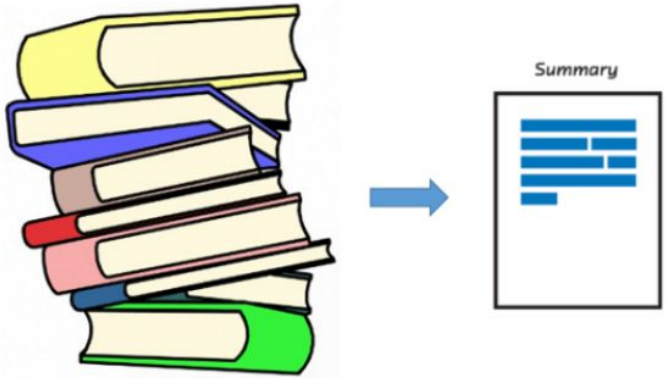
Abstractive Text Summarisation

“A Deep Neural Network Model based on NLP and LSTM to summarize text in an abstractive way by generating new sentences on its own”



CS 386 Project: Team 8

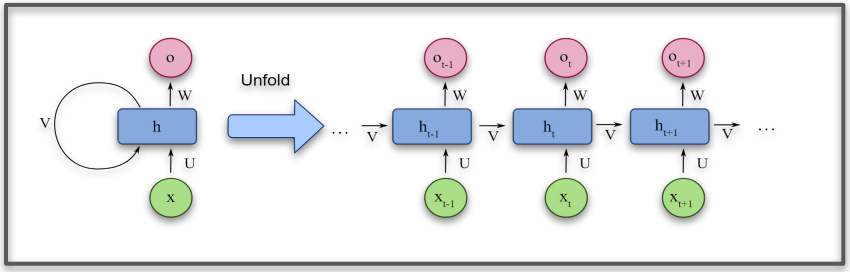
1. Shivam Jaiswal(180010026)
2. Viranch Patel(180010021)
3. Sohan Kshirsagar(180010016)

Abstract:

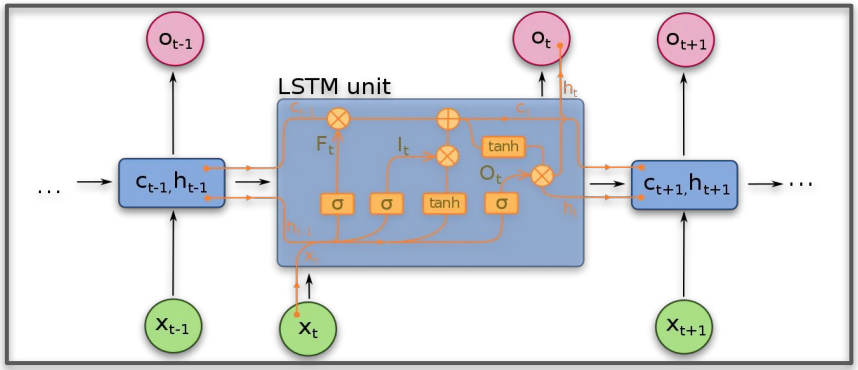
- In this project, we have implemented a standard Long Short Term Memory (LSTM) sequence-to-sequence attention model.
- This model utilises a global attention model to generate each word of the summary conditioned on the input sequence.
- We have trained this model on Amazon Food Reviews DataSet (Source).
- This model shows that neural models can encode texts in a way that preserve syntactic and semantic coherence.

Id	ProductId	UserId	ProfileName	Score	Time	Summary	Text
1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned dog food products and have found them all to be of good...

RNNs and LSTM:



- RNNs or Recurrent Neural Networks are used to process sequential data as each neuron uses its internal memory to store some information regarding the sequence.
- LSTMs or Long Short Term Memory units can store values for long and short times as values do not change in until the context is changed.
- Each LSTM Unit has the following
  - Logistic Functions
  - Input Gate
  - Forget Gate
  - Output Gate



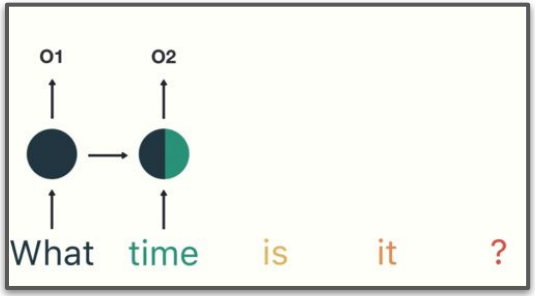
Our Solution

Pre-processing:

- Turned to lowercase, removed the stopwords, punctuations etc.
- Filtered out the texts and summaries taking the required range.
- Removed the rare words examining on the basis of their frequencies.
- Inserted Start and end tokens to the summaries so as the decoder can distinctly identify them.

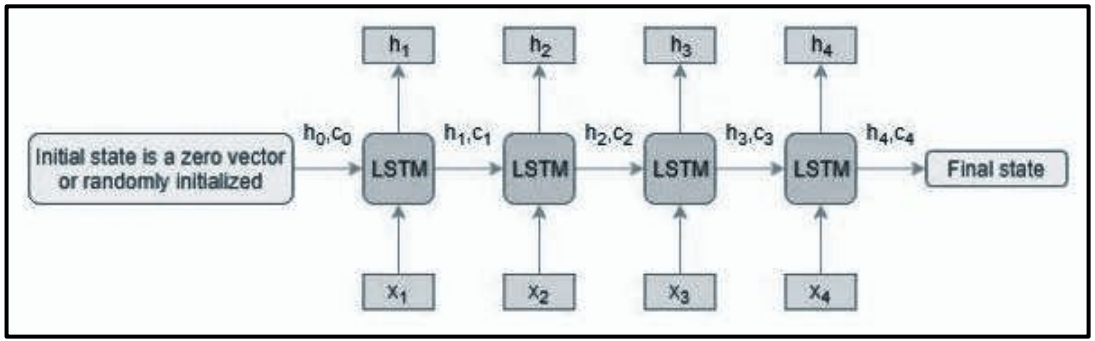
```
0 bought several vitality canned dog food products found good quality product looks like stew processed meat smells b
etter labrador finicky appreciates product better
1 product arrived labeled jumbo salted peanuts peanuts actually small sized unsalted sure error vendor intended repre
sent product jumbo
```

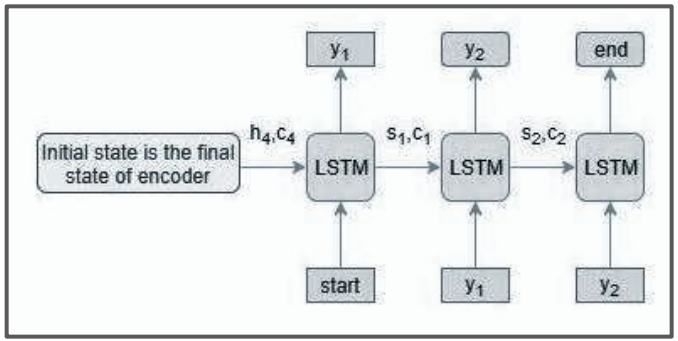
Tokenization and Embedding



- Tokenization basically assigns each word present in the data a specific code which with it can be identified for later stages as the machine cannot identify words it can just play with numbers!
- Inbuilt tokenization in keras can be used with either direct count of words or with tf-idf.
- In word embedding, we create a n❌1 vector for each of the words in the set. It is a representation of words that have similar meanings or are being used in similar contexts.
- Word Embedding matrix can be pre-filled [external matrix] or its parameters can be trained while fitting over the dataset using the inbuilt embedding layers in Keras.

Encoder, Decoder and Attention Layers





Our LSTM Model is based on following layers:

- **Encoder**: Reads a sequential input with one word at each timestep and encodes it into a fixed length vector. The hidden state (h_i) and cell state (c_i) of the last time step are used to initialize the decoder.
- **Decoder**: Decodes the fixed length vector and outputs the predicted sequence. It is trained to predict the next word from the context of the previous word. Takes input final state of encoder.
- **Attention Layer**: It aims to predict a word by looking at a few specific parts of the sequence only, rather than the entire sequence. We will be using global attention in this model.

Model Building, Inference and Model Evaluation:

1. **Model Building:** We package all the Encoder LSTM layers, embedding layers and concatenate the Decoder and Attention layer into one model.

2. **Inference:** We take the encoder layers from our defined models and give the required inputs and its output is given as input to the concatenation of attention layer and decoder.

In the **prediction phase**, we take the output of decoder and frame the sentence from it using reverse tokenization.

3. **Model Evaluation:** ROUGE evaluation is implemented as of now. It is preferentially used to compare large sentences.

Final Results:

```
Review: two love toy dino always top toy list stand aggressive chewing ordering second one
Original summary: it is a favorite
Predicted summary: my dog loves it


Review: say yuk horrible tasting even taste like coffee bitter taste br br words come
Original summary: yuk
Predicted summary: bitter


Review: mile chai wonderful right balance spiciness tried chai bit spicy one well balanced great almond milk bonus points
ch spicy side although still like peet may prefer flavor profile
Original summary: awesome chai
Predicted summary: delicious chai


Review: throw piece junk year service warranty nobody area second time breaks first time fixed made pay shipping charges ended
rvice
Original summary: a lemon do not waste your money
Predicted summary: do not buy


Review: really like cereal use water try milk super tasty agree needs bigger box go box days
Original summary: yum
Predicted summary: great breakfast cereal
```

References:

- https://www.di.ens.fr/~lelarge/dldiy/slides/lecture_8/index.html#87
- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://keras.io/api/layers/recurrent_layers/lstm/
- https://www.tensorflow.org/guide/create_op
- https://towardsdatascience.com/light-on-math-ml-attention-with-keras-dc8dbc1fad39

Contributions:

Viranch Patel: Searching of references, helped in pre-processing, embedding, model formation, model training and Inference of the model, read several research papers.

Sohan Kshirsagar: Searching of reading material, helped in implementing Keras pre-processing, embedding, model formation, model training and Inference of the model.

Shivam Jaiswal: Helped in embedding, model formation, model training and Inference of the model, Cuda implementation in our model, searched appropriate apis for project.

Thank You.