# MLP Coursework 1: Learning Algorithms and Regularization

s1866666
October 2018

## Abstract

This paper presents the results of primitive experiments and analysis on RMSProp, Adam, learning rate algorithms developed based on stochastic gradient, as well as cosine annealing, a learning rate scheduling method which adjusts learning rate during the training. The experiments were carried out on EMNIST dataset to investigate the performance in terms of test accuracy of these models.

## 1. Introduction

The goal is to investigate several methods in machine learning in terms of efficiency and accuracy. Specifically, How well do advanced learning rules like RMSprop and Adam perform compared to basic SGD? Does the employment of cosine annealing scheduler improves their performance? And is Adam with weight decay perform better than with L2 regularization. Section 2 presents a set of baseline system in using stochastic gradient descent (SGD) learning rule. Section 3 introduces RMSProp and Adam. Section 4 presents the cosine annealing scheduler with and without restarts. Section 5 presents a comparison between regularization and weight decay with Adam.

All the experiments are carried out on EMNIST dataset (Cohen et al., 2017), a well-labeled balanced dataset including images of handwritten letters and digits. A reduced dataset including 47 classes, instead of 62 classes (26 upper case letters, 26 lower case letters and 10 digits) is used in our experiments. This is because there are 15 letters which have small distinctions between their upper case and lower case, and thus their upper- and lower-case labels are merged. These letters are: C, I, J, K, L, M, O, P, S, U, V, W, X, Y, Z.

The training set contains 100,000 samples. The validation set and test set each contains 15,800 samples. Each sample consists of 784 float numbers representing the pixel values of a 28 by 28 image.

## 2. Baseline systems

The baseline systems are contructed using SGD. The networks comprise from 2-5 affine layers intersected by Relu layers. That is, for example, a 3-layer network includes 3 affine layers and 2 Relu layers seperating affine layers. Each hidden layer has 100 units in our experiments. The final error of the model is evaluated by cross entropy with softmax because the task of our model is classification. The optimal learning rate for each network was searched in the log scale. Shown in the figure 1 is the best validation accuracy achieved by each network in 100 epochs of training with different learning rate. Each model performed 4 trials of training with different initialization.

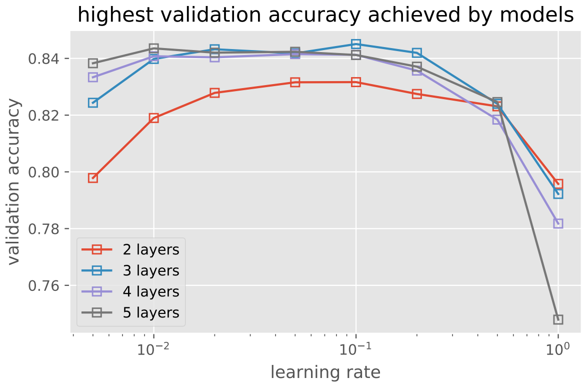

*Figure 1.* The average maximum validation accuracy of networks with 2-5 layers using various learning rate in 100 epochs of training. 4 trials of experiment were performed for each learning rate.

| NUM LAYERS | LEARNING RATE | EPOCHS | TEST ACCURACY |
|---|---|---|---|
| 2 | 0.1 | 47.50 ± 0.67 | 82.04% ± 0.11% |
| 3 | 0.1 | 21.25 ± 0.62 | 83.89% ± 0.26% |
| 4 | 0.05 | 34.75 ± 1.15 | 83.04% ± 0.08% |
| 5 | 0.01 | 96.25 ± 1.29 | 83.06% ± 0.14% |

*Table 1.* Test accuracy and epochs of the best SGD systems of from 2-5 layers. 4 trials were performed on each model

In general, 2-layer networks perform poorlier than others in cross-validation in terms of both accuracy and number of epoch. For each network, the model with optimal learning rate was selected to compare with other networks. The details are shown in the table 1. We see that 3-layer network outperforms others with higher test accuracy and less epochs.

## 3. Learning algorithms – RMSProp and Adam

RMSProp and Adam are methods for SGD with an adaptive learning rate for each parameter. Specifically, RMSProp (Hinton et al.) updates parameters as shown in algorithm 1: where $d_i$ is the gradient of the error function with respect to a weight $w_i$ at update time $t$, $S_i$ is the sum of moving average of squared gradients. There are three hyperparameters involved in the algorithm: learning rate $\alpha$, decay rate $\beta$, and $\epsilon$, which is a small number used to avoid division by zero. Using similar notations, Adam (Kingma & Ba, 2014) updates parameters as shown in algorithm 2: where $\hat{m}_i$ and $\hat{v}_i$ are bias corrected version of the vectors $m_i$, $v_i$, as the first few steps of updates are biasd at 0 due to the initialization. Adam involves four parameters: $\alpha$, $\alpha_1$, $\beta_2$ and $\epsilon$.

**Algorithm 1** RMSProp learning rule

```
$S_i(0) = 0$
$S_i(t) = \beta S_i(t-1) + (1-\beta) d_i(t)^2$
$\Delta w_i(t) = \frac{\alpha}{\sqrt{S_i(t)} + \epsilon} d_i(t)$

$w_i(t+1) = w_i(t) + \Delta w_i(t)$
```

**Algorithm 2** Adam learning rule

```
$m_i(0) = 0$
$v_i(0) = 0$
$m_i(t) = \alpha_1 m_i(t-1) + (1-\alpha_1) d_i(t)$
$v_i(t) = \beta_2 v_i(t-1) + (1-\beta_2) d_i(t)^2$
$\hat{m}_i(t) = \frac{m_i(t)}{1-\alpha_1^t}$
$\hat{v}_i(t) = \frac{v_i(t)}{1-\beta_2^t}$
$\Delta w_i(t) = \frac{\alpha \hat{v}_i(t)}{\sqrt{\hat{m}_i(t)} + \epsilon}$

$w_i(t+1) = w_i(t) + \Delta w_i(t)$
```

Instead of grid search, hyperparameters were optimised seperately in cross-validation starting from the most important ones to the least important ones. This was based on the assumption that each hyperparameters are independent to each other, which is often not the case. However, optimising hyperparameters seperately is less expensive. Shown in figure 2 is the highest validation accuracy achieved by RMSProp using various learning rate. The highest accuracy is obtained by setting learning rate 0.001, and thus we continued to optimise decay rate $\beta$ with this optimal learning rate unchanged. It turned out that $\beta = 0.99$ was optimal. 4 trials were performed for each hyperparameters.

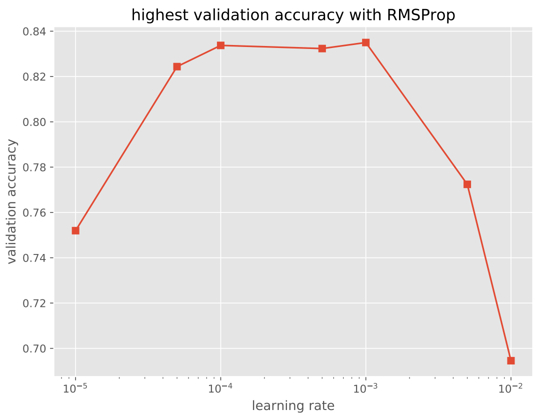

*Figure 2.* Average maximum validation accuracy with RMSProp using various learning rate during 100 epochs of training.

| ALGORITHM | NUM EPOCHS | TEST ACCURACY |
|---|---|---|
| SGD | 21.25 ± 0.62 | 83.16% ± 0.26% |
| RMSProp | 14.25 ± 1.03 | 82.57% ± 0.05% |
| Adam | 17 ± 1.91 | 82.36% ± 0.14% |

*Table 2.* Test accuracy and epochs of the best systems of SGD, MRSProp and Adam

The similar procedure was applied to finding the optimal hyperparameters in Adam. The learning rate was optimised first, and then $\alpha$ and $\beta$. The best values of the hyperparameters are $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999$. Shown in the table 3 is the test accuracy achieved by optimal models implementing RMSprop and Adam, listed together with the baseline system.

SGD outperforms both RMSProp and Adam in out experiment in terms of test accuray. However, RMSProp and Adam both reach its maximum validation accuracy faster than SGD does. Compared with Adam, RMSProp has slightly higher test accuracy.

## 4. Cosine annealing learning rate scheduler

The cosine annealing with restarts scheduler (Loshchilov & Hutter, 2017)updates at each epoch the learning rate multiplier $\eta_t$ which is multiplied by the initial learning rate $\alpha$ to otain the actual learning rate. $\eta_t$ is updated as follows using Loshchilov and Hutter's notations:

$$\eta_t = \eta_{min}^{(i)} + 0.5(\eta_{max}^{(i)} - \eta_{min}^{(i)})(1 + \cos(\pi T_{cur}/T_i))$$

where $T_{cur}$ is the number of epochs since the last restart, $i$ is the number of run. During the $i$th run the scheduler constantly decreases the learning rate until the the $(i+1)$th

| ALGORITHM | NUM EPOCHS | TEST ACCURACY |
|---|---|---|
| SGD | 21.25 ± 0.62 | 83.16% ± 0.26% |
| ADAM | 17 ± 1.91 | 82.36% ± 0.14% |
| SGD-COS | 25.5 ± 0.51 | 82.91% ± 0.06% |
| ADAM-COS | 16.75 ± 1.85 | 82.10% ± 0.15% |
| SGDR | 22.75 ± 0.23 | 83.45% ± 0.39% |
| ADAMR | 23.25 ± 0.62 | 83.48% ± 0.24% |

*Table 3.* Test accuracy and epochs of the best systems of SGD, Adam using cosine annealing scheduler with and without warm restarts. 4 trials were performed for each algorithm

warm restart is triggered when $T_{cur} = T_i$. Multiplying $T_i$ by an extension factor $T_{m}ult$ can adjust the epoch peroid of restarts for each run, and the similar procedure can be performed on $\eta^{(i)}_{min}$, $\eta^{(i)}_{max}$. In our experiments, only $T_i$ and $\eta^{(i)}_{max}$ are adjusted. Hence there are 5 hyperparameters considered: the initial number of epochs in a peroid $T_0$, the rate of expansion of the period epochs $T_{m}ult$, the minimum learning rate multiplier $\eta^{(i)}_{min}$, the initial maximum learning rate multiplier $\eta^{(0)}_{max}$ and the maximum learning rate discount factor $\eta_{max_{m}ult}$.

For a training of 100 epochs with warm restarts, we use $T_0 = 25$ and $T_{m}ult = 3$, hence there are 2 restarts in each training, that is, at 25 epoch and 100 epoch. For the experiments without warm restarts, multipliers are not an issue to be considered and $T_0 = 101$ is used to ensure the 100 epochs enjoy a whole iteration of cosine annealing. Tuning the other three hyperparameters one by one finally gave us the optimal hyperparameters $\eta^{(i)}_{min} = 0.1$, $\eta^{(0)}_{max} = 1$ and $\eta_{max_{m}ult} = 0.2$ for both Adam and SGD with restarts (AdamR, SGDR). For the versions without restarts (Adam-cos, SGD-cos), we find the optimal hyperparameters are $\eta^{(i)}_{min} = 0.1$, $\eta^{(i)}_{max} = 1$.

Shown in table 3 is the experiment results of applying cosine annealing scheduler with and without warm restarts on SGD and Adam learning rules. Applying the cosine annealing scheduler improves the performance of Adam by 0.1 0.3%, whereas dose not increase the performace of SGD in general. Compared to cosine annealing without warm restarts, we see that a scheduler with restarts performs better.

## 5. Regularization and weight decay with Adam

The difference of weight decay and L2 regularization on Adam is shown below: where $g_i(t)$ is the regularized gradient of a parameter $w_i(t)$, and $\eta_t$ is the learning rate multiplier at step t as discussed in section 4. Weight decay applies the decay term $\omega w_i(t)$ directly to the final updates of the parameters and thus avoids the coupling of $\omega$ with other hyperparameters $\alpha$ and $\beta$. This makes it possible for us to optimize decay rate more independently than it was with L2 regularization. Searching for the optimal decay rate $\omega$ of L2 regularization and weight decay on Adam using constant learning rate scheduler gave us $\omega = 0.001$ for L2 regularization and $\omega = 0.0001$ for weight decay.

**Algorithm 3** Adam with weight decay and L2 regularization

$$\begin{aligned}
g_i(t) &= d_i(t) + \omega w_i(t)\\
m_i(0) &= 0\\
v_i(0) &= 0\\
m_i(t) &= \beta_1 m_i(t-1) + (1-\beta_1) g_i(t)\\
v_i(t) &= \beta_2 v_i(t-1) + (1-\beta_2) g_i(t)^2\\
\hat{m}_i(t) &= \frac{m_i(t)}{1-\beta_1^t}\\
\hat{v}_i(t) &= \frac{v_i(t)}{1-\beta_2^t}\\
\Delta w_i(t) &= \eta_t\left(\frac{\alpha \hat{v}_i(t)}{\sqrt{\hat{m}_i(t)}+\epsilon} + \omega w_i(t)\right)
\end{aligned}$$

$$w_i(t+1) = w_i(t) + \Delta w_i(t)$$

| ALGORITHM | NUM EPOCHS | TEST ACCURACY |
|---|---|---|
| ADAM | 17 ± 1.91 | 82.36% ± 0.14% |
| ADAM-L2 | 28.5 ± 0.83 | 84.18% ± 0.03% |
| ADAMW | 88.75 ± 1.10 | 84.65% ± 0.27% |
| ADAMW-COS | 58 ± 1.85 | 84.73% ± 0.22% |
| ADAMWR | 73.5 ± 1.85 | 84.92% ± 0.07% |

*Table 4.* Test accuracy and epochs of the best systems of Adam using regularization/weight decay and different schedulers

Table 4 shows the test accuracy achieved by each algorithm with respective optimal hyperparameters. We see that the Adam versions with L2 regularization/weight decay perform better than normal Adam by  2% of accuracy. Adam with weight decay outperforms L2 regularization by about 0.5%. Using the cosine annealing scheduler improved the accuracy by  0.1% without warm restarts and  0.3% with warm restarts.

## 6. Conclusions

Based on the experiments, we conclude that Adam and RMSProp do not perform better than SGD without the use of scheduler and regularization/weight decay. Cosine annealing scheduler improves the performance of Adam and RMSProp, while is not so effective to SGD. In particular, cosine annealing with warm restarts improves learning more than cosine annealing without warm restarts does. Adam with weight decay performs better than Adam with L2 regularization. Overall, AdamWR is the best algorithm with a test accuracy of 84.92% ± 0.07%.

## References

Cohen, Gregory, Afshar, Saeed, Tapson, Jonathan, and van Schaik, André. EMNIST: an extension of MNIST to handwritten letters. feb 2017. URL http://arxiv.org/abs/1702.05373.

Hinton, Geoffrey, Srivastava, Ni@sh, and Swersky, Kevin. Neural Networks for Machine Learning Lecture 6a Overview of mini-Âŋ-batch gradient descent. Technical report. URL http://www.cs.toronto.edu/{~}tijmen/csc321/slides/lecture{_}slides{_}lec6.pdf.

Kingma, Diederik P. and Ba, Jimmy. Adam: A Method for Stochastic Optimization. dec 2014. URL http://arxiv.org/abs/1412.6980.

Loshchilov, Ilya and Hutter, Frank. Fixing weight decay regularization in Adam. *arXiv preprint arXiv:1711.05101*, 2017. URL https://arxiv.org/abs/1711.05101.