A deeper understanding of Recurrent Neural Networks and applications in Natural Language Processing

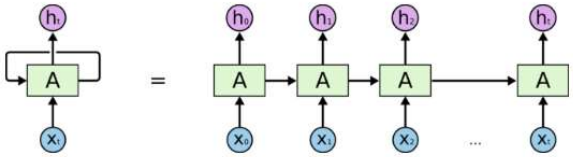
Sahith Reddy 1218622587, Siddhant Sangal, 1217670883, Supreet Palabatla 1218516039

***Abstract*— The report presented here has been written under the Artificial Neural Computation study project. The paper provides the reader with a comprehensive understanding of Recurrent Neural Networks (RNN) which hare Deep Neural Network (DNN) models used for computation f sequential data. The report deals with the specific application of Natural Language processing and how a Virtual assistant or a ChatBot can be give the ability to emulate the human behavior of conversation using complex sentences with grammatical accuracy. The paper dives deeper into one of the widely used models known as Long Short Term Memory (LSTM) models which are optimized versions and that have the ability to handle long sentences due to large memory space. The report provides the reader with understanding of how the LSTM model is structured, handles data and is able to perform the task of conversation. The report presented is based on modern research material and benchmarks from the last decade to show the progress of the community and provide the reader with intricate details about the problem approach of NLP, the data flow through the LSTM model, the computational requirements and benchmark driven comparative analysis of more State Of The Art (SOTA) models so that the reader receives a generalized overview of the subject matter.**

***Index Terms*— Recurrent, Neural, Deep Neural Network, LSTM, Natural Language Processing, SOTA.**

I. Introduction (Supreet)

The RNN has an internal memory, it is a derivative of a feed forward neural network. As the name suggests it is recurrent and it loops the same pattern or function. Here this function loops in and takes the input of the data. Every point from the data is considered as the input for this function which goes in a loop manner. The catch is that the output of the present input that has been retrieved is dependent on the past computation. The recent computation is considered, only the recent 1 computation is considered. [1] This recent output that is retrieved is taken and sent again back to the network. For the verdict to be concluded the present output and input are compared and considered. This is to show that the data procured and made is learned from the previous input. Here the RNN uses the internal memory to process the inputs it has retrieved. This is not like the usual feed forward network but a generalized and a derivative form it and it uses the memory that has it is internally. Since all the inputs are connected and related to each other here it is applicable to use in tasks such as speech recognition, or handwritten text recognition etc.



An unrolled recurrent neural network.

Here first the X(0) is taken as the input and it gives h(0) as the output and then from here the X(1) is taken as the input for the next step along with h(0). In the same manner h(1) is taken along with X(2) is considered as the input for the next one and it goes on in the similar way. This architecture keeps remembering the inputs and the outputs for getting itself trained. The current state is as follows:

$$h(t) = f(h(t-1), x(t))$$

Recurrent Neural Network has the ability to link the results and understand the input from the data, as each sample is dependent on the previous sample. RNN are also used with CNN as they can be effective in terms of understanding the pixels. Here there are a few drawbacks of RNN that is the gradient can be diminishing, the training of the RNN module is quite a difficult task. It is difficult to process the long sequences when activation functions are used.

II. Problem Description(R1) (Supreet)

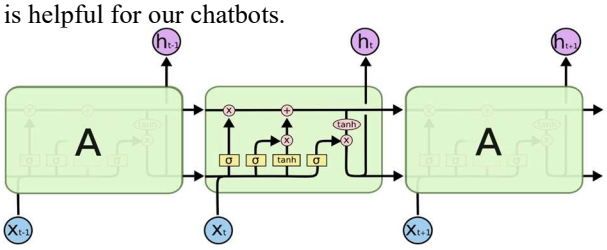
Natural Language Processing is the method to teach machines on how to understand the language spoken by humans. It works with the mixture of Artificial Intelligence and the language which has to be taught to the machine. This will help the machine to understand the language and the syntax spoken by the human training it and reply back to the human. There is no specific language that is to be trained here, the languages can be anything that the dataset has, to gain the dataset is the main task as the bigger data it is the better the training can be modelled. [2] The concept of the natural language processing is not just to communicate back but also to find the best relation or best answer to the questions asked. Such as the sequence to sequence model where the model can be trained. That is the syntax is pointed to the target, say that the input

and the output language is set and trained to the model to give the most accurate output. The dataset should be of that specific desired communicative language. For example the input is English and the output is French the input is given and the target that is the output is set to give the response back in French, this is just a simple example of how the sequence to sequence, that is the source to target pointer is worked out. There are many uses of natural language processing which are used in our day to day lives today. Google assistant, Cortana, Alexa, Siri are a few well known and famous ones made by Google, Microsoft for windows, Amazon, and Apple, respectively. These applications listen to humans or understand and interpret the text or voice and responses back to us. They are literally our assistants but virtually they help us to set our alarms, reminders when connected to smart devices they have the ability to turn them on and off, play music, make calls, send texts and messages and a lot more. All of this could be achieved because these have been trained to understand the human languages. They can also play games and speak different languages to communicate and keep the user occupied when needed. Recurrent Neural Networks is a section of ANN that is the artificial neural networks, they use their internal state memory which are derived from or in the form of a generalized form of the feedforward neural networks. This helps to understand the various inputs which are of different sequences. Recurrent Neural Networks has the ability to remember information that is has gathered in the history. This is useful for the answering of the model when the questions are given to the model as input. The problem of continuing the discussion with the chatbot is still around the corner. Every time we put a query into the chatbot, we need to specify the full information and it doesn't gather the same from the previous queries. It can be solved by using state of the art LSTM models. LSTMs are capable of remembering information and sequence of words for a long period of time. By using LSTM the chatbot can remember the information you have provided and might not require to provide to it again as long as the conversation is continuing. It helps for a faster response time and less effort from the humans.

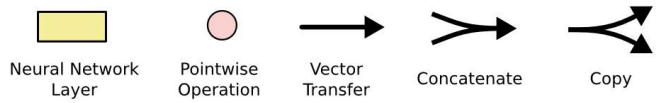
III. Solving with the LSTM model(R2) (Sahith)

A. *Design and Implementation*

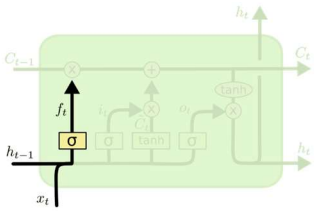
LSTM's are a special kind of RNNs which are capable in handling and learning long-term dependencies. LSTMs are designed to specifically learn and remember information throughout a long period of time. Every standard RNN have a repeating module, but the LSTM's repeating module is very different from RNN. Different from RNN, LSTM has four neural network layers, interacting in a very special way which is helpful for our chatbots.



In Fig 1 each line carries an entire vector, from the output of one node to the input of other nodes. The pink circles denote the pointwise operations like vector addition. The yellow rectangles are neural network layers. Vectors merging indicate input concatenation, while a vector splitting denotes its data being transferred and copied to different locations. The process can be clearly explained in the below Figure.

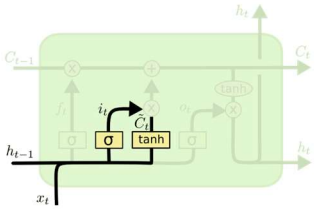


There are two kinds of information i.e., the information to remember and the information to forget. This decision is made by a sigmoid layer called the "forget gate layer". This forget gate layer is activated when the conversation with the human is stopped and the information is removed from the LSTM network.



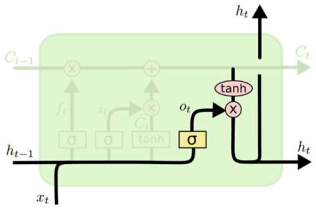
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

The lower lane decides what information must be retained. First, a sigmoid layer known as the "input gate layer" decides which values we'll update. Tanh layer decides what values are updated. This tanh layer in the LSTM of the chatbot updates the information provided in the chat if the user needs to modify any provided information. We'll concatenate these two tanh and sigmoid layer.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

In the previous step we already decided what to do in the present state 't'. We multiply the old state by f_t, then we add $i_t * C_t$. Here if we need to remember the value this function doesn't update the previous value, else the values are updated accordingly. To get the output finally, we need to get the sigmoid layer's output and send this through the tanh to normalize the values from -1 to +1. Now this tanh output is multiplied by the output of the sigmoid gate again, so that we only output the parts we decided to.



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

B. Hyperparameter selection

We use tanh in a LSTM neural network because tanh function has the ability to overcome the vanishing gradient problem. Moreover, tanh function's second derivative can sustain for a long range before going to zero. Sigmoid function is used in LSTM neural network because the sigmoid can output 0 or 1 which is very well used to define whether the information is to be remembered or to be forgot. To exclude the problem of overfitting, LSTM layer should have a Dropout layer. The final activation layer is chosen to be the softmax activation function because it allows us to interpret the outputs as probabilities. Loss function is selected in such a way that it complements the activation layer. As we have selected the softmax activation layer we are going to select cross-entropy as the loss function. These functions complement each other because the cross-entropy function cancels out the plateaus at each end of the soft-max function and therefore speeds up the learning process. We have selected the adaptive moment estimation also known as Adam optimizer as the optimizer function because of the faster learning curves it has produced in most learning applications. Adam is a combination of RMSprop and Stochastic Gradient Descent with momentum. Adam uses the estimation of first and second moments of gradient to adapt the learning rate for each weight of the neural network.

C. System requirements for implementation

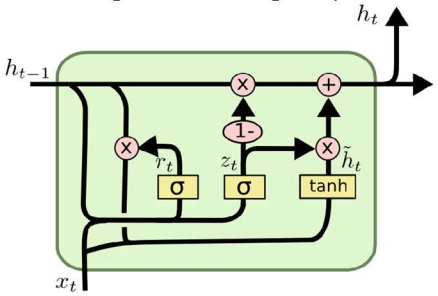
Though a CPU is sufficient to train a LSTM but a GPU is preferred for faster training because GPU cores are a streamlined version of the more complex CPU cores.. As we are going to use a large dataset to learn the words and predict the output to a question, we need to have a high-end GPU for an uninterrupted training process. There a Graphics cards which are high end such as Nvidia GTX 1080 with 8GB of VRAM and a CPU of intel's i7 which is from 7th generation or above would do the work. RAM must be above 8GB to get process flowing smoothly.

IV. Comparative Analysis (R3) (Siddhant)

One of the problems of LSTM models as it has been previously mentioned is the small amount of memory which can be overcome by using LSTM models with larger memory and hence the ability to handle long sentences[7]. The variations of RNN that we will be discussing in this section are known as Gated Recurrent Unit (GRU) and Bidirectional Encoder Representations from Transformers (BERT). Both the variations have different architectures and handle the same data with different strategies which have been discussed as following:

A. GRU

A gated recurrent unit is a modification of the LSTM architecture such that it requires much less number of parameters and has the unique characteristic of the absence of the output gate as it can be seen in the figure below[8]. This contributes to a lower cost of computation and can be considered an optimized or a lighter version of a classic RNN model. It has fewer number of parameters leading to a sparse topology and lower computational complexity.



(Gated Recurrent Unit architecture. Image source : data-blogger.com)

The GRU models holds an edge over LSTM models where the dataset is smaller and is less frequent. The GRU model is gaining popularity in the sequential modelling community and the most common applications for which it is used are Natural Language Processing as well as music and speech signal modelling. Based on various research papers, it has been found that LSTM models outperform GRU and can perform unbounded counting and learn simple languages which the GRU model does not have the ability to do [9]. The GRU network model can be used in scenarios where the speed of the model is the main focus as opposed to the accuracy. The implementation of this model can be done on hardware with lower computational capabilities and the specific version called as a minimal gated unit can also be implemented on mobile computational devices such as a raspberry pi for specific applications such as defense and exploration tasks. This is one of the classic examples of speed vs accuracy trade-offs. Newer research and optimization methods have shown that a GRU model has lower complexity and has accuracy comparable to LSTM models and hence has a big potential in the future [10]. The equations governing a GRU model are as following:

$$z_t = \sigma(W_z.[h_{t-1}, x_t]),$$

$$r_t = \sigma(W_r.[h_{t-1}, x_t]),$$

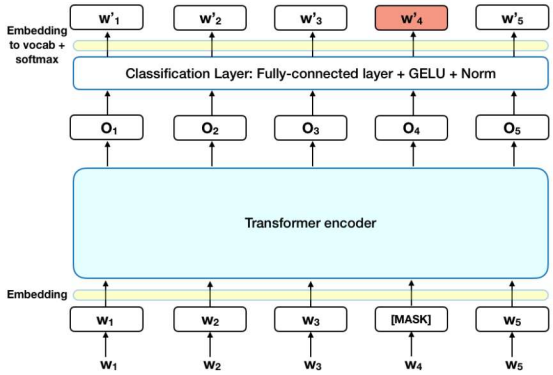
$$\tilde{h}_t = \tanh(W.[r_t * h_{t-1}, x_t]),$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t.$$

B. BERT

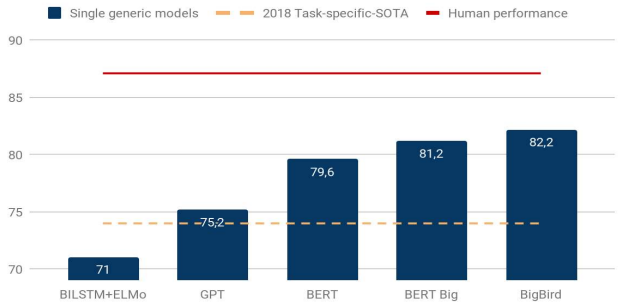
One of the most revolutionary innovations in the field of natural language processing and language modelling is known as Bidirectional Encoder Representations from Transformers which was proposed by the Google AI team in the year 2018. This method introduces transfer learning methods in the field of language modelling. As compared to the normal LSTM models, BERT has the capability of processing not only the previous token that would be received from the feedback loops but also the future tokens in the pre-processed dataset with tokenized vocabulary. Models based on BERT have the ability to perform long conversations with the user due to the ability

to predict a whole future sentence. The transformer architecture used by BERT for the encoding and decoding phases shows some state-of-the-art landmarks in performance [11]. It implements a masked LM approach in which the prediction or filling of a specific word in the sentence is based on a bidirectional traversal of the tokenized word structure. All the words in a sentence i.e. before and after a blank space which is to be predicted are used as base vectors for prediction. The approach and implementation is highly robust and generalized as it induces noise in the training phase by a random replacement of a portion of the tokenized data and eliminates the random states in the implementation phase. The model can be further optimized and fine-tuned using techniques such as average pooling, max pooling and dropout[12]. A diagrammatic representation of the architecture is as following:



(BERT architecture. Image source: towardsdatascience.com)

The BERT model places a large emphasis on the accuracy and generalization of a language model and hence has a tradeoff in terms of the computational complexity as compared to that of an LSTM model. The number of parameters range from 110 million in a BERT_base model to about 345 million in a BERT_large model with observably higher accuracy. Some very high powered hardware is required in order to implement this SOTA model since the average 1% increase in accuracy over the famous MNLI task requires 1 million steps. The convergence is extreemly slow. A real-time usage for this model could be in case of creating a much more human-like or emulated conversation with a virtual assistant such as Google assistant or Alexa wit hcloud computation and very fast internet capabilities. As of this moment, the current leader in the language model and NLP networks is the Microsoft Turing-NLG with 17 Billion parameters. The benchmarks for performance of various SOTA models has been presented in the graphical representation below.



(Performance benchmarks for SOTA NLP models. Image source: Medium.org)

Over the last two years, there have been many advancements and innovations in the natural language processing models community with growing funding and emphasis on bringing the models as close to human capabilities as possible. The computational complexity of these models has increased exponentially and cannot be implemented on minimal hardware resources and hence the models such as LSTM still hold significance and are being used in real-world applications such as the Google assistant that can conduct long conversations by holding memory but the Siri virtual assistant cannot.

V. Discussions and Conclusion (Siddhant)

The goal of a neural computations or the progress in the field of neural networks in general is to provide the computer with the ability to emulate or replicate human behavior. This report deals with one of the human characteristics of performing logical and long conversations in a grammatically accurate manner that can be understood by any human being that knows the language. One of the key aspects of the type of data required for emulating this behavior is the sequential nature of dataset or the language that we speak. Over many experiments and benchmarks, it was observed that a general RNN model is incapable of performing the complex task of long conversations due to the lack of a large memory space. This can be overcome with the state-of-the-art model that has been described in this report that is LSTM. With a long-term memory capability and special features such as the ability to get rid of unnecessary loaded data using the various gates explained above, it can be concluded to be the right model for the job. The LSTM model is widely popular in the language model community due to the low computational complexity and a respectable accuracy that can be replicated with the available minimal resources. As the computer cannot comprehend words and their significance, a method of tokenization proves to be extremely important where each significant word in the vocabulary dataset is assigned a numeric value and a predictive task of producing a layout of a combination or these tokenized datapoints is done to generate English language sentences that have to abide by the laws of grammar in order to make sense. The model that is split into the encoder and decoder part works in conjunction to convert the dataset into a real-world like implementation. We can conclude that Natural Language processing is one of the most important abilities that a computer can possess as it goes in tandem with one of the classic measures of the intelligence of a machine called as the Turing test. The growth in the NLP model accuracy community is accelerated at this point in time with the current absolute state of the art models lying the greater than 90% accuracy range and constitute multiple billion parameters which cannot be replicated by a general user but are certainly the guides to further innovation in the field as the available computational hardware becomes more and more powerful following the Moore's law. The virtual assistant that utilizes this LSTM network has a wide range o application that can incorporate multiple modules and perform various tasks by performing speech processing on the input sequential data where the output can be in the form of text, voice output or execution of the requested task such as sending an email, setting a reminder and many more.

VI. Acknowledgements

We would like to thank the instructor for the subject Dr Jennie Si for her valuable inputs and insights provided during the lectures and the doubt solving sessions which lead to this project being at the stage that it is.

VII. References and Literature(Siddhant,Supreet,Sahith)

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7283081/

[2] https://towardsdatascience.com/personality-for-your-chatbot-with-recurrent-neural-networks-2038f7f34636

[3] https://en.wikipedia.org/wiki/Recurrent_neural_network

[4] https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

[5] http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

[6] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7283081/

[7] Gers, Felix A., Jürgen Schmidhuber, and Fred Cummins. "Learning to forget: Continual prediction with LSTM." (1999): 850-855.

[8] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

[9] Weiss, Gail; Goldberg, Yoav; Yahav, Eran (2018). "On the Practical Computational Power of Finite Precision RNNs for Language Recognition"

[10] Britz, Denny; Goldie, Anna; Luong, Minh-Thang; Le, Quoc (2018). "Massive Exploration of Neural Machine Translation Architectures"

[11] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017.

[12] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[13] Shreya Ghenali, et al. " Breaking BERT down" Towardsdatascience (2019)

[14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735– 1780, 1997.

[15] Y. Miao, M. Gowayyed, and F. Metze, “EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding,” in IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), pp. 167–174, 2015.

Credit report:

Identify research topic/reference(s) (team to provide further detailed division)	study deep architecture & design (R1 & R2) (further detailed division)	compare with another deep architecture (further detailed division)	compare different development tools (further detailed division)
Siddhant-identify topic, key papers	Siddhant- key references,	Siddhant – key references, comparative analysis, benchmarking, optimization, discussions	NA
Supreet- identify topic, problem and approach description, analysis with RT usages	Supreet – key references,	Supreet – key references	NA
Sahith – identify topic , key papers	Sahith – key references, architecture, approach, tuning, requirements, analysis	Sahith- key references	NA