” categorize many questions into different groups, such as “withdraw”, “currency exchange” and “card lost”. There are totally 77 types of questions we formulated and we compile each type of categories in specific content manually so that our chatbots’ responses to customers are more effective, increasing users experiences. The following steps is a process to reveal that how our chatbots work.

Receiving a sequence from customers - Labels the sequence - Chatbots response

Two Examples also given below

Customers: “ my pin code still doesn’t arrive” or “I don’t receive my pin code”.

**Response; “Please wait for a second, I will check your delivery information for you.”**

When customers ask these two questions, our dataset will label them as category of “Check delivery” and give counter-part response.

**Customers: “ why did they charge me an extra $1? ” or “what is this extra pound charge for?”.**

**Response; “This is the service fee.”**

When customers ask questions about one extra dollar transaction, our dataset will label them as category of “extra\_charge\_on\_statement” and response is to tell them that it may be the service fee.

**Here is the output of training our model on train-set**

Chart, line chart

Description automatically generated

Here is our test-set accuracy:

Text

Description automatically generated

We see that our model has over-fitting issues. We will take note of this and work in further optimizing the model to reduce the gap between the training and test-set accuracy.

**Implementation of Model Architecture 2: Using Bidirectional LSTM**

Text

Description automatically generated

1. **Embedding Layer**: The model begins with an embedding layer which turns the input integer indices into the corresponding word vectors.
2. **Dropout Layer**: Added the dropout layer to combat overfitting.
3. **BiDirectional with LSTM Layer:** For the purpose of our project, we are using Bidirectional wrapper class from Keras.
4. **Dense:** Added another Dense layer with number of units same as embed\_dim=128.
5. **Dropout Layer**: Added another dropout layer to combat overfitting.
6. **Dense layer**: This the final layer with values equal to target\_length
7. **Activation function**: softmax for multi-class classification.
8. **Loss\_function**: Because it is a multi-class classification problem, categorical\_crossentropy is used as the loss function

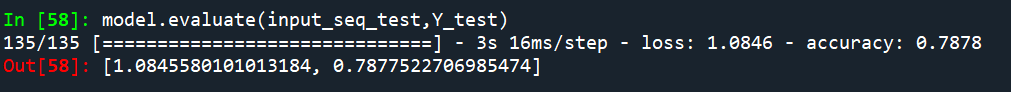
Here is the output of training our model on trainset, the **train-set accuracy**

**has peaked at 0.8831**

Chart, line chart

Description automatically generated

Here is our test-set accuracy:



By this Bidirectional model architecture, we have somewhat managed to combat the problem of overfitting.

# 7. Business Solution & Recommendation

Customers will obviously have a lot of questions as and when their hard-earned money is involved. This keeps banking agents forever busy with queries, not giving them ample time to ensure good experiences for everyone. Thanks to AI bots, banks can think of automating support for common queries and focus more on serving the customers in a better way.

Firstly, our [customer service chatbot](https://www.revechat.com/blog/customer-service-chatbots/) is smart enough to successfully handle various non-complex queries related to banking products and customer accounts. It can answer queries related to 77 types of issues with an 78% test accuracy.

Examples of common customer queries which can be answered by our chatbot:

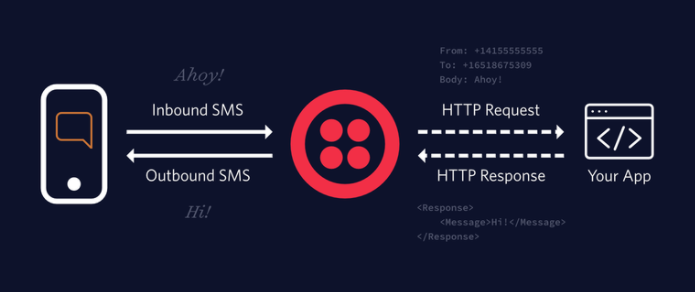
* Card not working
* Information about the status of a transaction
* List of supported ATMs
* Exchange rate

Secondly, using a free API for WhatsApp called ‘Twilio’, we are able to provide chatbot services to anyone with a WhatsApp messenger account.

**Deployment**

The chatbot is deployed on the WhatsApp messenger by converting it into a Flask web application. This web application is connected to a public URL using a utility called ‘ngrok’. Then using ‘Twilio’, which provides a free WhatsApp sandbox our chatbot can communicate with anyone through a number provided by Twilio through the WhatsApp messenger.

How the chatbot works on WhatsApp:



WhatsApp

Twilio

Flask App

1. **Flask App:** The Twilio API for WhatsApp uses a webhook to notify an application when there is an incoming message. We defined a webhook using Flask as given below:

from flask import Flask

app = Flask(\_\_name\_\_)

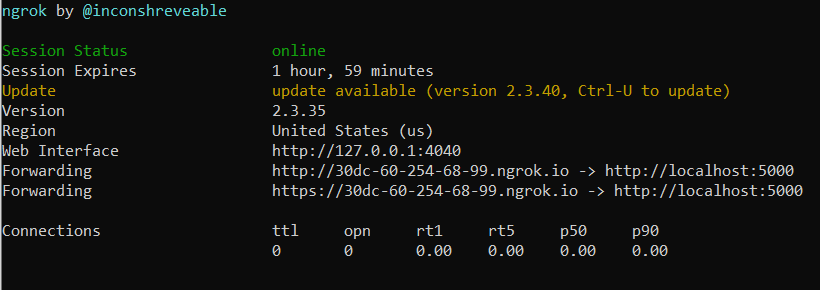
@app.route('/bot’, methods=['POST'])def bot():

# add webhook logic here and return a response

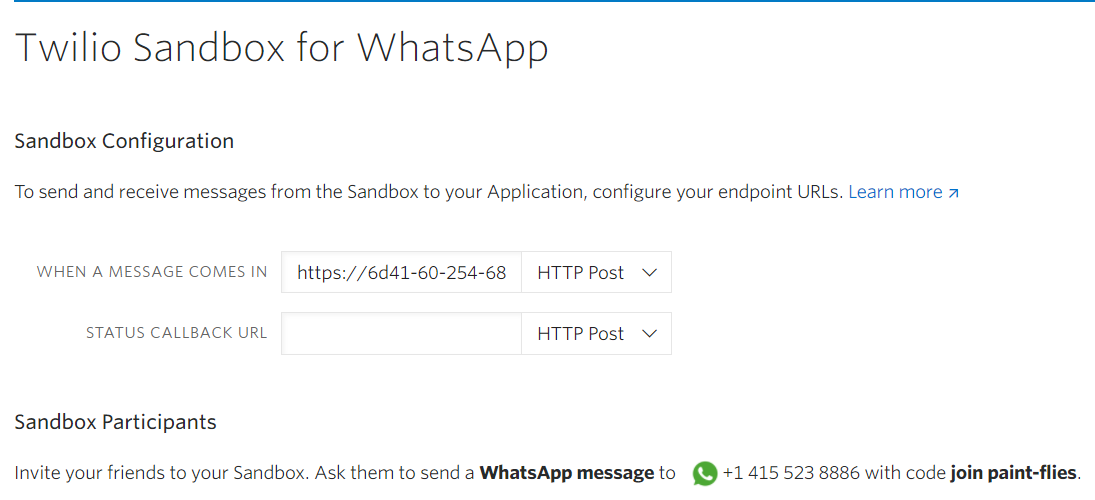
if \_\_name\_\_ == '\_\_main\_\_':

app.run()

1. **Ngrok:** Our Flask App runs as a private service on port 5000 inside our computer and will sit there waiting for incoming connections. To make this service reachable from the Internet we need to use ngrok. It allocates a temporary public domain that redirects HTTP requests to our local port 5000.



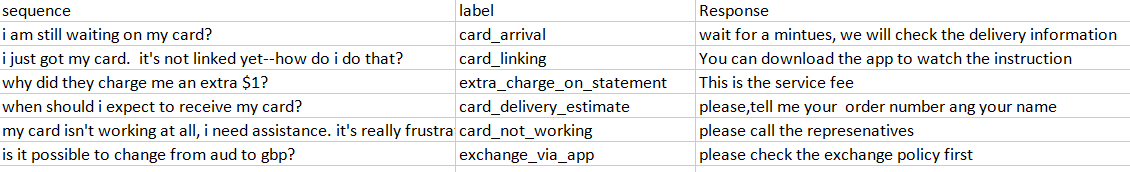
1. **Twilio:** Twilio is a cloud communications platform which allows software developers to programmatically make and receive phone calls, send and receive text messages, and perform other communication functions using its web service APIs. We make use of a free Twilio WhatsApp sandbox which can communicate with our chatbot through the temporary public domain given to us via ngrok.



Anyone can communicate with our chatbot by sending a WhatsApp message ‘join paint-flies’ to this number provided by Twilio: +14155238886.

# Results

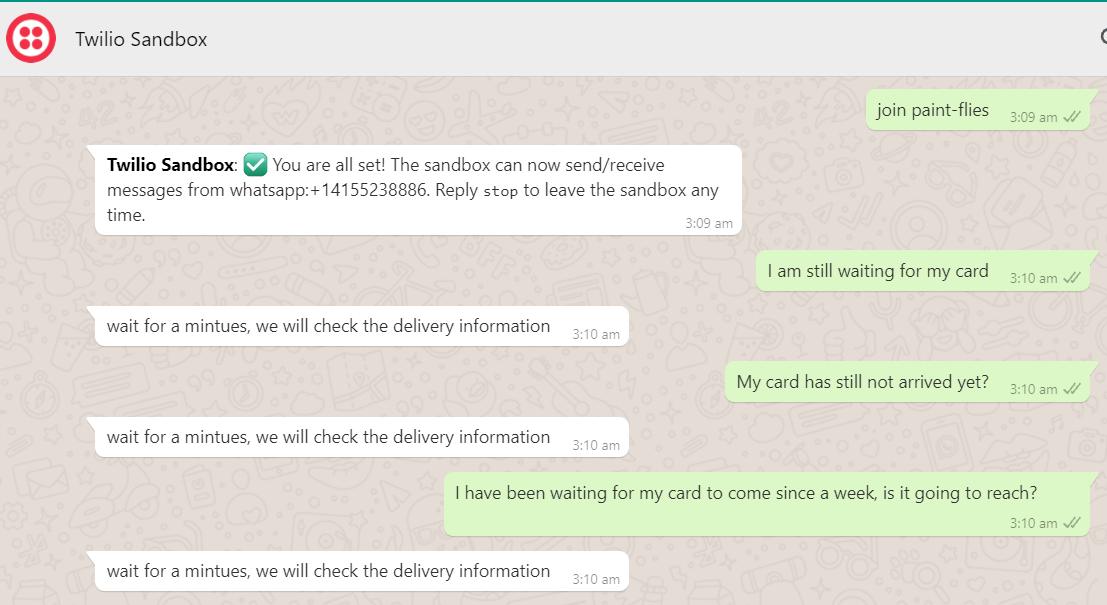
The model assigns a category to the query and based on the category a response is sent back. Each category has one type of response. Some of the responses based on the categories are given below:



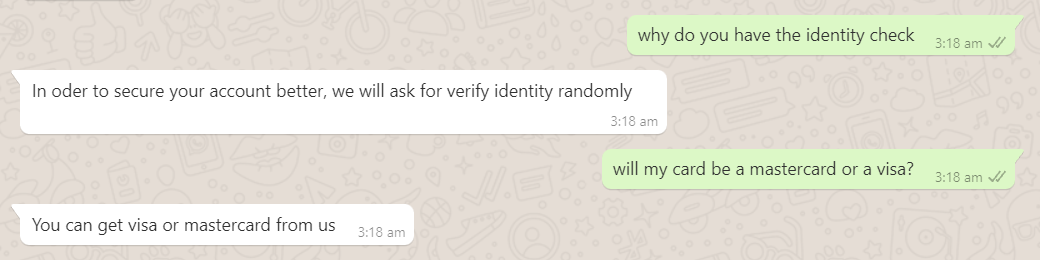
There could be different ways to say the same thing, our model can recognize this and return the correct category for example:

|  |  |
| --- | --- |
| sequence | label |
| i am still waiting on my card? | card\_arrival |
| what can i do if my card still hasn't arrived after 2 weeks? | card\_arrival |
| i have been waiting over a week. is the card still coming? | card\_arrival |

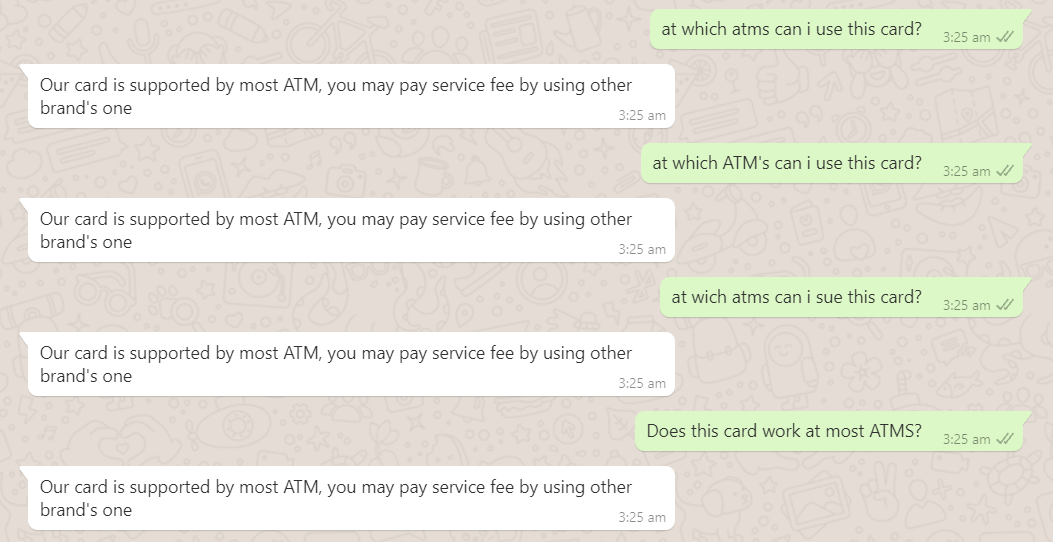
**Examples:**



1. All queries are of the category: card\_arrival and the chatbot is able to deduce it.



1. Different categories have a different response



1. Same question asked in different ways which the chatbot recognizes



4. Overall the chatbot is fairly accurate in determining the category. However it must be noted that if we ask unrelated questions, the chatbot will still try to assume it is related to the banking data and assign a category.

# 8. Conclusion and Future Study

We started with intent recognition dataset which was a particularly smaller dataset to help predict the responses in the category of the intent/emotion as learnt by the machine. From there, we set out to built a chatbot that will prove useful in automating responses to customer queries in the banking sector that will heavily reduce the workload. We went a step ahead and also deployed our model to WhatsApp so that the service is available at the tip of customer’s fingers. Through the high-performing LSTMs, we managed to achieve a high accuracy ~80% on test-set

However, there is room for improvement and as part of our future work we would like to work in below areas to enhance the model accuracy and reduce the problem of overfitting which our model is currently facing.

1. We look forward to trying out pre-trained word embeddings such as GloVe and compare the performance with the current Embedding layer in Keras.
2. We would like to explore other optimizer settings such as the embeddings\_regularizer and

embeddings\_constraint as well as other regularizes available as parameters of LSTM layer in Keras such as kernel**\_regularizer** , **recurrent\_regularizer** to study the impact on learning of model.

1. We would further like to develop a Seq2seq model for creating a generative chatbot that as the names implies generate responses which will be a massive improvement on our many to 1 text-classification chatbot.

# 9. References

1. ILLUSTRATED GUIDE TO LSTM’S AND GRU’S: A STEP BY STEP EXPLANATION In-text: (Illustrated Guide to LSTM’s and GRU’s: A step by step explanation, 2021) Your Bibliography: Medium. 2021. Illustrated Guide to LSTM’s and GRU’s: A step by step explanation.

Available at: <https://towardsdatascience.com/illustrated-guide-to-lstms-and-  gru-s- a-step-by-step-explanation-44e9eb85bf21>

MULTI-CLASS TEXT CLASSIFICATION WITH LSTM.In-text: (Multi-Class Text Classification with LSTM, 2021) Your Bibliography: Medium. 2021. Multi-Class Text Classification with LSTM. [online] Available at: <https://towardsdatascience.com/multi-class-text-classification-with-lstm-1590bee1bd17>

3. MULTI CLASS TEXT CLASSIFICATION WITH KERAS AND LSTM In-text: (Multi Class Text Classification with Keras and LSTM, 2021)Your Bibliography: Medium. 2021. Multi Class Text Classification with Keras and LSTM. [online] Available at: <https://djajafer.medium.com/multi-class-text-classification-with-keras-and-lstm-4c5525bef592>

4. Chatbot market: 2021 - 26: Industry share, size, growth - mordor intelligence. Chatbot Market | 2021 - 26 | Industry Share, Size, Growth - Mordor Intelligence. (n.d.). Retrieved October 17, 2021, from https://www.mordorintelligence.com/industry-reports/chatbot-market.

5. Roznovsky, A., & Kovtun, A. (2021, June 15). Chatbots in banking sector: Benefits, examples, and future. Chatbots in Banking Sector: Benefits, Examples, and Future. Retrieved October 17, 2021, from https://light-it.net/blog/chatbots-in-banking-industry-benefits-forecasts-and-more/.

6.Jassova, B. (2021, March 10). Chatbot statistics 2021: Market & Opportunities: Landbot. Landbot.io. Retrieved October 17, 2021, from https://landbot.io/blog/chatbot-statistics-compilation#section-2.

7. Banking Training Data for Chatbot: <https://github.com/jianguoz/Few-Shot-Intent-Detection>

8. WhatsApp Deployment: <https://www.twilio.com/blog/build-a-whatsapp-chatbot-with-python-flask-and-twilio>

9.Tokenization and text data preparation with TensorFlow & Keras. KDnuggets. (n.d.). Retrieved October 18, 2021, 10.https://www.kdnuggets.com/2020/03/tensorflow-keras-tokenization-text-data-prep.html.

11.Aggarwal, R. (2019, July 4). Bi-LSTM. Medium. Retrieved October 18, 2021, from https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0.