Tech Report of IAI Project 2

Jingze Zhang, jz-zhang21@mails.tsinghua.edu.cn

Tsinghua University, Department of Automation

1 Assignment1

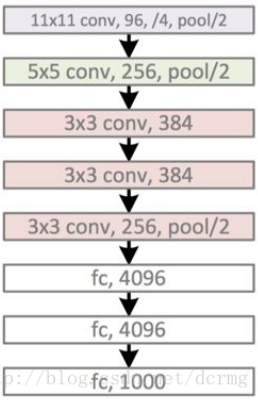
In this assignment, we designed various network architectures and tested their performance on the test set, validation set, and training set using a given dataset. All data were obtained after training the neural networks for 50 epochs. The experimental results are as follows:

Table 1 Accuracy on Different Datasets

Network	Trainable Params	train/acc	val/acc	test/acc
MLP	667K	80.0%	62.6%	61.8%
AlexNet	57.0M	97.5%	84.3%	83.5%
Resnet50 (w/o pretrained weights frozen)	23.5M	**99.9%**	93.1%	**92.9%**
Resnet50 (w weights pretrained frozen)	12.3K	96.2%	**93.3%**	92.5%

1.1 AlexNet

In the experiment, we used the following AlexNet structure.



1.2 Resnet50

To enhance the accuracy of experimental recognition and test the effectiveness of a pretrained model on this dataset, we conducted a series of experiments using a ResNet50 model equipped with pretrained weights. The results of these experiments are recorded in Figure 1.

2 Assignment2

2.1 Impact of Optimizer Selection on Model Training Outcomes

Table 1: Experimental Results of Different Optimizers

优化器	学习率	train/acc	val/acc	test/acc	epoch
Adam	0.001	97.5%	84.3%	83.5%	50
AdamW	0.001	96.1%	82.8%	83.0%	50
SGD	0.001	43.8%	42.7%	42.6%	50
SGD	0.001	63.1%	62.1%	63.6%	150
SGD	0.03	**98.8%**	**87.4%**	**87.5%**	150
Adam	0.03	18.1%	17.4%	17.4%	50

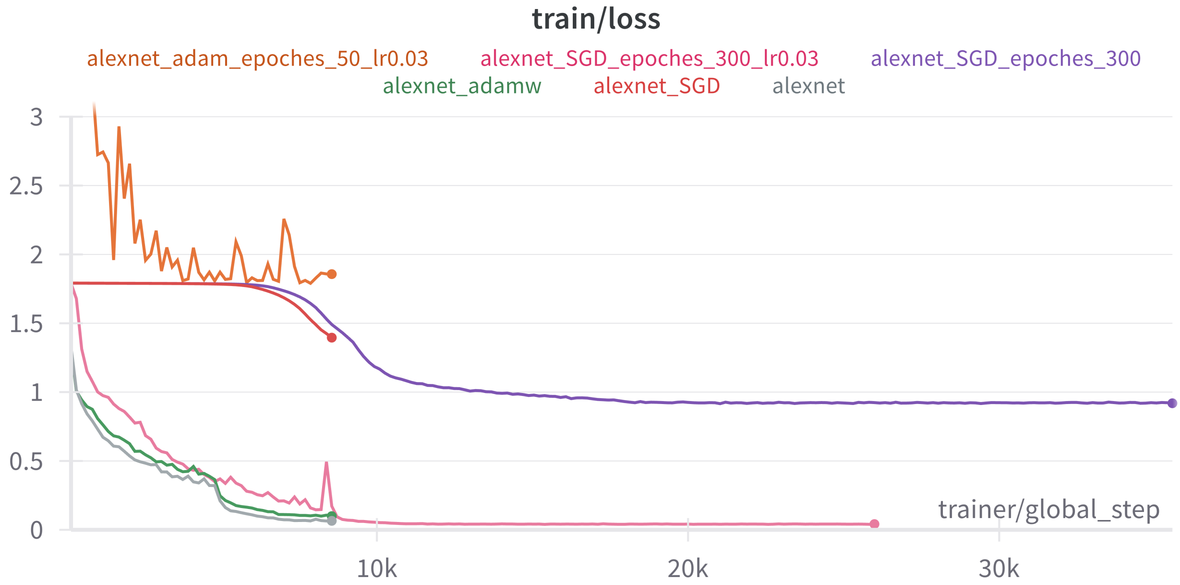


Fig 1 training loss

Fig 2 validation loss

Fig 3 training accuracy **Fig 4 validation accuracy**

This table compares the performance of different optimizers (Adam, AdamW, SGD) with varying learning rates, as demonstrated by the accuracy on training, validation, and test datasets over specific training epochs:

1. **Adam Optimizer**: Performs well at a learning rate of 0.001, achieving over 83% accuracy on validation and test sets. However, increasing the learning rate to 0.03 drastically reduces performance to around 18%, indicating instability at higher rates.

2. **AdamW Optimizer**: Shows slightly inferior performance to Adam at a learning rate of 0.001, with minimal differences, suggesting comparable generalization capabilities.

3. **SGD Optimizer**: Exhibits poor performance at a low learning rate of 0.001, but significantly improves at a higher rate of 0.03, achieving 87.5% accuracy, which highlights its high sensitivity to learning rate adjustments.

These findings underscore the critical role of optimizer selection and learning rate tuning in enhancing model training effectiveness and generalization.

2.2 MobileNet

```python

def conv_bn(inp, oup, stride):
    return nn.Sequential(
        nn.Conv2d(inp, oup, 3, stride, 1, bias=False),
        nn.BatchNorm2d(oup),
        nn.ReLU(inplace=True)
    )

def conv_dw(inp, oup, stride):
    return nn.Sequential(
        nn.Conv2d(inp, inp, 3, stride, 1, groups=inp, bias=False),
        nn.BatchNorm2d(inp),
        nn.ReLU(inplace=True),

        nn.Conv2d(inp, oup, 1, 1, 0, bias=False),
        nn.BatchNorm2d(oup),
        nn.ReLU(inplace=True),
    )
```

Using the method described above, we compressed the parameters of AlexNet from 54MB to 0.7MB. After training for 50 epochs, its test accuracy was essentially the same as that of the original AlexNet.

2.3 The impact of certain model structures on performance

2.3.1 Dropout

	00.0%	**10.0%**	**20.0%**	**30.0%**	**40.0%**	**50.0%**	**60.0%**	**70.0%**	**80.0%**	**90.0%**
Test/Acc	83.30%	83.53%	80.01%	84.21%	82.77%	84.24%	83.77%	81.54%	84.12%	79.42%

The data indicates that the test accuracy varies between 79.42% and 84.24% across different test segments, generally maintaining a stable performance above 80%. The highest accuracies are observed around the 30.0% and 50.0% intervals. The lowest performance occurs at the 90.0% mark, suggesting some limitations in that condition.

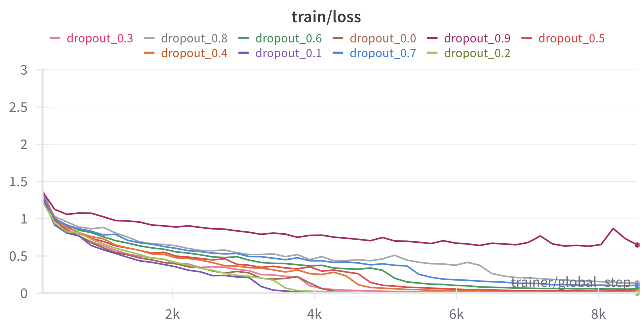


Fig 1 training loss

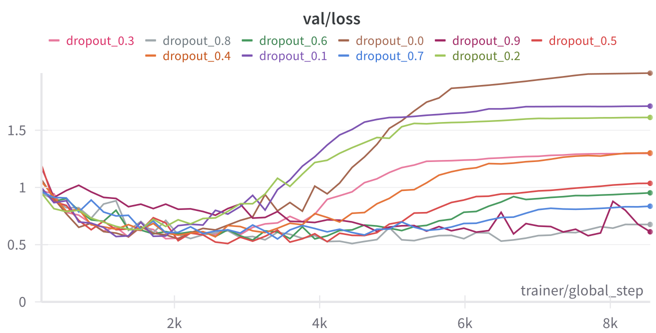
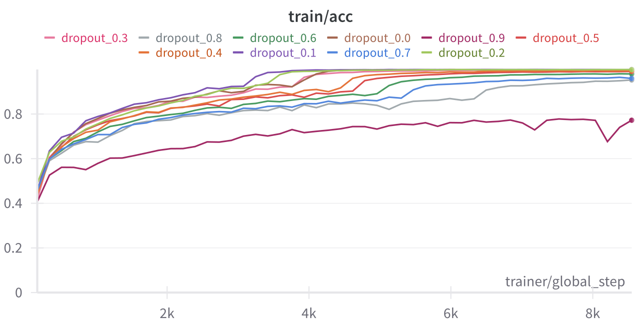


Fig 2 validation loss



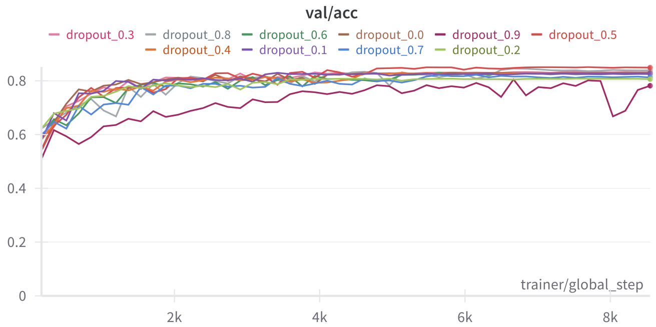


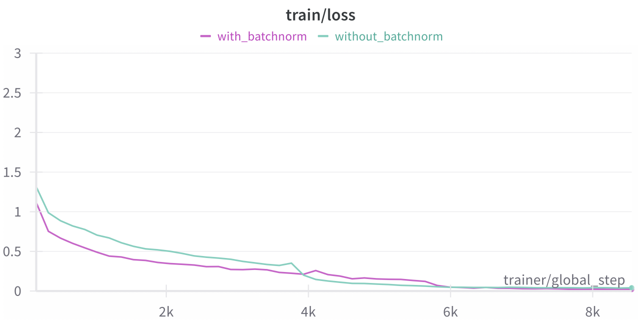
Fig 3 training accuracy

Fig 4 validation accuracy

2.3.2 Batch Normalization

Table 3: Experimental Results of Batch Normalization

Setting	train/acc	val/acc	test/acc	epoch
w Batch Normalization	**99.4%**	**89.8%**	**89.2%**	50
w/o Batch Normalization	98.7%	83.2%	83.5%	50



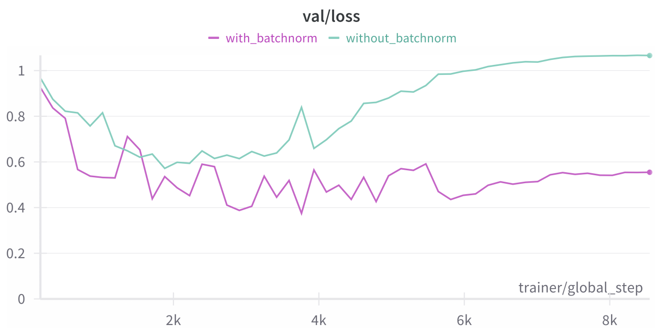
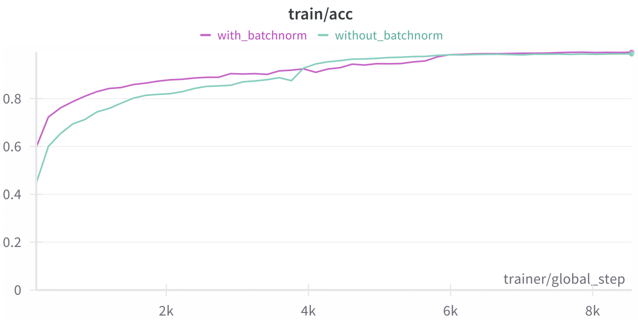


Fig 1 training loss

Fig 2 validation loss



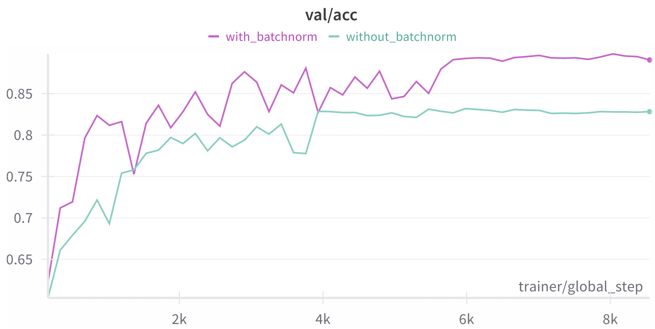
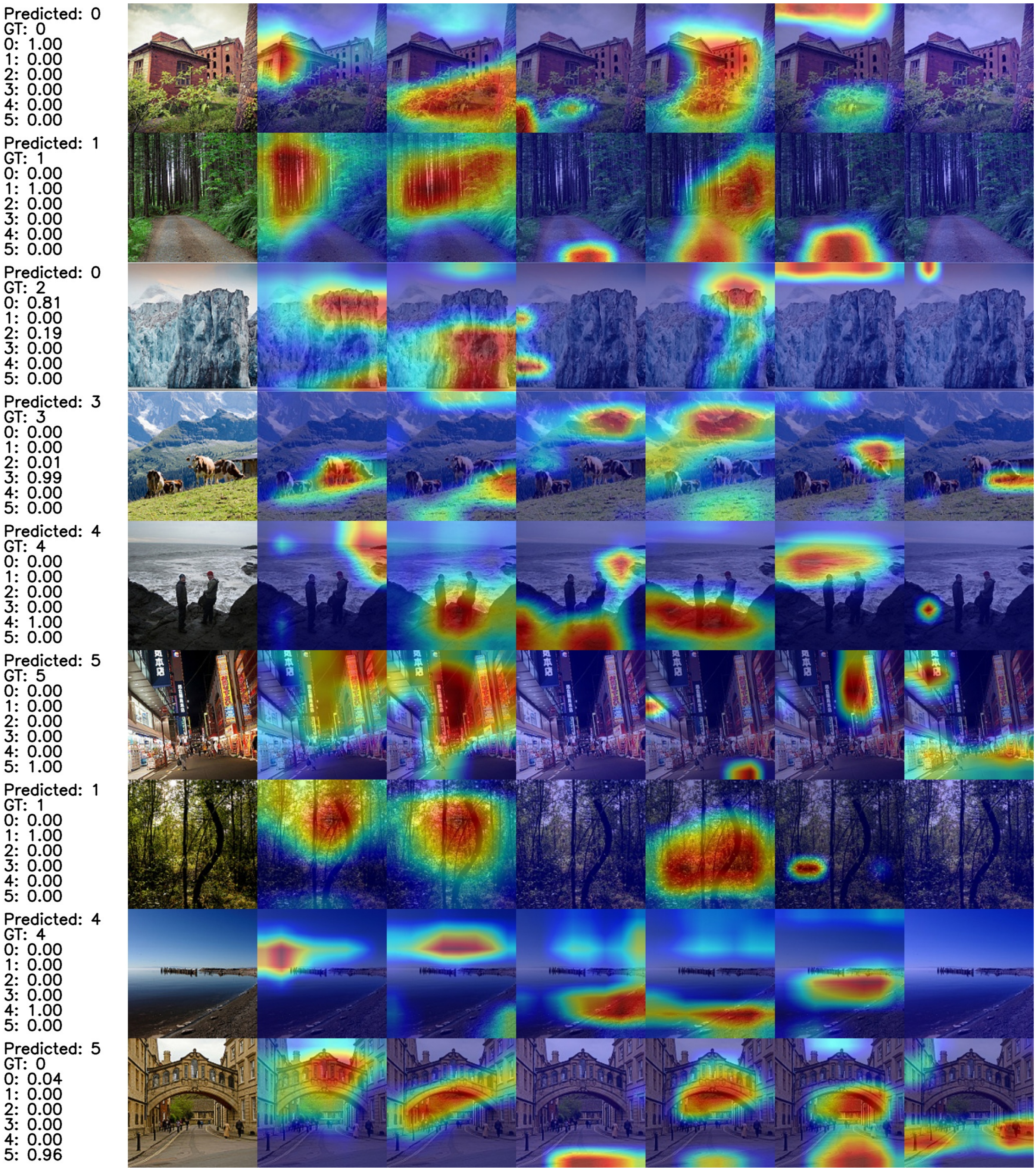


Fig 3 training accuracy

Fig 4 validation accuracy

From the curve above, we can see that batch normalization can quickly help the model converge at a faster rate, and to some extent, prevent overfitting and enhance the model's performance.

2.4 CAM Analysis for Model Interpretability



This image showcases a series of heat maps generated through Gradient-weighted Class Activation Mapping (Grad-CAM), which visualizes the areas of input that are important for predictions in a convolutional neural network. Each row corresponds to different input images and their corresponding Grad-CAM heat maps across various layers of the network. The images include diverse scenes such as buildings, landscapes, and urban environments.

From the heat maps, it is evident that the model focuses on different regions depending on the specific features of the scene. For example, in architectural images, the model tends to focus on structural elements like windows and facades, while in natural landscapes, it highlights areas with distinct topographical features. The "Predicted" and "GT" labels, likely referring to the predicted class and ground truth, indicate how well the model's predictions align with the actual class labels. The scores next to these labels could be the confidence scores of the predictions.

The conclusion from analyzing these Grad-CAM visualizations is that the model is effectively learning relevant features for different categories, as indicated by the focused activation in meaningful areas of the images. These heat maps are instrumental in understanding model behavior, identifying which parts of the input images influence the neural network's predictions, and ensuring that the model is not focusing on irrelevant features for its decisions.

This result can only show that the cnn network pays different attention to different parts. **However, it is also obvious that the neural network can easily learn some features different from our original intention.**

2.5 Ablation Study: Distribution of Classes in the Training Set

In this problem, we use confusion matrix as our metric. Here are the relevant results.

Table 4: Experimental Results of Varition in Training Set Distribution

Setting	train/acc	val/acc	test/acc	epoch
[1.00, 1.00, 1.00, 1.00, 1.00, 1.00]	99.4%	89.8%	89.2%	50
[1.00, 1.00, 0.30, 1.00, 1.00, 1.00]	98.7%	89.6%	88.6%	50
[0.50, 0.50, 0.50, 1.00, 1.00, 1.00]	98.8%	89.2%	89.2%	50
[0.02, 0.02, 0.02, 1.00, 1.00, 1.00]	99.2%	79.9%	79.8%	50

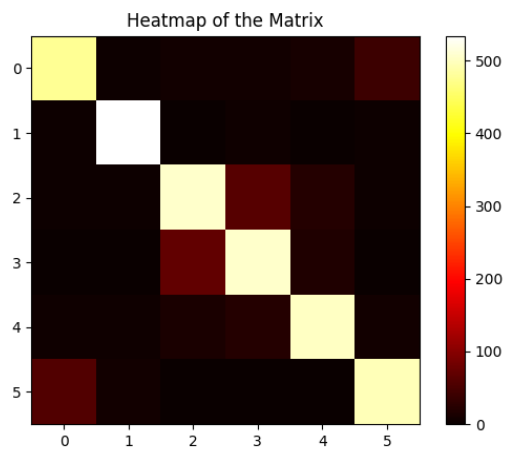


Fig 1 initial settings(baseline)

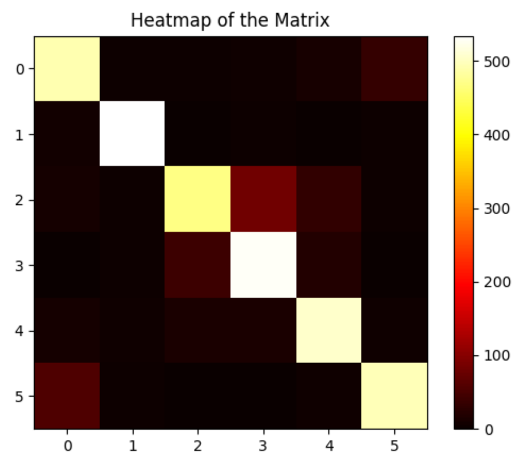


Fig 2 class 2 70% decline

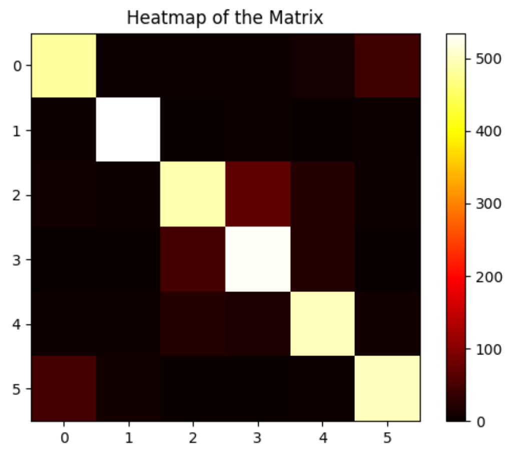


Fig 4 class 0 to 2 98% decline

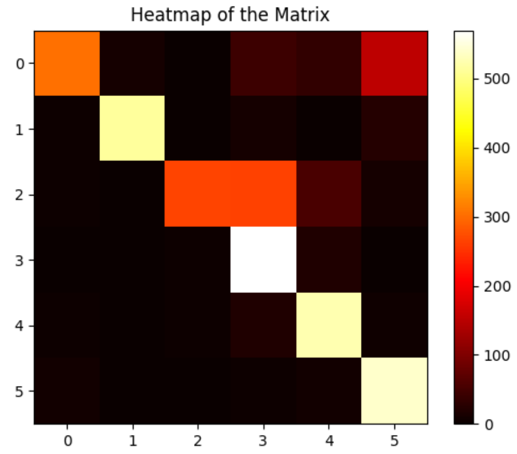


Fig 2 validation loss

2.5.1 Initial Settings

```
[473   1   8   6  15  45]
[  4 531   1   6   2   6]
[  5   3 499  61  28   3]
[  1   0  53 520  20   0]
[  5   1  17  13 513   5]
[ 52   3   2   0   3 502]
```

2.5.2 Fewer Samples in Class 2

```
[491   3   4   5  11  34]
[  8 534   0   4   1   3]
[  9   3 468  82  33   4]
[  2   3  40 529  20   0]
[  9   5  14  14 506   6]
[ 56   4   2   0   5 495]
```

- In this experiment, the decline ratio of each class is set to `[1.0, 1.0, 0.3, 1.0, 1.0, 1.0]`

2.5.3 Fewer Samples in First 3 Classes

```
[484   3   3   6   9  43]
[  4 535   2   6   0   3]
[  7   3 493  68  22   6]
[  0   0  46 529  19   0]
[  6   5  21  16 498   8]
[ 47   8   2   0   4 501]
```

- In this experiment, the decline ratio of each class is set to `[0.5, 0.5, 0.5, 1.0, 1.0, 1.0]`

2.5.4 No Samples in First 2 Classes

- In this experiment, the decline ratio of each class is set to `[0.02, 0.02, 0.02, 1.0, 1.0, 1.0]`

```
[305  10   2  44  35 152]
[  3 513   0  10   1  23]
[  3   2 266 263  55  10]
[  0   0   3 570  19   2]
[  3   0   3  18 524   6]
[  7   2   1   4   7 541]
```

2.5.5 Discussion

Based on the confusion matrix data provided, some preliminary patterns can be identified, which reflect the performance of the classification model under various conditions. Confusion matrices serve as an essential tool in assessing the accuracy of classification models, where each element corresponds to the count of predictions for each actual class. Here are key observations from the data:

1. **Variability in Diagonal Elements**: The diagonal elements, representing the true positive counts for each class, exhibit variations across different datasets, especially when data volume is reduced. This generally indicates a potential decline in model accuracy with reduced data.

2. **Group 0 Data (Class 0)**: There is a noticeable decrease in classification accuracy for the first group when data is reduced (e.g., from 50% to 2%), with the true positive count dropping from 484 to 305. This suggests that the identification of Class 0 is particularly sensitive to the volume of data.

3. **Group 1 Data (Class 1)**: Despite reductions in data volume, the classification accuracy for the second group remains relatively stable (with diagonal values around 535), indicating robustness of the model for this class against changes in data volume.

4. **Impact of Data Reduction on Misclassification**: Significant data reductions, especially to 2%, lead to an increase in misclassifications (off-diagonal elements), particularly noticeable in Group 0 and Group 2. In Group 0, misclassifications across other classes increase substantially.

These observations can guide further model adjustments and data management strategies, enhancing the understanding of the model's performance under varied data volumes. For more detailed analysis, such as statistical tests or complex data visualizations, further methodologies can be applied.