Personified Autoresponder

Arun Mahendra
Stanford University
arunmahe@stanford.edu

Abstract

Question answering is an exciting area of research in natural language processing and recently deep learning models based on Recurrent Neural Networks (RNN) involving memory and attention mechanisms have shown some very promising results. In this project I attempt to develop a personified auto responder that can generate responses to email messages by learning from a person's prior emails. By training on an individual's prior emails, the hope is that models can learn some individual-specific characteristics and use that in generating responses. I applied Seq2Seq [Cho+14], [SVL14] model with Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU) to learn and generate sequences. I evaluated their performance and present the results.

1 Introduction

Recently there has been a lot of interest in developing conversational models based on Deep Learning Neural Networks (DNN). One useful and practical application of conversational models (or question answering in general) is a personified email response generator. Such a response generator would learn from a person's prior email conversations with other people and generate responses when presented with a new email message. This application, if able to generate reasonable responses, can benefit anyone who uses email and especially people who receive large volumes of emails everyday. The goal is not to automatically send an email like the standard out-of-office autoresponder but rather to automatically generate a response that the user can review and edit before it is sent. In the most complete and optimal system, the application would be able to generate responses that require no editing at all. Although, currently that goal is very challenging to achieve, in this research project I hope to make some progress in the direction.

2 Problem statement

Generating responses to email messages can be posed as a general problem of question answering. Generating longer responses with high quality is challenging. My approach to question answering problem is to build a Seq2Seq model [Cho+14], [SVL14]. There has been some interesting results in question answering with short sentences but question answering on longer sentences using simpler models based on seq2seq has been challenging as show by Vinyals et al. [VL15]. Regular RNNs suffer from vanishing gradients which causes it to perform poorly on longer sequences. In order to overcome that issue I base my model on LSTM and GRU.

The goal of my project is to apply existing deep learning neural network architectures and evaluate their performance in generating responses to email messages that reflects the persona of the actual email user.

3 Related work

Based on my research, I have found no other prior work dedicated to developing a personified email response generator using DNN. There has been several prior research work in the general problem of question answering such as: [VL15], [SLL15], [Li+16], [Ser+16].

A conversational model proposed by Vinyals et al. [VL15] uses an encoder-decoder approach to learn from movie subtitle dataset. This model shows interesting results achieved by training a seq2seq model on large movie subtitle dataset. But the model does not perform well on conversations involving longer sentences. This model also does not account for persona of a particular speaker as the model is trained on data irrespective of the author of the conversations.

Unlike the neural conversational model proposed by Vinyals, the persona based conversational model proposed by Li et al captures author specific characteristics. Li et al. proposes two models that learns persona's: a Speaker model that models a speaker without taking into account the other person in the conversation and a Speaker-Addressee Model that also learns the distinguishing characterise that a speaker employs while addressing certain people [Li+16]. One of the datasets the models were trained on was Twitter conversations. The advantage of using Tweets is the large amount of data that is available. But one should note that tweets are very short sentences and often contain abbreviations and intentional truncation of messages by the author in order to meet the character limits of a tweet. It is reasonable to think that this form of character limit reduces expressiveness in tweets. The reduced expressiveness could result in less persona characteristics being present in tweets. The other dataset that was used was television series transcripts [Li+16], although a rich dataset, since the dialogs in TV shows are written for the sole purpose of dramatization, it could be missing subtle characteristics of persona and style that naturally occurs during spoken (or written) conversations.

4 Dataset

In this project I used the Enron email dataset [Coh15]. It is the only publicly available large collection of emails from many users. For this research, I will choose few candidate email users that have the highest number of email communication in the dataset. Here are some statistics on the data, in the interest of privacy placeholders A,B,C... etc. will be used in place of actual user names.

User	Total Email Count	Emails with Conversations
User A	6272	3282
User B	5100	2654
User C	4797	1543
User D	4437	1693
User E	3686	1458

The email data is in its raw format, a custom email parser was implemented to parse the email body and extract conversations between people. Additionally, manual human hand picking was used to pick out and organize conversations from the body of the email. Extracting data is a challenge since the email messages are of varying types such as broadcast group emails, forwarded mail, emails with attachments, one way messages and spam. In order to objectively categorize email content into question and answers, here are some definitions:

Answer: An answer is a message sent by the candidate user u at time t in response to a message sent by a random user x at time $t-1$.

Question: The message sent by a random user x at time t becomes the question. Note that "question" and "answer" according to these definitions does not necessarily have be question and answer in the true sense of the words.

An email thread with multiple conversations involving candidate user u and others can we written as:

$$E_{even} = \langle Q_{t-n}^{(x)}, A_{t-n-1}^{(u)}, Q_{t-n-2}^{(x)}, A_{t-n-3}^{(x)}, Q_{t-n-4}^{(u)} ... A_{t}^{(u)} \rangle \tag{1}$$

$$|E| = n, \qquad n \in 2\mathbb{Z}^+$$

Odd combinations of the sequence mentioned above is also possible. The emails are preprocessed into $(Q^{(x)}, A^{(u)})$ pairs for selected candidate users.

4.1 Interesting visualization

A random sample of 50 email messages from random user X and corresponding responses from two candidate users $U0$ and $U1$ are chosen. Each message was treated as a long string but with all the punctuations in place. For each set of tokens (resulting from splitting a string on space and few special characters) it's corresponding 300-dimension word vectors were averaged to form a single vector $I \in \mathbb{R}^{300}$ each for a message and its response. A set of average vectors of n samples is $L \in \mathbb{R}^{n \times 300}$. The following plot shows the first two principal components of set L. It is interesting to note that even after averaging and taking only the first principal components, we can see a separation between email messages and responses.

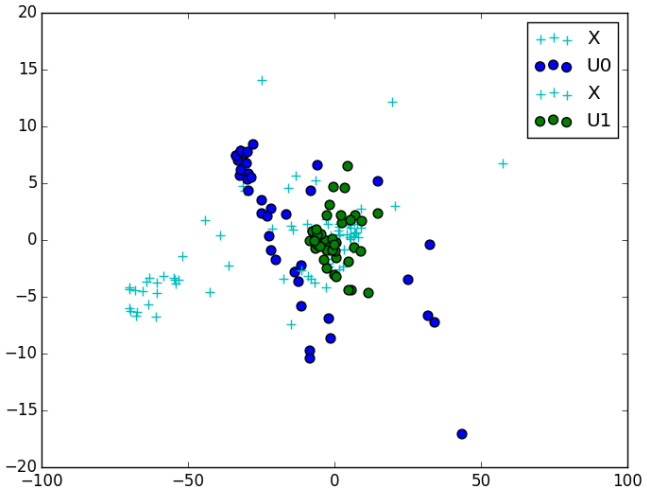

Figure 1: Separation between email messages and responses in vector space.

The 300-dimension word vectors were pre-trained on Google news dataset. This dataset contains 3 million words and phrases, it was created by Mikolov et al. [Mik+13]. In order to store and efficiently look up word vectors from this large dataset, these word vectors are stored in LevelDB. LevelDB is a fast key value persistent library written in C++. [San].

5 Approach and Models

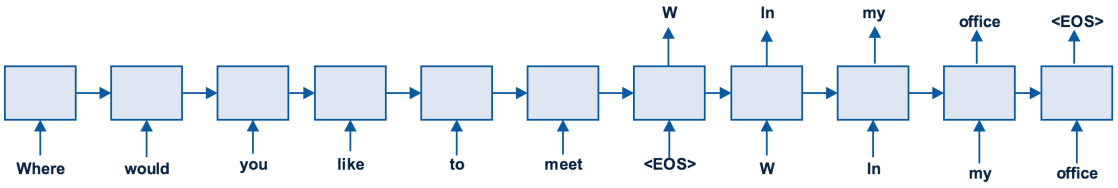

Figure 2: Representation of Seq2Seq model with a question and answer

5.1 Model: Seq2Seq

Seq2seq model was introduced by Cho et al. in which two RNNs were used, one to encode varying length sequences into a fixed size embedding matrix and the other RNN is conditioned on the embedded weight from the first RNN to generate output sequences.[Cho+14]

This approach is important for this project as it allows the model to be trained on varying length sequences. But RNNs suffer from vanishing gradient issue which means that Seq2Seq model based on simple RNNs will perform poorly on longer sequences. Sutskever et al. introduced multilayer LSTM based Seq2Seq model and have shown some promising results in [SVL14].

5.2 Long Short-Term Memory

LSTM network, first introduced by Hochreiter et al. in [HS97] is a complex recurrent neural network that contains memory units. This architecture solves the problem of vanishing gradients that are present in the basic RNN. LSTM contains four important components:

Input gate $i^{(t)} = \sigma\ \left(W_i x^{(t)} + U_i h^{(t-1)}\right)$

Forget gate $f^{(t)} = \sigma\ \left(W_f x^{(t)} + U_f h^{(t-1)}\right)$

Output gate $o^{(t)} = \sigma\ \left(W_o x^{(t)} + U_o h^{(t-1)}\right)$

Memory cell $\tilde{c}^{(t)} = tanh\left(W_c x^{(t)} + U_c h^{(t-1)}\right)$

The input gate decides which value should or should not go into the memory cell. The forget layer decides whether or not to keep the past information. The output gate acts as a filter for the cell's hidden state.

5.3 Gated Recurrent Unit

GRU was intorduced by Cho et al. in [Cho+14]. Unlike basic RNN where activation is computed directly based on the input from the $t-1$ step, GRU incorporates information from time steps back in the sequence. Two main components of a GRU unit are the Update and Reset gates, formally they are represented by the following equations:

Update gate: $z^{(t)} = \sigma\ \left(W^{(z)} x^{(t)} + U^{(z)} h^{(t-1)}\right)$

Reset gate: $r^{(t)} = \sigma\ \left(W^{(r)} x^{(t)} + U^{(r)} h^{(t-1)}\right)$

Notice that the weights used to compute and set the update and reset gates are different. If the reset gate is 1 then the GRU unit keeps the previous state. Even if the reset gate is set, the amount of information from the previous state that must be carried over is based on the output of the update gate. [Cho+14]

5.4 Projection and Loss

I added a projection layer to my model to transform the output probabilities from the decoder RNN to a distribution over vocabulary. For calculating loss I used weighted cross entropy and Adam optimizer for batch gradient decent optimization.

I used perplexity as a metric for quantitative evaluation, it can be written in terms of cross entropy loss function as :

$$CE(y^{(t)}, \hat{y}^{(t)}) = -\sum_{j=1}^{|V|} {y_j}^{(t)}\ \log \hat{y}_j^{(t)}$$

$$PP(y^{(t)}, \hat{y}^{(t)}) = 2^{CE(y^{(t)}, \hat{y}^{(t)})}$$

The derivation of the above equation is not presented here, however it is straight forward to derive if you consider the fact that $y^{(t)}$ is a one-hot vector.

6 Experiments & Results

I implemented and trained two variations of Seq2Seq model: LSTM and GRU based. I tried various combination of hyper parameters and deeper layers, the results presented below are from the best performing models.

6.1 Seq2Seq with LSTM units

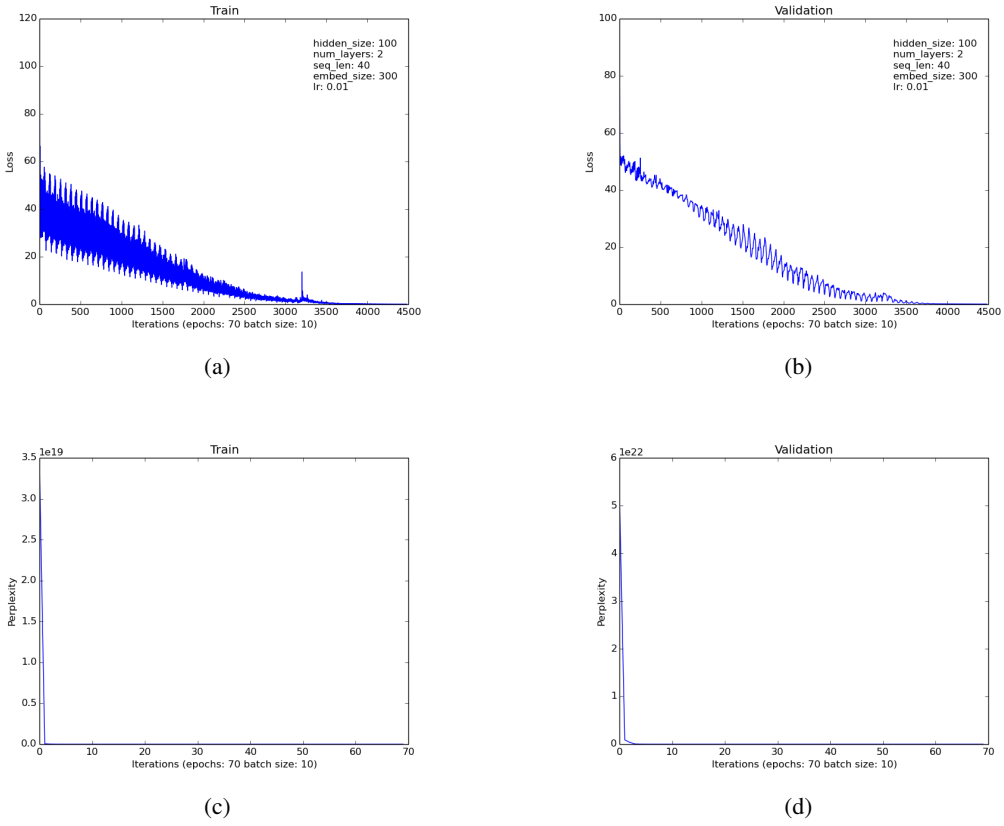

Figure 3: Seq2Seq with LSTM units

The models were trained on question answer pairs extracted from the email dataset. As described in the previous section, a questions were asked by a random user X and the answers were given by the the candidate user U. Question and answers where treated and along string and tokenized on space and some special characters. For each word in the question a 300-dimension word embedding was looked up and a sequence of word vectors were created. A batch of k such lists of word vectors were sliced to form n time steps. I used a fixed sequence length for sentences, shorter sentences were padded with a chosen padding token. The longer sentences were truncated, if they were longer than a threshold it was discarded. Similarly, a batch of corresponding answer sequences were created. The questions were trained in reverse as it had yielded good results in [SVL14].

The model was implemented in Tensorflow using the built in libraries. Several experiments were run to tune the hyperparameters. I used a forget gate bias of 1, as it yield good results in [JZS15]. I experimented by initializing the model with different values: zeros, small positive and negative numbers and ones, ones worked the best. LSTM based Seq2Seq model over-fitted when trained without dropout. Without dropout, the loss converged much sooner on the training dataset but the validation perplexity was quite high even after 35 epochs. The models were trained with high learning rate first and when the loss plateaued, the training was stopped and learning rate was reduce by half until the desired loss was achieved.

6.2 Seq2Seq with GRU units

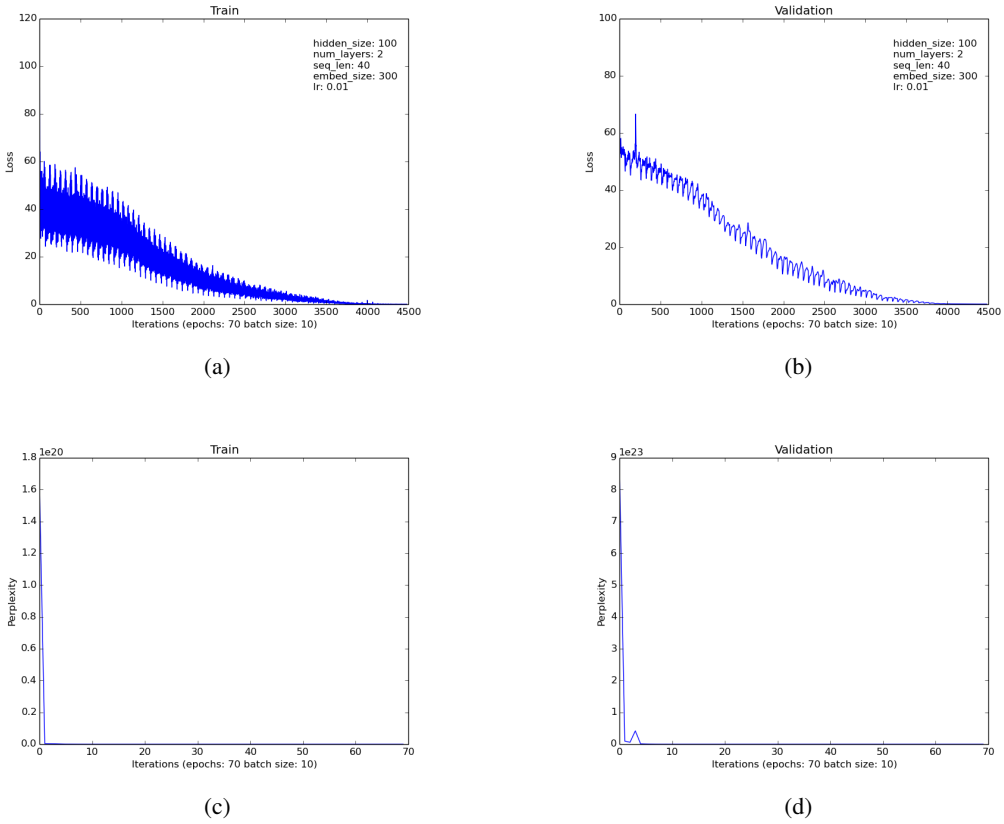

Figure 4: Seq2Seq with GRU units

GRU generally converged much faster with less compute time than LSTM on training and the performance was similar. For both GRU and LSTM based models I was able to achieve loss ¡= 0.5 and very low perplexity on both training and validation datasets.

6.3 Qualitative Analysis: Response Generation

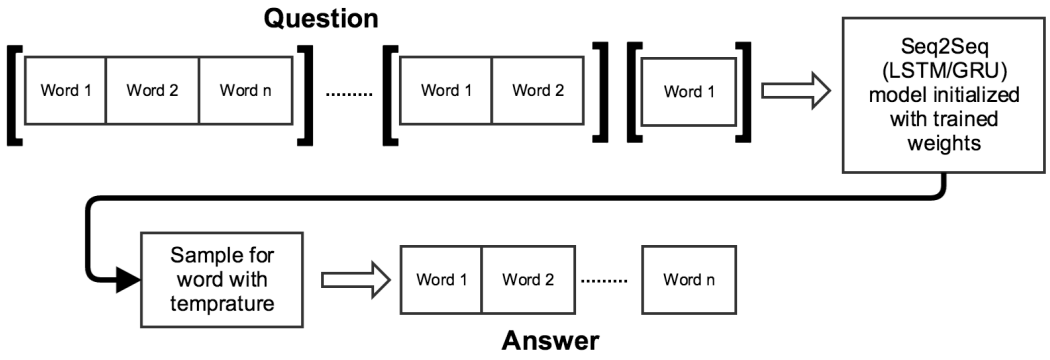

Figure 5: Response generation process

An RNN decoder with either LSTM or GRU units was initialized with the final time-step state from the trained Seq2Seq decoder. A new question was tokenized and fed into the decoder as a progressive sequence of word vectors as shows in the figure 5. In each iteration a word is sampled with a certain temperature from the output of the decoder. A response sequence was generated up to a desired length or until <EOS>. Figure 5 shows an illustration of this process.

6.3.1 Generated Response Samples

```
Q: enron quarterly profits
A: outstanding growth two; volumes in to East 1999
Q: how does enron make money?
A: from and successful in merchant gas factor natural
Q: what does enron do ?
A: transactions energy Some daily Eastern sale
Q: how do you make money ?
A: electricity Power forecast
Q: Enron profits this year
A: Power transactions OnLine increase
```

7 Conclusion

Recurrent neural networks are quite powerful, with the right architecture these neural nets can learn complex tasks. I was able to apply LSTM and GRU recurrent networks to produce some reasonable results with limited dataset in a short period of time. But for a practical application the responses need to be of higher quality and consistency. I have discussed how the quality can be improved in the next section.

8 Future work

There were two limiting factors that left a need for further evaluation of the models and room for improvement: small size of dataset and training time. Small data size is an inherent problem of emails as people do not send hundreds of thousands of emails. This issue is compounded by the scarcity of freely available datasets for research. Developing models that can learn well from small datasets is a good solution. One other approach is to incorporate supplemental information that is available in large quantities which can offset the data scarcity issue by providing a model with enough information to perform well. One such model is Dynamic Memory Networks (DMN).

DMN was introduced by Kumar et al. in [Kum+15]. Unlike Seq2Seq model [Cho+14], Kumar et al. proposes a model that is trained on a triplet of raw input, question and answer. The episodic memory module of DMN iteratively looks at various slices of input information and updates a memory vector. After memory iteration, this vector is used to generate the answer [Kum+15]. For generating email responses, DMN can be trained using all email communication and notes pertaining to a candidate user as input, email messages and responses as questions and answers respectively.

9 Acknowledgement

I would like to thank Superna Sharma for helping with pre-processing the data.

References

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[Mik+13] Tomas Mikolov et al. "Distributed Representations of Words and Phrases and their Compositionality". In: *CoRR* abs/1310.4546 (2013). URL: http://arxiv.org/abs/1310.4546.

[Cho+14] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *EMNLP*. 2014.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". In: *CoRR* abs/1409.3215 (2014). URL: http://arxiv.org/abs/1409.3215.

[Coh15] William W. Cohen. *Enron Email Dataset*. 2015. URL: https://www.cs.cmu.edu/~./enron/ (visited on 05/07/2015).

[JZS15] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. "An Empirical Exploration of Recurrent Network Architectures". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. Ed. by David Blei and Francis Bach. JMLR Workshop and Conference Proceedings, 2015, pp. 2342–2350. URL: http://jmlr.org/proceedings/papers/v37/jozefowicz15.pdf.

[Kum+15] Ankit Kumar et al. "Ask Me Anything: Dynamic Memory Networks for Natural Language Processing". In: *CoRR* abs/1506.07285 (2015). URL: http://arxiv.org/abs/1506.07285.

[SLL15] Lifeng Shang, Zhengdong Lu, and Hang Li. "Neural Responding Machine for Short-Text Conversation". In: *ACL*. 2015.

[VL15] Oriol Vinyals and Quoc V. Le. "A Neural Conversational Model". In: *CoRR* abs/1506.05869 (2015).

[Li+16] Jiwei Li et al. "A Persona-Based Neural Conversation Model". In: *CoRR* abs/1603.06155 (2016).

[Ser+16] Iulian Vlad Serban et al. "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models". In: *AAAI*. 2016.

[San] Jeffrey Dean Sanjay Ghemawat. *LevelDB*. URL: https://github.com/google/leveldb,%20http://leveldb.org.