Introduction to Deep Learning using pyTorch

Dennis Köhn

Wednesday 6th June, 2018

Tutorial at HWR Berlin

Table of contents

1. Introduction to pyTorch
2. Scientific Computing on GPUs
3. Feed-Forward Network
4. Recurrent Neural Networks
5. Application: Sequence Classification in E-Commerce

Introduction to pyTorch

What is pyTorch?

PyTorch is a python package that provides two main functionalities:

- data structures to performing scientific computing on GPUs
- modules useful for building Deep Learning model

Installation

- Much simpler than comparable libraries (Tensorflow, Keras, etc.)
- Generate pip/conda command here

Scientific Computing on GPUs

Feed-Forward Network

Starting Point

A single Neuron

- linear transformation wrapped into an activation function
- $f(x) = \sigma(xw + b)$
- $w \in \mathbb{R}^d$ weight vector representing the weights of each of the d features
- $b \in \mathbb{R}$ the bias term
- σ is the activation, .e.g. sigmoid $\sigma(z) = \frac{1}{1+exp(-z)}$

Basic Neural Network: General Idea

- Stack multiple neurons above and next to each other (neurons in hidden layers)
- Formally: $f(x) = y(\sigma_h(xW^{(1)} + b^{(1))}W^{(2)} + b^{(2)})$
- Multiple hidden layer (deep learning)
- Update weights by propagating the error through the layers

Feed Forward Neural Network

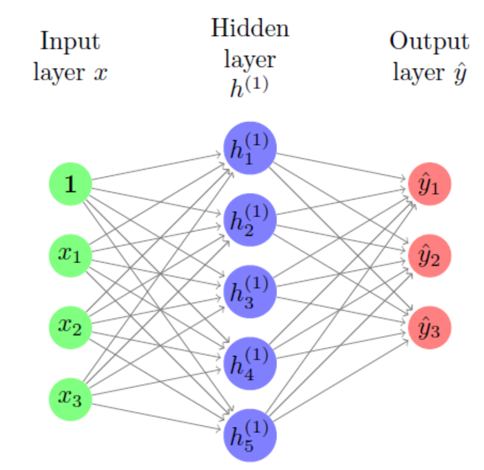

Figure 1: Feed-Forward Neural Network

Recurrent Neural Networks

Motivation:

- Modeling temporal dependencies between current and previous data points
- Example: Predict next word of a incomplete sentence

Solution:

- add loop into hidden layers

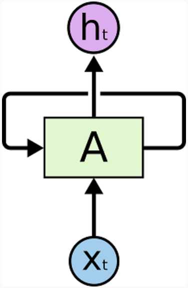

Figure 2: RNN Loop [2]

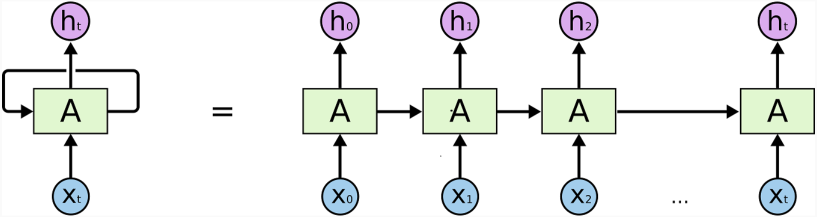

Figure 3: unrolled RNN [2]

Problems of RNNs

- Sequential dependencies of Layers
  $\rightarrow$ hard to parallelize
- Problems handling long term dependencies (*Vanishing Gradient Problem*)
  $\rightarrow$ Solution: LSTMs

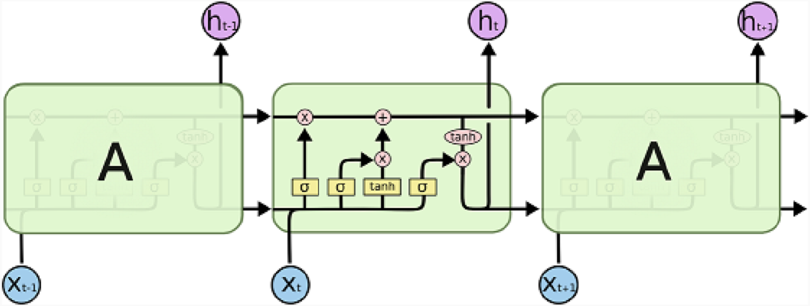

Figure 4: The four layers of the LSTM [2]

LSTM: *Forget Layer*

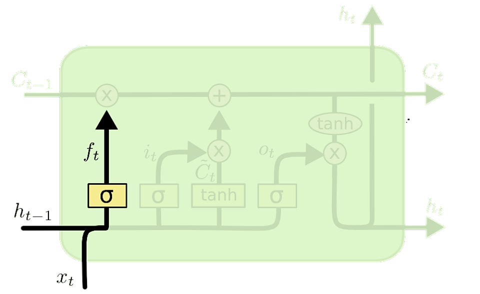

Figure 5: *Forget Layer*[2]

- Which information to pass from previous to current layer
- $f_t = \sigma(W_f * [h_{t-1}; x_t] + b_f)$

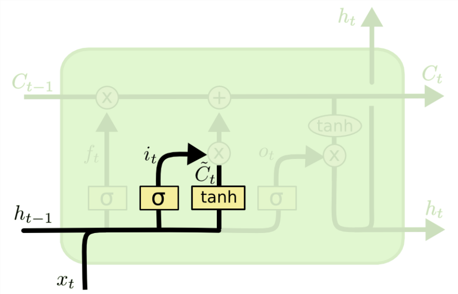

Figure 6: *Update Layer*[2]

- which information to update
- update values: $i_t = \sigma(W_i * [h_{t-1}; x_t] + b_i)$
- new candidates: $\tilde{C}_t = tanh(W_C * [h_{t-1}; x_t] + b_C)$

LSTM: Updating the Cell State

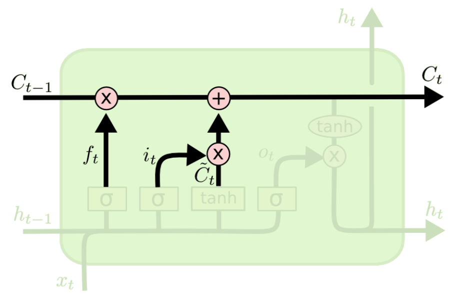

Figure 7: *Update Layer*[2]

- the actual update happens here
- $C_t * f_t + i_t * \tilde{C}_t$

LSTM: *Output Layer*

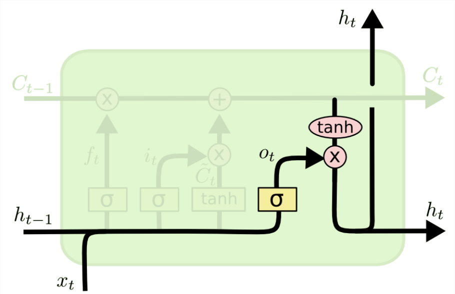

Figure 8: *Output Layer*[2]

- filters the output based on C_t
- $o_t = \sigma(W_o * [h_{t-1}; x_t] + b_o)$
- $h_t = o_t * tanh(C_t)$

Application: Sequence Classification in E-Commerce

Problem Statement

Given an incomplete trace of a user session from an e-shop, predict the whether the user will buy or not.

- user session trace = sequence of page views called clickstream
- train an algorithm to predict the outcome of clickstream
- training data: clickstreams of past sessions with their corresponding outcome
- page view at time t depends on page view at time $t-1 \rightarrow$ LSTM to model this dependency

References I

S. Hochreiter and J. Schmidhuber.
Long short-term memory.
Neural computation, 9(8):1735–1780, 1997.

C. Olah.
Understanding lstm networks (rnn graphiken), 2015.
[Online; accessed Juli, 2015].