MLP Coursework 1

s1862323

Abstract

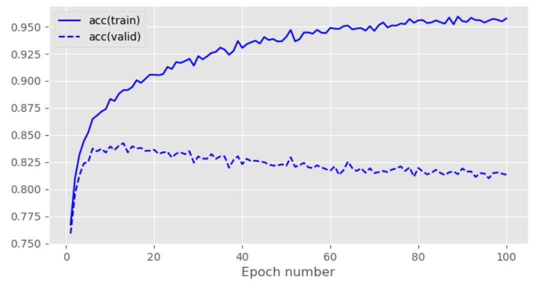
In this report we study the problem of overfitting, which is the training regime where performance increases on the training set but decrease on validation data. **[Because the noise or random fluctuations on the training set learned as concepts by the model. But these concepts will negatively impact the models ability to generalize the new data. In addition, the validation accuracy will not increase, and the generalization performance will decrease.]** . We first analyse the given example and discuss the probable causes of the underlying problem. Then we investigate how the depth and width of a neural network can affect overfitting in a feedforward architecture and observe that increasing width and depth **[As the increase of the width and depth, the complexity of model will increase. The wider model will have more parameter or probabilities to learn the noise or random fluctuations;the deeper model will have more layer to increase the weight factors of the noise or random fluctuations. Eventually, increasing of width and depth will negatively impact the model ability to generalize the new data, and causing the overfitting.]** . Next we discuss why two standard methods, Dropout and Weight Penalty, can mitigate overfitting, then describe their implementation and use them in our experiments to reduce the overfitting on the EMNIST dataset. Based on our results, we ultimately find that **[As the width increase, the training accuracy of the model will increase and the training error of the model will decrease. However, This positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. As the width increase, the validation accuracy of the model will not increase and the validation error of the model will increase(supposed to decrease). In addition, as the depth increase, the training accuracy and the training error of the model almost are the same. However,as the depth increase, the validation error of model will increase(supposed to decrease). It illustrates the noise or random fluctuations on the training set learned as concepts by the model because of the deeper layer and the wider layer. But these concepts will negatively impact the models ability in the validation set, and resulting in more severe overfitting.]** . Finally, we briefly review a technique that studies use of weight decay in adaptive gradient algorithms, discuss its findings, and conclude the report with our observations and future work. Our main findings indicate that **[Firstly, on the EMINIST dataset, simply adding the width and depth will not make the performance of the model better. Although the training accuracy and training error will performance better, this positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. Moreover, because the noise or random fluctuations on the training set learned as concepts by the model. It will result in validation accuracy decrease, validation error increase, and overfitting. These will negatively influence the model performance and generalization ability. Secondly, increasing of width and depth will consume higher resources to process. Thirdly, the complexity of the model will increase, which is harder to regularize the model.About the Dropout and Weight Penalty, if the dropout value is extremely large, it will not help with the validation accuracy,but increasing the generalization gap. The optimal dropout value should be from 0.7 to 0.85. For the L1 and L2 penalty, if the value is small, it would contribute negative effect on the validation accuracy. The L1 and L2 penalty should be from 1e-3 to 5e-4. In general, L2 regularization is a better choice compared to L1 regularization. And the optimal width and depth should be 128ReLU and 1 depth in the early stopping condition otherwise it would negatively influence the model performance. Because the overfitting would occur.**

] .

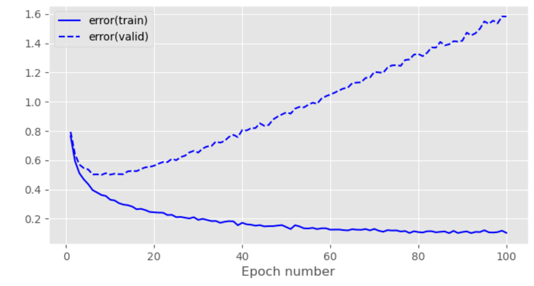
1. Introduction

In this report we focus on a common and important problem while training machine learning models known as overfitting, or overtraining, which is the training regime where performances increase on the training set but decrease on unseen data. We first start with analyzing the given problem in Fig. 1, study it in different architectures and then investigate different strategies to mitigate the problem. In particular, Section 2 identifies and discusses the given problem, and investigates the effect of network width and depth in terms of generalization gap (see Ch. 5 in Goodfellow et al. 2016) and generalization performance. Section 3 introduces two regularization techniques to alleviate overfitting: Dropout (Srivastava et al., 2014) and L1/L2 Weight Penalties (see Section 7.1 in Goodfellow et al. 2016). We first explain them in detail and discuss why they are used for alleviating overfitting. In Section 4, we incorporate each of them and their various combinations to a three hidden layer[1] neural network, train it on the EMNIST dataset, which contains 131,600 images of characters and digits, each of size 28x28, from 47 classes. We evaluate them in terms of generalization gap and performance, and discuss the results and effectiveness of the tested regularization strategies. Our results show that **[As the width increase, the training accuracy of the model will increase and the training error of the model will decrease. However, This positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. As the width increase, the validation accuracy of the model will not increase and the validation error of the model will increase(supposed to decrease). In addition, as the depth increase, the training accuracy and the training error of the model almost are the same. However,as the depth increase, the validation error of model will increase(supposed to decrease). It illustrates the noise or random fluctuations on the training set learned as concepts by the model because of the deeper layer and the wider layer. But these concepts will negatively impact the models ability in the validation set, and resulting in more severe overfitting.]** . In Section 5, we study a related work that studies weight decay in adaptive gradient algorithms.[2] Finally, we conclude our study in section 6, noting that **[Firstly, on the EMINIST dataset, simply adding the width and depth will not make the performance of the model better. Although the training accuracy and training error will performance better, this positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. Moreover, because the noise or random fluctuations on the training set learned as concepts by the model. It will result in validation accuracy decrease, validation error increase, and overfitting. These will negatively influence the model performance and generalization ability. Secondly, increasing of width and depth will consume higher resources to process. Thirdly, the complexity of the model will increase, which is harder to regularize the model.About the Dropout and Weight Penalty, if the dropout value is extremely large, it will not help with the validation accuracy,but increasing the generalization gap. The optimal dropout value should be from 0.7 to 0.85. For the L1 and L2 penalty, if the value is small, it would contribute negative effect on the validation accuracy. The L1 and L2 penalty should be from 1e-3 to 5e-4. In general, L2 regularization is a better choice compared to L1 regularization. And the optimal width and depth should be 128ReLU and 1 depth in the early stopping condition otherwise it would negatively influence the model performance. Because the overfitting would occur.**

] .



(a) accuracy by epoch



(b) error by epoch

Figure 1. Training and validation curves in terms of classification accuracy (a) and cross-entropy error (b) on the EMNIST dataset for the baseline model.

2. Problem identification

Overfitting to training data is a very common and important issue that needs to be dealt with when training neural networks or other machine learning models in general (see Ch. 5 in Goodfellow et al. 2016). A model is said to be overfitting when as the training progresses, its performance on the training data keeps improving, while its is degrading on validation data. Effectively, the model stops learning related patterns for the task and instead starts to memorize specificities of each training sample that are irrelevant to

[1]We denote all layers as hidden except the final (output) one. This means that depth of a network is equal to the number of its hidden layers + 1.

[2]Instructor note: Omitting this for this coursework, but normally you would be more specific and summarise your conclusions about that review here as well.

new samples.

[On the EMINIST dataset, overfitting will make the training accuracy and training error performance better. But this positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. In addition, the overfitting will negatively impact the model performance and generalization ability. Moreover, overfitting will consume higher resources to process. Thirdly, the complexity of the model will increase, which is harder to regularize the model.] .

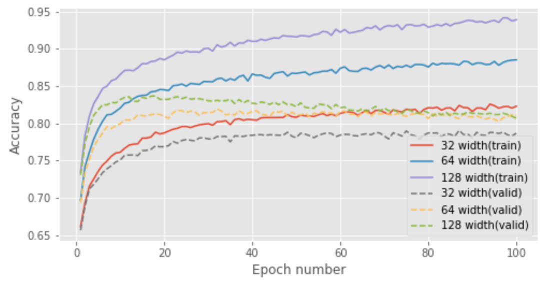
Although it eventually happens to all gradient-based training, it is most often caused by models that are too large with respect to the amount and diversity of training data. The more free parameters the model has, the easier it will be to memorize complex data patterns that only apply to a restricted amount of samples. A prominent symptom of overfitting is the generalization gap, defined as the difference between the validation and training error. A steady increase in this quantity is usually interpreted as the model entering the overfitting regime.

[Firstly, higher network capacity will make the model have more free parameters, it will easier to memorize complex data patterns that only apply to a restricted amount of samples. However, complex restricted data features will make the model lose generalization abilities and negatively influences the model performances. Hence, a higher network capacity will result in a higher chance of overfitting. In addition, higher network capacity means higher complexity of the model, which will make the model harder to regularize it and avoid or mitigate the overfitting.] .

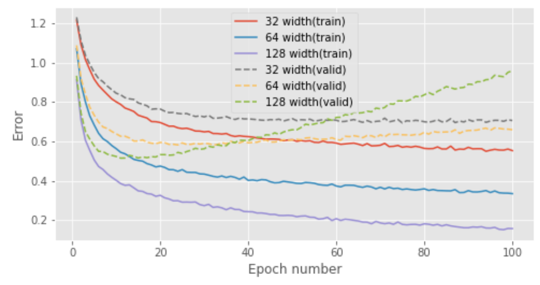
Fig. 1a and 1b show a prototypical example of overfitting. We see in Fig. 1a that **[The training accuracy gradually increase by the epoch. However, the validation accuracy increase gradually before 17 epoch, and then gradually decrease after 17 epoch. The maximum validation accuracy point is at 17 epoch, Thereafter, the trend shows the opposite direction. The magnitude starts to decrease. Also, in the velocities perspective, the velocities is positive and significant before 15 epoch, and then velocities decrease to around zero and then decrease to negative. Thereafter, the trend shows the opposite direction. Hence, 17 epoch should be the turning point, after this point, the model begin to be overfitting. The error of the model shows the similar situation. The training error gradually decrease by the epoch. In contrast, the validation error decrease at the first 15 epoch and then gradually increase(supposed to decrease) until the end. The minimum valid error point is at 15 epoch. Thereafter, the trend shows the opposite direction. The magnitude starts to increase. Also, in the velocities perspective, the velocities is negative and significant before 15 epoch, and then velocities increase to around zero and then increase to positive. Thereafter, the trend shows the opposite direction. It means overfitting occurs. Hence,the turning point is around 15 epoch, and the overfitting occurs around 15 epoch.]** .

# Hidden Units	Val. Acc.	Train Error	Val. Error
32	78.6%	0.554	0.706
64	81.1%	0.336	0.660
128	80.6%	0.158	0.960

Table 1. Validation accuracy (%) and training/validation error (in terms of cross-entropy error) for varying network widths on the EMNIST dataset.



(a) accuracy by epoch



(b) error by epoch

Figure 2. Training and validation curves in terms of classification accuracy (a) and cross-entropy error (b) on the EMNIST dataset for different network widths.

The extent to which our model overfits depends on many factors. For example, the quality and quantity of the training set and the complexity of the model. If we have sufficiently many diverse training samples, or if our model contains few hidden units, it will in general be less prone to overfitting. Any form of regularisation will also limit the extent to which the model overfits.

2.1. Network width

[Question Table 1 - Fill in Table 1 with the results from your experiments varying the number of hidden units.] [Question Figure 2 - Replace the images in Figure 2 with figures depicting the accuracy and error, training and validation curves for your experiments varying the number of hidden units.]

First we investigate the effect of increasing the number of

# Hidden Layers	Val. Acc.	Train Error	Val. Error
1	81.0%	0.160	0.965
2	81.6%	0.098	1.569
3	81.9%	0.116	1.690

Table 2. Validation accuracy (%) and training/validation error (in terms of cross-entropy error) for varying network depths on the EMNIST dataset.

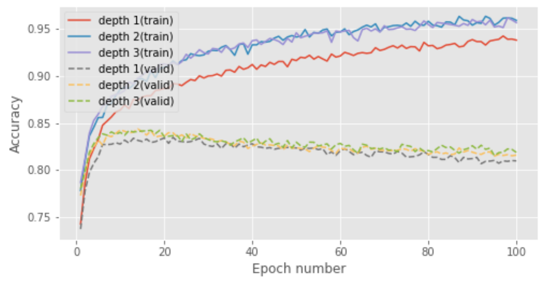
hidden units in a single hidden layer network when training on the EMNIST dataset. The network is trained using the Adam optimizer with a learning rate of 9×10^{-4} and a batch size of 100, for a total of 100 epochs.

The input layer is of size 784, and output layer consists of 47 units. Three different models were trained, with a single hidden layer of 32, 64 and 128 ReLU hidden units respectively. Fig. 2 depicts the error and accuracy curves over 100 epochs for the model with varying number of hidden units. Table 1 reports the final accuracy, training error, and validation error. We observe that **[As the hidden units(width) increase(from 32 ReLU hidden units to 128 ReLU hidden units), the validation accuracy increase slightly(78.6%, 81.1%, 80.6%). The validation accuracy growing trend is similar(at first growing strong at first 10 epoch and then gradually decrease after 20 epoch). Also, as the hidden units increase, the growing rate is increase as well. The 128 ReLU hidden units have the biggest growing rate at first 10 epoch. The validation accuracy declining rate are similar in different ReLU hidden units. In contrast,as the hidden units(width) increase(from 32 ReLU hidden units to 128 ReLU hidden units), the 32 ReLU hidden units have the biggest validation error at first 10 epoch.The 128 ReLU have the smallest validation error at first 10 epoch. However, after 10 epoch, the situation change completely. The 128 ReLU validation error growing strongly after 20 epoch, but the 64 ReLU and 32 ReLU validation error didn't grow strongly. It means after 20 epoch, the 128 ReLU is overfitting. Hence, the validation error in 64 ReLU is smallest, and 128 ReLU is biggest.]** .

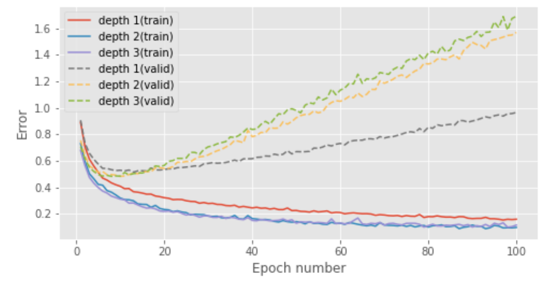
[Yes, the results match well with the prior knowledge. On the EMINIST dataset, singly adding the width will not make the performance of the model better. Moreover, the validation error of the model will increase. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. Eventually, these will negatively influence the model performance and generalization ability.].

2.2. Network depth

[Question Table 2 - Fill in Table 2 with the results from your experiments varying the number of hidden layers.] [Question Figure 3 - Replace these images with figures depicting the accuracy and error, training and validation curves for your experiments varying the number of hidden layers.]



(a) accuracy by epoch



(b) error by epoch

Figure 3. Training and validation curves in terms of classification accuracy (a) and cross-entropy error (b) on the EMNIST dataset for different network depths.

Next we investigate the effect of varying the number of hidden layers in the network. Table 2 and Fig. 3 depict results from training three models with one, two and three hidden layers respectively, each with 128 ReLU hidden units. As with previous experiments, they are trained with the Adam optimizer with a learning rate of 9×10^{-4} and a batch size of 100.

We observe that **[As the hidden layout(depth)increase the validation accuracy doesn't increase a lot (81.0%, 81.6%, 81.9%).The validation accuracy trend is similar in different depth. The validation accuracy grow fast in first 10 epoch and then decrease gradually after 20 epoch. The growing rate is similar as well. As the depth increase, the validation error increase. The trend of validation error is similar even in different depth. However,in depth 2 and depth 3, the validation error will increase a lot after 20 epoch. The error growing rate of depth 2 and depth 3 are much higher than depth 1. It means when the depth is big, it should use early stopping otherwise it will influence the validation error a lot.]** .

[Yes, the results match well with the prior knowledge.On the EMINIST dataset, singly adding the width will not make the performance of the model better. Moreover, the validation error of the model will increase. Eventually, these will negatively influence the model performance and generalization ability.] .

[As the previous experiments results shown, the increasing width and height will have better performances in the training set, but doesn't perform well in the validation set. It illustrates higher width and height will make the model overfitting. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. In addition, these will negatively influence the model performance and generalization ability. Also, it will increase model complexity and make it harder to regularize it. If the model do the early stopping at the turning point, it will help the model performance better in training set and the validation set as well.] .

3. Dropout and Weight Penalty

In this section, we investigate three regularization methods to alleviate the overfitting problem, specifically dropout layers and the L1 and L2 weight penalties.

3.1. Dropout

Dropout (Srivastava et al., 2014) is a stochastic method that randomly inactivates neurons in a neural network according to an hyperparameter, the inclusion rate (*i.e.* the rate that an unit is included). Dropout is commonly represented by an additional layer inserted between the linear layer and activation function. Its forward pass during training is defined as follows:

$$\text{mask} \sim \text{bernoulli}(p) \quad (1)$$

$$\boldsymbol{y}' = \text{mask} \odot \boldsymbol{y} \quad (2)$$

where $\boldsymbol{y}, \boldsymbol{y}' \in \mathbb{R}^d$ are the output of the linear layer before and after applying dropout, respectively. mask $\in \mathbb{R}^d$ is a mask vector randomly sampled from the Bernoulli distribution with inclusion probability p, and $\odot$ denotes the element-wise multiplication.

At inference time, stochasticity is not desired, so no neurons are dropped. To account for the change in expectations of the output values, we scale them down by the inclusion probability p:

$$\boldsymbol{y}' = \boldsymbol{y} * p \quad (3)$$

As there is no nonlinear calculation involved, the backward propagation is just the element-wise product of the gradients with respect to the layer outputs and mask created in the forward calculation. The backward propagation for dropout is therefore formulated as follows:

$$\frac{\partial \boldsymbol{y}'}{\partial \boldsymbol{y}} = mask \quad (4)$$

Dropout is an easy to implement and highly scalable method. It can be implemented as a layer-based calculation unit, and be placed on any layer of the neural network at will. Dropout can reduce the dependence of hidden units between layers so that the neurons of the next layer will not rely on only few features from of the previous layer. Instead, it forces the network to extract diverse features and evenly distribute information among all features. By randomly dropping some neurons in training, dropout makes use of a subset of the whole architecture, so it can also be viewed as bagging different sub networks and averaging their outputs.

3.2. Weight penalty

L1 and L2 regularization (Ng, 2004) are simple but effective methods to mitigate overfitting to training data. The application of L1 and L2 regularization strategies could be formulated as adding penalty terms with L1 and L2 norm square of weights in the cost function without changing other formulations. The idea behind this regularization method is to penalize the weights by adding a term to the cost function, and explicitly constrain the magnitude of the weights with either the L1 and L2 norms. The optimization problem takes a different form:

$$\text{L1: } \min_{\boldsymbol{w}} E_{\text{data}}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}) + \lambda \|w\|_1 \quad (5)$$

$$\text{L2: } \min_{\boldsymbol{w}} E_{\text{data}}(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{w}) + \lambda \|w\|_2^2 \quad (6)$$

where E_{data} denotes the cross entropy error function, and $\{\boldsymbol{X}, \boldsymbol{y}\}$ denotes the input and target training pairs. λ controls the strength of regularization.

Weight penalty works by constraining the scale of parameters and preventing them to grow too much, avoiding overly sensitive behavior on unseen data. While L1 and L2 regularization are similar to each other in calculation, they have different effects. Gradient magnitude in L1 regularization does not depend on the weight value and tends to bring small weights to 0, which can be used as a form of feature selection, whereas L2 regularization tends to shrink the weights to a smaller scale uniformly.

[Method1: Combining L2 and L1 through Elastic Nets gets both accuracy and sparsity.L1 regularization gives output in binary weights from 0 to 1 for the model's features and is adopted for decreasing the number of features in a huge dimensional dataset. L2 regularization disperse the error terms in all the weights that leads to more accurate customized final models. The potential benefits of this approach would be getting the model both accuracy and sparsity. And it would performance better to mitigate overfitting. Method2: When the weight value is small, using L1 regularization, and using L2 regularization when the weight value is large. The potential benefits of this approach: The combination of L1 and L2 regularization will be efficient compared to singly L1 or L2 regularization, the time taken to shrink the weights to 0 will be faster. And it would performance better to mitigate overfitting.] .

4. Balanced EMNIST Experiments

[Question Table 3 - Fill in Table 3 with the results from your experiments for the missing hyperparameter

values for each of L1 regularisation, L2 regularisation, and Dropout (use the values shown on the table).]

[Question Figure 4 - Replace these images with figures depicting the Validation Accuracy and Generalisation Gap (difference between validation and training error) for each of the experiment results varying the Dropout inclusion rate, and L1/L2 weight penalty depicted in Figure 3 (including any results you have filled in).]

Here we evaluate the effectiveness of the given regularization methods for reducing the overfitting on the EMNIST dataset. We build a baseline architecture with three hidden layers, each with 128 neurons, which suffers from overfitting as shown in section 2.

Here we train the network with a lower learning rate of 10^{-4}, as the previous runs were overfitting after only a handful of epochs. Results for the new baseline (c.f. Table 3) confirm that lower learning rate helps, so all further experiments are run using it.

Then, we apply the L1 or L2 regularization with dropout to our baseline and search for good hyperparameters on the validation set. We summarize all the experimental results in Table 3. For each method, we plot the relationship between generalisation gap and validation accuracy in Figure 4.

First we analyze three methods separately, train each over a set of hyperparameters and compare their best performing results.

[Based on the Table 3 and Figure 4,In the dropout model, dropout:0.7 will be the best in the dropout model. Because when the dropout value increase, the generalization performance will become worse and the validation accuracy didn't increase a lot. It means dropout value increase to 0.85 or 0.97, the model trend to be more overfitting. Hence, dropout:0.7 is a good choice rather than 0.85 or 0.87. In addition, among the L1 penalty model, L1 penalty:1e-3 would be the best choice. Because as the L1 penalty increase to 5e-3, although the validation accuracy will increase slightly, the generalization performance would relatively become worse. If the L1 penalty decrease to 5e-3 or 5e-2, the validation accuracy will dramatic decline. It violates what we want(validation accuracy should be bigger than baseline otherwise it's meaningless) Hence, L1 penalty:1e-3 would be best choice in the L1 penalty model. Moreover, in the L2 penalty model, the best choice would be 1e-3 as well. If the L2 penalty decrease to 5e-4, although the validation accuracy would increase, the generalization would increase more. Also, if the L2 penalty decrease to 5e-3 or 5e-2, the validation accuracy will decrease as well. Also because the L2 penalty:5e-3,5e-2, the validation accuracy didn't pass over the baseline. It's not a good choice to choose a smaller L2 penalty. Overall, the best model will be Dropout:0.7. Because the validation accuracy is relatively high and the generalization gap is relatively low. Totally, this model will be the best model based on validation accuracy and generalization performance.] .

[The first two experiments I want to choose is Dropout:0.7 + L1 penalty:1e-3 and Dropout:0.7 + L2 penalty:1e-3. Because it's the best value in every single model. I want to figure out based on the same condition of L1,L2 penalty and the dropout, L2 penalty would still better than L1 penalty or not. Because based on the previous observation, L2 penalty has the better validation accuracy and better generalization performance. And then the 3,4 experiments, I want to choose dropout:0.7 + L2 penalty:1e-3 and dropout:0.85 + L2 penalty:1e-3. Because I want to control L2 penalty as constant, and change the dropout value to ensure whether dropout:0.7 is a good choice or I should choose dropout:0.85 in the future. Because in the previous observation, dropout:0.7 and dropout:0.85 didn't show significant difference in validation accuracy and generalization performance. If the dropout value:0.85+L2:1e03 show better performance, I would like to choose a larger dropout value because it would help the validation accuracy. In the 5,6 experiments, I want to choose same dropout value:0.7 and the different L2 penalty:5e-4 and 1e-3. To compare whether smaller L2 penalty would improve the validation accuracy and generalization performance or not. In the 7,8 experiments, I want to choose dropout:0.97,L1 penalty:5e-4 and dropout:0.97,L2 penalty. Because I want to figure out in the extreme condition, whether L2 penalty still better than L1 penalty. If the L2+dropout model would be better, I think the best combination would be L2 penalty and dropout instead of L1 penalty and dropout.] .

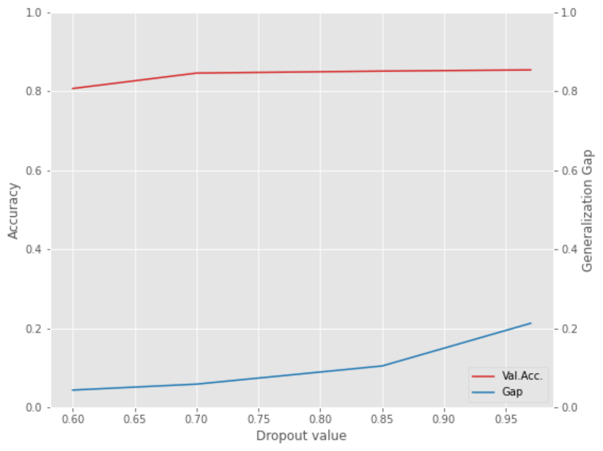
5. Literature Review: Decoupled Weight Decay Regularization

Summary In this section, we briefly study a method (Loshchilov & Hutter, 2019) that decouples the weight penalty from the weight update. The authors shed light onto the relation between weight decay and L2 weight penalty for SGD and Adam optimizers, pointing out that weight decay and L2 weight penalty are not equivalent when used with the Adam optimizer.

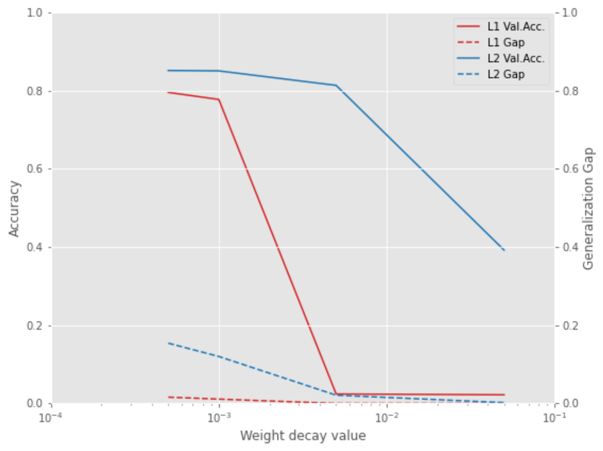
In particular, the authors claim that SGD with L2 weight penalty and weight decay are equivalent when their coefficients have the relation $\lambda' = \frac{\lambda}{2}$ (see their **Proposal 1**), which they prove with Equations (5) and (6) (proof is in their Appendix A). In addition, they claim that such a simple scalar relation does not hold in Adam optimization (see **Proposal 2**) and show, in their Appendix A, that the necessary relation for equivalence requires use of a preconditioner matrix $\boldsymbol{M}_t$. This requirement can be explained with the fact that **[L2 regularization and weight decay are not identical. The two techniques can be made equivalent for SGD by a reparameterization of the weight decay factor based on the learning rate; Adam can substantially benefit from a scheduled learning rate multiplier. The fact that Adam is an adaptive gradient algorithm and as**

Model	Hyperparameter value(s)	Validation accuracy	Train Error	Validation Error
Baseline	-	0.837	0.241	0.533
Dropout	0.6	80.7	0.549	0.593
	0.7	**84.6**	**0.444**	**0.503**
	0.85	85.1	0.329	0.434
	0.97	85.4	0.244	0.457
L1 penalty	5e-4	79.5	0.642	0.658
	1e-3	*77.7*	*0.723*	*0.734*
	5e-3	2.41	3.850	3.850
	5e-2	2.20	3.850	3.850
L2 penalty	5e-4	85.1	0.306	0.460
	1e-3	*85.0*	*0.333*	*0.453*
	5e-3	81.3	0.586	0.607
	5e-2	39.2	2.258	2.256

Table 3. Results of all hyperparameter search experiments. *italics* indicate the best results per series (Dropout, L1 Regularization, L2 Regularization) and **bold** indicates the best overall.



(a) Accuracy and error by inclusion probability.



(b) Accuracy and error by weight penalty.

Figure 4. Accuracy and error by regularization strength of each method (Dropout and L1/L2 Regularization).

such adapts the learning rate for each parameter does not rule out the possibility to substantially improve its performance by using a global learning rate multiplier, scheduled, e.g., by cosine annealing. The gradient of the L2 regularization will be included in the computation of the adaptive phrases. Therefore, a more complex non-linear pre-processing matrix M is needed to adjust the pace between these two optimizers, instead of a scalar relationship] .

[Because grads_wrt_params already include the L2 regularization term in the layers python file. Moreover, the grads_wrt_params was passed to function update_params for the Adam optimizer in the learning_rules python file. Hence, we implemented the Adam with L2 regularization instead of AdamW.J.] .

Finally the authors validate their findings and proposed decoupled optimization strategies in various learning rate schedules, initial learning rates, network architectures and datasets. [3]

6. Conclusion

[Firstly, on the EMINIST dataset, simply adding the width and depth will not make the performance of the model better. Although the training accuracy and training error will performance better, this positive correlation does not exist in the validation set. Because the model didn't get extra positive generalization ability to improve the validation accuracy, but getting the ability to generalize the noise or random fluctuations in the training data. Moreover, because the noise or random fluctuations on the training set learned as concepts by the model. It will result in validation accuracy decrease, validation error increase, and overfitting. These will

[3]Instructor note: Omitting this for this coursework, but normally you would give an evaluation of the experiment setup in (Loshchilov & Hutter, 2019) here.

negatively influence the model performance and generalization ability. About the Dropout and Weight Penalty, if the dropout value is extremely large, it will not help with the validation accuracy,but increasing the generalization gap. The optimal dropout value should be from 0.7 to 0.85. For the L1 and L2 penalty, if the value is small, it would contribute negative effect on the validation accuracy. The L1 and L2 penalty should be from 1e-3 to 5e-4. In general, L2 regularization is a better choice compared to L1 regularization. And the optimal width and depth should be 128ReLU and 1 depth in the early stopping condition otherwise it would negatively influence the model performance. Because the overfitting would occur.] .

References

Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Loshchilov, Ilya and Hutter, Frank. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Ng, Andrew Y. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 78, 2004.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958, 2014.