CSE847 Project Proposal

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

Eric 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 an LSTM.

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 was that the gradients (of the error with respect to a particular weight) end up becoming very large or small (Schmidhuber 2015, p. 16). This was overcome by the introduction of the long short-term memory network (LSTM RNN), whose activation function has a constant derivative and thus does not explode or vanish (Schmidhuber 2015, p. 19).

{Next: explain weakness of LSTMs and how introduction of fast associative memory addresses that weakness.} LSTM networks are composed of memory blocks, each of which contain ’memory cells’. Each cell has an activation over a unit and is referred to as the ’cell state’. Due to this linear nature of cell connections, cell states are prone to grow in a linear manner as time units are expended. There have been many attempts to overcome this and other shortcomings of LSTMs. see last citation

Hopfield nets, associative memory: (Mackay 2003)
Layer normalization: (Ba et al. 2016b)
Grid search: (Goodfellow et al. 2016)
Adam optimizer: (Kingma and Ba 2014)
IRNN definition: (Talathi and Vartak 2015)
LSTM: (Gers et al. 2000), (Hochreiter and Schmidhuber 1997)

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 (iRNN) 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 LSTM networks. To do so, we must first perform a thorough proof and derivation of the research and developments that led to the creation of LSTM networks. This will form the initial stage of our project. Some of the components that need to be proven include the following:

- Speed at which weight matrix is diminished by BPTT corresponds to the shortness of memory.
- Given that expression of Hidden unit at time t is:

$$h_t \quad g(\mathbf{W} \cdot x_t \quad \mathbf{U} \cdot h_t - 1 \quad b_h) \tag{1}$$

after t time steps, we get :

$$h_t \quad g(\mathbf{W} \cdot x_t \quad \mathbf{U} \cdot g(\cdots g(\mathbf{W} \cdot x_t - T \quad U \cdot h_t - T \quad b_h) \cdots) \quad b_h) \tag{2}$$

Because of the T nested multiplications by $\mathbf{U}$, the effect of $h_t - T$ (”Long Term Memory”) is negligible.

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*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 4331–4339. http://papers.nips.cc/paper/6057-using-fast-weights-to-attend-to-the-recent-past.pdf

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016b. Layer Normalization. (2016). arXiv:1607.06450

Felix A. Gers, Jürgen Schmidhuber, and Fred A. Cummins. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural Computation* 12, 10 (2000), 2451–2471. DOI: http://dx.doi.org/10.1162/089976600300015015

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.

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

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. (2014). arXiv:arXiv:1412.6980 (published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015)

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.* 1017–1024 pages.

Sachin S. Talathi and Aniket Vartak. 2015. Improving performance of recurrent neural network with relu nonlinearity. (2015). arXiv:arXiv:1511.03771