Summary of "Visualizing and Understanding Recurrent Networks"

Anuar Maratkhan

Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM) in particular has showed an effective performance in application that involve sequential data, such like language modeling, handwriting recognition, machine translation, speech recognition, video analysis, and image captioning.

However, the performance and drawbacks of the LSTM remain poorly understood. But the most cited convenience of the LSTM is the ability to store and retrieve data over long time scales. This ability was tested mostly in toy problems quite well. Although, it hasn't been studied clearly that the LSTM can perform as well in real-world problems.

The authors of the paper has explored the predictions of the LSTMs on real-world data by illuminating long-range dependencies learned by LSTMs. Moreover, the researchers measured the predictions of LSTMs and compared that to n-gram models in the same environment. As a result, the researchers has found that LSTM outperformed n-gram significantly. In the end, the authors conducted error analysis of that performance of LSTM.

In general, RNN has been studied a lot. However, the most succesful and popular are LSTM networks. In addition, there exist improvements of the basic architecture such as Gated Recurrent Units (GRU). Further, in this paper the authors used different approach in studying LSTM.

In tthe paper the authors discussed three recurrent network architectures that are RNN, LSTM, and the GRU, which are the most commonly used architectures. The simplest description of a deep recurrent networks include hidden state vectors h, where time t and depth l are taken as parameters. This paper however omits the bias vectors in the Vanilla RNNs for brevity. LSTM in its turn, were designed to show the difficulties of training RNNs. For instance, the backpropogation flow was the reason of vanishing of gradients. The difference of LSTM is that it maintains a memory vecor c, other than hidden state vectors. As a consequence, LSTM can choose to read from, write to, or reset the cell using gate activations that are based on a sigmoid function. The use of such feature in LSTM is that it enables to distribute gradients during the backpropogation. The GRU forms less complex alternative to the LSTM.

The researchers used character-level language modeling in testing the networks for sequence learning. To be explicit, the sequence of characters was given on input, and classifiers' goal was to predict the next character in that sequence. This was obtained by illustrating vectors that were projected from the top hidden layer of the network and held unnormalized log probability of the next character, and whose objective was to minimize the average cross-entropy loss over all targets.

During the work, researchers initialized all parameters uniformly, used mini-batch stochastic gradient descent. The models were trained for 50 epochs.

The datased used in the experimental part was different than those in previous works. Two datasets with more than 6 million characters in total were chosen, namely "War and Peace" and the source code of Linux Kernel were used.

Comparison of the recurrent network models showed that RNNs performed relatively less efective than LSTM and GRU, which showed relatively identically better performance. LSTM cells performed well on real-world data through memorizing longer range dependencies (e.g. 230 characters). The distribution of saturation regime mostly appears to be right-saturated (activation is more than 0.9), which is interesting because the cells in this model remember their values in long range periods of time, and do not activate their forget gates so often. Since, there are no cells showed up as left-saturated (less than 0.1), the cells do not act as in feed-forward fashion. Futhermore, the authors compared LSTM to n-gram models to show that first has a capability to keep track of longer range interactions. In particular example of closing brace, the LSTM performed better than n-grams on more than 20 character sequence.

The researchers, further in the paper, has analyzed most common errors of recurrent network models that were used. In particular, they proposed several oracles, which are n-gram oracle, dynamic n-long memory oracle, rare words oracle, word model oracle, punctuation oracle, and boost oracles that describe the errors occured and suggest eliminations of them. Moreover, the authors provide readers with the number and type of errors made during testing, and improvement that will happen when certain oracles applied. The results show that n-gram oracle will produce the best improvement.

References

[1] Andrej Karpathy, Justin Johnson, Li Fei-Fei *Visualizing and Understanding Recurrent Networks*. Stanford University, 2015