Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models

Bogdan-Adrian Muşat
Xperi
bogdan.musat@xperi.com

Abstract

This project is an attempt to implement the ideas and replicate the results of Serban et al. [2015]. The paper introduces a new idea for preserving the context of an open domain, conversational dialogue system based on large dialogue corpora using generative models. The main technical challenge was to accurately implement and train the network, as described in the paper, using the Keras framework. Although I successfully implemented the paper's novel proposal, I was only partially able to replicate the training results that the original authors report.

1 Introduction

The impressive achievements of Recurrent Neural Networks (RNNs) have revolutionized the field of natural language processing in recent years. Due to their improved performance on tasks such as sentiment analysis, language translation, language modeling, they have become a staple approach for the industry and research community. Cho et al. [2014] proposed a novel network architecture called RNN Encoder-Decoder that consists of two RNNs, later known as seq2seq (Sutskever et al. [2014]). As the name suggests, one RNN encodes a sequence of symbols into a fixed length vector representation, and the other decodes the representation into another sequence of symbols. This kind of architecture represented a real breakthrough for the field of machine translation, and a big step in the direction of automated chatting systems. What this model lacks when it comes to building such a system is the preservation of the conversational context. While in machine translation the actual translation depends only on the current utterance, in a real life conversation you have to keep track of previous information obtained along the way. Sordoni et al. [2015] proposed a model called Hierarchical Recurrent Encoder-Decoder (HRED), which builds on top of the seq2seq idea. It uses a horizontal stack of seq2seq models, binded by a contextual RNN. This allows the information to flow from the past, like in the case of an RNN model, where here each utterance is a different time step. The model architecture was extended to better suit the dialogue task. To carry out experiments, the MovieTriples dialogue dataset was used, which is based on movie scripts[1].

2 Model and Training

A representation of the HRED model architecture is given in Figure 1. A training sample, in this case, consists of a three-turn utterances coming from a dialogue. Since vanilla RNN suffers from the well-known vanishing/exploding gradient problem, the GRU cell proposed by Cho et al. [2014] was used for this network. The encoder is responsible of representing the input phrase as a fixed length vector. To better capture the full understanding of the utterance, the encoder uses a Bidirectional RNN (Schuster and Paliwal [1997]). The output is obtained as the concatenation of the forward and backward runs of the Bidirectional RNN. The context RNN (middle part) maintains a history of the whole dialogue, by taking into consideration all the previous utterances that appeared up until now. The decoder's responsibility is to reproduce as accurately as possible the reply attached to the encoder's input. At each time step of the decoder, the network is trying to predict the next word in

[1] Available upon request from the authors

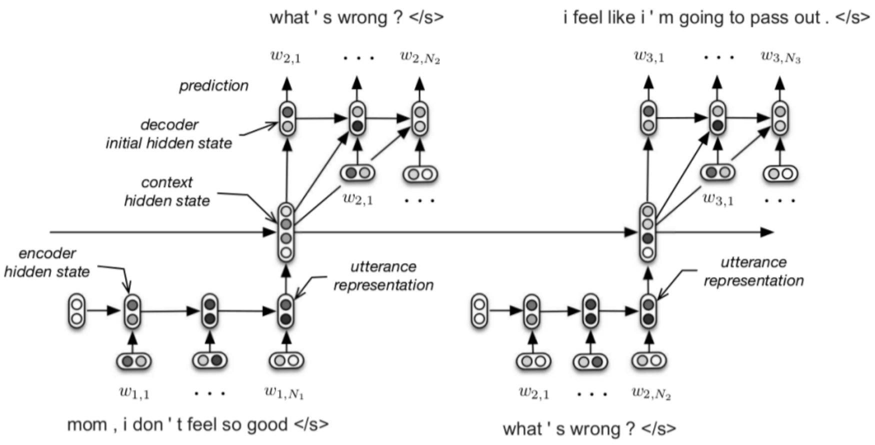

Figure 1: The computational graph of the HRED architecture for a dialogue composed of three turns.

the utterance, using the current input word and the information from the context RNN. Thus, the decoder RNN models a conditional distribution based on a input x_t and a fixed length vector c. For the embeddings of the input words, the model uses pretrained weights from word2vec (Mikolov et al. [2013]), for a better semantical representation. At inference time, beam search with a width of 3 was chosen as the sampling algorithm, as it is common in practice.

The implementation and training of the model was done using Keras (Chollet et al. [2015]) with TensorFlow (Abadi et al. [2015]) as its back-end. The optimizer used was Adam (Kingma and Ba [2014]) with the default hyperparameters and a learning rate decay upon reaching a plateau. The available hardware was an Nvidia GeForce GTX 1080 Ti chipset, and the training took approximately two days until convergence.

3 Results

The principal evaluation metric that was used to measure the quality of the system's replies was perplexity, which is a measure coming from information theory and it's value is e^{xent}, where $xent$ stands for cross entropy. The authors reported a perplexity of **26.31**, whilst this implementation produced a perplexity of **81.74**. Table 1 presents a sample conversation produced by the system:

Table 1: Sample conversation

Real person	Chatbot
Are you a robot?	What are you doing?
Do you have feelings?	I don't know what you are talking about.
Tell me about yourself.	I know. It's not logical, it's not what I can do. Forget it.

4 Conclusion

This project's purpose was learning to work with various state of the art methods involving natural language generation and at the same time getting a grasp on what it means to replicate some paper's results. While the final result did not match the one reported by the authors, this project represented a great learning experience.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL `http://arxiv.org/abs/1406.1078`.

François Chollet et al. Keras. `https://github.com/keras-team/keras`, 2015.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL `http://arxiv.org/abs/1412.6980`.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL `http://arxiv.org/abs/1301.3781`.

M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, Nov 1997. ISSN 1053-587X. doi: 10.1109/78.650093.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Hierarchical neural network generative models for movie dialogues. *CoRR*, abs/1507.04808, 2015. URL `http://arxiv.org/abs/1507.04808`.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. *CoRR*, abs/1507.02221, 2015. URL `http://arxiv.org/abs/1507.02221`.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014. URL `http://arxiv.org/abs/1409.3215`.