Assignment 14

LSTM - A Search Space Odyssey

Nisim Hurst

Sunday 22 April 2018

Abstract

The paper [1] is the first large-scale comparison of 9 distinct variants of LSTMs in 3 dissimilar domains. The authors used random search for hyperparameter tunning. Training time amounts to an equivalent of 15 years of a single-CPU time over 5400 experiments.

Hypothesis

This paper has no hypothesis to prove, it is just a comprehensive review of eight LSTM variants on three representative tasks: speech recognition, handwriting recognition and polyphonic music modeling. The objective is to test how well LSTM's perform without falling in any assumptions. However, the authors rely heavily on random search and the fANOVA framework to weight the parameters influence in each variant.

Evidence and Results

LSTM vanilla is taken as reference which is the most common architecture used in the literature. The variants that where tested are:

- **NIG.** No input gate.
- **NFG.** No forget gate.
- **NOG.** No output gate.
- **NIAF.** No input validation function.
- **NOAF.** No output validation function.
- **CIFG.** Coupled Input and Forget gate.
- **NP.** No peepholes.
- **FGR.** Full gate recurrence.

Datasets

1. **TIMIT.** TIMIT is a frame-wise speech recognition dataset in which the task is to recognize phonemes. It is divided into 3696 for training, 400 for testing and 192 for validation with 304 frames on average.
2. **IAM Online.** IAM-OnDB is a handwriting recognition dataset in which the task is to map pen movements to characters. It contains a set of *boards* that are divided into one training set of 775, two validation sets of 192 and 216, and one testing set of 544. A *board* is an image with multiple handwritten lines and sentences.
3. **JSB Chorales.** A music dataset where the task is to predict the next note. It consist on 229 training sequences, 76 validation sequences and 77 test sequences.

Hyperparameter tunning

These variants were tuned by using 200 trials on 27 random searches (3 datasets per 9 variants). The following table describe the search space:

Characteristic	Range
# LSTM blocks per hidden layer	log uniform [20, 200]
Learning Rate	log uniform $[10^{-6}, 10^{-2}]$
Momentum	log uniform [0.01, 1.0]
Standard deviation of Gaussian noise	uniform [0, 1]

Results

Table 2: Results vs State-Of-The-Art methods

Dataset	LSTM	State-of-the-art
TIMIT classification error (CIFG)	29.6%	26.9%
JSB Chorales log likelihood (NIAF)	-8.38	-5.56
IAM Online character error rate	9.26%	11.5%

Table 2 above shows the results presented in the paper vs other state of the art methods mentioned in the paper [1].

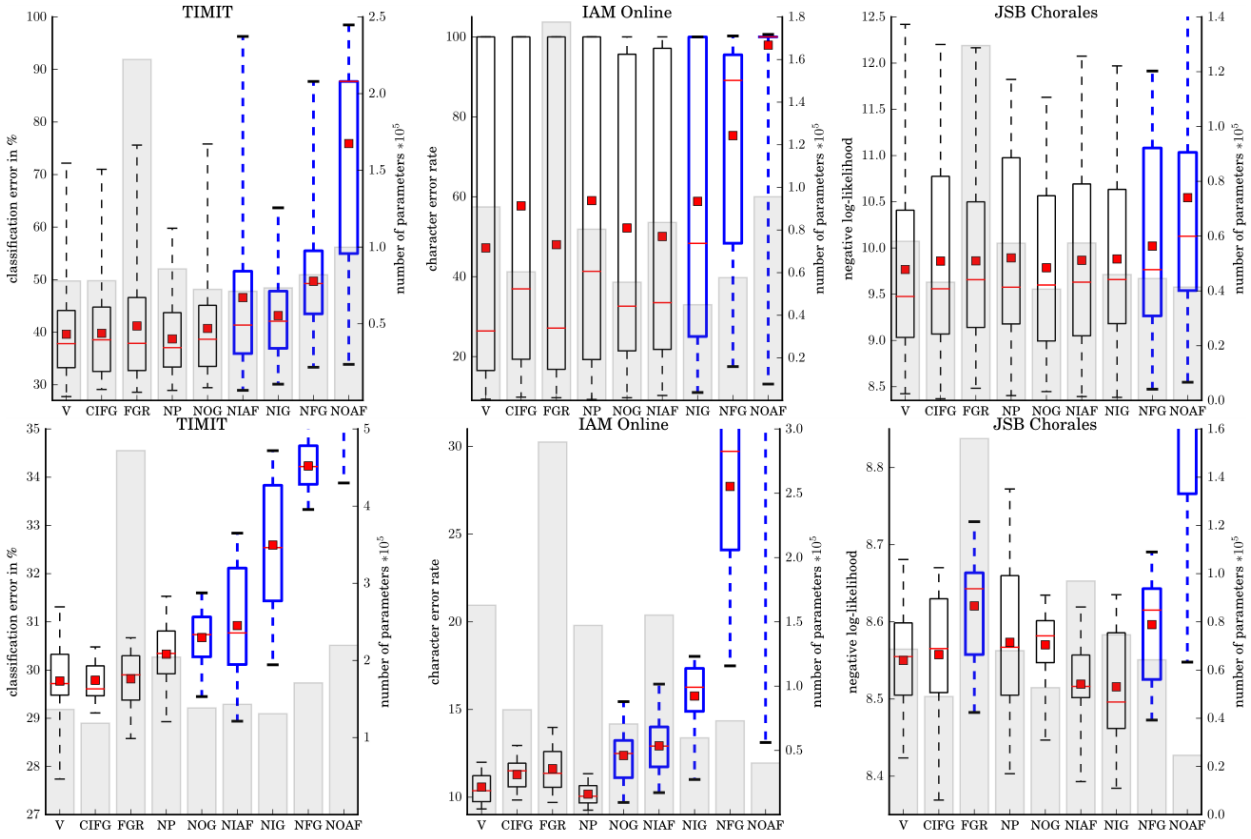

Figure 1: Boxplot results

Figure 1 (taken from [1]) shows the variance on different metrics for all the LSTM variants over all the datasets. The first row show the variance over the 200 trials and the bottom row only over the 10% bests

results. The contribution section explains the observations that constitute the contribution of these results.

Contributions

First, the authors start by describing the Vanilla LSTM architecture which is the most common architecture used in the literature, both for the forward pass and the backpropagation through time. They bring about the related formulas and how gradient propagation change in each case. This is only an introduction to understanding the stemming origin of the eight other variants.

Then, the history of the LSTM is presented through its variants:

1. **The Initial version.** It had no forget gate and peephole connections. Training used Real Time Recurrent Learning and Backpropagation Through Time on the gradient cell. It also had *full gate recurrence*, i.e. all gates received recurrent inputs from all gates.
2. **Forget gate.** Allowed an LSTM to flush its inner state and be used in online learning**.**
3. **Peephole connections.** Allowed precise timings easier to learn by giving the cell direct control over the output.
4. **Full gradient.** By using full backpropagation through time it allowed the gradients to be checked using finite differences.
5. **Other variants.** Most of them were developed as alternatives to BPTT. An extended Kalman Filter was used to allow the net be trained on pathological cases at the cost of high complexity. GRU (Gate Recurrent Unit) is a variant of LSTM but doesn't uses neither peephole connection nor output activation functions and coupled the input and the forget gate into an update gate. Evolutionary, linear projection and trainable slope modification were also made with good results for specific tasks.

Then, the authors presents their empirical results versus state-of-the-art-methods and over 3 distinct dataset of different domains for the 8 variants. This is a good contribution that can be summarized in the following set of observations:

- Removing the output activation function (which bounds the output) or the forget gate (allows to forget information) hurt performance in all datasets.
- Input and forget gate coupling only improved music modeling and peephole connections only improved handwriting recognition.
- Other tasks were harmed by some other variants, e.g. full gate recurrence hurt performance on music modeling.

Finally, one of the best contributions of this article is the use of random search for weighting and finding the best hyperparameters in each domain. The following list summarize the weight impact of each hyperparameter:

- **Learning rate.** The most important hyperparameter affecting accuracy. It can be tuned in a small network and then be used on a larger one.
- **Hidden layer size.** The most important parameter affecting performance (training time).
- **Input noise.** Hurts performance and training time.
- **Momentum.** It had less than 1% accountability for the total variance so it is not so important.

ANOVA was used to measure variance percentage per hyperparameter and for higher order interactions. These were the results:

	TIMIT	IAM-OnDB	JSB Chorales
higher order interactions	20%	7%	5%
learning rate	67%	89%	91%
hidden size	10%	2%	<1%
input noise	1%	<1%	2%
momentum	<1%	<1%	<1%

Weaknesses

The accuracy results are shown in boxplots and then the authors claimed to have significant results based on their observations. However, only the Welch's test vs the vanilla architecture was made and no all-vs-all approach was taken for example using a Shaffer or Bergmann-Hommel tests [2]. Even though this is comprehensible because there are not enough datasets (30 is the least recommended in the literature [3]), it would be desirable to repeat this test on more datasets to be able to perform other statistical tests.

Future Work

Through all the testing the authors made, some progress on characterizing the problem state has been achieved. Further studies could be performed to find a good set of intrinsic variables in the dataset to predict hyperparameter accountability of the variance. Having handled that first step, a good heuristic could be derived to make LSTMs converge faster or to find out in which datasets they outperform other methods.

References

[1] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[2] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[3] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.