MLP Coursework 1: Learning Algorithms and Regularization

s1678478

Abstract

In this coursework, an extended version of MNIST database (EMNIST) is used. The database contains handwritten letters, both upper and lower case, and handwritten digits. Several different machine learning algorithms are used here to classify the handwritten letters and digits, including Stochastic Gradient Decent (SGD), RMS-Prop and Adam Learning Algorithms. In the later part of the coursework Adam learning rule has been further explored by applying different optimization methods to it. The three main optimization methods used are Cosine Annealing Learning Rate Scheduler, L2 Regularization and Weight Decay. Results from all the different learning rules and optimization methods are compared regarding the accuracy on successfully classify the handwritten digits and letters from EMNIST database.

1. Introduction

The aim of this coursework is to explore RMS-Prop and Adam learning algorithms. The motivation for the coursework comes from the paper by Loshchilov and Hutter [2017] on using L2 regularization and weight decay as Adam optimizer.

All the experiments are carried out on the EMNIST Balanced dataset which contains 47 potential classes. There are 10 digits, 26 lower case letters, 26 upper case letters where the lower-case and upper-case labels are merged for the following letters: C, I, J, K, L, M, O, P, S, U, V, W, X, Y, Z. The conversion process used results in centred images of dimension 28×28. The EMNIST dataset is split into training set, validation set and test set, each of them takes up 75%, 12.5% and 12.5% of the whole database respectively. The batch size for all three sets is 100 and there are 1000 datapoints in the training set, 158 datapoints in both the validation set and test set. The number of epochs is set to 100 for all the experiments. The input dimension is 784 (28×28) which is the number of pixels contain in each image, the output dimension is 47 which corresponding to the 47 classes in the EMNIST and the hidden layer dimension is set to 100 ReLU hidden units per layer.

The first task is to set a baseline system on EMNIST using Stochastic Gradient Decent (SGD). Second, we implement the RMSProp and Adam learning rule. Test them on the test set and compare their accuracy with the baseline. Then we investigate optimization methods presented by Loshchilov and Hutter [2017] involving using a scheduler called Cosine Annealing Learning Rate Scheduler. After the implementation of the scheduler, we apply it to both SGD and Adam to compare their performance. Last, followed the idea in Loshchilov and Hutter [2017], we explore the two different Adam optimizer using L2 regularization and weight decay. The experiments are carried out to compare between

- L2 regularization vs. weight decay
- constant learning rate vs. cosine annealing schedule
- no restarts in the scheduler vs. use of a warm restart

SGD ACCURACY				
LEARNING RATE		0.1	0.05	0.01
NUMBER	2	0.8375	0.8428	0.8382
OF	3	0.8421	0.8429	0.8428
HIDDEN	4	0.8410	0.8431	0.8410
LAYERS	5	0.8418	0.8385	0.8431

Table 1. Classification accuracies for different number of hidden layers using SGD on validation set

2. Baseline systems

In order to set up a baseline system for the EMNIST database, we recording the performance of models with 2-5 hidden layers and select the models with highest accuracy on the test set. Total of 3 groups of experiments are carried out on validation set, each group contains 4 experiments. The first group using learning rate of 0.1 on each of the model with 2-5 hidden layers. The second and third group using learning rate of 0.05 and 0.01 respectively. The highest accuracy of each experiment is shown in table 1. From which we can observe that there is no distinct difference among the results. Thus, the models with the highest accuracy from each experiment group are selected and then their performance are compared on the test set as shown in table 2. The results show that the model with 3 hidden layers with learning rate of 0.05 achieves the highest accuracy on the test set using SGD learning rule. Figure 1 shows the performance of the baseline system over 100 epochs, the error is at its minimum around epoch 20 and starts increasing after that. This implys that the model starts to be overfitting after epoch 20. The baseline for the database is then the highest accuracy over 100 epochs which is 0.8340.

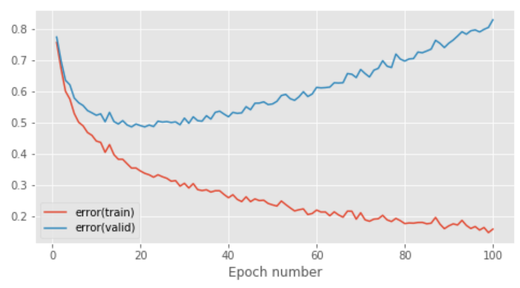

Figure 1. Error of SGD baseline system on validation set.

NUMBER OF HIDDEN LAYER	LEARNING RATE	VALIDATION ACC	TEST ACC
3	0.1	0.8421	0.8338
3	0.05	0.8429	0.8340
5	0.01	0.8431	0.8339

Table 2. Different models using SGD on test set

3. Learning algorithms – RMSProp and Adam

The RMS-Prop can be viewed as a stochastic gradient descent version of RProp normalized by a moving average of the squared gradient. The pseudocode for RMS-Prop is shown in Algorithm 1. The recommend value for β is 0.9 which is used for all RMS-Prop model in this coursework.

Adam can be viewed as a variant of RMS-Prop with momentum. The exponential moving average of the gradient, m_t, is the estimate of the 1st moment which is also the mean of the gradient. The squared gradient, v_t, is the estimates of the 2nd raw moment which is the uncentered variance of the gradient. Both m_t and v_t are updated during the process and hyperparameter β_1 and β_2 are used to control the exponential decay rates of them respectively. The pseudocode for Adam is shown in Algorithm 2. The recommend value for β_1 is 0.9 and for β_2 is 0.999 which are used for all Adam model in this coursework.

The major difference between RMS-Prop and Adam is that RMS-Prop updates its parameter using a momentum on the rescaled gradient where Adam using a running average of the first and second moment of the gradient. In theory, the Adam learning rule should achieve a higher accuracy on test set compared to RMS-Prop.

Three different learning rates are used to test the performance of RMS-Prop and Adam. The accuracy of the model with 3 hidden layers perform on validation set are listed in table 3. Then the appropriate learning rates for RMS-Prop and Adam are obtained which iare 5×10^{-4} and 1×10^{-3} respectively. Then the behaviour of the two learning algorithms are compared on test set. Figure 2 and figure 3 shows the error of the model using Adam learning rule and

LEARNING RATE	1E-3	5E-4	1E-4
RMS-PROP	0.8372	0.8395	0.8377
ADAM	0.8379	0.8125	0.8103

Table 3. Accuracy of Adam and RMS-Prop on validation set

LEARNING RULE	LEARNING RATE	ACCURACY
SGD	0.05	0.8340
ADAM	0.001	0.8298
RMS	0.0005	0.8294

Table 4. Accuracy of Adam and RMS-Prop on validation set

RMS-Prop learning rule perform on validation set respectively. The model using Adam complete learning process around epoch 25 and starts to be overfitting, the error rate is below 0.7 at epoch 100. For RMS-Prop the learning process is completed around epoch 18 and after that the error rate starts to grow very quickly which exceed 1.0 at epoch 100.

Table 4 conclude the results and compare them to the accuracy achieved by SGD learning rule which is also the baseline of the dataset. The performance of Adam is better than RMS as expected. The performance of SGD is better than both that of Adam and RMS-Prop.

4. Cosine annealing learning rate scheduler

As we can see from the previous experiment, change the learning rate have big influence on the results. It is important to decide how to set the learning rate during the learning process. Using a scheduler can help us achieve adjusting the learning rate from batch to batch or from epoch to epoch. There are two approach, one is early stopping approaches which is very useful when the training set is very large. The training process is stopped when no improvement on accuracy can be observed, this can help us save computational resources. Another approach is to assume a fixed number of training epochs and optimize the learning rate within this range.

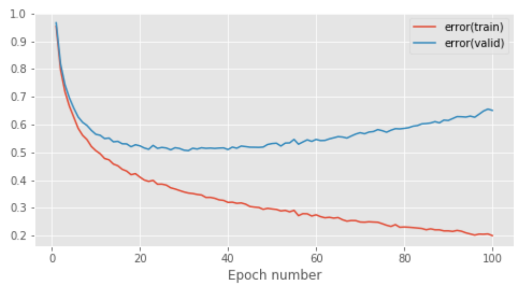

Figure 2. Error and accuracy plot of Adam learning rule on test set.

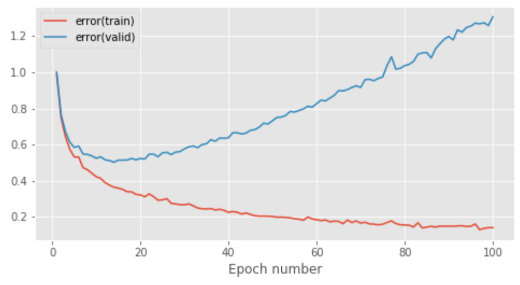

Figure 3. Error and accuracy plot of RMS-Prop learning rule on test set.

Algorithm 1 RMS-Prop Learning Rule

```
**given:** $\alpha, \beta$(decay rate), $\epsilon$
**initialize:** parameter vector $w \in \mathbb{R}^n$, weighted average vector $v_{dw} \leftarrow 0$
**repeat**
    $dw \leftarrow$ corresponding gradient on current batch
    $v_{dw} \leftarrow \beta \cdot v_{dw} + (1-\beta) \cdot dw^2$
    $w \leftarrow w - \alpha \cdot (dw/\sqrt{v_{dw}} + \epsilon)$
**until** stopping criterion is met
**return** optimized parameters $w$
```

Loshchilov and Hutter [2017] present a learning rate scheduler called cosine annealing scheduler algorithm which belong to the second approach. The epoch number is set to 100 in this coursework. This scheduler contains a cosine function which allow the learning rate to decay through time and there are periodic restart points along the process. At each restart point, the learning rate is increased and after which is decreased again. Hyperparameters are used to control when a restart happens. The implemented fuction is shown below:

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

Within the i-th run, the value of the learning rate, η_t, decays according to a cosine annealing scheduler for each batch. The range of learning rate is provided by $\eta_m in^{(i)}$ and $(\eta_{max}^{(i)}$. T_{cur} holds the value of epoch number that has been performed since the last restart which is set to zero at each restart point. T_i is the total epochs per period. The restart condition is $T_{cur} = T_i$. In order to achieve good anytime performance, initially T_i can be set to a small number and multiply it by a factor of T_{mult} at every restart. A maximum learning rate discount factor is added to decrease the learning rate at each restart point. This can potentially improve the performance.

This scheduler can be applied to both Adam and SGD. The baseline case for each algorithm use a constant learning rate and no scheduler. Here the systems with best performance from the previous experiments are selected as baseline. Thus, the baseline for Adam and SGD is 0.8298 and 0.8340 here. The total epochs per period T_i is set to 25, the expansion factor T_{mult} is set to 3. The experiments

Algorithm 2 Adam Learning Rule

```
**given:** $\alpha, \beta_1, \beta_2, \epsilon$
**initialize:** time step t←0, parameter vector $w_{t=0} \in \mathbb{R}^n$, first moment vector $m_{t=0} \leftarrow 0$, second moment vector $v_{t=0} \leftarrow 0$
**repeat**
    t ← t+1
    $\nabla f_t(w_{t-1}) \leftarrow$SelectBatch($w_{t-1}$)
    $g_t \leftarrow \nabla f_t(w_{t-1})$
    $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$
    $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
    $\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$
    $\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$
    $w_t \leftarrow w_{t-1} - \alpha \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$
**until** stopping criterion is met
**return** optimized parameters $w_t$
```

LEARNING RULE	RESTART	LEARNING RATE	ACCURACY
ADAM	NO	0.0001-0.001	0.8292
ADAM	YES	0.0001-0.001	0.8315
SGD	NO	0-0.05	0.8335
SGD	YES	0-0.05	0.8363

Table 5. Adam and SGD with cosine annealing scheduler

are performed on SGD with/without restart and on Adam with/without restart. The error rate plot for Adam learning rule is shown by Figure 4. There is a bump in the curve in both plot with restart. The bump is where the restart happens. In the Adam no restart case, the overfitting is much obvious compared with the case with restart. The results for the 4 experiments mentioned are listed in table 5. It is clear that for both Adam and SGD, the system with restart achieves higher accuracy than systems with no restart.

5. Regularization and weight decay with Adam

Loshchilov and Hutter [2017] **?** proposed that L2 regularization and weight decay have different impact on the Adam optimizer. They are the same when applying to SGD learning rule, but for other learning rules, the regularization hyperparameter and the learning rate hyperparameter can interact with each other. And this may lead to worse accuracy.

In the experiment, the L2 regularization for Adam is applied through adding a penalty to Affinelayers. Another learning rule is implemented for weight decay for Adam which is called Adam learning Rule with Weight Decay (AdamW). The AdamW and the original Adam differs in the functions they used to update the parameters. The equation in AdamW is shown below:

$$w_t \leftarrow w_{t-1} - \eta_t(\alpha \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon) + c w_{t-1}),$$

where η_t is the scheduler multiplier which can be fixed or

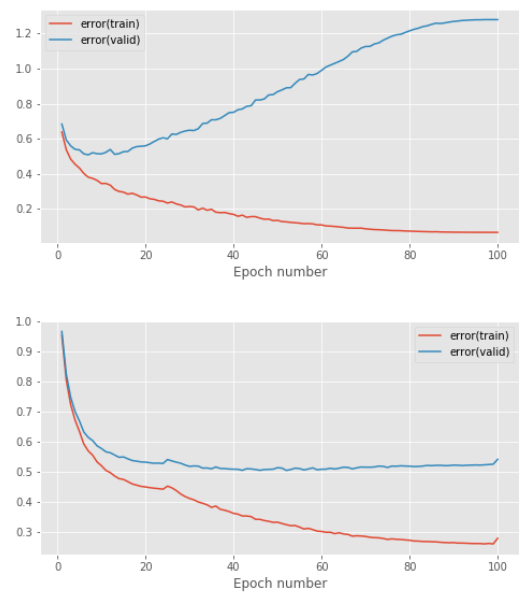

Figure 4. Error rate of Adam learning rule no restart(upper)/with restart(lower) on validation set.

LEARING RULE	LEARNING RATE	ACC
L2	0.001	0.8401
AdamW	0.001	0.8480
AdamW+sch	0-0.001	0.8382
AdamW+restart	0-0.001	0.8377

Table 6. L2 regularization and AdamW

change according to the scheduler and c is the weight decay coefficient.

There are total of 4 experiments carried out, L2 regularization with constant learning rate, AdamW with constant learning rate, AdamW with scheduler no restart and AdamW with restart. The error rate of models with constant learning rate are shown in Figure 5. The constant learning is set to 0.001. We can see in L2 regularization, the learning process completed around epoch 80 which is slower than that of the AdamW which finished around 40. Thus, the AdamW can learn faster than L2 regularization.

The accuracy of the models perform on test set are listed in Table 6. The accuracy of AdamW with constant learning rate 0.001 is higher than that of the L2 regularization with the same learning rate. Thus, the hypothesized Loshchilov and Hutter proposed is right within this coursework, the weight decay optimizer can be a solution to L2 regularization performed on Adam. Weight decay method using the separate hyperparameters to control learning rate which is proved to be useful here.

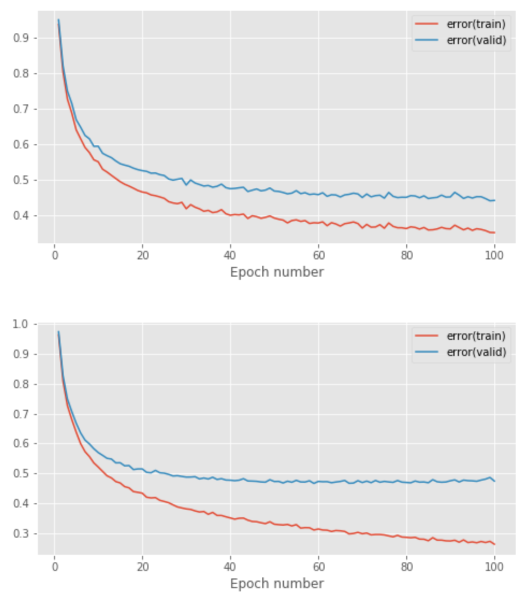

Figure 5. Error rate of L2(upper)/AdamW(lower) on validation set.

6. Conclusions

The baseline of the EMNIST dataset is the accuracy achieved by model with 3 hidden layers and learning rate of 0.05. The performance of Adam and RMS-Prop learning rules are below the baseline where Adam learning rule is slightly better than the RMS-Prop. After applying cosine scheduler with no restart to Adam and SGD, the accuracy dropped compared with the original Adam and SGD. However, after applying the scheduler with restart to both Adam and SGD, the accuracy not only surpass the results without restart but also the original Adam and SGD learning rule. The results illustrate that by applying cosine annealing with warm restart scheduler, the accuracy of the system can be increased.

Among all the experiments conducted in this coursework, the highest accuracy is achieved by AdamW with learning rate 0.001, followed by L2 regularization, AdamW with no restart and AdamW with restart. The highest accuracy is 0.8480 compared with the baseline 0.8340 in section 1, there is a 1.68% increase. All the 4 experiments conducted in section 5 regarding L2 regularization and AdamW achieve higher accuracy than the baseline system.

References

[1] Ilya Loshchilov, Frank Hutter. *FixingWeight Decay Regularization in Adam* arXiv preprint arXiv:1711.05101, 2017. URL http://arxiv.org/abs/1711.05101