

Designer Chatbots for Lonely People

Roy Chan

roychan@stanford.edu

Abstract

Two slightly different architectures for a seq2seq neural network based chatbot were tested for their efficacy in generating human understandable English dialogue. In the first design, the chatbot accepted user dialogue in the form of pretrained word vectors. In the second design, user input was broken down into individual characters and fed into the neural net as randomly initialized vectors corresponding to single characters. Both versions of the chatbot were then evaluated on real humans and found to be ineffective in holding reasonable dialogue.

1 Introduction

Artificial neural networks have seen tremendous progress in the field of Natural Language Processing. The many variants of artificial neural networks, from simple to recurrent to convolutional have collectively surpassed the performance of and thus rendered obsolete traditional NLP algorithms. In this study, a recurrent neural network based on GRUs (Gated Recurrent Units) was pieced together in an attempt to allow conversations between human and machine. GRUs were chosen among the different types of recurrent artificial neurons as they have been touted by several studies to outperform the LSTM, which itself has shown state of the art performance on Natural Language Processing tasks. It is hoped that the model would provide good conversational performance on text based English dialogue with humans. In an age where people are fully immersed in a digital life, facilitated by the advent and widespread adoption of virtual reality, there is the plausibility of nonsuperficial, face to face communication between humans becoming a thing of scarcity. Human individuals with reduced exposure to genuine human dialogue might develop emotional loneliness, which is an unpleasant psychological response to the perception of isolation or lack of companionship. Convincing chatbots with good social skills can provide therapeutic conversations with humans, thereby alleviating emotional loneliness or other negative mental states.

2 Background/Related Work

Microsoft recently developed an online chatbot named Tay, which could get smarter by learning from new conversations [1]. However, it was quickly put down due to the unexpected profanities which arose in its language, learnt from the people who interacted with the bot.

Google has also launched its chatbot, titled “A Neural Conversational Model” [2], based on the seq2seq architecture by Sutskever et al., 2014 [3]. The Google bot was a single LSTM layer with 1024 memory cells and was trained on an IT helpdesk dataset to provide replies to customer queries.

The character level chatbot in this study was inspired by Andrej Karpathy’s github release titled “Char-RNN” in which single characters were fed into the neural network as input. Char-RNN was demonstrated to generate entire text documents that resembled its training data [4].

3 Technical Approach and Models

Reddit comment and reply pairs were downloaded from www.reddit.com with a custom python script. Only comments with one or more reply pairs were downloaded and tokenized according to the chatbot input requirements. Start, middle and end tokens were inserted respectively at the beginning of each comment-reply pair, between each comment and reply, and at the end of the reply. A total of 1000 comment and reply pairs were procured with the script and utilized as the training dataset.

The WordGRU chatbot

Google’s TensorFlow framework was used to construct both chatbot models. The WordGRU chatbot comprises 3 GRU layers with 700 memory cells each, arranged in a seq2seq format. The output of each layer is squeezed via an affine layer, and then passed into a softmax layer to give word predictions. In the seq2seq model, the predicted word at each timestep could be fed into the neural network at the next timestep to give a new word prediction. WordGRU accepts word vectors corresponding to individual word tokens as input. The word vectors were derived from a Word2vec model pretrained on a Google News dataset containing 100 billion words [5]. The python Gensim library was used to extract relevant word vectors from the Word2vec embedding matrix [6], and NLTK punkt parser was used to tokenize input sentences into individual words for word vector lookup [7]. As the entire word2vec embedding matrix was large (3.6gb), to speed up loading times word vectors corresponding to encountered words in the training dataset were extracted into a smaller resizable array. This smaller array was programmed to expand and incorporate newer words as they are encountered. For novel words not present in the pretrained Word2vec embeddings, the word vectors were randomly initialized.

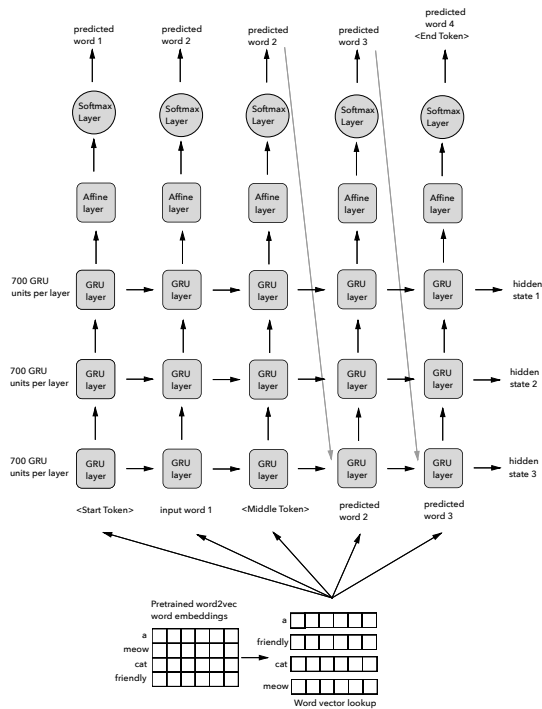
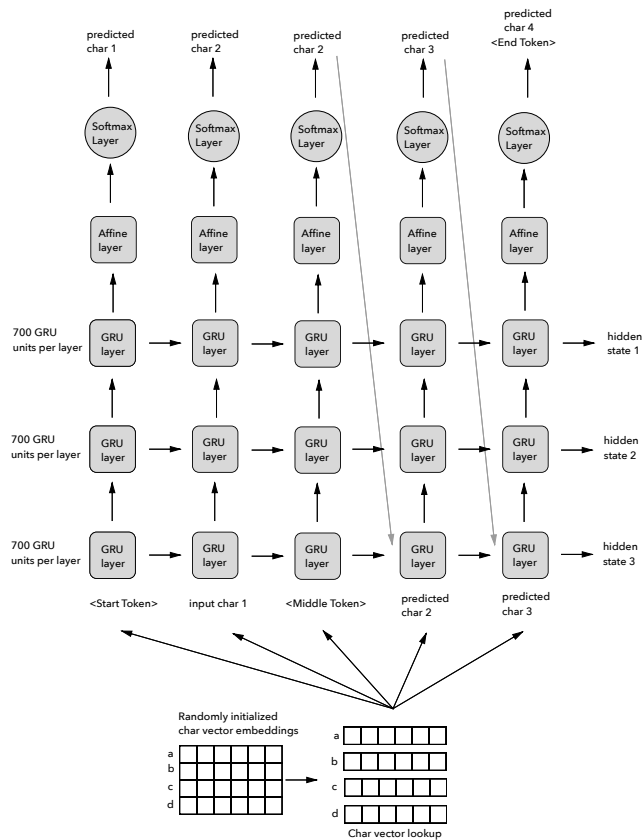


Figure 1: The neural network architecture for WordGRU chatbot.

72 The CharGRU chatbot

73



74

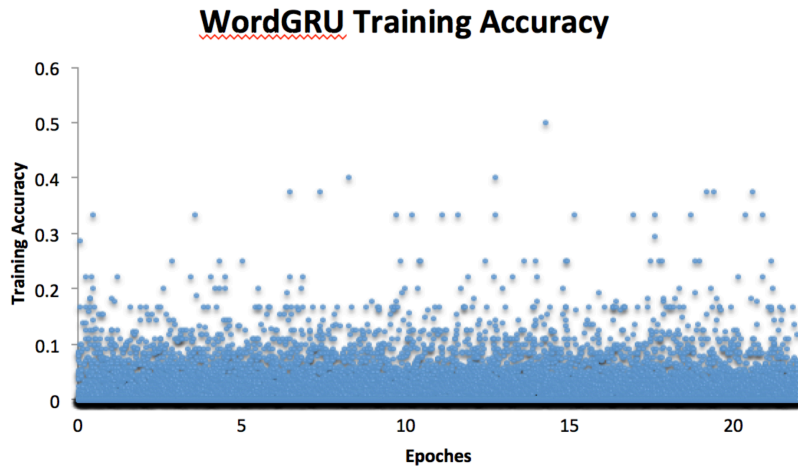
75 Figure 2: The neural network architecture for CharGRU chatbot.

76 The CharGRU chatbot has an identical neural network architecture to the WordGRU chatbot,
 77 except that the accepted inputs are character vectors instead of word vectors. For training,
 78 Reddit comment-reply pairs were tokenized into individual ASCII characters. Each character
 79 is assigned a size 30 randomly initialized vector, or “character vector”. The character vector
 80 embedding array is expandable and accommodates new characters as they are encountered.

81 4 Experiment

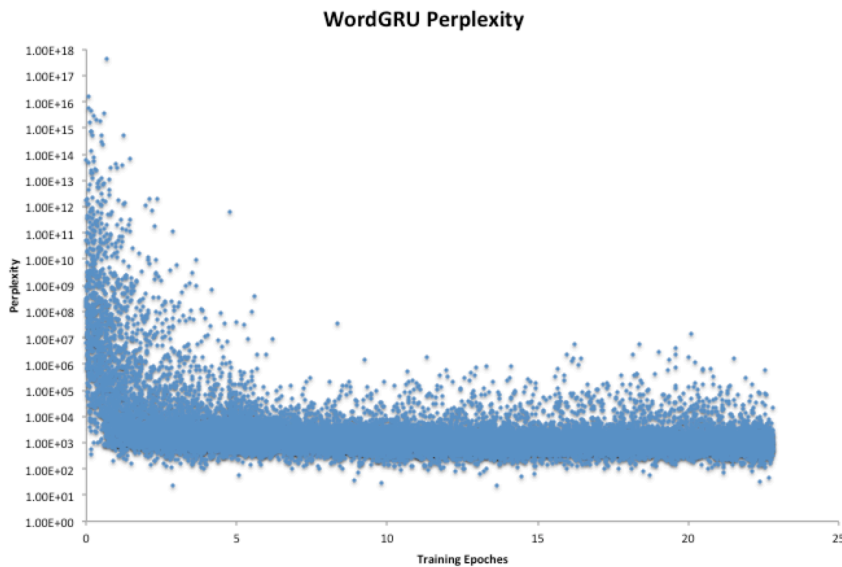
82 Training of both WordGRU and CharGRU chatbots on the reddit comment-reply pair dataset
 83 occurred on an Amazon Elastic Cloud 2 g2.2x large GPU instance. WordGRU managed to
 84 achieve an average training accuracy of 0.05 (baseline chance accuracy is 0.0002 with 4981
 85 words in vocabulary). The test perplexity of about 1000 was achieved after 22 epochs.

86



87

88 Figure 3: Training accuracy of WordGRU over 22 epochs.



89

90 Figure 4: Perplexity of WordGRU over 22 epochs.

91 Sample text generated by the chatbot during training are as follows:

92 **WordGRU:** *brainfarts that that that that that that . that that sincerely sincerely that .*

93 **WordGRU:** *presumably so the door this the a door the door knock in in case the searching*
 94 *you also ?*

95 **WordGRU:** *still not sure why i why the why to make ramen noodles*

96 **WordGRU:** *`` , has been up for up hours and already has 11000 11000 . 11000 the 11000*
 97 *the awnstart . no , it had in 2 hours then the had had had in and does something to it . do n't*
 98 *know what .*

99 **WordGRU:** *what do 's high . she does n't know she 's a awnstart . she 's a told i 'm a dog] (*
 100 *http : awnend)*

101 **WordGRU:** *i worked there there nights where i was there there and very very , so i would*
 102 *make some some some on some , and the next morning i 'd find my find find in the car . still*
 103 *not sure why i why the why to make ramen noodles . i ca ? ? ? hot awnend ?*

104

105 *WordGRU: i it it the day i moved to moved from from , i was very very i think very the fuck*
106 *? you guys told me this me does n't awnend here ! awnend awnend ! "*

107 A human attempt to communicate with the WordGRU chatbot uneventful:

108 **User input:** *Hello friend!*

109 **WordGRU:** *, be former former former flint flint flint*

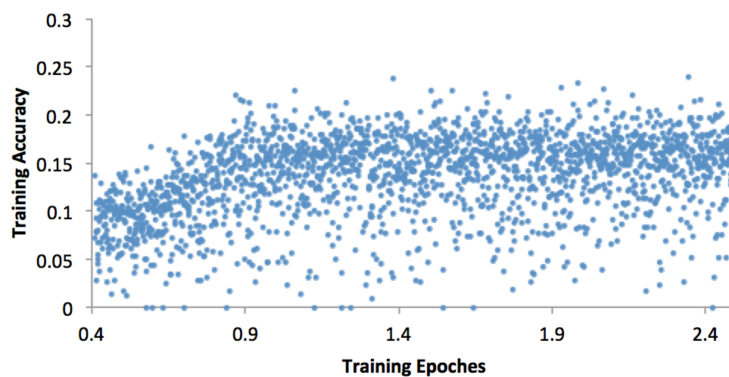
110 **User Input:** *No, I thought your perplexity was low.*

111 **WordGRU:** *literally be to flint flint flint*

112

113 On the other hand, CharGRU managed to achieve an average training accuracy of 0.15
114 (baseline chance accuracy is 0.01 with 96 characters in vocabulary). An average perplexity
115 of 10 was achieved after 2.5 training epochs.

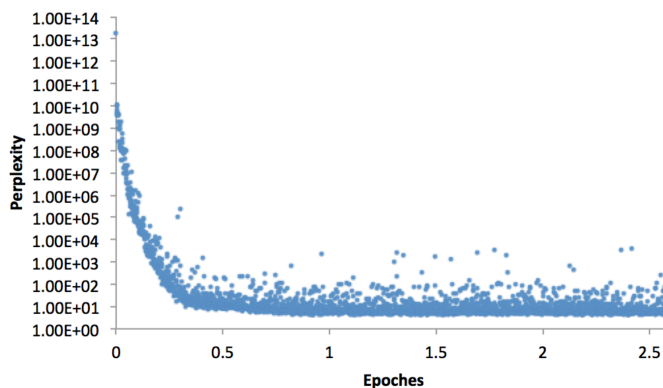
CharGRU Training Accuracy



116

117 Figure 4: Training accuracy of CharGRU over 2.5 epochs

CharGRU Perplexity



118

119 Figure 4: Perplexity of CharGRU over 2.5 training epochs

120 A human attempt to communicate with the CharGRU chatbot gave an unwieldy feeling that
121 the chatbot was unintelligent. It knew it had to place lots of spaces between words, and tried
122 to mimic words with more commonly encountered characters such as the vowels ‘a,e,i,o,u’:

123 **User Input:** *hello friend!*

124 **CharGRU:** *m o n n o o oo fh oe on*

125 **User Input:** *How are you doing today?*

126 **CharGRU:** *hh0reI ooo oh ae / eo eea cxY eh ou ooe aI Mo Fo ee*
 127 *(he th oh Or bab ooe wot ro Pow*
 128 **User Input:** *I'm pretty sure that's not English*
 129 **CharGRU:** *oe ge wT or o o d op hot6)o ^o wodpe io he haa hr Sob ia o oo*
 130 *hhE oo e*
 131 **User Input:** *You gotta be kidding me, I thought your training accuracy was pretty high.*
 132 **CharGRU:** *h A n a _e) 9T h Oto e oeil = BP O / ou uu oj ho oot oo*

134 4 Conclusion

136 The human brain contains about 100 billion neurons, a large fraction of which is involved in
 137 language processing. It was unlikely from the start that a shallow network with 3 GRU
 138 layers of 700 memory cells each was going to result in a successful chatbot with human
 139 level performance. Perhaps a much larger neural network with improved architecture on the
 140 order of a billion neurons and a more sizeable training dataset would result in genuinely
 141 successful chatbots with convincing dialogue.

143 5 References

- 144 [1] Microsoft's chatbot, Tay. [https://en.wikipedia.org/wiki/Tay_\(bot\)](https://en.wikipedia.org/wiki/Tay_(bot))
- 145 [2] A Neural Conversational Model, Vinyals, Oriol; Le, Quoc. arXiv:1506.05869
- 146 [3] Sequence to sequence learning with neural networks. Sutskever, I., Vinyals, O., and Le, Q. V.
 147 In *NIPS*, 2014.
- 148 [4] Andrej Karpathy's CharRNN. <http://karpathy.github.io/2015/05/21/rnn-effectiveness/>
- 149 [5] Google News pretrained Word2Vec. <https://code.google.com/archive/p/word2vec/>
- 150 [6] The Gensim library. <https://radimrehurek.com/gensim/>
- 151 [7] NLTK library. <http://www.nltk.org>