NANYANG TECHNOLOGICAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

AI6103 Deep Learning and Application

Group Project:

Zheng Weixiang	Zeng Tian	Ao Yichen	Cai Xiaobing
wzheng014@e.ntu.edu.sg	tzeng005@e.ntu.edu.sg	aoyi0001@e.ntu.edu.sg	caix0028@e.ntu.edu.sg
G2103278G	G2102023C	G2102383H	G2102065L

Abstract

The paper is a group project of AI6103 Deep Learning and Application. It discusses the effect of 6 different regularization techniques including data augmentation, mixup, weight decay, dropout, label smoothing and stochastic depth. Among them, data augmentation and mixup are data dependent regularization, the others are data independent regularization techniques.

The Effect of Label Smoothing & Data Augmentation

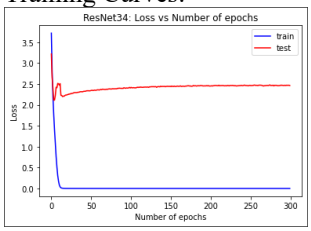
The basic model for label smoothing and data augmentation is configured as follows:

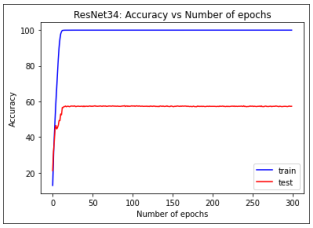
Neural Network:	ResNet-34
Loss Function:	Cross-Entropy
Initial Learning Rate:	0.01
Learning Rate Schedule:	Cosine Annealing
Momentum:	0.9

Performance:

	Train Loss	Train Accuracy	Test Loss	Test Accuracy
Baseline	0.0003	99.98	2.4624	57.28

Training Curves:





We can conclude from the graphs that the test loss is increasing while the training loss is decreasing. Also, the test accuracy shows a reducing trend with the epoch. This means that the model is overfitting the data.

Label Smoothing

Label smoothing is one of the regularization techniques that can make the model predict the output with less confidence. In this way, the model is expected to have a better generalization ability and avoid overfitting the training data.

Equation of Label Smoothing:

$$\boldsymbol{y}_s^{(i)} = [\frac{\epsilon}{C-1}, \dots, \frac{\epsilon}{C-1}, 1-\epsilon, \frac{\epsilon}{C-1}, \dots \frac{\epsilon}{C-1}]$$

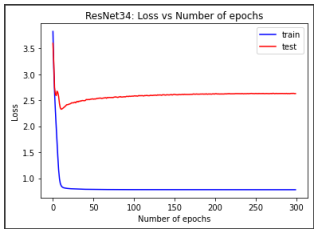
$$\ell(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}, w) = -\boldsymbol{y}_s^{(i)^\top} \log P(\hat{y}|\boldsymbol{x}^{(i)}, w)$$

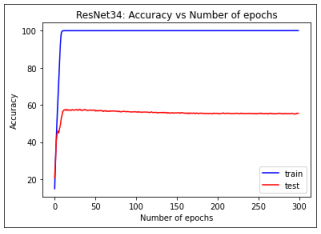
Performance:

	Train Loss	Train Accuracy	Test Loss	Test Accuracy
Smooth factor = 0.1	0.7798	99.99	2.6283	55.46
Smooth factor = 0.2	1.4092	99.98	2.9672	55.49

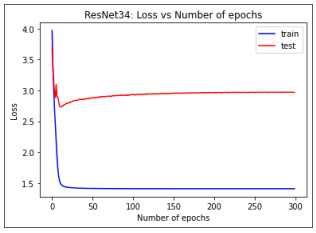
Training Curves:

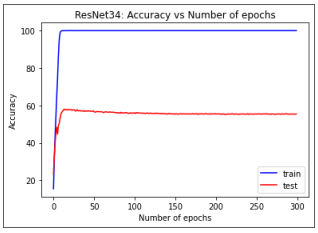
Smooth Factor = 0.1





Smooth Factor = 0.2





From the results above, the effect of label smoothing on the performance of the model is not prominent. The test loss is increasing with the epoch and the test accuracy is decreasing with the epoch, which means the model is overfitting the data. Furthermore, the final test accuracy is even lower than that of the baseline. However, to some extent, the label smoothing method still improves the test accuracy of the model at the early stage of the training process. For the smooth factor = 0.1, the highest test accuracy appears at the 32nd epoch and is equal to 57.64%. Also, for the smooth factor = 0.2, the highest test accuracy appears at the 14th epoch and is equal to 58.01%. While the highest test accuracy for the baseline is at the 91st epoch and is equal to 57.60%.

This phenomenon might show that the label smoothing method can somehow increase the generalization ability of the model as the two configurations both have their maximum test accuracies higher than that of the baseline model. Nonetheless, as the training epoch goes larger, the effect of the label smoothing method becomes negative to the test performance of the model.

Data Augmentation

Data augmentation is a regularization technique that modifies the input data into different variants for the model to be trained on. This technique reduces the chance of overfit-

ting by expanding the capacity of training data while enhancing the robustness of the model by providing variants of the same image.

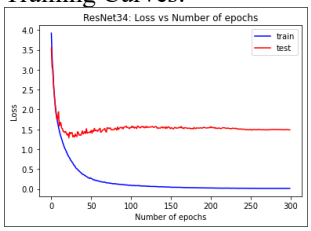
Data augmentation techniques used in this experiment include random crop, random horizontal flip, and random erasing.

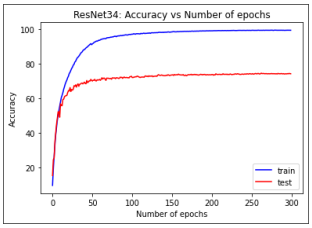
The random crop method creates a random subset of the original image, which helps the model generalize better because the random subsets may contain details that are not always visible in the original image scale. In addition, the random horizontal flip method creates a variant by flipping the original image horizontally. This can help the model recognize better the different poises of the same object. On the other hand, the random erasing method will create a variant by randomly cutting out a portion from the original image, which forces the model to recognize the object based on other features of the image. This further enhances the robustness of the model in image classification.

Performance:

	Train Loss	Train Accuracy	Test Loss	Test Accuracy
With Data Augmentation	0.0166	99.51	1.4874	74.32

Training Curves:





The diagrams show that the test loss converges eventually and the test accuracy increases slowly with the epoch, which means the model is not overfitting the data.

Finally, we can confirm that the data augmentation methods are effective in terms of boosting test performance and preventing overfitting. Though on the cost of a little training accuracy, the data augmentation methods improve the test accuracy of the model by 29.7%.

The Effect of Mixup Regularization

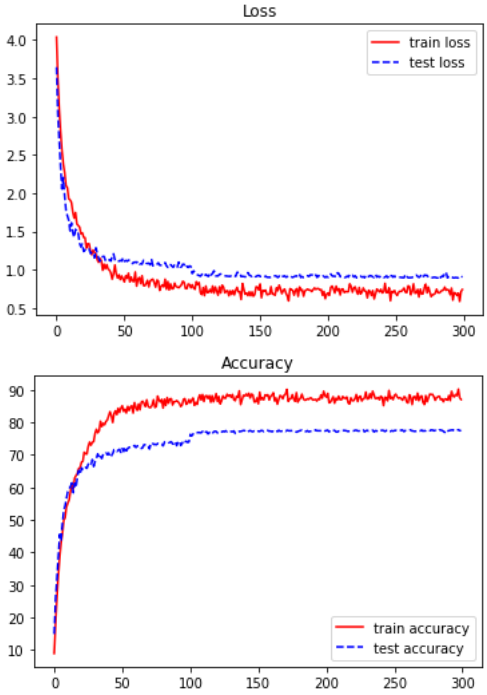
The baseline model for evaluating the effect of Mixup is:

Network	ResNet-34
Loss	Cross-Entropy
Initial LR	0.01
LR Scheduler	None
Momentum	0.9
Weight Decay	1e-4
Transform	RandomCrop+RandomHorizontalFlip

Performance:

	Train Loss	Train Acc	Test Loss	Test Acc
Baseline	0.5877	90.2308	0.9020	77.74

Plot:



Mixup

Mixup is a data augmentation technique that that generates a weighted combinations of random image pairs from the training data. Given two images and their ground truth labels: (x_i, y_i), (x_j, y_j), a synthetic training example (x_{hat}, y_{hat}) is generated as:

$$x_{hat} = \lambda x_i + (1 - \lambda)x_j$$

$$y_{hat} = \lambda y_i + (1 - \lambda)y_j$$

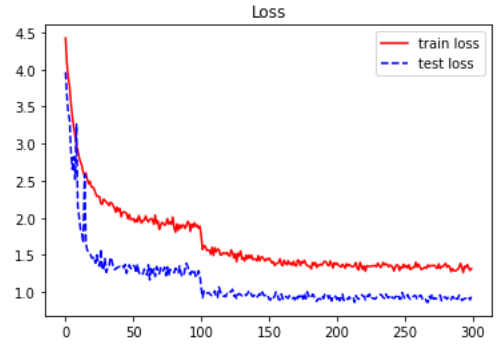
Where $\lambda \sim Beta(\alpha = 0.2)$ is independently sampled / for each augmented example.

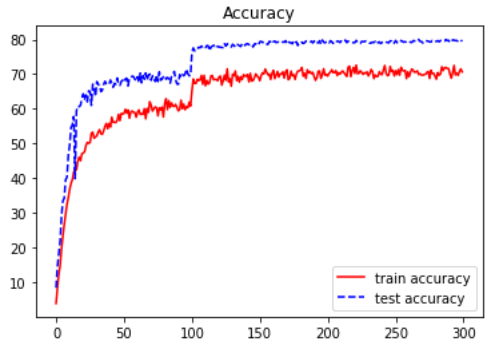
Performance:

	Train Loss	Train Acc	Test Loss	Test Acc
$\alpha = 1$	1.3221	70.9455	0.8698	80.1

An interesting observation about this regularization is that after applying Mixup, the training loss will be higher than testing loss, the training accuracy will be lower than testing accuracy. This is because only during the training process are two images mixed up, during the testing process, the

images will not be mixed up. Therefore, the model only feels confused when training, during the testing process, the model will only see one complete image to label.





We can see clearly from the data and the plot that, during the training process, because of the Mixup, the model seems more confused about the training data, i.e., higher training loss than baseline, lower training accuracy than baseline. However, the test result is improved, this is because we force the model to learn more about feature of every single picture, and because there is a process of comparing different data points and differentiate them, the model can understand the feature better than baseline. Overall, we can see that Mixup is a stable booster for improving test accuracy at the cost of lower training accuracy and higher training loss.

The Effect of Weight Decay & Dropout

The basic model for evaluating the effects of Weight Decay and Dropout regularization techniques is configured as follows:

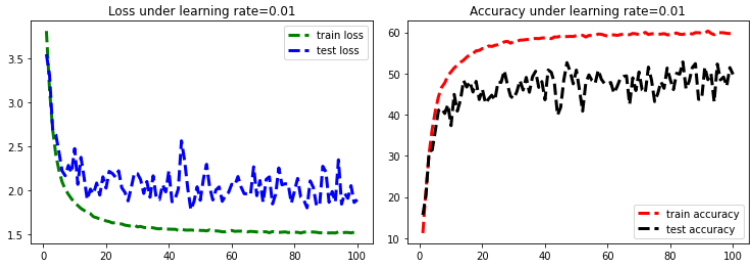
Network	ResNet-34
Loss	Cross-Entropy
Initial LR	0.01
LR Scheduler	ConstantLR
Momentum	0.9
Transform	None
Epoch	100

Performance of different weight decays after 100 epochs:

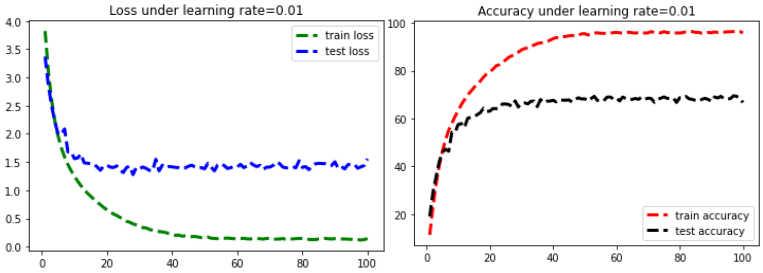
	Train Loss	Train Acc	Test Loss	Test Acc
1e-2	1.5189	59.71	1.8973	49.78
5e-4	0.1449	95.77	11.5548	66.69

The following experiments are conducted with 100 epochs. One of the reasons is that when experimenting with 300 epochs, the loss keeps growing. The preliminary reasoning is due to overfitting therefore we decrease the number of training epochs to 100.

The plot of weight decay = 1e-2



The plot of weight decay = 5e-4

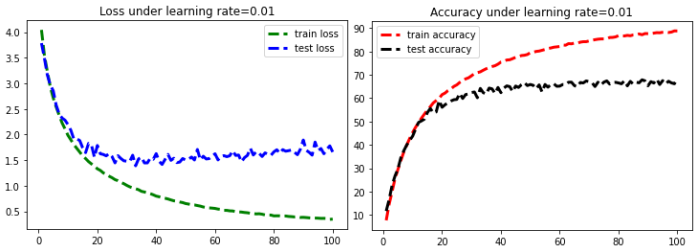


A higher weight decay (1e-2) means stronger regularization, but stronger regularization is not necessarily a good thing. We can see clearly from the above plots that smaller weight decay (5e-4) has a smoother test curve and at the same time the test accuracy is higher, i.e., around 60 versus 50 with higher weight decay. The stronger regularization also result in greater gap between training and testing curve, can be observed both in loss and accuracy plots. It means that even higher weight decay will result in better training statistics, it is not necessarily reflected on the testing dataset.

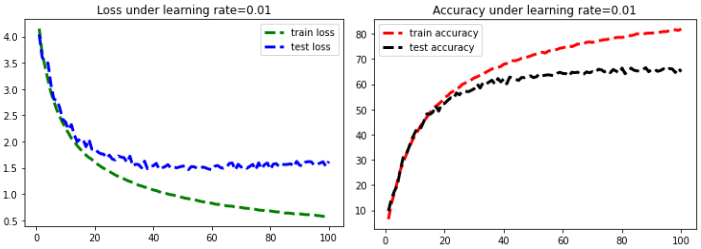
Performance of different dropouts after 100 epochs:
(We use Dropout to substitute Batch Normalization, there are two layers of BN, therefore two layers of Dropout.)

	Train Loss	Train Acc	Test Loss	Test Acc
0.5+0.5	0.5703	81.91	1.6227	64.98
0.2+0.5	0.3469	88.79	1.6588	68.13

The plot of dropout 0.2 + 0.5 after 100 epochs



The plot of dropout 0.5 + 0.5 after 100 epochs



We can see from the above plots that 0.2+0.5 dropout tends to have more fluctuation during the testing process. It means that the model's performance is not very stable. On the other hand, when using 0.5+0.5, the testing curve is smoother and contain fewer fluctuations. Another interesting point is that when using 0.2+0.5, the training accuracy is higher up to 90, but when using 0.5+0.5, the highest point is around 80. However, the difference of their testing accuracy is not that obvious.

The Effect of Stochastic Depth

Stochastic Depth (stochastic depth network) is a regularization method which is to add a random variable during training, the probability distribution of it satisfies a Bernoulli distribution, and then multiplies and randomly discards the residual part. If this structure is the ResNet structure, and at that time, the residual branch is not activated, and the whole structure degenerates into an identity function. (below fig)

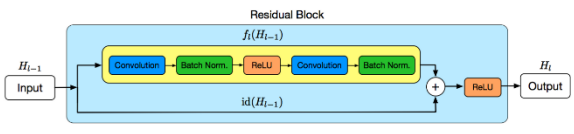


Fig. 1. A close look at the ℓ^{th} ResBlock in a ResNet.

This process can be represented by the following equation:

$$H_\ell = \mathrm{ReLU}(b_\ell f_\ell(H_{\ell-1}) + \mathrm{id}(H_{\ell-1}))$$

There is also a survival probability assigned for each block which defines the probability to use that block. The probability will be different for each mini batch. The author has tried different distribution on the survival probability and found that linear decay performs the best.

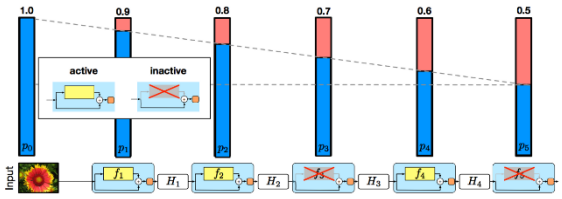


Fig. 2. The linear decay of p_ℓ illustrated on a ResNet with stochastic depth for $p_0 = 1$ and $p_L = 0.5$. Conceptually, we treat the input to the first ResBlock as H_0, which is always active.

For the optimization of the residual module, since a very deep ResNet usually takes a long time to train (that is, the training is very slow), Stochastic Depth is a method like dropout which randomly discards sublayers during the training process (randomly drop a subset of layers) and use the full graph normally when inferencing.

During training, randomly drop each ResBlock (by Bernoulli distribution) and directly output the previous ResBlock to the next ResBlock, and the dropped ResBlock does nothing and does not update. Additionally, the input of the network is regarded as the first layer and will not be dropped. The difference with Dropout is that this method drops the entire ResBlock, while Dropout only drops a part of the unit during training. Thus, this method greatly reduces the training time, and even deletes some layers after the training is completed without affecting the accuracy.

Weight Decay/ `Baseline`	**Train**		**Test**	
	Loss	**Accuracy (%)**	**Loss**	**Accuracy (%)**
5e-4	1.0064	73.80	1.5424	59.55
`Baseline`	0.4989	85.24	1.6992	58.92

Table 1 Weight Decay vs `Baseline` Loss & Accuracy

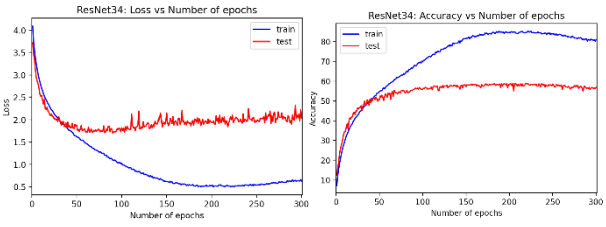


Figure 1: **Bassline** Loss **(Left) vs** Accuracy **(Right)**

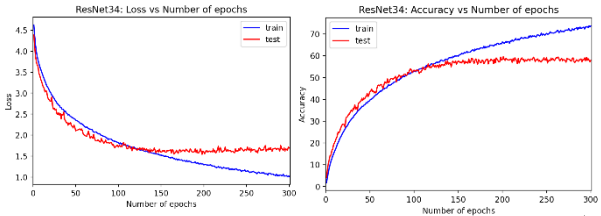


Figure 2: **Weight Decay** Loss **(Left) vs** Accuracy **(Right)**

We run 300 epochs for Stochastic Depth network, From **Table 1** we found that for **Baseline**, the best loss is 0.4989 & the best accuracy is 85.24% for training, and for testing those are 1.6992 and 58.92. But we add **weight decay** = 5e-4, For training, those are 1.0064 and 73.80%. For testing, those are 1.5424 & 59.55%. Based on the training and testing curve above **Figure 1**, It shows quite good and converged to some points for training loss & accuracy, but testing loss and accuracy are quite bad which means that there still has overfit issue. but for **Figure 2**, they have relatively lower gap between training & testing curves. Thus, it has lower overfitting issue.

References

Boyang Li, Albert. (2022). AI 6103: DEEP LEARNING & APPLICATIONS, lecture slides. Retrieved from https://ntulearn.ntu.edu.sg/webapps/blackboard/content/listContent.jsp?course_id=_2606899_1&content_id=_2784948_1

Papers with Code - Mixup Explained. (n.d.). Papers with Code - Mixup Explained. Retrieved April 19, 2022, from https://paperswithcode.com/method/mixup

Kulakov, A. (n.d.). 2 reasons to use MixUp Augmentation when training your Deep Learning models. Medium. Retrieved April 20, 2022, from https://towardsdatascience.com/2-reasons-to-use-mixup-when-training-yor-deep-learning-models-58728f15c559

Team, K. (n.d.). Keras documentation: MixUp augmentation for image classification. Keras. Retrieved April 20, 2022, from https://keras.io/examples/vision/mixup/

Zhang, H. (n.d.). mixup: Beyond Empirical Risk Minimization. arXiv.Org. Retrieved April 20, 2022, from https://arxiv.org/abs/1710.09412

Liu, Z. (n.d.). Unveiling the Power of Mixup for Stronger Classifiers. arXiv.Org. Retrieved April 20, 2022, from https://arxiv.org/abs/2103.13027

Huang, G. (n.d.). Deep Networks with Stochastic Depth. arXiv.Org. Retrieved April 20, 2022, from https://arxiv.org/abs/1603.09382

ResNet 为什么不用 Dropout? (n.d.). 知乎. Retrieved April 20, 2022, from https://www.zhihu.com/question/325139089