Deep Learning Course 2018 *Vanditha Chandrashekar Rao*
Autonomous Intelligent Systems
University Freiburg *Date: 13.11.2018*

Deep Learning Lab Course 2018 - Exercise 2

Introduction

In this exercise the goal is to implement a convolutional neural network (CNN) in TensorFlow and optimizing the hyperparameter using Random search. This report will give a short overview over the implementation of CNN using Tensorflow and the effect of hyperparameter optimization on the accuracy. The experiment was conducted with TensorFlow running on the GPU.

The exercise consists of four experiments. In the first task, the network is trained several times, with default learning rate of 1e-3. The second and the third task examines the influence of the different learning rates and different filter sizes [1,3,5,7] on the network performance. All other parameters such as number of training epochs in the convolutional layers, and batch size in the stochastic gradient descent are fixed.

In the fourth task the hyperparameters are optimized using Random search.

Convolution Neural Network

The network for classifying the MNIST data set is based on LeNet and consists of two convolutional layers with 3x3 filters, (number of filters is 16 and stride of 1) where each layer is followed by a ReLU activation function and a max pooling layer (pool size of 2). After the convolutions, the output of the last max pooling layer is used as the input of a fully connected layer with 128 units followed by a softmax layer with 10 units. The optimization is performed with stochastic gradient descent with cross-entropy.

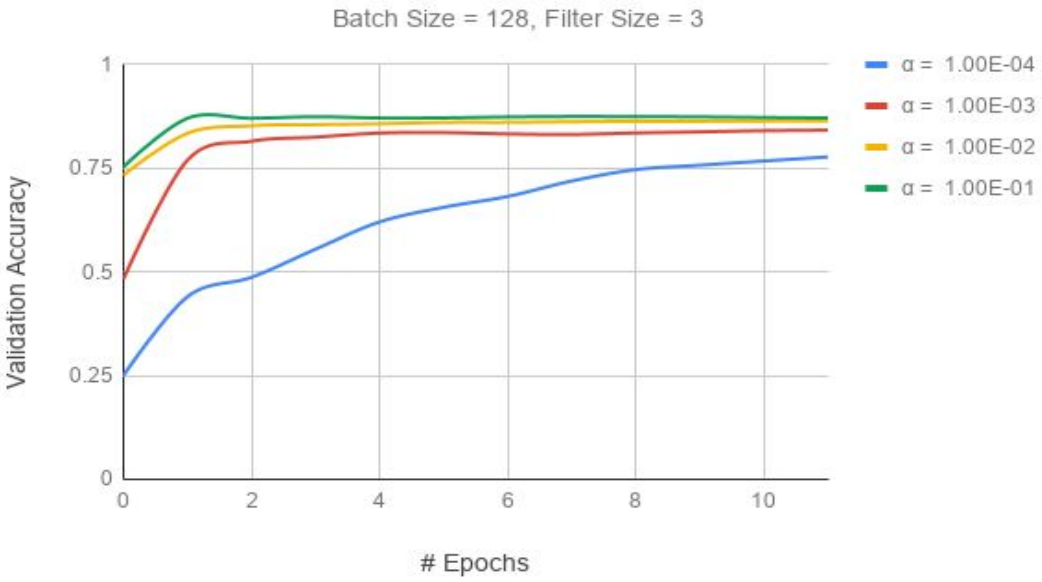
Experiments and Results

Varying Learning Rates

The network was trained using the following learning rates during training: 1e-4, 1e-3, 1e-2, 1e-1. The network was trained for a total of 12 epochs. The highest learning rate of 1e-1 results in the best performance during training on the validation set with an accuracy of 87.139% and the lowest learning rate of 1e-4 we obtain a validation accuracy of 77.77%. With decreasing learning rate the performance on the validation set decreases

and with increasing learning rate, the learning curve gets steeper resulting in a faster convergence.

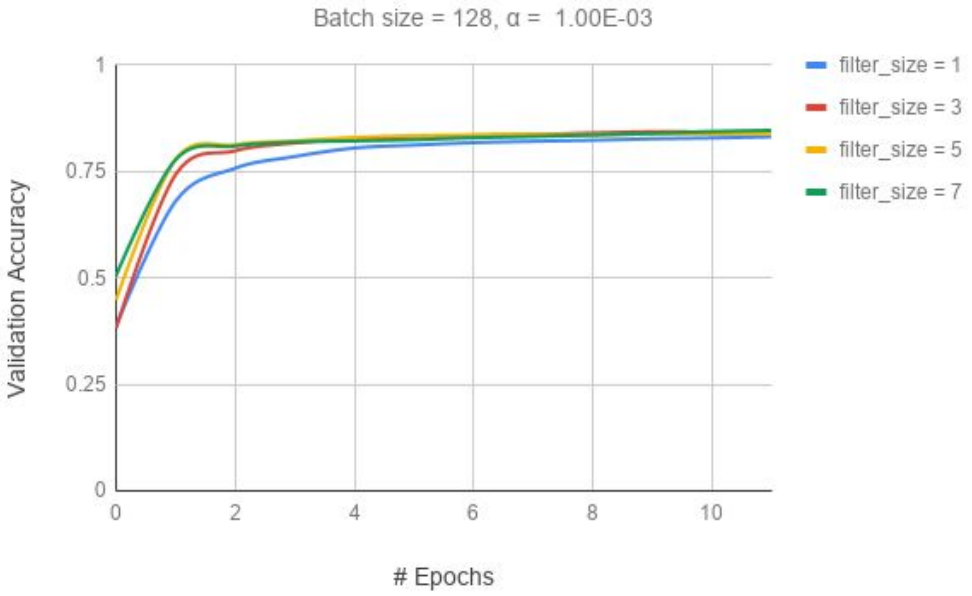
Learning rate	Validation Accuracy	Test Accuracy	Test Error
1e-4	77.77%	79.83%	0.7578
1e-3	84.02%	89.46%	0.4459
1e-2	86.42%	89.01%	0.313
1e-1	87.139%	92.76%	0.307



The figure above shows the number of epochs (epochs = 12) and the resulting validation accuracy on the validation set for each training epoch

Varying Filter Size

In CNN the size of the filter is a hyperparameter. The graph shows the influence of different filter sizes. Smaller filters (3x3 or 5x5) usually perform better than larger filters.



Filter_size	Validation Accuracy	Test Accuracy	Test Error
1	83.18%	86.89%	0.337
3	84.30%	87.94%	0.318
5	83.92%	89.46%	0.314
7	84.68%	87.33%	0.382

Random Search

Random search is the algorithm of drawing hyperparameter assignments from that process and evaluating them. It randomly samples the search space and evaluates sets from a specified probability distribution. Tuning hyperparameter is most important in training neural network and also is is time consuming procedure. HpBandSter

implements recently published methods for optimizing hyperparameters of machine learning algorithms.

The CNN implementation works well in cnn_mnist_solution.py, but the same implementation when called in random_search.py gives an error *"TypeError: 'float' object cannot be interpreted as an integer"*. Due to this reason, random search couldn't be implemented.

References:

1. https://automl.github.io/HpBandSter/build/html/quickstart.html
2. http://machinelearninguru.com/deep_learning/tensorflow/neural_networks/cnn_classifier/cnn_classifier.html
3. https://github.com/automl/HpBandSter/blob/master/hpbandster/examples/example_5_keras_worker.py
4. https://www.tensorflow.org/tutorials/estimators/cnn#getting_started