Proceedings of the 2nd International Conference on Communication and Electronics Systems (ICCES 2017)
IEEE Xplore Compliant - Part Number:CFP17AWO-ART, ISBN:978-1-5090-5013-0

A Neural-network based Chat Bot

[1]Milla T Mutiwokuziva, [2]Melody W Chanda, [3]Prudence Kadebu, [4]Addlight Mukwazvure, [5]Tatenda T Gotora

[1,2,4,5] *Software Engineering Department*
Harare Institute of Technology
P O Box BE 277, Belvedere, Harare Zimbabwe

[3]*ASET,*
Amity Univeristy Gurgaon
Pachgaon, Manesar, Haryana 122413, India

[1]tidymilla@gmail.com, [2]wchanda@hit.ac.zw, [3]pkadebu@hit.ac.zw , [4] amukwazvure@hit.ac.zw, [5]tgotora@hit.ac.zw

Abstract- In this paper, we explore the avenues of teaching computers to process natural language text by developing a chat bot. We take an experiential approach from a beginner level of understanding, in trying to appreciate the processes, techniques, the power and possibilities of natural language processing using recurrent neural networks (RNN). To achieve this, we kick started our experiment by implementing sequence to sequence long short-term memory cell neural network (LSTM) in conjunction with Google word2vec. Our results show the relationship between the number of training times and the quality of language model used for training our model bot affect the quality of its prediction output. Furthermore, they demonstrate reasoning and generative capabilities or RNN based chat bot

1. INTRODUCTION

Let's imagine for a minute world where instead of a human being at a customer support centre, a chat bot helps us fix our router and the internet starts working again. Such an invention would be of great convenience in this ever connected world where we cannot afford to wait several hours per year to fix our internet connection, it is the internet. But is it possible? Simple answer is yes. Today's applications and technologies like Apple's Siri, Microsoft Cortana, are at the forefront of highly personalised virtual assistants. They do not do what we have imagined but they are, by no reasonable doubt, a stone throw away from being able to do so. Now where do we begin?
The most obvious solution that leads us one step closer to living in our imaginary world is knowing that the chat bot must be able to understand messages we present it and how to respond appropriately. But computers are dumb. For starters, they use numbers raised to the powers of two, which is binary and that is all they know, whilst humans normally use decimal numbers [1], expressed as powers of ten, humans can read, write, and are intelligent. How do we make them understand our natural language when all they know is 0 and 1? Luckily for us there is a field of computer science called Natural Language Processing (NLP) and linguistics that comes to our rescue. As the name suggests, NLP is a form of artificial intelligence that helps machines "read" text by simulating the human skill to comprehend language. Given the benefit of time, driven by cognitive and semantic technologies, natural language processing will make great strides in human-like understanding of speech and text, thereby enabling computers to understand what human language input is meant to communicate.
A chat bot is a computer program capable of holding conversations in a single or multiple human languages as if it were human. Applications can range from customer assistance, translation, home automation to name just a few. This paper explores the technologies behind such innovation, implements a simple model and experiments with the model.
Recent research has demonstrated that deep neural networks can be trained to do more than classification but mapping of complex functions to complex functions. An example of this mapping is the sequence to sequence mapping done in language translation. This same mapping can be applied to conversational agents, i.e. to map input to responses by using probabilistic computation of occurrence of tokens in sequences that belong to a finite set of formally defined model of utility such as a language model. We develop our own sequence to sequence model with the intention of experimenting with this technology in hope of understanding the underlying functionality, concepts and capabilities of deep neural networks.
We trained our model on a small conversational dataset to kick start a series of experiments with the model.

2. LITERATURE REVIEW

This research is inspired by the investigation initiated by a group of researchers [2] who treated generation of conversational dialogue as a statistical machine translation (SMT) problem. Previously dialogue systems were based on hand-coded rules, typically either building statistical models on top of heuristic rules or templates or learning generation rules from a minimal set of authored rules or labels[3], [4]. The SMT is data driven and it learns to converse from human to human corpora. [1] SMT researched on modelling conversation from micro blogging sites. In their research they viewed response generation as a translation problem where a post needed to be translated into a response. However, it was discovered that response generation is considerably more difficult than translating from one language to another due to lack of a plausible response and lack of phrase alignment between each post and response [5]. [5] Improved upon [2] by re-scoring the output of a phrasal SMT-based conversation system with a SEQ2SEQ model that incorporates prior context. They did this using an extension of (Cho et al 2014) Hierarchical Recurrent Encoder Decoder Architecture (HRED). Hidden state LSTMs that implement GRUs were the ingredient that improved on [5] STM. Both the encoder and decoder make use of the hidden state LSTM/GRUs. However in their paper they were using triples not full length dialogue conversations which they mentioned in their future work.
One of the biggest problems was that conversational NLP systems could not keep the context of a conversation but was only able to respond from input to input. Hochreitern et al. wrote a paper on long short memory (LSTM) retention in deep nets [6]. At the time of publication of

paper, they had not found practical solutions for the LSTM but now it is being used by conversational NLP [2], [5] systems to keep the context of a conversation as it uses the LSTM to keep previous responses and inputs of text into a deep net. [6] Training proposes techniques of improving training in deep nets [7], [8], [9] which overcome the vanishing gradient or exploding gradient in Neural nets by using a stack of restricted Boltzman machine. The vanishing gradient would make the layers closer to the output learn faster than ones closer the input. This would end up with a network that has the output layers trained whilst the input layers were not trained. Tsung-Hsien Wen at al. proposed a paper [8] on NLG using LSTM for spoken dialog system (SDS) which help in the structuring of Chatbot. Finally, Mikolov et al. at google produced a paper that describes the efficient use of word vectors for large data sets which is currently being used by google for its Deep mind machine. Our Chat Bot will use Word2vec which is based on the publication of Mikolov et al. to convert words in its language into vectors representation.

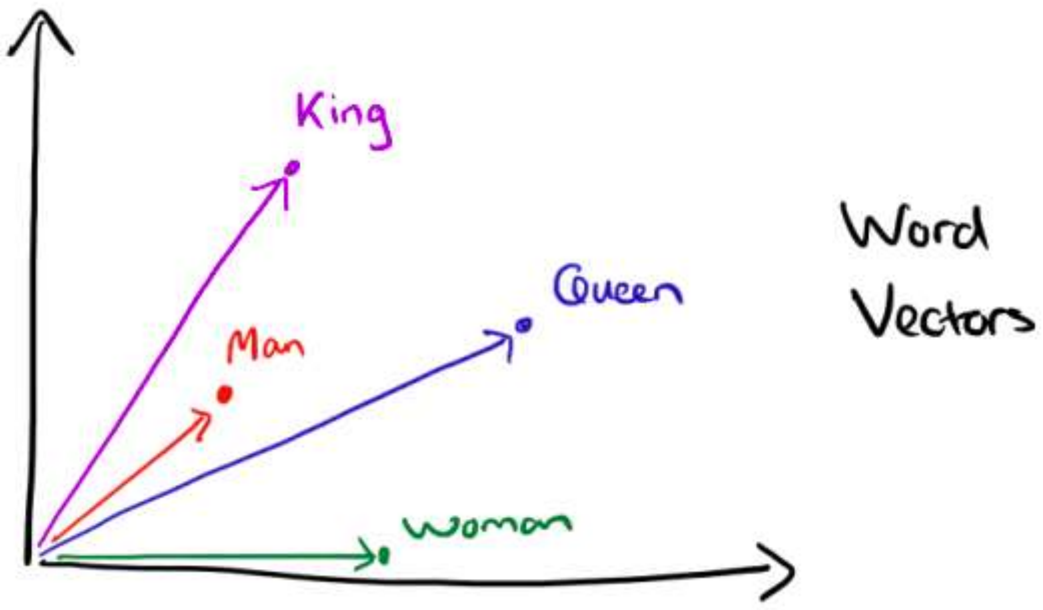


Figure 1: 2-D illustration of a continuous word representation model [10]

Chat bot models come packaged in different shapes and sizes. To begin with, there are two types of Chat bots i.e. [5] *Retrieval-based and Generative*.

The *Retrieval based* bots come with a set of written responses to minimize grammatical errors, improve coherence, and avoid a situation where the system can be hacked to respond in less appropriate ways. *Retrieval based* chat bots best suit closed domain systems.

Closed domain chat bot systems are built to specifically solve simple recurring problems for instance an elevator voice assistant. The draw backs with closed domain chat bots is that the set of data they come with does not come with responses for all possible scenarios.

Generative model conversational agents are more intelligent but more difficult to implement than retrieval based chat bots. With generative [5] chat bots, responses are generated by a program without any active human assistance. Their advantage over retrieval based is that they can easily be adapted or trained to be used in diverse domains.

Open domain systems include popular virtual assistance services like *Siri* and *Cortana* which are not constrained to a particular type of a conversation domain. Apple users can ask *Siri* anything from time of day to business advice and still get reasonable human like responses.

3. METHODOLOGY

Here we explain the breakdown of our solution into simple components that can be developed and built separately.

A. Word embeddings

Converting words from natural language text into a format that computers can efficiently work with is a challenging problem. One way of doing so is to use one hot vectors whereby each individual word in a language is represented by a vector of (n * n) where n is the size of the vocabulary. This clearly poses a huge challenge as the vocabulary expands. For each vocabulary update the whole set of vectors would have to be re-developed. Also the size of the vectors become unmanageable [5], [11] say for a million words in a vocabulary, we would have a (1,000,000 * 1,000,000) vectors to represent the whole vocabulary space.

A different method called skip-gram [5] model is a more efficient way of performing word embedding. It only focuses on feature relationships between words. If there are 300 features, every word in the vocabulary would be represented by a 300 dimensional vector no matter how big the vocabulary is. Also, skip-gram models have the ability to utilise transfer learning, i.e when we need to add new words to the vocabulary all we need to do is to train the old vocabulary with the new words. To do this, we use the tool developed by Google called Word2vec. We then downloaded a pre-trained APN news language model to use in converting our text into vectors. Word2vec is very efficient when storing relationships of words in a vocabulary.

B. LSTM

To process word vectors output from word2vec, we use LSTM. LSTM [12], [13], [14] which are a form of RNN that learn long term dependencies of sequence inputs, thus they remember information for long periods of time. They act as giant probability calculators for predicting the likelihood of a word to show up in a sentence given sequence of previous input words.

(1)

$$P(w_t|\text{context})\,\forall t \in V$$

Where

P is the probability function,
wt is the preceding word and context is the set or previous input words and,
V is the set of all words in the given vocabulary for instance

context = (Mary, is, walking)

w_t = (home)

An LSTM [12], [13], [14] has four layers of interacting networks as follows:

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

(2)

f, is the **forget gate**. It is used to filter the amount of past information the LSTM should keep such as when the subject of interest changes during a conversation, the networks forget about the previous subject. For instance, if the next sentence in our conversation says: ***"John is waiting for Mary",*** the primary subject switches form Mary to John and the forget gate is responsible for doing that.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

(3)

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

(4)

i, the **input gate** and ***Ć*** the **candidate state,** are responsible for updating the state of the neural network.

Input gate, decides which information should be input into the network. For instance stop words like "a, is, of" are not really considered to influence the meaning of a statements thus the input gate filter out the influence of such input.

Candidate state generates a vector of potential candidate. Continuing with our example, when we are forgetting the gender of Mary getting that of John, the input gate and the candidate state outputs get multiplied to update the state of the current subject.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

(3)

C_t, is the **current state**. We get it by multiplying the previous time step state with the value of the forget gate and addition of the update state as explained above. The state runs through all time steps and is responsible for passing down context information such as subject, gender and so forth.

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

(5)

$$h_t = o_t * \tanh\left(C_t\right)$$

(6)

We finally calculate the ***o_t*** output of the cell, which is a weighted product of influence of the current input and previous hidden state called the **cell state**. We filter the output of the cell state by multiplying it with the current state of the system in order to produce a context based output based on the current and previous inputs.

4. MODEL

We developed a sequence-to-sequence [4], [14], [15], [16], [17] LSTM model which reads inputs one token at a time and outputs a prediction sequence one token at a time. Each token is a word vector of 300 dimensions. We trained the model by back propagation using ADAM, minimizing the cosine distance between the predicted token and the target token. We used a small dataset with only 68 words and 97 lines of conversation and ran 10 000 epochs. We used the APN news pre-trained language for converting words into vectors. Our model is simple but yet effective and some researchers haves used the same model for translation and question answering with minimal modifications. Since the length of sentences differ, we clipped or padded all the sentence to 15 tokens. The draw backs to our model is that it is not best suited to hold human like conversations due to its simplicity. It does not carry along context, attention which are used in long term multi-turn human dialogues. The best the model can do is to predict the next step in a sequence.

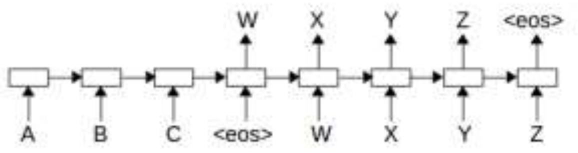


Figure 2: sequence-to-sequence LSTM Model

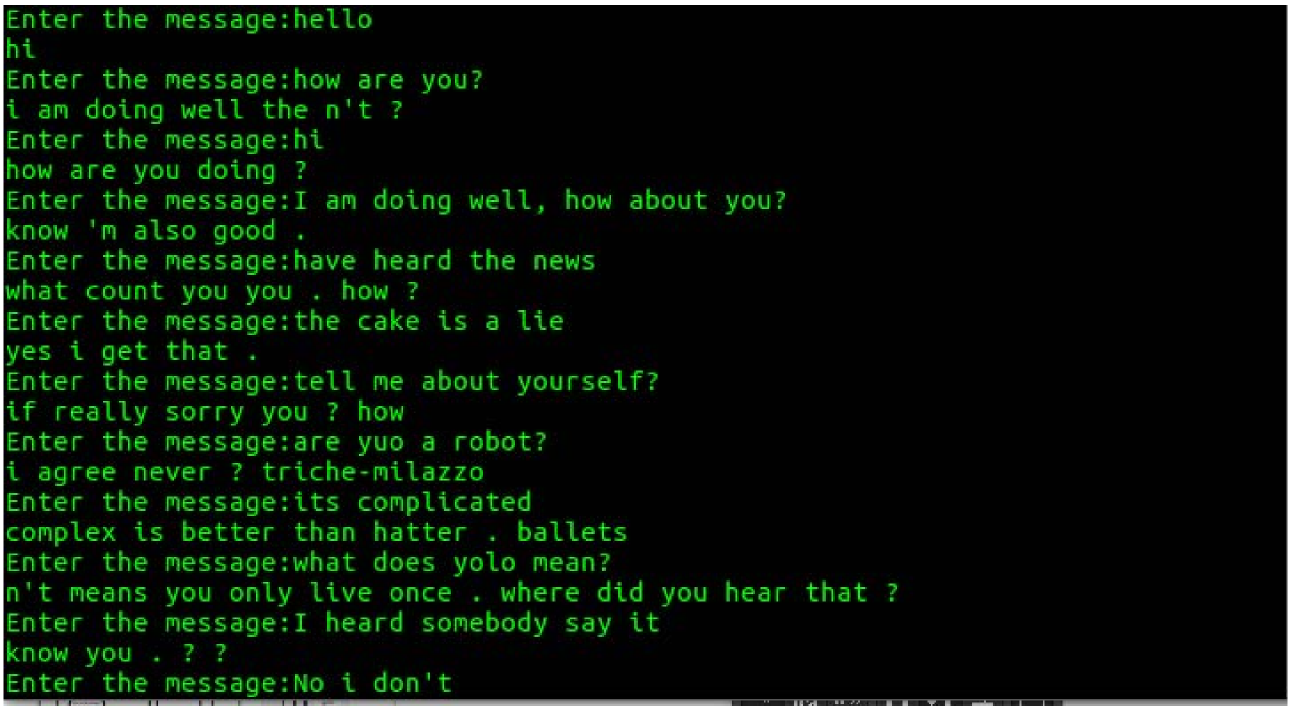
5. RESULTS

We only managed to train the bot up to 10000 epochs on a really small conversational dataset of 68 unique words. As can be observed from the table 1 below, accuracy improves with more training epochs. Of more interest is a case, given this bot was only trained on a vocabulary of 68 words, whereby it managed to spew a grammatically and syntactically correct sentence which does not exist in the dataset that was presented to it as shown in Fig 3 below.

Table 1: Training Results

Number of training epochs	**Input**	**Expected output**	**Actual Output**
5	Hello	Hi	
50	Hello	Hi	silked poh-leet
500	Hello	Hi	know n't n't
1000	Hello	Hi	n't
2000	Hello	Hi	n't
5000	Hello	Hi	hi ballets
5000	Hi	how are you doing?	how are you doing ?
5000	can I help you with anything?	yes, I have a question	tell me . ? ?
7000	Hello	Hi	hi ballets
7000	can I help you with anything?	yes, I have a question.	know me . ? ?
10000	Hello	Hi	Hi
10000	Hi	how are you doing?	how are you doing ?
10000	can I help you with anything?	yes, I have a question	tell me . ? ?
10000	who are you?	who? Who is but a form following the function of what	what ? who is but a form following the function of what
10000	so what are you then?	A man in a mask	blue ballets so n't
10000	What is your favourite book?	I can't read	n't do n't read .

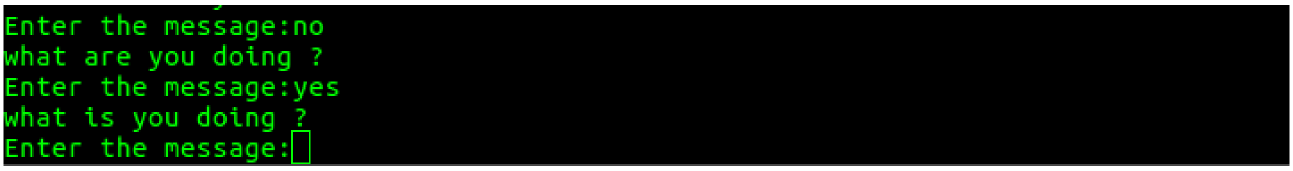
Below is a screen shot of a conversation with the bot trained with 10000 epochs . The statement ***"yes i get that ."*** is not in the conversation dataset that was presented to the network and yet it is an appropriate response to the preceding statement ***"the cake is a lie".***



```
Enter the message:hello
hi
Enter the message:how are you?
i am doing well the n't ?
Enter the message:hi
how are you doing ?
Enter the message:I am doing well, how about you?
know 'm also good .
Enter the message:have heard the news
what count you you . how ?
Enter the message:the cake is a lie
yes i get that .
Enter the message:tell me about yourself?
if really sorry you ? how
Enter the message:are yuo a robot?
i agree never ? triche-milazzo
Enter the message:its complicated
complex is better than hatter . ballets
Enter the message:what does yolo mean?
n't means you only live once . where did you hear that ?
Enter the message:I heard somebody say it
know you . ? ?
Enter the message:No i don't
```

Figure 3: Conversational Results after Training

It keeps getting interesting. When we type in ***"no"*** and ***"yes"***, it responds with ***"what are you doing?"***, and ***"what is you doing ?"*** respectively. These statements were never presented to the bot in the training set. The generated sentences are even punctuated which is quite interesting. The bot knows that how to generate and punctuate questions.



```
Enter the message:no
what are you doing ?
Enter the message:yes
what is you doing ?
Enter the message:
```

Figure 4: Sentence Generation and Punctuation

6. Challenges

Several challenges were faced in the development of our chat bot, including the following:-

i. The biggest challenge is that there aren't much available introductory resources for new learners to deep learning

ii. The little available resources are complex and skip the basics to building a full understanding of concepts

iii. There are very few code examples that do a step by step introductory approach to concepts

iv. A more technical challenge we faced was the unavailability of computational power. We had prepared a training dataset from the Cornel Movie dialogue corpus, with vocabulary size of 300 000 words and 300 000 conversation exchanges. However to train such a model required high end multiple GPU's which we could not get access of publishing the paper.

7. Conclusion/Future Work

Our experimental research was a success. This simple but powerful bot is a proof of concept of even bigger possibilities thus we would like to continue working on the implementation to fix the code and improve the performance of the model. There is still much to be understood about LSTM's both in theory and in practice. Learning through experimentation can really make difference between success and failure there however in our case, all went well. In our future, we plan to train our bot on bigger datasets. So far we already have the Cornel Movie Dialogue Corpus which has more than 300 000 unique words in its vocabulary. Also, we plan to include voice recognition and voice synthesis to as input and outputs to the system.

References

[1] K. Marcus, *Introduction to linguistics*. Los Angeles: University of California, 2007.

[2] A. Ritter, C. Cherry, and W. B. Dolan, "Data-Driven Response Generation in Social Media," *Proc. Conf. Empir. Methods Nat. Lang. Process.*, pp. 583–593, 2011.

[3] A. H. Oh and A. I. Rudnicky, "Stochastic Language Generation for Spoken Dialogue Systems," *Proc. 2000 ANLP/NAACL Work. Conversational Syst.*, vol. 3, pp. 27–31, 2000.

[4] J. Li, M. Galley, C. Brockett, G. P. Spithourakis, J. Gao, and B. Dolan, "A Persona-Based Neural Conversation Model," *CoRR*, 2016.

[5] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models," in *AAAI*, 2016.

[6] M. Boden, "A Guide to Recurrent Neural Networks and Backpropagation," *Dallas Proj.*, no. 2, pp. 1–10, 2002.

[7] D. Britz, "Deep Learning for Chatbots, Part 1 – Introduction," 2016. [Online]. Available: http://www.wildml.com.

[8] T.-H. Wen, M. Gasic, N. Mrksic, P.-H. Su, D. Vandyke, and S. Young, "Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[9] CWoo, "Context-Sensitive Language," 2006. [Online]. Available: http://www.planetmath.org.

[10] A. Coyler, "The Amazing Power of Word Vectors," 2016. [Online]. Available: https://blog.acolyer.org.

[11] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "From Word Embeddings to Document Distances," *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37, pp. 957–966, 2015.

[12] G. Klaus, S. Rupesh K, K. Jan, S. Bas R, and S. Jürgen, "LSTM: A Search Space Odyssey," *IEEE Trans. Neural Networks Learn. Syst.*, vol. PP, no. 99, pp. 1–11, 2016.

[13] Colah, "Understanding LSTM Networks," 2015. [Online]. Available: http://colah.github.io.

[14] I. Sutskever, O. Vinyals, and Q. V Le, "Sequence to Sequence Learning with Neural Networks," in *Advances in neural information processing systems.*, 2014, pp. 3104–3112.

[15] Suriyadeepan Ramamoorthy, "Chatbots with Seq2Seq," 2016. [Online]. Available: http://suriyadeepan.github.io.

[16] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan, "A Diversity-Promoting Objective Function for Neural Conversation Models," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 110–119.

[17] O. Vinyals and Q. Le, "A Neural Conversational Model," *arXiv Prepr. arXiv*, vol. 37, 2015.