**Data:**

The data being used in this project is created manually based on the data which is available on the pace.edu website already. The responses that we will be getting are the links for the query that one is looking for which will be discussed while the presentation. Initially, the data was uploaded in the .json format and then it was converted into a data frame as it will be an easier way to manipulate and transform the data. The example format used in the dataset is as follows.

{

"tag":"greeting",

"input":["hello","hi","hey","hey there","how's it going"],

"response":["hola","hi, how are you","all good, how about you","what's up"]

},

{

"tag":"about pace",

"input":["i want to know about pace","how is pace university","about pace university"],

"response":["please go through the following link https://www.pace.edu/about-pace"]

}, and so on…...

So, “Tag” in the dataset can be considered as the heading for that part inside the flower brackets, whereas input is the expected input that the user will be inputting, and responses are the output that the chatbot will give after referencing the input that was entered by the user. Mian thing to mention here is that “tag” and “input” are part of the training dataset as well as responses can be considered as the target or the predicted value by the model.

**Preprocessing:**

The first step of the preprocessing was to remove the punctuations from the data which was split into training data that is in form of the data frame, to achieve this .apply() was the sting function used. The next step was to concentrate on tokenizing the data using the Tokenizer() function that is already provided by TensorFlow.

**Tokenizing the data:**

Why are we using the Tokenizer in the first place? The reason is that machines do not understand the data, which is in text format, so for machines to understand the data we are passing to is to be converted into the numerical format for which we use the tokenizer and feed the tokenized data into the model. The key point to mention is that, if the parameter “num\_words” is being used in the tokenizer, it is suggested as per the observation that it should be somewhere near to the value which is the total number of words as it varies the usage of the number of epochs (one training iteration). Less number of epochs will cost less money. So, consider if there are 100 words in the dataset, the suggested number of words to keep inside the tokenizer is 99(n-1) as per the tensorflow.org website. But as per the observation in this project that number should be somewhere around 120 ish. This was a bit of an explanation about the parameter being used and the usage of the tokenizer() function. In addition, the pad\_sequences function is being used to convert the tokenized data into the list of 2D NumPy arrays.

**Neural Network:**

Usually, Neural networks are the algorithms that are used to mimic the functionalities of the human brain. They are being used in ML, DL, and AI in recent days. They are highly relied on the training data to learn and improve their accuracy over time as well as they have an input layer, output layer, and hidden layers in between the input and output layers so that they can better understand the data to predict the outcome. In this project Neural network is basically used to recognize the text data that we are creating for the chatbot and give us a response. There are not only two hidden layers in the neural network there are many, but just for understanding purposes, below is the sample picturization of the Neural Network.

**Hidden layer 2**

**Hidden Layer 1**

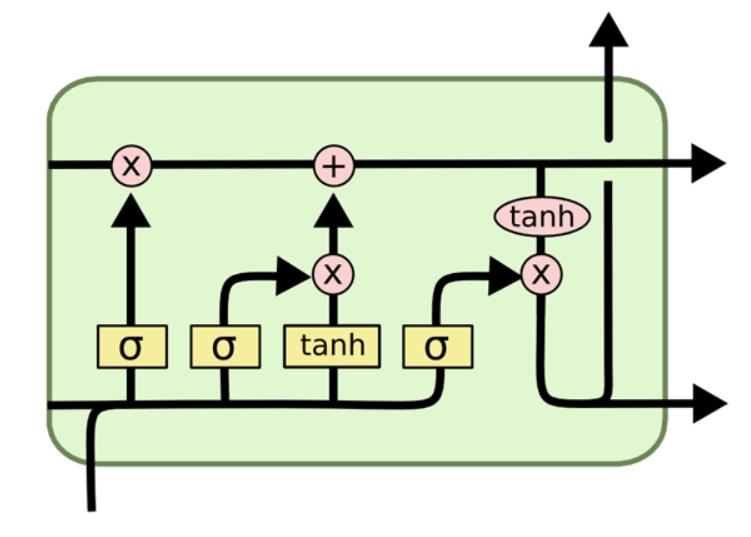
**Input Layer**

**Output Layer**

As already mentioned above there are just 4 layers in the neural network but in general, there are many hidden layers involved in the network so that it understands the data easily.

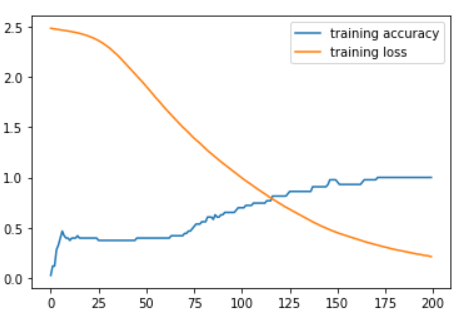
**Model Used: LSTM (Long short-term memory):**

Basically, LSTM can be referred to as one of the artificial neural networks that are used in tasks specifically related to text data as well as they can learn the long-term dependencies usually in sequence prediction problems. LSTMs are also called a special type of Recurrent Neural Networks. In addition, they are useful in ignoring the useless data in the network which is referred to as Gradient vanishing. The layers being used in this project are input, embedding, Flatten, and Dense. More layers can be used if required and based on the research if found something before the submission date. Below is what the one LSTM cell or network looks like.



**Results:**

Accuracy is the metric being used to measure the correctness of this model and the loss function to measure the loss. Right now, I have achieved accurate results for the chatbot on the training data at around 179 epochs where the accuracy is 1.0, initially at epoch 1 the accuracy was 0.02 whilst I ran the notebook before submitting this document. The responses to the chatbot are also executed successfully but with minimal errors which will be demonstrated while presenting the project.



References:

1. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>
2. <https://www.bmc.com/blogs/neural-network-introduction/>
3. <https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer>