Deep Learning Lab - Exercise 1

Baohe Zhang

27.10.2018, Uni Freiburg

1 Introduction

In this exercise we implement a multi-layer perception model, which is also called neural network and use it to classify on the most commonly used dataset - MNIST. Besides that, we will try to tune the hyperparameters of our model and implement different optimization and regularization techniques to achieve a good result on test set. The code is based on the framework provided by AIS Lab of University Freiburg.[1]

2 Neural Network

2.1 Concept

A neural network consists of three parts: input layer, hidden layer, output layer. For hidden layer, it usually also contains two parts: linear function and activation function, which is designed to add non-linearity to fit data. In our implementation, we provides Linear, Sigmoid, Relu and Tanh, four types of activation functions and use softmax and cross-entropy as output layer.

2.2 Result

Our resulting model is a neural network with 3 hidden layer. The first layer has 300 units and the second layer has 64 units. Their output will all go through a Relu activation function. The last layer has 10 units as output layer without activation function. Finally we use softmax to produce the output and calculate the cross-entropy loss. Furthermore, we use Dropout[2] as regularization method and use Adam[3] as optimizer. By training on 60000 samples with 20% dropout, our model achieves 1.76% on test set. The error-epochs curves are in Figure 1.

[1] Andreas Eitel. *Deep Learning Lab*. `https://github.com/aisrobots/dl-lab-2018`. 2018.

[2] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[3] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

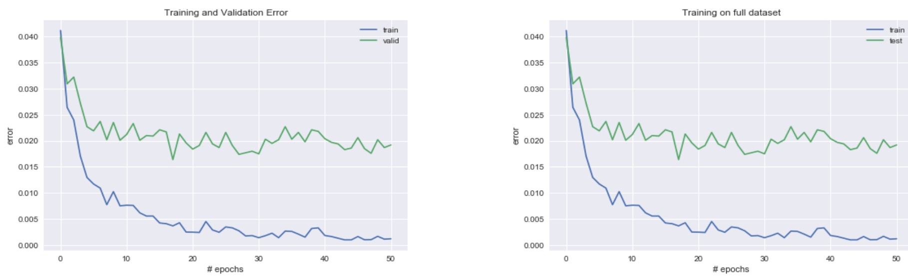

Figure 1: (a) Train-Valid Curve (b) Train on full dataset

3 Optimization and Regularization

Optimization and regularization also play important roles in neural network based problem. Here we will briefly discuss about the influence of them.

3.1 Optimization

Optimization has huge influence on the training procedure. Choosing one suitable optimization method will significantly reduce the epochs we need to ensure that our model has converged. Here we compare three different types of methods:

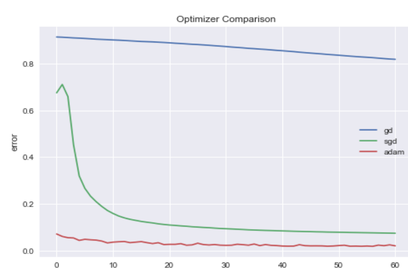

Figure 2: Learning Curve of Three Optimizers

3.2 Regularization

regularization is necessary to avoid overfitting in complex neural network. In our implementation, we provide L2-Regularization[4] and Dropout to use. We train our model on a subset of MNIST with 1000 training data. We can see, regularization will reduce the performance on the training data, but perform better in validation set. Figure 3 shows the impact of regularization:

[4]Anders Krogh and John A Hertz. "A simple weight decay can improve generalization". In: *Advances in neural information processing systems*. 1992, pp. 950–957.

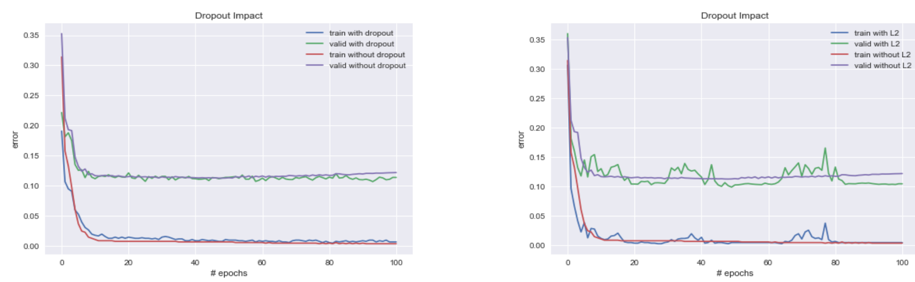

Figure 3: (a) Dropout (b) L2-Regularization

4 Conclusion

With this experiment we can find that neural network achieves a very good result on MNIST task. Moreover, choosing good optimizer will significantly reduce the training time, and for task with less training data, it's better to add regularization techniques into implementation. But it will also need user spending more time on tuning the hyperparameters.