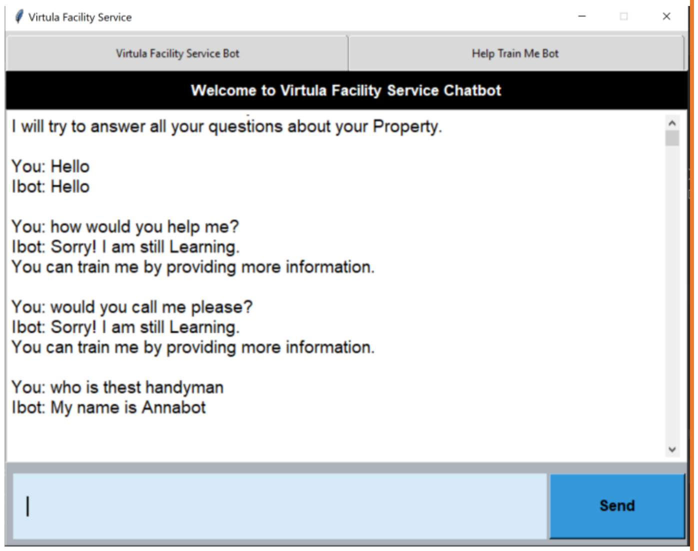

Virtula Facility Service
Virtula Facility Service Bot
Help Train Me Bot
Welcome to Virtula Facility Service Chatbot
Send

```
I will try to answer all your questions about your Property.

You: Hello
Ibot: Hello

You: how would you help me?
Ibot: Sorry! I am still Learning.
You can train me by providing more information.

You: would you call me please?
Ibot: Sorry! I am still Learning.
You can train me by providing more information.

You: who is thest handyman
Ibot: My name is Annabot
```

Virtual Facility Service Chatbot

ABSTRACT

Together with Artificial Intelligence and Machine Learning chatbots can interact with humans like how humans interact with each other. The implementation of chatbots is helpful in many cases from customer support to personal assistants. When you first start in virtual facility service, you have a lot of questions. This chatbot will help to answer those questions.

Yousof Rahimian

Belleview University, DSC 680 – Applied Data Science

June, 2022

Virtual Facility Service Chatbot

Topic

Together with Artificial Intelligence and Machine Learning chatbots can interact with humans like how humans interact with each other. The implementation of chatbots is helpful in many cases from customer support to personal assistants. When you first start in virtual facility service, you have a lot of questions. This chatbot will help to answer those questions.

Business Problem

When you are first learning a hobby, you have a lot of questions. Some go to classes, some go to books, and some go online to find answers. The problem arises when a student goes home from a class or reads a book or blog, they still have questions. This chatbot is aimed to help consolidate a lot of questions the user may have into a learning environment where the chatbot can help answer those questions. In this article, I'm going to build a simple but efficient AI Chatbot using Python, NLTK, TensorFlow, and Neural networks. This chatbot is highly customizable and can make changes as you want. So, building my own chatbot for my personal uses or for business makes sense.

Data Explanation

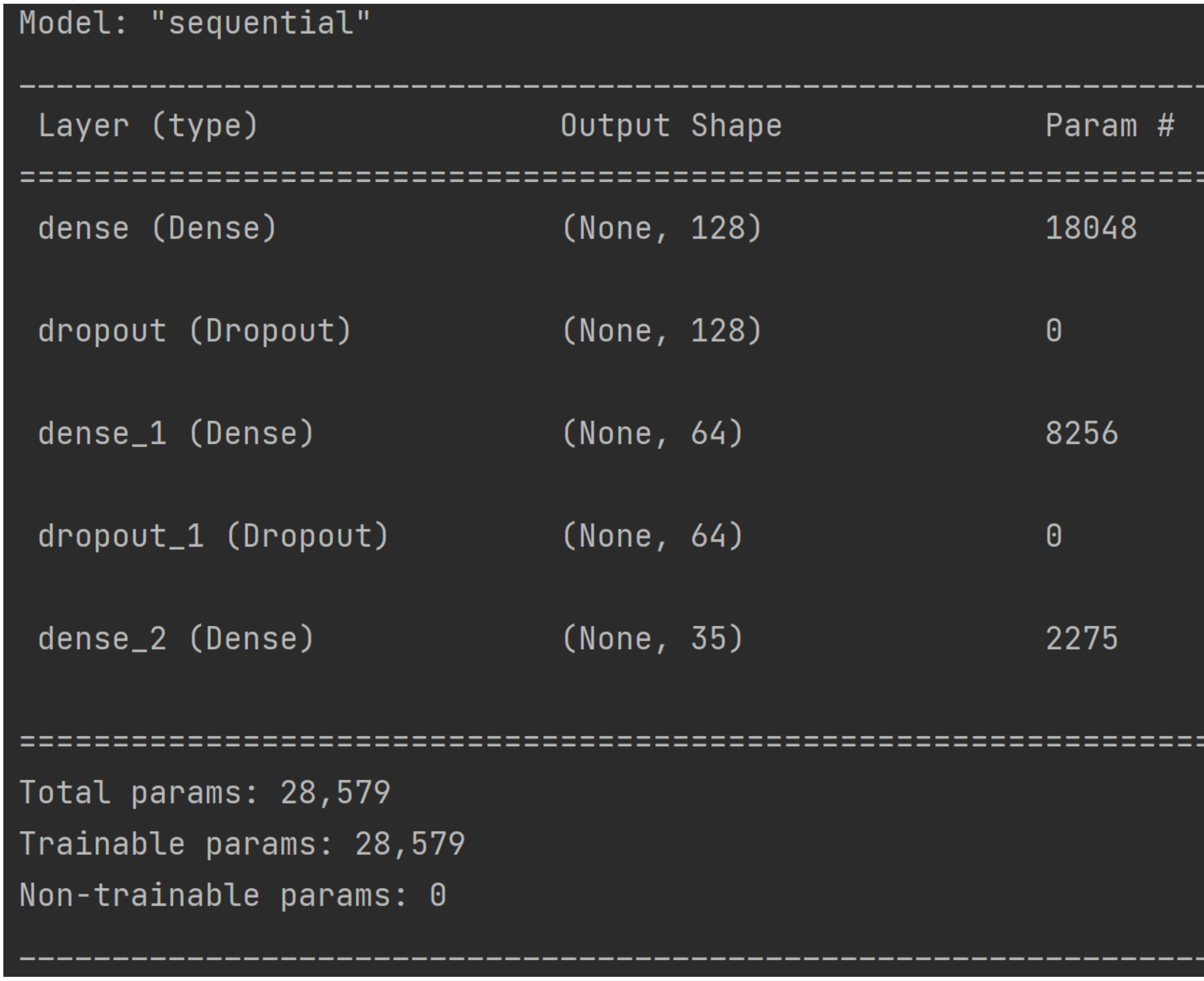
```
{
 "intents": [
  {
   "tag": "greeting",
   "patterns": [
    "Hi",
    "Hello",
    "Hey"
   ],
   "responses": [
    "Hello",
    "Hi",
    "How are you?"
   ]
  },
  {
   "tag": "goodbye",
   "patterns": [
```

For the chatbot to learn, initial data needs to be created. For this, I used Google. Initial questions I had about virtual facility service chatbot were transformed into a JSON file broken down into tags, patterns, and responses. Tags are topics that a usermay query about. For example, if the user asks how are you is, “greeting” would be the tag since it is thetopic of the query. Patterns are the queries that a user may ask about the tag. For example, regarding the tag “name”, a user may ask “What is your name” or “Do you have a name?”. These questions are different ways of asking the chatbot what their name is. These known types of questions would then become patterns. These patterns then will be used for the models to learn from so the user doesn’t have to ask these specific questions to get the response they are looking for. Responses are static responses that will be used for answers to the queries about the tags.

Methods

To create the data, we will use it for training the model, I will use the tags, patterns, and responses I created in the intents JSON file. A vocabulary of all the words will be created from the patterns. Classes will be created from the tags. A list of all the pattern words will be created along with their associated tags. Once this is complete, we can take this data into the Keras model for training. We first con the words into a numerical value using the bags of words technique. The numerical values are then usedto look at all the features (words) to predict the tag (or subject) it is associated with. Once the highest probability is chosen, we can then randomly use the responses to present to the user. If no tag was found, a “help teach me” response is displayed so the user can use the “Train Me” tab to help create a better model to use.

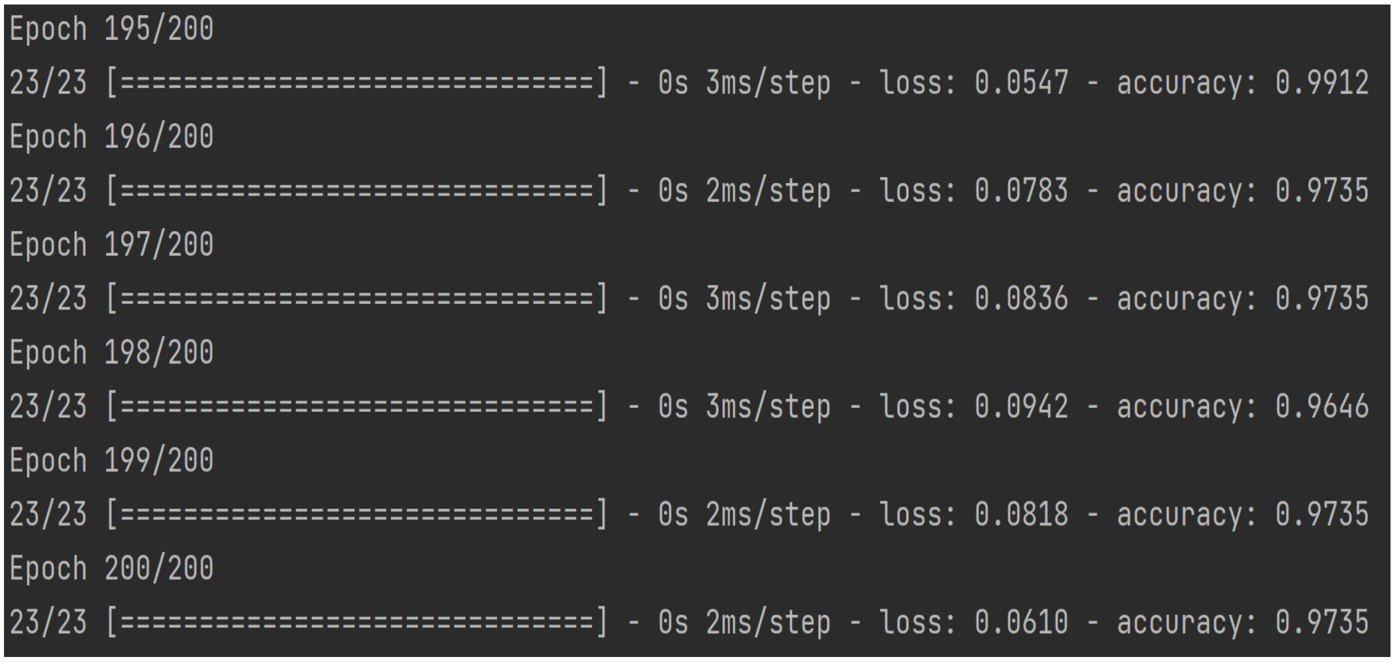
Analysis



```
Model: "sequential"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               18048

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 35)                2275

=================================================================
Total params: 28,579
Trainable params: 28,579
Non-trainable params: 0
_________________________________________________________________
```

We used the Sequential class with three layers and used some dropout layers which help with overfitting the data between layers. The model was then compiled using the categorical cross-entropy for the loss function and the optimizer used was the Stochastic Gradient Descent (SGD). A loss function is used to calculate the quantity that the model should seek to minimize while training your data. Here we are using the categorical cross-entropy loss function as it is used for training categorical data, which we are with the chatbot, to predict the probability of whether the data is part of one class (tag) or another class (tag). The optimizer determines how the network will be updated based on the loss function. With SGD, just a few samples are selected randomly by the model instead of using the whole training dataset for each iteration.

We trained 21,781 words of which none were non-trainable. After training the model, we received between 98 and 100% accuracy with 200 epochs (passes over the training dataset).



```
Epoch 195/200
23/23 [==============================] - 0s 3ms/step - loss: 0.0547 - accuracy: 0.9912
Epoch 196/200
23/23 [==============================] - 0s 2ms/step - loss: 0.0783 - accuracy: 0.9735
Epoch 197/200
23/23 [==============================] - 0s 3ms/step - loss: 0.0836 - accuracy: 0.9735
Epoch 198/200
23/23 [==============================] - 0s 3ms/step - loss: 0.0942 - accuracy: 0.9646
Epoch 199/200
23/23 [==============================] - 0s 2ms/step - loss: 0.0818 - accuracy: 0.9735
Epoch 200/200
23/23 [==============================] - 0s 2ms/step - loss: 0.0610 - accuracy: 0.9735
```

Now that the data has been properly trained, the chatbot will then use the model created from the user's queries (testing). A user types in a question and hits the "SEND" button. Once that is done, the query will be lemmatized and vectorized with the bag of words method as we did with our training data patterns. The model will then predict the tag (or class) of the user's query. The highest probability will be the response from that tag

Here is an example of a user's query:

```
1/1 [==============================] - 0s 54ms/step
hi [('greeting', '0.9994406')]
Query: hi
Bot: Hi

1/1 [==============================] - 0s 14ms/step
What kind of questions do you answer? [('occupation', '0.9987835')]
Query: What kind of questions do you answer?
Bot: I can answer your questions about Stained Glass.

1/1 [==============================] - 0s 14ms/step
How do i start? [('supply', '0.99996984')]
Query: How do i start?
Bot: At Anything Stained glass you can get a starter kit. It has everything you need.

1/1 [==============================] - 0s 14ms/step
Can I be a pro after 1 class? [('grinder', '0.53233564')]
Query: Can I be a pro after 1 class?
```

Going through the example:

Question 1: hi
Response: Hi
This correctly predicted the tag 'greeting' with a 99.9% probability.

Question 2: what kind of questions do you answer?
Response: I can answer your questions about Stained Glass.
This correctly predicted the tag "occupation" with a 98.9% probability

Question 3: how do I start?
Response: At Anything Stained glass you can get a starter kit. It has everything you need.
This correctly predicted the tag "supply" with a 99.9% probability

Question 5: Can I be a pro after 1 class?
Response: I like the Gryhon Studio Pro grinder.
This incorrectly predicted the tag "grinder" with a 53.2% probability, but we could change the threshold to higher than 50% so incorrect answers do not show up.

Question 6: lets go to the moon!
Response: none
This correctly determined that there was no correct tag that matched so the response from the model was an empty array

Conclusion

Although this is a very primitive version of what a chatbot should be, it answers questions to the best of its ability (as any normal human would). As users input more of their expertise into the training tab, the model will be able to have a broader and more decisive model to draw from so as time goes on the chatbot will be a good tool for any beginning-stained glass artist.

Assumptions

This chatbot only knows as much as I can input into its initial learning JSON data. It will give a lot of incorrect responses as it initially learns as not all questions or inquiries will have been considered.

Limitations

The chatbot can only answer questions from the initial data I provided. As more and more users use the chatbot and help it with its learning, this chatbot can only have the limitations of the technology it has been provided. With more upgrades and modifications, there is no limit to its usefulness.

Challenges

The biggest challenge faced was creating a dataset robust enough to answer the many questions a user may have without having to reply that it doesn't know the answer. I think with a good beta environment, stained glass experts can input their knowledge into the model, so it has a better knowledge pool to learn from.

Future Uses/Additional Applications

Once this chatbot is up and running, upgrades to the base learning dataset will need to be done.

Currently, the chatbot only allows additions to the learning dataset. In later upgrades, the chatbot should allow users to modify its current “tags” and their associated “patterns” and “responses” to allow the model to learn with patterns or responses that are not currently there.

Another great addition in future upgrades may be having the ability to display images to the user based on specific queries for visual guidance.

Recommendations

Learning more about Neural Networks and Deep Learning will help in enhancing this chatbot to become “smarter” and thus helping more beginning stained-glass artists. Also, having the ability to save a user’s chats to use to help the chatbot learn (for those who don’t enter their expertise in the training tab) will help.

Implementation Plan

To implement a chatbot, the application would first need to be beta tested with actual stained-glass artists to help build a good foundation for the model to be trained with. Once a few iterations of the beta test have been done, the application should be able to be uploaded to a host for public usage.

Ethical Assessment

As I will be taking questions and answer from users in Facebook groups, there are the ethical considerations of their privacy and not making responses personal of any sort. Also, since I will be creating the answers for this chatbot, it is my personal opinions and responses will be generated.

References

Create Chatbot with Python & Artificial Intelligence. (n.d.). Retrieved from TechVidvan: https://techvidvan.com/tutorials/chatbot-project-python-ai/

Kumar, A. (2020, October 28). *Keras – Categorical Cross Entropy Loss Function*. Retrieved from Data Analytics: https://vitalflux.com/keras-categorical-cross-entropy-loss-function/

Sidharth. (2022, May 31). *How to build a AI chatbot using NLTK and Deep Learning*. Retrieved from PyCodeMates: https://www.pycodemates.com/2021/11/build-a-AI-chatbot-using-python-and-deep- learning.htm