Project Report

Chat-Bot Model Using Keras
(INT 248)

Submitted To
Ms. Ankita Wadhawan

Submitted By
Aakash
11804088
KM086 - B59

Introduction

In this project, I have worked on a Yes/No answering bot. Where the bot is supposed to give answers w.r.t the story provided. I have used Keras library to implement the idea. In easy terms I can say that I have implemented an NLP model. RNNs are used to build this Chat-Bot. RNN used in this project, has been taken from the paper *“End to End Memory Networks”*. This paper implements an RNN like structure that uses an attention model to compensate for the long-term memory issue about RNNs i.e., the problem of vanishing

Literature Review

1. End-To-End Memory Networks

Implementation of neural network with a recurrent over a large external memory. The architecture is a form of Memory Network which is trained end-to-end that requires significantly less supervision during training, making it more applicable in realistic settings.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, Rob Fergus
https://arxiv.org/pdf/1503.08895.pdf

2. An intelligent Chatbot using deep learning with Bidirectional RNN and attention model ("Apr 8th Journal Club | Research Blog")

This paper shows the modelling and performance in deep learning computation for an Assistant Conversational Agent (Chatbot). The utilization of TensorFlow software library, particularly Neural Machine Translation (NMT) model. Acquiring knowledge for modelling is one of the most important tasks and quite difficult to pre-process it. The Bidirectional Recurrent Neural Networks (BRNN) containing attecontainingrs is used, so that input sentence with considerable number of tokens (or sentences with more than 20–40 words) can be replied to with more appropriate conversation.

Manyu Dhyani, Rajiv Kumar
https://www.sciencedirect.com/science/article/pii/S221478532034030X

3. Seq2Seq AI Chatbot with Attention Mechanism

Intelligent Conversational Agent development using Artificial Intelligence or Machine Learning technique is an interesting problem in the field of Natural Language Processing. In many research projects, they are using Artificial Intelligence, Machine Learning algorithms and Natural Language Processing techniques for developing conversation/dialogue agent. "In the past, methods for constructing chatbot architectures have relied on hand-written rules and templates or simple statistical methods."

Abonia Sojasingarayar
https://arxiv.org/pdf/2006.02767.pdf

4. Chatbot Analytics Based on Question Answering System: Movie Related Chatbot Case Analytics

"Question Answer (QA) systems are established to retrieve accurate and concise answers to human queries posted in natural language." The primary focus of the QA system is to achieve efficient and natural interaction between machines and humans. To achieve the above several researchers are directed towards Natural Language Processing (NLP) based deep learning. With the rise of a variety of deep NLP models, it is now possible to obtain a vector form of words and sentences that stores the meaning of the context. NLP aids deep learning-based mathematical models in understanding the semantic and syntax of natural human language.

Jugal Shah, Dr. Sabah Mohammed
Link

5. Towards Building a Neural Conversation Chatbot Through Seq2Seq Model

Improvements in computation and processing power paved a way for Machine learning to be applied more efficiently in real-time and in a lot of applications. In which most prominent area is Natural Language Processing and Natural Language Understanding, which helps the computer to process and understands the natural language used by people.

J.Prassanna, Khadar Nawas K, Christy Jackson J, Prabakaran R, Sakkaravarthi Ramanath
Link

Dataset Used

In 2015, Facebook produced a bAbI dataset and 20 tasks for testing text understanding and reasoning in the bAbI project. “The tasks are described in detail in the paper here.”

The goal of each task is to challenge a unique aspect of machine-text related activities, testing different capabilities of learning models. In this post we will face one of these tasks, specifically the “*QA with single supporting fact*.”

FACT: Sandra went back to the hallway. Sandra moved to the office.

QUESTION: Is Sandra in the office?

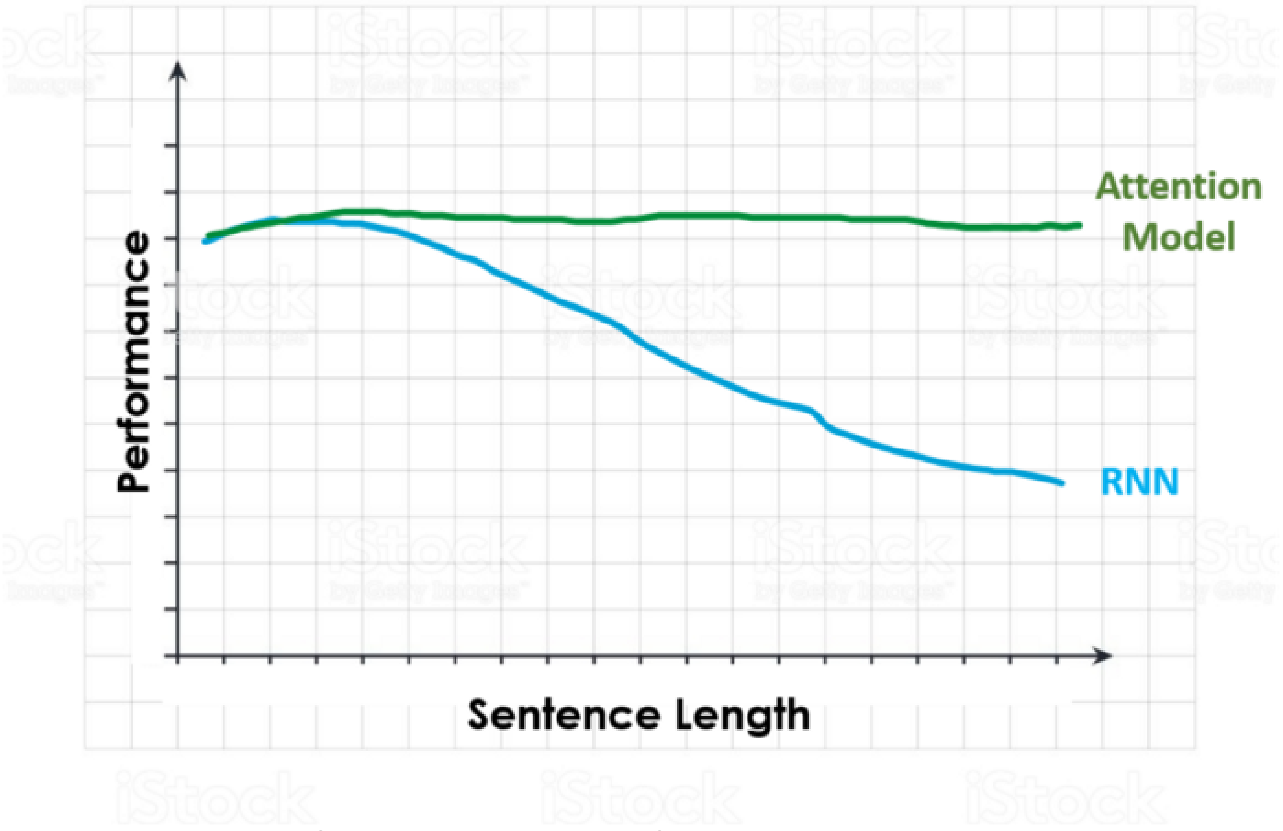
ANSWER: yes

The dataset comes already separated into training data (10k instances) and test data (1k instances), where each instance has a fact, a question, and a yes/no answer to that question.

Link: https://research.fb.com/downloads/babi/

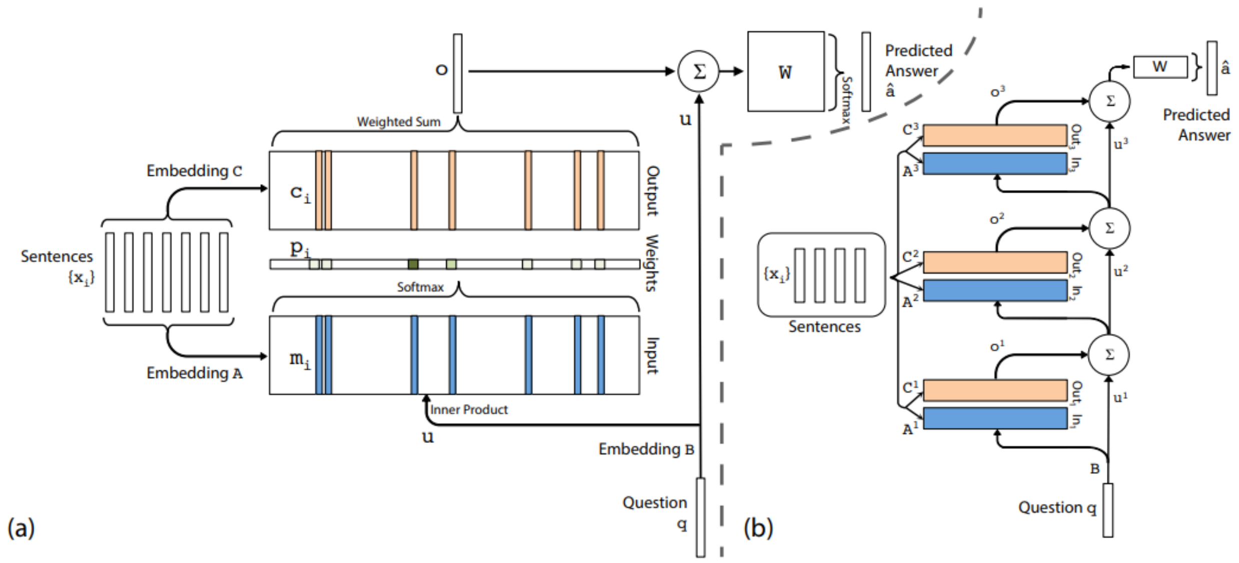
Proposed Architecture

RNNs are used to build this Chat-Bot. RNN used in this project has been taken from the paper *“End to End Memory Networks”*. This paper implements an RNN like structure that uses an attention model to compensate for the long-term memory issue about RNNs i.e., the problem of vanishing gradients.



Performance vs Sentence Length for RNNs and Attention models.

The model takes a discrete set of inputs x1, ..., xn that are to be stored in the memory, a query q, and outputs an answer a. Each of the xi, q, and a contains symbols coming from a dictionary with V words. The model writes all x to the memory up to a fixed buffer size, and then finds a continuous representation for the x and q. The continuous representation is then processed via multiple hops to output a. This allows backpropagation of the error signal through multiple memory accesses back to the input during training



On the left (a) is a representation of a single layer of the model. On the right (b) 3 of these layers are stacked together.

On the left part we can see a representation of a single layer of this model. Two different embeddings are calculated for each sentence, *A* and *C*. Also, the query or question q is embedded, using the *B* embedding.

The A embeddings mi are then computed using an inner product with the question embedding *u* (this is the part where the attention is taking place, as by computing the inner product between these embeddings what we are doing is looking for matches of words from the query and the sentence, to then give more importance to these matches using a *SoftMax* function on the resulting terms of the dot product).

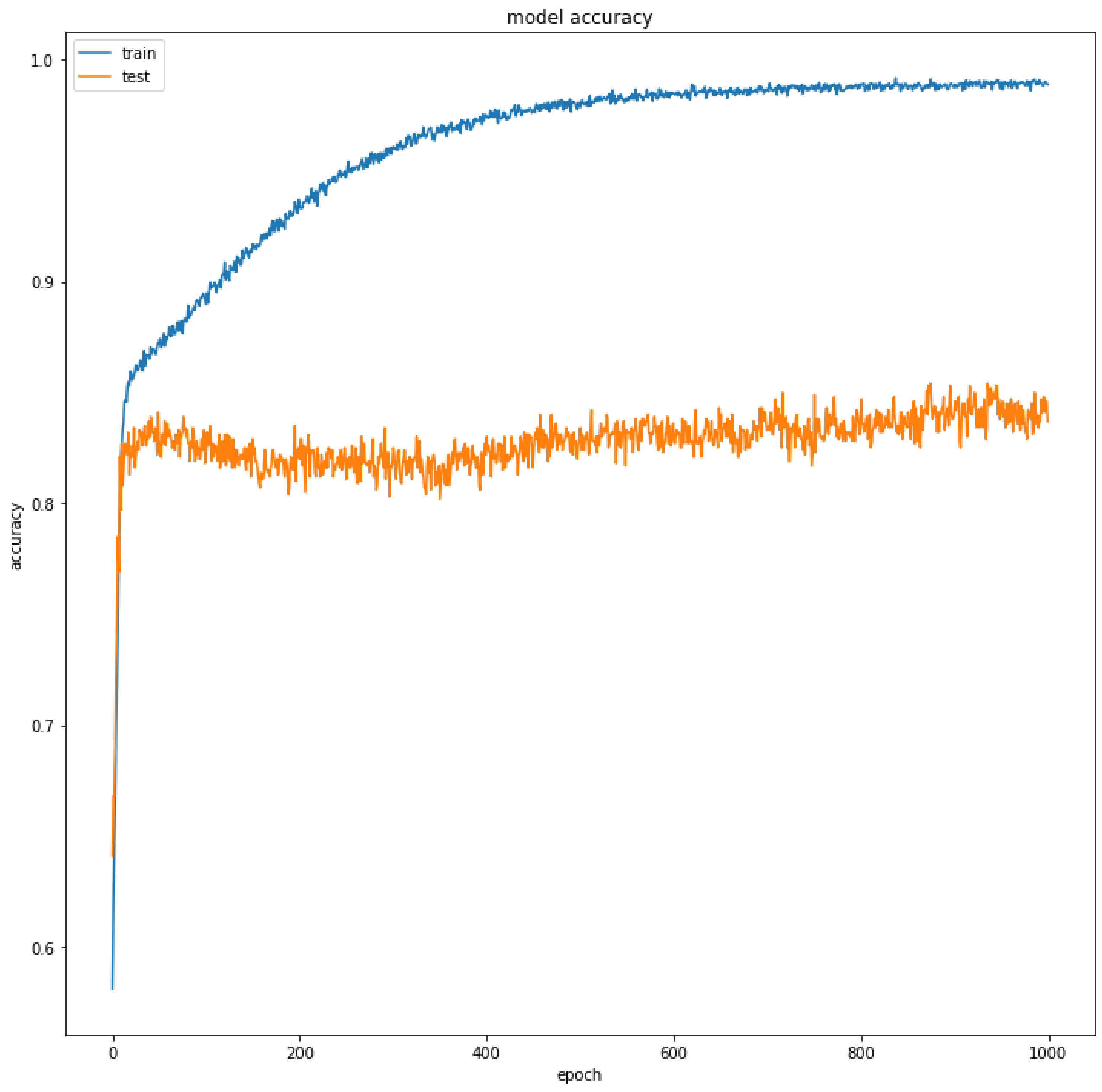
Lastly, we compute the output vector *o* using the embeddings from C *(ci)*, and the weights or probabilities *pi* obtained from the dot product. “With this output vector *o*, the weight matrix *W*, and the embedding of the question *u*, we can finally calculate the predicted answer *a hat*.”

To build the entire network, we just repeat this procedure on the different layers, using the predicted output from one of them as the input for the next one. This is shown on the right part of the previous image.

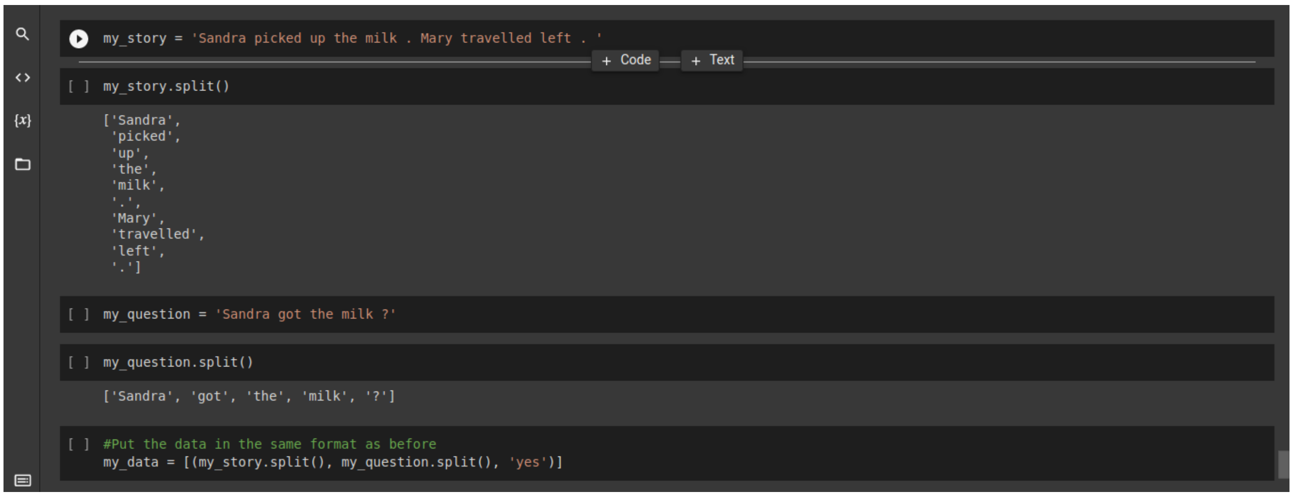
Result and experiment analysis

I have implemented a manual process to vectorize the data. And later automated it by creating functions. Then I have built the network. And later Trained and testes the same.

The model was trained for 1000 epochs. Whereas, without any training also the model was at around 50% accuracy. And after training, the model was at around 100% accuracy.



Screenshots



```
my_story = 'Sandra picked up the milk . Mary travelled left . '
```

```
[ ] my_story.split()
```

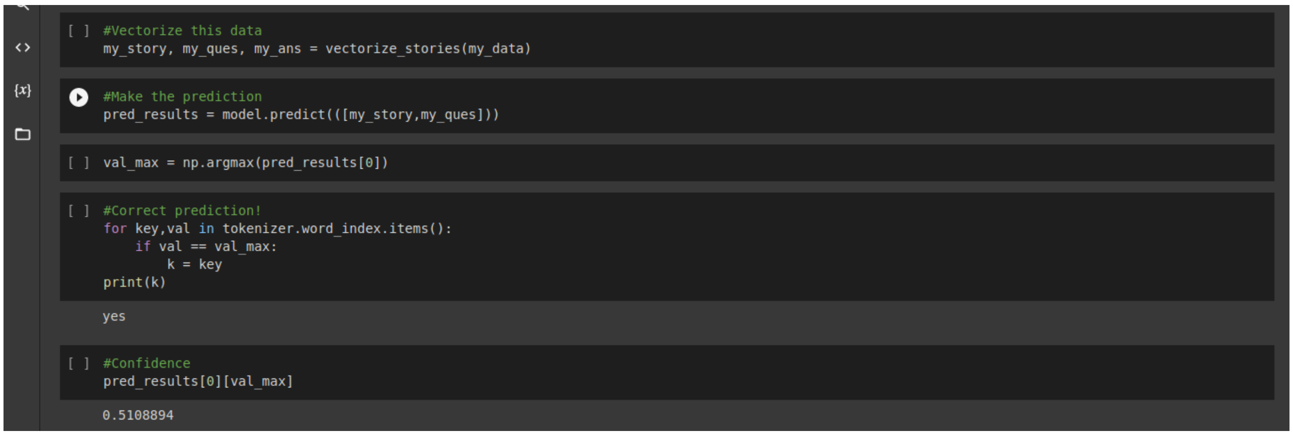
```
['Sandra',
 'picked',
 'up',
 'the',
 'milk',
 '.',
 'Mary',
 'travelled',
 'left',
 '.']
```

```
[ ] my_question = 'Sandra got the milk ?'
```

```
[ ] my_question.split()
```

```
['Sandra', 'got', 'the', 'milk', '?']
```

```
[ ] #Put the data in the same format as before
    my_data = [(my_story.split(), my_question.split(), 'yes')]
```



```
[ ] #Vectorize this data
    my_story, my_ques, my_ans = vectorize_stories(my_data)
```

```
#Make the prediction
pred_results = model.predict(([my_story,my_ques]))
```

```
[ ] val_max = np.argmax(pred_results[0])
```

```
[ ] #Correct prediction!
    for key,val in tokenizer.word_index.items():
        if val == val_max:
            k = key
    print(k)
```

```
yes
```

```
[ ] #Confidence
    pred_results[0][val_max]
```

```
0.5108894
```

Conclusion and Future Scope

Though the bot can give response in the form of “yes/no” only. Since, it is trained with very less vocab, there will be chances where the bot might give wrong answers.

Talking about the future scope of my project, it can be improved and can be made capable of giving answers to a greater number of questions. Some workarounds and some more training would surely do the job.

References

https://arxiv.org/pdf/1503.08895.pdf
https://arxiv.org/abs/1502.05698#
https://towardsdatascience.com/creating-a-chatbot-with-keras-da5ca051e051