Recurrent Neural Networks

Gregory Feldmann

February 25, 2020

Abstract

1 A Vanilla RNN Cell

1.1 Overview

The following definition is based on [1]. Suppose that at time t we have an input sequence $\mathbf{x}_t = (x_1, x_2, x_3, ...x_m)$ of length m. A simple RNN cell with one hidden unit h first does the following

$$h_t = f_{hidden}(W_{in} \bullet \mathbf{x}_t + H \bullet h_{t-1} + b_{hidden}) \tag{1}$$

where h_t is the value of the hidden state h at time t, W_{in} is a vector of weights applied to $\mathbf{x}$, H is a vector of weights applied to h_{t-1}, b_{hidden} is the bias and f_{hidden} is a non-linearity such as tanh. The output of the vanilla RNN cell is then computed as follows

$$\hat{y}_t = f_{out}(W_{out} \bullet h_t + b_{out}) \tag{2}$$

where $\hat{y}_t$ is the output at time t, W_{out} is a vector of weights applied to h_t, b_{out} is the bias and f_{out} is an activation function.

We can see that h_t stores information about previous computations in the RNN. This allows it to 'remember' information and re-use it in future calculations.

1.2 Vanishing and Exploding Gradients

2 Gated Recurrent Units

Gated recurrent units (GRUs), proposed in [2], are more complex than RNNs, but simpler than LSTMs (more on LSTMs below). For this reason they are discussed before LSTMs, despite being discovered after them.

GRUs build on Vanilla RNNs by adding *reset* and *update* gates. As in the

vanilla RNN case, we assume only one hidden unit. In this case, the reset gate is given by

$$r = f_\sigma\Big(W_r \bullet \mathbf{x}_t + H_r \bullet h_{t-1}\Big) \tag{3}$$

where r is the value of the reset gate, W_r is a vector of weights applied to x_t, H_r is a vector of weights applied to the hidden state h_{t-1} and f_σ is the sigmoid activation. The update gate is given by

$$z = f_\sigma\Big(W_z \bullet \mathbf{x}_t + H_z \bullet h_{t-1}\Big) \tag{4}$$

where z is the value of the update gate and the remaining variables are defined similarly to those in the reset gate.

The reset and update gates are then used to calculate the hidden state as follows

$$h_t = zh_{t-1} + (1-z)\tilde{h}_t \tag{5}$$

where $\tilde{h}_t$ is the *candidate hidden state* and is defined as follows

$$\tilde{h}_t = f_{hidden}(W_{\tilde{h}} \bullet \mathbf{x}_t + w_{\tilde{h}} r h_{t-1} + b_{\tilde{h}}) \tag{6}$$

When the reset gate r is close to 0, $\tilde{h}$ becomes independant of h_{t-1} and there is greater emphasis on calculating $\tilde{h}$ using information from the input values $\mathbf{x}_t$. The reset gate then determines how much the candidate hidden state relies on the previous hidden state. This helps to capture short-term dependencies [3].

When the update gate is close to 1, the hidden state becomes independant of $\tilde{h}$. In other words, the update gate determines how much of the old state to retain in the new state. This helps to capture long-term dependencies [3].

3 Long Short-Term Memory Cells

Long short-term memory (LSTM) neural networks build on the complexity of GRUs with an input gate, forget gate, output gate and memory cell. The input, forget and output gates are all calculated similarly as follows

$$I_t = f_{sigma}(W_{input} \bullet \mathbf{x}_t + H_{input} \bullet h_{t-1} + b_{input}) \tag{7}$$

$$F_t = f_{sigma}(W_{forget} \bullet \mathbf{x}_t + H_{forget} \bullet h_{t-1} + b_{forget}) \tag{8}$$

$$O_t = f_{sigma}(W_{output} \bullet \mathbf{x}_t + H_{output} \bullet h_{t-1} + b_{output}) \tag{9}$$

The candidate memory cell is given by

$$\tilde{C}_t = f_{tanh}(W_{\tilde{C}_t} \bullet \mathbf{x}_t + H_{\tilde{C}} \bullet h_{t-1} + b_{\tilde{C}_t}) \tag{10}$$

and the memory cell by

$$C_t = F_t C_{t-1} + I_t \tilde{C}_t \tag{11}$$

Finally, the hidden state is given by

$$H_t = O_t f_{tanh}(C_t) \tag{12}$$

4 Backpropagation Through Time

5 Bi-directional RNNs

References

[1] John A Bullinaria. Recurrent neural networks neural computation : Lecture 12, 2015. Available at https://www.cs.bham.ac.uk/ jxb/INC/l12.pdf.

[2] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[3] Aston Zhang, Zack C Lipton, Mu Li, and Alex J Smola. 9.1. gated recurrent units (gru). Available at https://d2l.ai/chapter$_r ecurrent - modern/gru.html$.