CSE847 Project Proposal

An analysis and verification of the Fast Associative Memory method for RNNs

Eric Alan Wayman
Department of Computer Science & Engineering
Michigan State University
waymane1@msu.edu

Adi Mathew
Department of Computer Science & Engineering
Michigan State University
mathewa6@msu.edu

ABSTRACT

This paper proposal describes our goal to perform an analysis and reproduction of tests using the "fast weights" method defined by Ba et al.

KEYWORDS

Recurrent Neural Nets, LaTeX, LSTM

1 PROBLEM DESCRIPTION

Until recently, recurrent neural networks (RNNs), used for sequential processing, were limited by the fact that within-sequence memory was limited to short-term only (long-term memory was limited to between-sequence memory). The "fast weights" method introduced in the paper to be analyzed in this project (Ba et al. 2016a) addresses this limitation by providing the network with the capacity to store information about a given sequence during its duration (to be used during each step in the hidden layers). We will provide a full analysis and explanation of the methology, and replicate one of the empirical tests of the method, which compares its performance on an associative retrieval task to that of an iRNN and a long short-term memory network, or LSTM (Hochreiter and Schmidhuber 1997).

2 SURVEY OF PRIOR WORK

Recurrent neural networks (RNNs) are well-suited for learning from sequential data since weights are shared among different stages of the sequence (Goodfellow et al. 2016, p. 373). In particular, Recurrent Neural Nets have been shown to perform well in tasks of Speech to Text conversion, creation of Language models for both characters and words (Sutskever et al. 2011) and even frame by frame video analyses (Mnih et al. 2014). In RNNs, hidden states essentially act as short-term memory for previous states with inputs and hidden states helping to define the input and future hidden state. One major issue in training RNNs with many layers is that the error gradients end up becoming very large or small (Schmidhuber 2015, p. 16) which implies that even if the network can be trained, the effect of hidden cells corresponding to much earlier values of the sequence is almost non-existent. This was overcome by the introduction of the long short-term memory network (LSTM network), whose activation function has a constant derivative and thus does not explode or vanish (Schmidhuber 2015, p. 19). Unfortunately, the LSTM's memory is still limited to an amount proportional to the number of hidden units in a sequence (Ba et al. 2016a, p. 1). Ba et al. propose the Fast Associative Memory method to allow sequence-to-sequence memory in a recurrent networks.

Hopfield nets, associative memory: (Mackay 2003)

3 PRELIMINARY PLAN

Our term paper will first present the fast associative memory methodology and place it in the context of methods that led to its development. We will provide an extended description and derivation of the methodology for the purpose of verifying its properties. Our goal will be to also replicate section 4.1 of the paper, which compares the fast associative memory method's performance on an associative retrieval task with that of an Identity-RNN, or iRNN (Talathi and Vartak 2015), and LSTM (Ba et al. 2016a).

This project is intended to understand the foundational math and reasoning behind pursuing the use of Fast Weights in a network. The initial stage of our project will be to perform a thorough proof and derivation of the equations for RNNs, and clearly explain the issues that led to the creation of LSTM networks. For instance, we will explain the "long-term memory issue" in RNNs. The expression of the hidden unit h_t at time t is:

$$h_t = g(\boldsymbol{W} \cdot x_t + \boldsymbol{U} \cdot h_{t-1} + b_h)$$

After t time steps, we get:

$$h_t = g(\boldsymbol{W} \cdot x_t + \boldsymbol{U} \cdot g(\cdots g(\boldsymbol{W} \cdot x_{t-T} + \boldsymbol{U} \cdot h_{t-T} + b_h) \cdots) + b_h)$$

Because of the T nested multiplications by $\mathbf{U}$, the effect of $h_t - T$ on h_t is negligible (namely, the network does not have "long-term memory"). We will provide a full exposition of how this problem manifests during training of the network.

The next stage of the project will involve explaining LSTM networks, their improvements on RNNs, and their limitations. We will then explain the mathematics of Fast Weights and the Fast Associative Memory Network, as well as several methodologies used in their implementation in the paper being studied such as layer normalization (Ba et al. 2016b), grid search (Goodfellow et al. 2016), and the Adam optimizer (Kingma and Ba 2014).

Following that, we will implement the Fast Associative Memory Network in MATLAB.

Table 1: *Project timeline*

Week	Dates	Task	Deliverable
Week 1	(2/6 - 2/13)	Background Reading	
Week 2	(2/13 - 2/20)	Background Reading	Proposal
Week 3	(2/20 - 2/27)	Data collection	
Week 4	(2/27 - 3/6)	Background, foundational proofs	
Week 5	(3/6 - 3/13)	Compile data and preliminary run	
Week 6	(3/20 - 3/27)	Report prep	Intermediate Report
Week 7	(4/3 - 4/10)	Full implementation	
Week 8	(4/10 - 4/17)	Run with Data and comparisons	
Week 9	(4/17 - 4/24)	Report prep	
Week 10	(4/24 - 5/1)	Report prep + Rehearse	Final Report Presentation

REFERENCES

Jimmy Ba, Geoffrey E. Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. 2016a. Using Fast Weights to Attend to the Recent Past. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. 4331–4339. http://papers.nips.cc/paper/6057-using-fast-weights-to-attend-to-the-recent-past

Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. 2016b. Layer Normalization. *CoRR* abs/1607.06450 (2016). http://arxiv.org/abs/1607.06450

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. DOI:http://dx.doi.org/10.1162/neco.1997.9.8.1735

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). http://arxiv.org/abs/1412.6980

David J. C. Mackay. 2003. *InformationTheory, Inference, and Learning Algorithms.*

Volodymyr Mnih, Nicolas Heess, Alex Graves, and others. 2014. *Recurrent models of visual attention*. 2204–2212 pages.

Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61 (2015), 85–117. DOI:http://dx.doi.org/10.1016/j.neunet.2014.09.003

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating Text with Recurrent Neural Networks. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*. 1017–1024.

Sachin S. Talathi and Aniket Vartak. 2015. Improving performance of recurrent neural network with relu nonlinearity. *CoRR* abs/1511.03771 (2015). http://arxiv.org/abs/1511.03771