

NATURAL LANGUAGE PROCESSING

CONVERSATIONAL CHATBOTS USING TRANSFORMERS

PROFESSOR: VAHID BEHZADAN

TEAM MEMBERS: Sai Priya Mallu

Kavya Pothula

**Abstract:**

Many organizations in a variety of industries are using artificial intelligence into their workflows. We can now design technology capable of replicating human-like interactions, including speech and text recognition, thanks to breakthroughs in Natural Language Processing (NLP) and Natural Language Understanding (NLU).

Use of chatbots is widely increasing now and training a computer to have a conversation like humans is very difficult and requires complicated language models.

**Introduction:**

In place of direct communication with a live human agent, a chatbot or chatterbot is a software application that conducts an online chat conversation using text or text-to-speech. A chatbot is software that may assist clients by automating conversations and interacting with them via messaging networks.

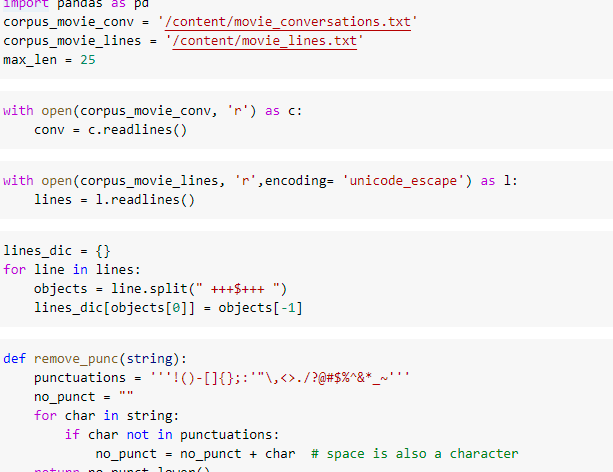
A transformer is a deep learning model that uses the self-attention mechanism to weight the importance of each element of the input data differently. Its primary applications are in natural language processing (NLP) and computer vision (CV).

The Transformer NLP model included a "attention" mechanism that considers the relationship between all of the words in a sentence. It generates differential weightings that show which other parts in the sentence are more important for understanding a problematic word.

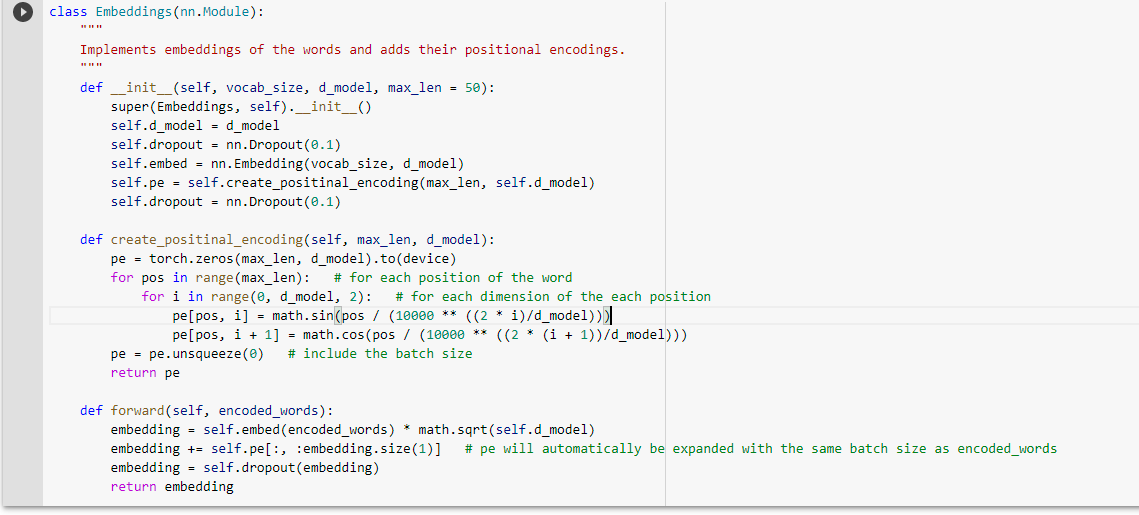
Transformer, like many other sequence-to-sequence models, has an encoder and a decoder. Transformer, on the other hand, uses multi-head attention layers, which are made up of many scaled dot-product attention layers.

In this first we preprocess the cornell movie-dialogs corpus using the datasets.Next we implement multihead attention and a transformer with transformer API.

During this the conversations are taken by the transformer and trained so that we don’t have to give any input while executing.



This consist of embedded inputs and outputs**.** They convert sequence of words into numbers.They represent a three-dimensional space in which tokens with similar meanings are close to each other**,** but do not encode the relative positions of words in the sentence. After embedding, there is a position encoding, and the words approach each other based on their meaning and position.



It produces a 512 dimenstional embedding vector for each word in a sentence and passes to the next layer which is positional encoding.

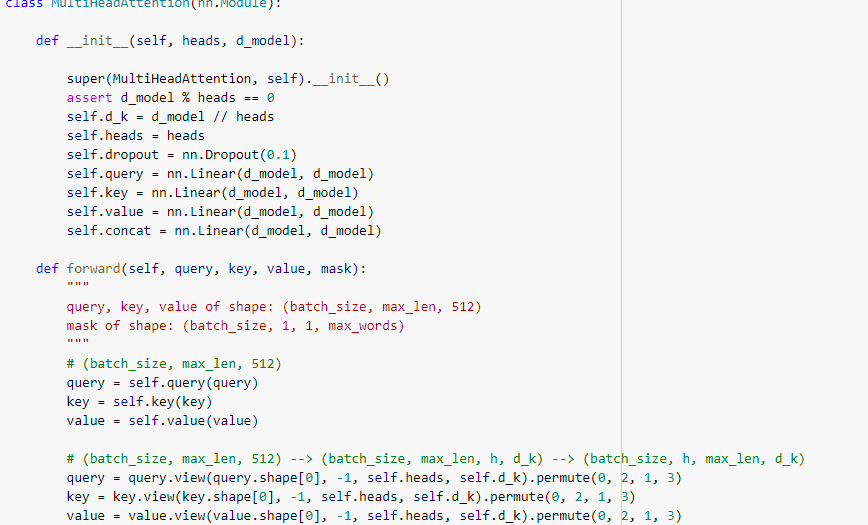
Positional encoding some information about positions before sending the embedding to the encoder.Positional encoding should satisfy the following criterias

* It should output a unique encoding for each time step
* Distance between any two time step shoul be constant across sentences with different length

In this process we assign numbers to each time step i.e first word is given as I and second is given II .. so that it would provide unique encoding for each time step.

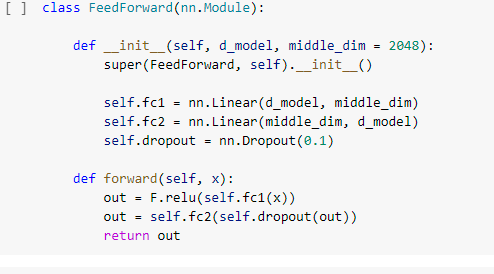
**Multihead Attention**:

with the help of multihead attention we maintain separate queries,keys and value weight matrices for each head resulting in different query,key and value matrices.



**FEEDFORWARD:**

This layer expects a single matrix(a vector for each word)

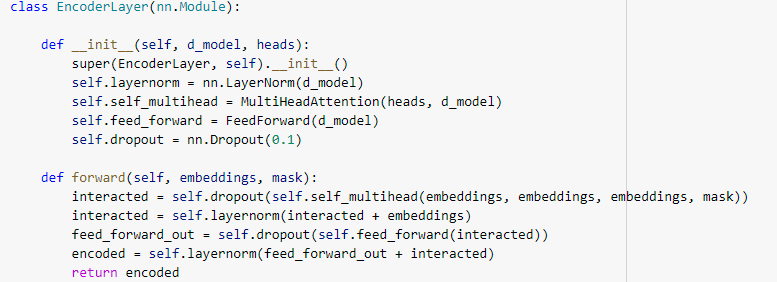


**ENCODING LAYER**

Encoder takes the sequence of input and maps it into a higher dimensional space.The output of the encoder works as input to the decoder which turns into output sequence. A basic choice for encoder and decoder of this model is a single LSTM.

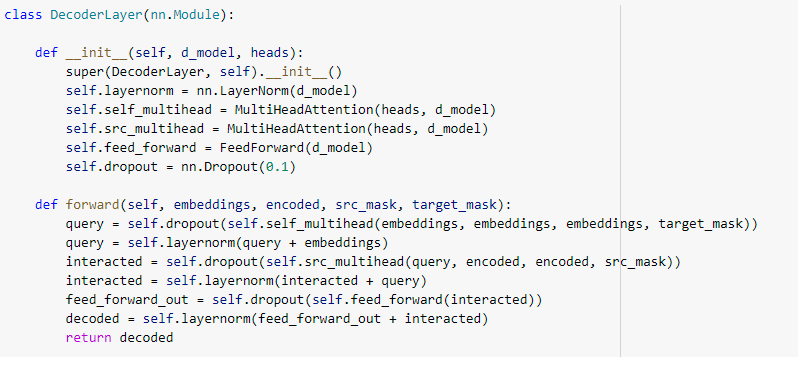
It consists of

* Input embeddings
* Positional Encoding
* N encoder layers



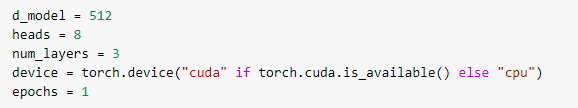
**DECODING LAYER**

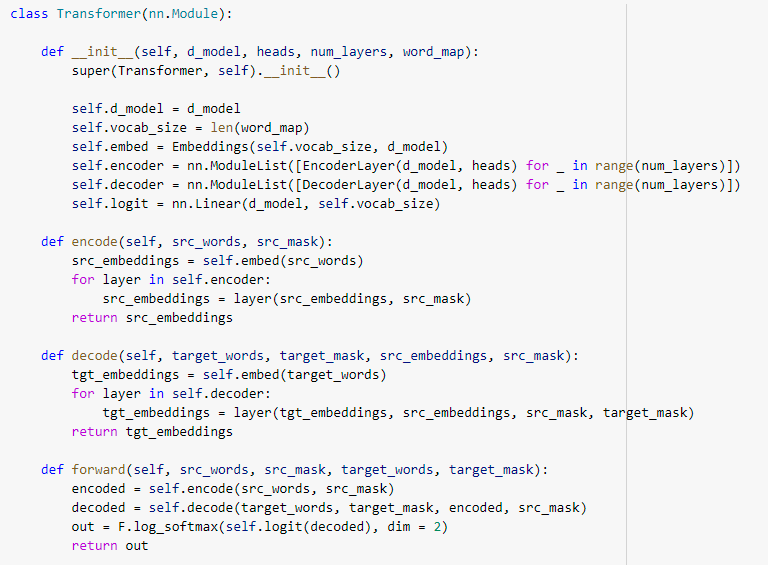
The encoder has to process the entire sentence at a time and produces output which acts as an input to the decoder since decoder has not translated anything, we simple give a small sentence as a token to start working.based on the two inputs which are produced from encoder and a token, the decoder then chooses the words.



**TRANSFORMER**

The output of the decoder is taken by the linear layer and the final output is returned and also we calculate the loss.The model is trained as follow





After this we train and evaluate the model so that when we ask a question to the transformer the chatbot replies with the respective answer.

As we don’t have GPU in our systems, we didn’t train the model enough but if we run it in a good environment we can obtain maximun accuracy.

**Conclusion:**

In this we are able to create a chatbot which can reply to the questions.As we have trained it using cornell movie dataset, for now we can only ask those related questions.

**GITHUB LINK:**

https://github.com/kavyapothula/Conversational-chatbots-using-transformers.git

**References:**

<https://arxiv.org/abs/1706.03762>

https://towardsdatascience.com/transformers-89034557de14

<https://huggingface.co/docs/transformers/index>

https://www.drift.com/learn/chatbot/ai-chatbots/