swissQuant
Mastering Complexity

Mathematical Challenge February 2017

Recurrent Neural Networks

References

- [1] Bengio, Y.; Goodfellow, I. J. & Courville, A. “Deep learning” *An MIT Press book in preparation,* 2015
- [2] Jaeger, H. & Haas, H. “Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication”, *Science, American Association for the Advancement of Science,* 2004*, 304*, 78-80
- [3] Olah, C. “Understanding LSTM Networks”, http://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015
- [4] Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio. "On the difficulty of training recurrent neural networks." *ICML (3)* 28 (2013): 1310-1318.
- [5] S. A. Billings. "Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains, Wiley, ISBN 978-1-1199-4359-4, 2013.
- [6] Karpaty, Andrej. “The Unreasonable Effectiveness of Recurrent Neural Networks”, http://karpathy.github.io/2015/05/21/rnn-effectiveness/, 2015

Description

Introduction

In this challenge, we succinctly present old as well as new considerations about Recurrent Neural Networks, and the main practical challenges when fitting them.

Recurrent Neural Networks (RNNs)

RNNs as an idea are nearly as old as the Neural Networks are themselves. They were introduced approximately in 1980s with the birth of the idea of parameter sharing [1]. In its purest form, RNN is

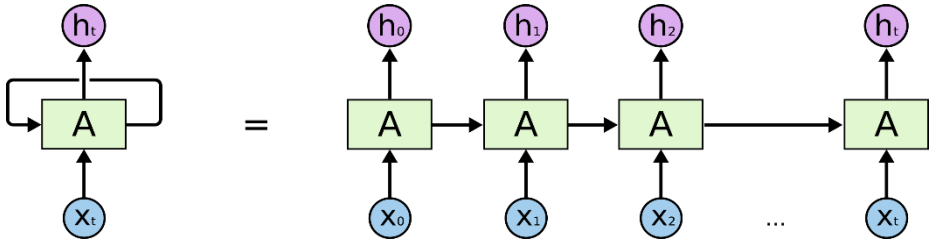

Figure 1. RNN Topology; taken from [3].

a simple feedforward neural network which takes two inputs, and outputs a result (h_t) along with a feedback that is then further used as an input the next time the network is used. This idea is visualized in Figure 1, where **A** denotes an arbitrary function that handles the input and the feedback to output a prediction h_t.

Let s_t denote the feedback (also a hidden state), from the input x_t, then we can define the output of the network as follows

$$h_t = \sigma(\boldsymbol{X} s_{t-1} + \boldsymbol{Y} x_t)$$
$$s_t = \sigma(\boldsymbol{W} s_{t-1} + \boldsymbol{Z} x_t)\,,$$

where $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{W}$ and $\boldsymbol{Z}$ are parameter matrices to be determined and σ represents a non-linear element-wise function. In the machine learning language x_t represents features and h_t represents the predicted labels.

RNNs have shown to be very good models of temporally linked data such as language or speech among many others [1, 3]. Particularly, one type of RNN called Long-short term memory (LSTM) caused a breakthrough in the field recently [6]. In this model, changes to s_t are only additive (no scaling), and procession of the hidden state is heavily regulated. Further information regarding the informal interpretation of LSTM can be found in [3].

Reservoir Computing

The applications to the speech and language are novel, but RNN have had their breakthroughs in literature before. Already in the early 2000s, a RNN was used as a black-box estimator of a dynamical system that had either very non-linear or chaotic dynamics to be modelled by classical methods such as ARIMA [2]. This approach, to usually time series prediction, is often called Reservoir Computing in the scientific literature. In [2] authors show successful applications of RNNs on prediction of chaotic time series with possible application to denoising of wireless communication protocols. Additionally, in finance field, the reservoir computing is more often called Non-linear autoregressive exogenous model (NARX) [5], however the principles of recurrent systems are still very similar. These uses demonstrate that RNNs is already an established technology disguised under different names that has been recently dramatically improved to tackle speech and language with the invention of several new model dynamics such as LSTM [1, 6].

The training of RNNs and other recurrent systems

One of the main difficulties while working with RNNs is their training. In other words, by training we mean the optimal choice of $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{W}$ and $\boldsymbol{Z}$. A classical approach is to formulate a loss function with a sum structure on the samples (x_t, y_t, h_t), and subsequently minimize it with first order (gradient descent variants) or second order (e.g. L-BFGS) methods [1].

$$Loss(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}) = \sum_{t=1}^{T} (y_t - h_t(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}))^2$$

As any summand in the cost function is dependent on all the previous updates through s_t, taking a gradient often leads to a repeated multiplication of the same term, which can lead either to convergence towards zero or divergence. This phenomenon is known as the vanishing or exploding gradient problem. The exploding gradient problem can be solved by enforcing a maximum length of a vector – so called gradient clipping, whereas the vanishing gradient problem is more subtle.

Possible remedies for vanishing gradient problem include efficient initialization and truncated backpropagation through time [4]. Additionally, more advanced recurrent architectures were proposed in order to minimalize this effect such as LSTM by enforcing additive updates to the recurring state.

Questions:

- Q1. Demonstrate the vanishing gradient problem on a simple one hidden layer RNN with the history of length T. Assume no non-linearity (σ is the identity), and a quadratic loss function. *Hint:* Use the equation in the first section, and consider the bound on the expression $\frac{\delta s_{t+k}}{\delta s_t}$.
- Q2. Building upon on the result from Q1, can you propose an efficient random initialization of a matrix $\boldsymbol{W}$ such that the implication of vanishing gradient problem can be minimalized? *Hint:* Think about the initial spectral radius of the matrix $\boldsymbol{W}_0$.
- Q3. What other possible applications of RNNs or more generally recurrent systems, apart from wireless communication, power systems, speech and language can you think of? Is there a novel time series dataset with sufficiently complex temporally dependent structure that can be exploited by these highly parametrized recurrent models (such as LSTM)?