Rahul Thapa
Paper: LSTM: A Search Space Odyssey

Summary

The central idea behind the LSTM architecture is a memory cell that can maintain its state over time, and non-linear gating units that regulate the information flow into and out of the cell. Many modifications have been made to the LSTM architecture since its original formulation. However, a systematic study of the utility of various computational components that comprise LSTMs was missing. In this paper, the authors present a large-scale analysis of eight LSTM variants on three representative tasks: speech recognition, handwriting recognition, and polyphonic music modeling. They optimized the hyperparameters for all LSTM variants for each task separately using random search and assessed their importance using the fANOVA framework.

Vanilla LSTM is the most commonly used LSTM in literature. The authors of this paper use this as a reference for comparison of all the variants. It features three gates: input, forgets, and output. It also comprises of block input, a single cell, an output activation function, and peephole connections. The activations functions used in the model are logistic sigmoid and hyperbolic tangent functions.

The initial version of the LSTM block included cells, input, and output gates, but no forget gate and no peephole connections. The authors of this initial version trained the model using a mixture of Real-Time Recurrent Learning (RTRL) and Backpropagation Through Time (BPTT). Only the gradient of the cell was propagated back through time. Therefore, the authors did not use the exact gradient for learning.

The first paper to suggest a modification of the LSTM architecture introduced forget gate which enabled the LSTM to reset its own state. Similarly, peephole connections were added to architecture in order to make precise timing easier to learn. The final modification towards the vanilla LSTM was to do full backpropagation through time (BPTT) training. There were multiple other variants suggested by many other researchers. This paper focuses on comparing different LSTM variants and not to achieve state-of-art results. Each variant adds, removes, or modifies the baseline (Vanilla LSTM) in exactly one aspect, which allows isolating their effect.

The authors used Bidirectional LSTM which consists of two hidden layers, both connected to a single softmax output layer for TIMIT and IAM Online tasks dataset. For the JSB Chorales task, they used a normal LSTM with one hidden layer and a sigmoid output layer. Gradients were computed using full BPTT for LSTMs. Some of the variants of the V architecture that the authors use in this paper are No Input Gate (NIG), No Output Gate (NOG), and Full Gate Recurrence (FGR). While there are other methods to efficiently search for good hyperparameters, the authors used the random search approach as they saw some advantage in it. First, it is easy to implement, trivial to parallelize, and covers the search space more uniformly. There were following hyperparameters that they reached randomly in their experiment: number of LSTM blocks per hidden layer, learning rate, momentum, and standard deviation of Gaussian input noise.

The authors used Welch's t-test at a significance level of $p=0.05$ to determine whether the mean test set performance of each variant was significantly different from that of the baseline. They found out that the most commonly used LSTM architecture (vanilla LSTM) performs reasonably well on various datasets and using the eight possible modifications does not significantly improve its performance. The authors also realized that the forget gate and output

activation functions are the critical components of the LSTM block. Similarly, the learning rate and network size are the most crucial tunable LSTM hyperparameters. However, the analysis of hyperparameter interactions revealed that the interaction between even these crucial hyperparameters is quite small which implies that the hyperparameters can be tuned independently.

Strengths

- The paper does a good job of incorporating really nice visualizations such as pie charts and box plots with color codes. The figure showing the architecture of LSTM is also self-descriptive.
- The supplementary material section provides further insight into their experiment for those who want to understand it in depth or replicate it.
- The authors have done a good job of testing their model on various datasets with various hyperparameter tuning to be able to assert their conclusions confidently.
- Unlike other papers, the authors ran their experiments multiple times and calculated p-value to establish confidence in their model.

Weaknesses

- The math is not explained well in the paper.
- Some of the captions in the figures are not very descriptive. It s very hard to understand what the figure is showing exactly.

Confusions

- I could not fully understand the schematic of the Long Short-Term Memory block shown in figure 1.
- I also find it hard to understand the vector formula for various gates explained in Section 2.

Discussions

- How is Long Short-Term Memory able to capture the long-term dependencies?
- What is the function of various gates such as forget gate in LSTM?
- How does gradient descent work in RNN/LSTM?