Paper: *LSTM: A Search Space Odyssey*

Nirajan Koirala

CSC 9010_001

## Summary

In this paper several variants of LSTM architecture for RNNs have been experimented with and discussed. In general, a RNN has a recurrent core that takes some input x and feed that input into itself and has some internal hidden states which gets updated every time that RNN reads a new input and it produces some outputs at every time-step. Since, RNNs suffer from gradient exploding/vanishing problem, LSTM architecture is particularly a RNN architecture developed to combat these kind of gradient flow problems. The central idea behind LSTM is a memory cell which can maintain its state over time and non-linear units which regulate the information flow in and out of the cell. A systematic study of the utility of various components comprising LSTM is done in this paper which ultimately helps in improving these kind of architectures.

An overview of the vanilla LSTM is provided in section 2 where all the functions used for different gates are discussed and various formulas are provided for computing the output of the ifog gates. In section 3, history of LSTM is discussed. The reason behind using different components of LSTM like forget gate, peephole connection is thoroughly discussed. After this discussion, the authors proceed to the evaluation section where they use the vanilla LSTM as a baseline for evaluating eight variants of LSTM. For choosing the best hyperparameters, random search is used and analyses were focused on the 10% best performing trials for each variant and dataset.

In section 4, authors discusses about the datasets, training method, type of LSTM variants and the method of searching the hyperparameters. Datasets used in this paper are TIMIT Speech corpus, IAM Online Handwriting Database and JSB Chorales. A bidirectional LSTM which consists of two hidden layers, one processing the input forwards and the other one backwards in time, both connected to a single softmax output layer is used for the TIMIT and IAM Online tasks. But, a normal LSTM with one hidden layer and sigmoid output layer was used for the JSB Chorales task. Different methods for evaluating the errors are used for each dataset and stochastic gradient descent is used for training them. The first six variants of the LSTM omits each of the gates, input/output activation functions and peephole connection in a vanilla LSTM. The last two variants of the network are CIFG which uses only one gate for gating both input and the cell recurrent connection and FGR which adds recurrent connections between all the gates. Random search is used for the hyperparameters and for TIMIT dataset two additional hyperparameters are considered since it contains acoustic data.

The section 5 consists of results and discussion where the authors compare different variants and discuss the impact of hyperparameters on each of these variants. Welch’s t-test at a significant level of $p = 0.05$ is used to determine whether the mean test set performance of each variant was significantly different from that of the baseline. It is found that removing the output activation function or the forget gate significantly hurts the performance on all three datasets. CIFG and peephole connections are found to not significantly change the mean performance on any of the datasets. Adding the FGR doesn’t significantly change the performance on TIMIT or IAM Online datasets but it leads to worse results on the JSB Chorales dataset. Removing the input gate, output gate and input activation function leads to a significant reduction on performance on speech and handwriting recognition but for music modelling there is no significant effect. Later in the sub-sections, in order to find the impact of hyperparameters fANOVA method is used. Average performance for any slice of the hyperparameters space is obtained by first training a regression tree and then summing over its predictions along the corresponding subset of dimensions. Learning rate is found to be the most important hyperparameter followed by hidden layer size and input noise. They discuss how to search for a good learning rate for each type of datasets and they found that it is better to start with a high value of learning rate. They also found that momentum is not affecting the performance or training time in any significant way and suggest that it does not offer any substantial benefits when training LSTMs with online stochastic descent. For the interaction of hyperparameters, they found that learning rate and hidden size has the strongest one among others and they suggest that they need more samples to properly analyze the fine interplay between the hyperparameters.

Finally they conclude that vanilla LSTM performs reasonably well on various kind of datasets and using any of the eight possible modifications does not significantly improve the performance. Their study find that learning rate and network size are the most important hyperparameters and we can make the LSTM architecture less complex by simply coupling the input and forget gates or removing peephole connections as they don’t significantly hurt the performance of the networks.

**Strengths**

- All the main points made in the paper are concise and to the point.
- Authors did a great job while choosing the various type of datasets for this study.
- Evaluation is done thoroughly as they have used specific type of evaluation methods and architecture for each type of dataset.

**Weaknesses**

- I think the description provided for the vanilla LSTM block was very brief. They could have elaborated the block more in detail as it is the core on which variants are made.

**Points of Confusion**

- I am confused why they used different type of LSTMs for TIMIT and JSB Chorales datasets since both of the datasets are dealing with acoustics data.
- Peephole connections were confusing as well and I couldn’t find any real purpose of using them in LSTM.

**Discussion Questions**

- What are CECs and how it helps LSTM to combat the gradient flow problem?
- What are higher-order interactions and why they play an important role in case of TIMIT but not in case of other two datasets?
- A discussion on how momentum plays a role in training time would be helpful.