Project 3

Qiyu Chen

March 18, 2023

Abstract

This project trains MLP on Taiji key pose classification task and CNN on the wallpaper images classification task. Different architectures are experimented to compare their performance. On wallpaper task, improved CNN architecture is trained under the technique of data augmentation in order to have better performance on the challenge test subset.

Contents

1 Taiji Key-pose Classification with MLP

In this part, a baseline and improved architectures of MLP are trained on the Taiji Key-pose Classification task.

1.1 Baseline and updated architecture of MLP

The baseline MLP architecture consists of single hidden layer, and the number of neurons in this hidden layer is 1024. Compared to the baseline architecture, updated MLP architecture inserts one additional hidden layers in-between, and the dimension of this hidden layer is also 1024. What's more, several Relu layers are added after the output of every layers except the output layer.

1.2 Model evaluation

There are two different datasets to train and test: the "full" version and the "lod4" version. Performance of both baseline and updated architectures are evaluated on them.

Full dataset
Figure 1 and 2 tells us that the accuracy of the updated architecture is better on both training and testing subsets. Even, the standard deviation of the updated model is smaller than the baseline model. These two observations validate that the improved model has a better learning ability among all provided features.

Figure 3 and 4 shows test classification accuracy of both models for each class. It is obvious that both models are having great performance on classes 33, 34 and worst performance on classes 16, 27. Relative low std on class 33 of both models, shown by Figure 5 and 6, further support the claim that classification on class 33 is great and stable.

Figure 7 and 8 are confusion matrices of subject 8 in test subsets. The fact that there are more scatter off-diagonal "pixels" in baseline architecture supports the previous claim that updated architecture has better performance and is more stable.

lod4 dataset
For the lod4 dataset, where each sample is having much less input features, we can see from figure 9 and 10 that the training and testing accuracy gap between the baseline and updated model is much smaller than the gap in full dataset, though the improved architecture still slightly performs better. This observation illustrates that the lod4 dataset already extracts useful input features from the full version, such that subsequent analysis on features have smaller impact on the overall accuracy.

Figure 11 and 12 still supports the same observation that both architectures perform worst in class 27. However, more other classes reach the same accuracy level as classes 33 and 34. Similarly, the low standard deviation of class 33 in figure 13 and 14 validates that prediction on this class sample is quite stable.

Lastly, figure 15 and 16 are confusion matrices of subject 8 in test subsets. Comparing those in full dataset, we observe that confusion matrix in both models are now much similar, and this supports the claim that lod4 already extracts very useful input features, both models on subsequent feature interpretation are similar, and thus have much smaller impact on class prediction.

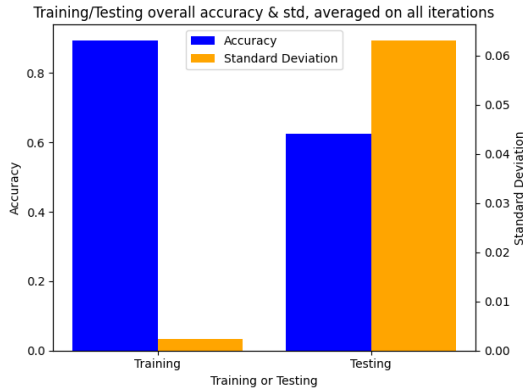

Figure 1: Training and testing overall accuracy and standard deviation of baseline architecture, averaged on all LOSO iterations

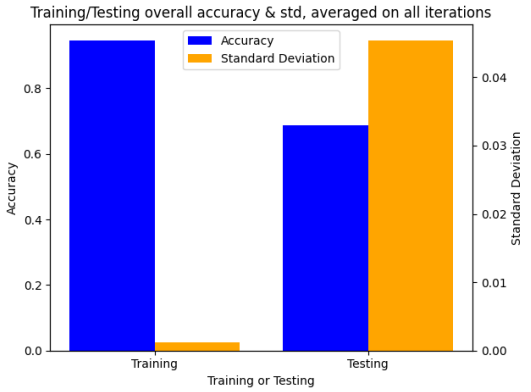

Figure 2: Training and testing overall accuracy and standard deviation of updated architecture, averaged on all LOSO iterations

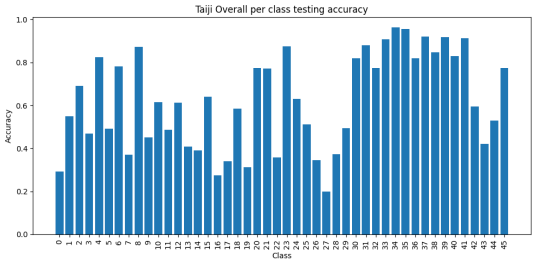

Figure 3: Test classification accuracy of baseline model for each class, averaged on all LOSO iterations

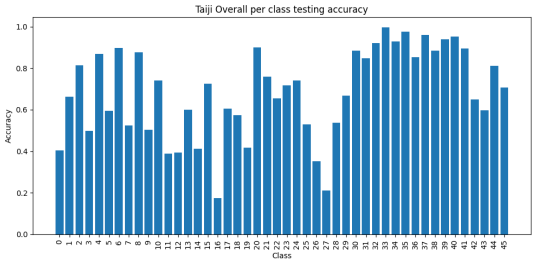

Figure 4: Test classification accuracy of improved model for each class, averaged on all LOSO iterations

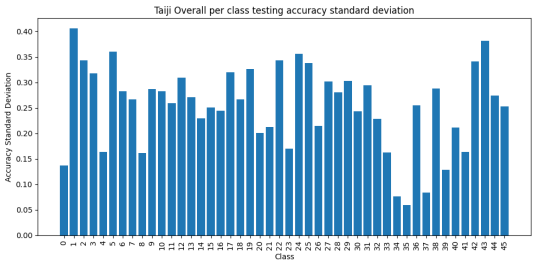

Figure 5: Test classification accuracy std of baseline model for each class, averaged on all LOSO iterations

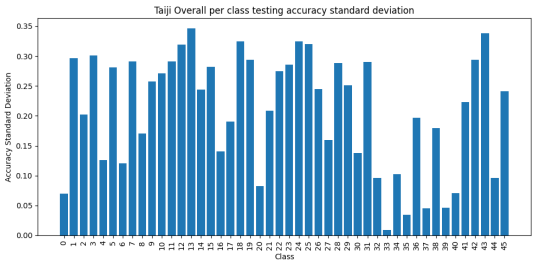

Figure 6: Test classification accuracy std of improved model for each class, averaged on all LOSO iterations

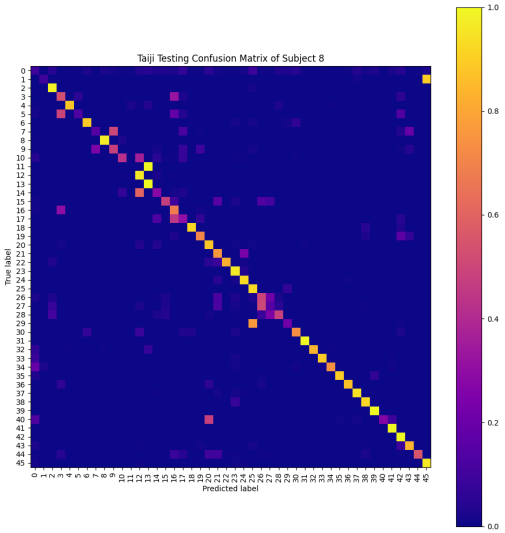

Figure 7: Confusion matrix of subject 8 in test subset (baseline architecture)

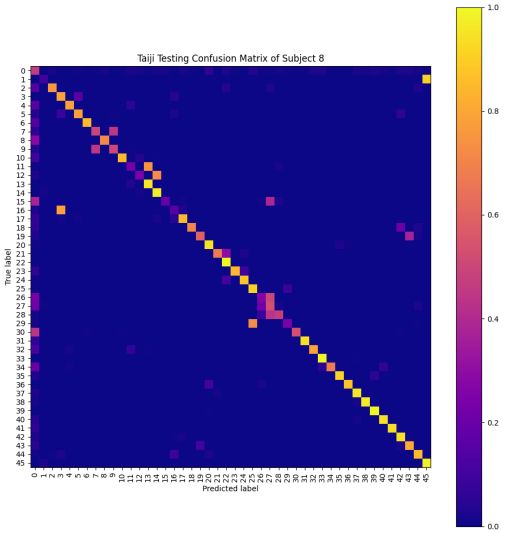

Figure 8: Confusion matrix of subject 8 in test subset (improved architecture)

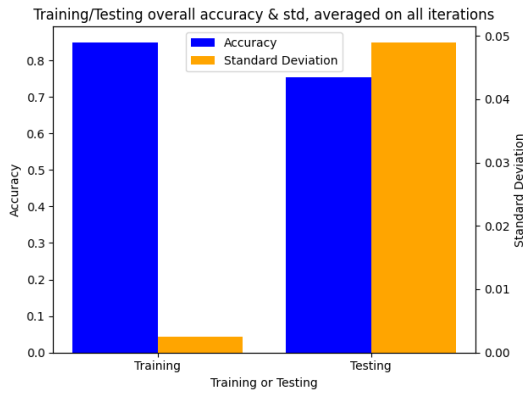

Figure 9: Training and testing overall accuracy and standard deviation of baseline architecture, averaged on all LOSO iterations

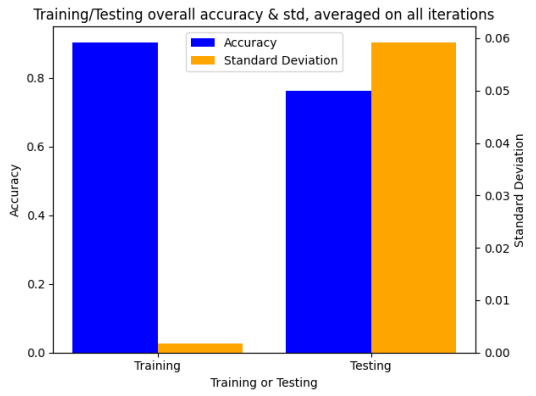

Figure 10: Training and testing overall accuracy and standard deviation of updated architecture, averaged on all LOSO iterations

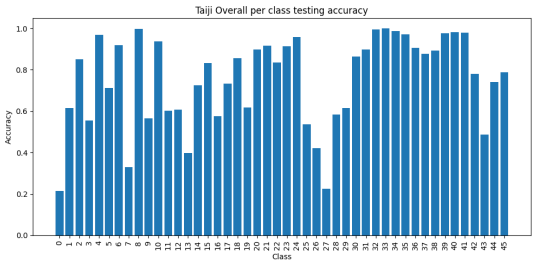

Figure 11: Test classification accuracy of baseline model for each class, averaged on all LOSO iterations

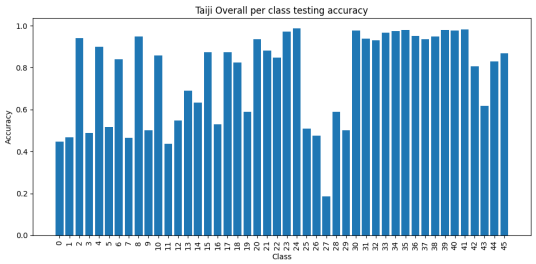

Figure 12: Test classification accuracy of improved model for each class, averaged on all LOSO iterations

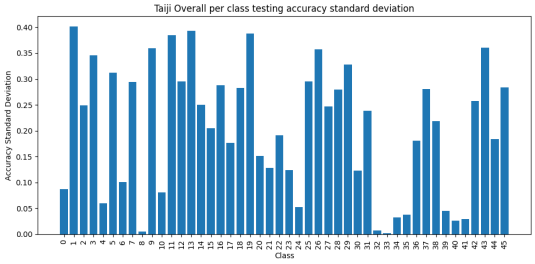

Figure 13: Test classification accuracy std of baseline model for each class, averaged on all LOSO iterations

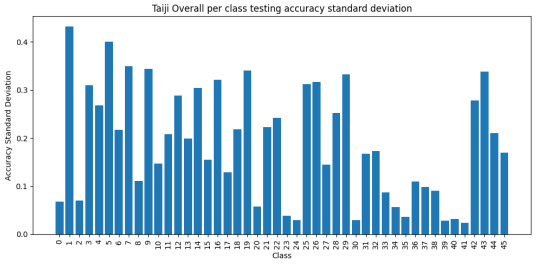

Figure 14: Test classification accuracy std of improved model for each class, averaged on all LOSO iterations

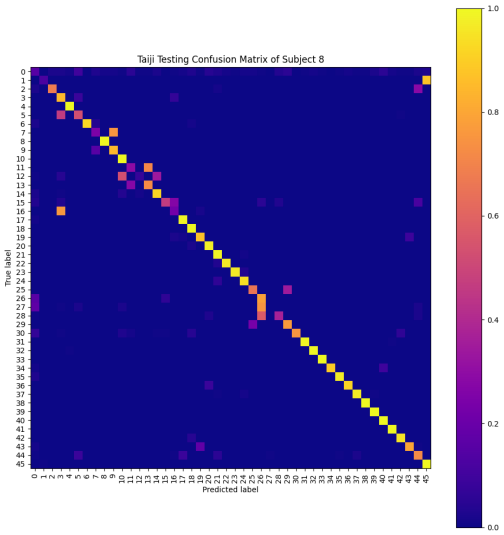

Figure 15: Confusion matrix of subject 8 in test subset (baseline architecture)

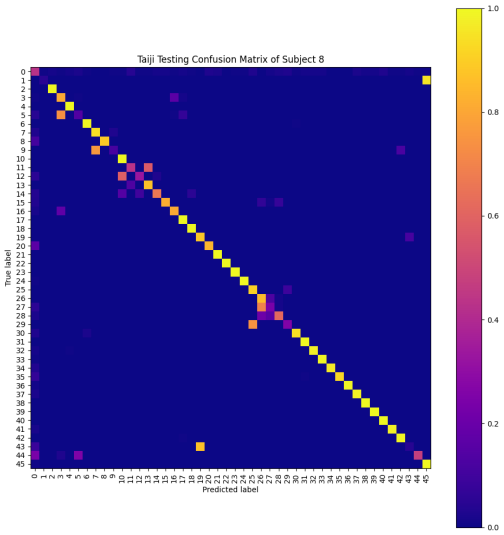

Figure 16: Confusion matrix of subject 8 in test subset (improved architecture)

2 Wallpaper Classification with CNN

In this part, a baseline and improved architectures of CNN are trained on the Wallpaper Class Classification task.

2.1 Baseline and updated architecture of MLP

The baseline CNN architecture is described in project 3 manual, except that the Dropout layer is not added between last two linear layers. In improved architecture, a Relu layer and Dropout layer are added after every Conv2d layers. In addition, one additional Conv2d layer (also Relu, Dropout, and MaxPool2d after) with kernel size 3, padding 1, is added after the last MaxPool2d layer in the baseline architecture, which maps the number of channels from 128 to 256. The second to last linear layer then adjusts its input dimension accordingly. Lastly, the output of the CNN model do not go through log softmax calculation. This is because the function used for loss computation is CrossEntropyLoss, which already computes log softmax after passing raw logits output into it. If the output from CNN model is processed after log softmax, it is better to use the NLLLoss according to documentation.

2.2 Model evaluation

There are three different datasets for model evaluations: train, test and test challenge. To tackle the test challenge dataset, we also try using data augmentation to improve the training on models. Performance of both models without data augmentation is evaluated. After applying data augmentation in training, the performance of the improved model is also evaluated.

Data augmentation
In my experiment, different data augmentation strategies are tried, consisting of rotation (random from 0 to 360 degrees), uniform scaling (random scaling factor (0.5, 2)), translation (random translation using factors (0.5, 0.5)), vertical/horizontal flipping, and cropping (random cropping, then resize to input image size).

Specifically, I prepared several different augmentation difficulties. The simplest one randomly select any of the above augmentation strategy and perform on the input image. The next level difficulty then tries any combination of two strategies mentioned above (do not consider their orders) with same probability. In this way, the hardest difficulty applies all above strategies on input images. To ensure original images are trained, above strategies only applied with certain probability, said 0.8, on input images. Hence, by training enough epochs, original input images can still be trained. The training starts from the simplest difficulty and the difficulty is increased gradually, once the model starts to perform much worse in the test challenge subset. By experiments I found that training in this way leads better performance on test challenge dataset comparing to directly training on any single level augmentation strategy.

Evaluation without data augmentation

Figure 17-22 indicates the better performance of improved architecture on train, test, and test-challenge datasets. This fact is further supported by figure 23 and 24, such that the improved architecture has relatively great performance over all classes.

Figure 25 and 26 give another view of classification rate per class. As we can see, bright pixels are mainly laying on diagonal, meaning overall classification of both models on test dataset is great.

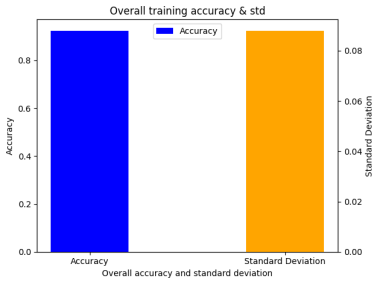

Figure 17: Overall accuracy and standard deviation on train dataset of baseline architecture

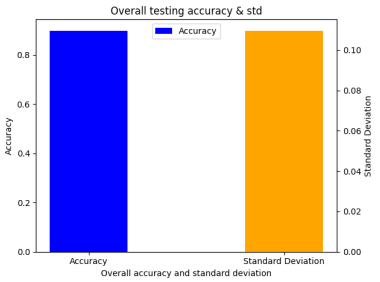

Figure 18: Overall accuracy and standard deviation on test dataset of baseline architecture

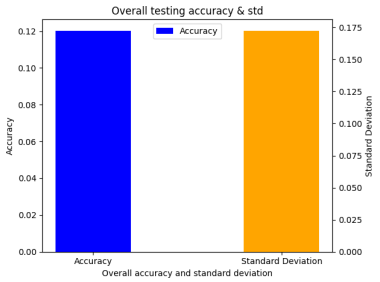

Figure 19: Overall accuracy and standard deviation on test challenge dataset of baseline architecture

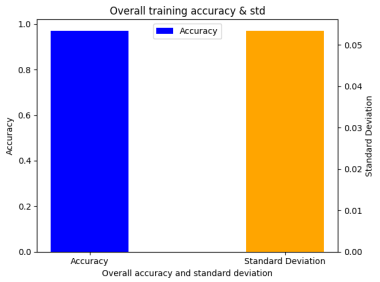

Figure 20: Overall accuracy and standard deviation on train dataset of improved architecture

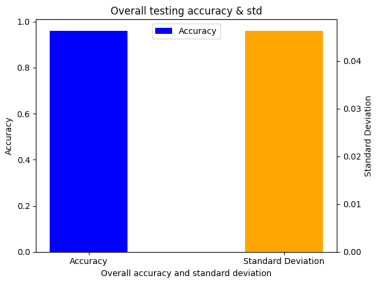

Figure 21: Overall accuracy and standard deviation on test dataset of improved architecture

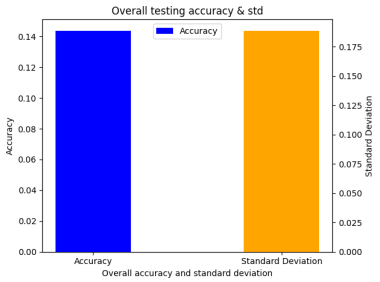

Figure 22: Overall accuracy and standard deviation on test challenge dataset of improved architecture

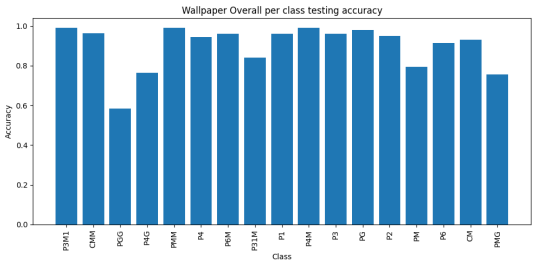

Figure 23: Test classification accuracy of baseline architecture for each class

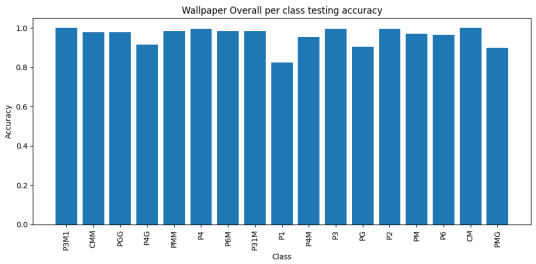

Figure 24: Test classification accuracy of improved architecture for each class

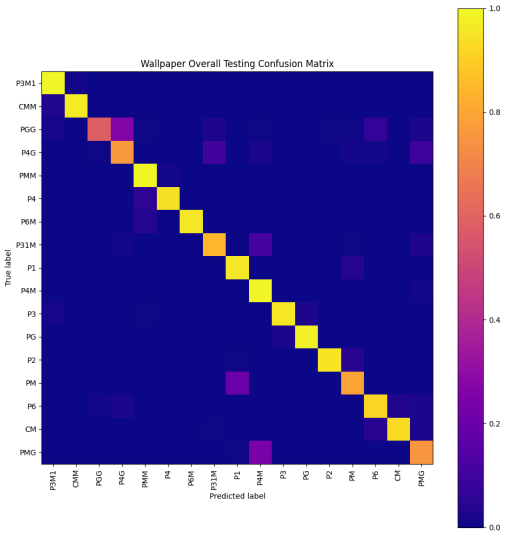

Figure 25: Test confusion matrix of base architecture for each class

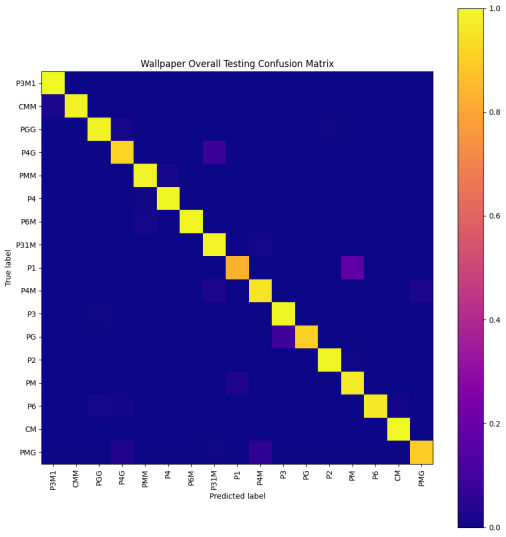

Figure 26: Test confusion matrix of improved architecture for each class

Evaluation with data augmentation

From figure 27 and 28, we observe that the accuracy of the improved model on train and test datasets drops compared to the one trained without augmentation. This is possible, as training data are now becoming more challenging after augmentation, making the model less likely to overfit. Also, as most epochs are trained on augmented images, performance of the model will slightly favor augmented samples. From figure 29, we can see that accuracy on test-challenge dataset increases dramatically, from 14.38% to 38.77%. The effect of data augmentation is significant as indicated.

Figure 30 indicates that the improved architecture has uneven classification rates over different classes, where samples from P4G class has lowest accuracy, and samples from PM class has highest accuracy. The brightness of pixels among diagonal on figure 31 supports the same claim.

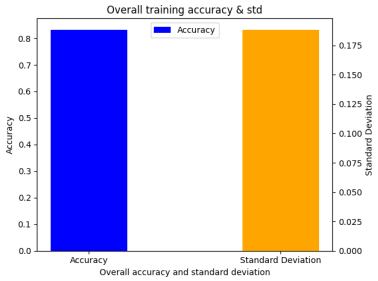

Figure 27: Overall accuracy and standard deviation on train dataset of baseline architecture

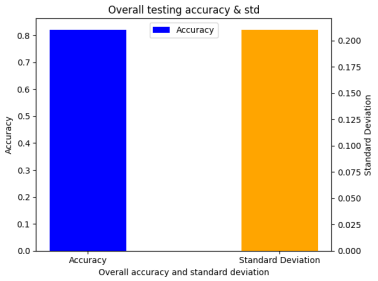

Figure 28: Overall accuracy and standard deviation on test dataset of baseline architecture

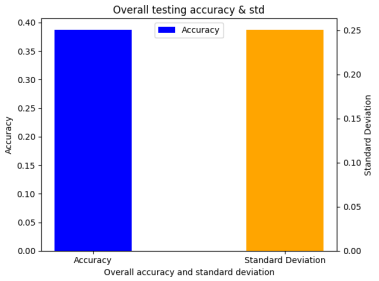

Figure 29: Overall accuracy and standard deviation on test challenge dataset of baseline architecture

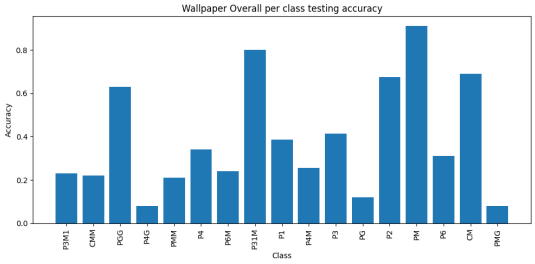

Figure 30: Test confusion matrix of base architecture for each class

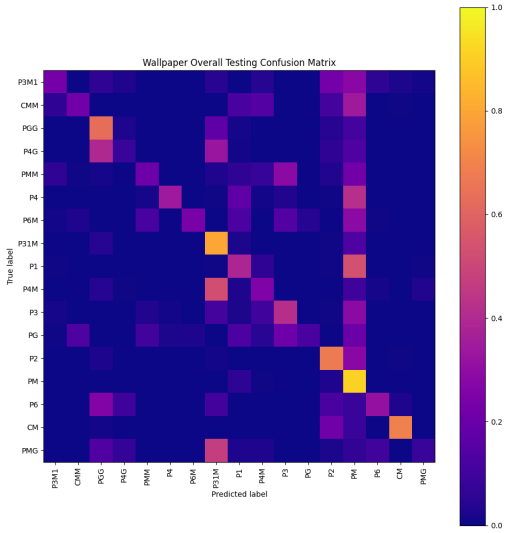

Figure 31: Test confusion matrix of improved architecture for each class

Feature and t-SNE visualization

Figure 32 is the feature maps of the first convolutional layer from 17 wallpaper groups in improved architecture. As there are 32 output channels from that layer, the first channel is selected as the representation output.

Figure 33 is the visualization of t-SNE after performing it on the input of the last fully connected layer of the improved architecture. 20 input images are selected for each wallpaper group, and they are plotted with the same color.

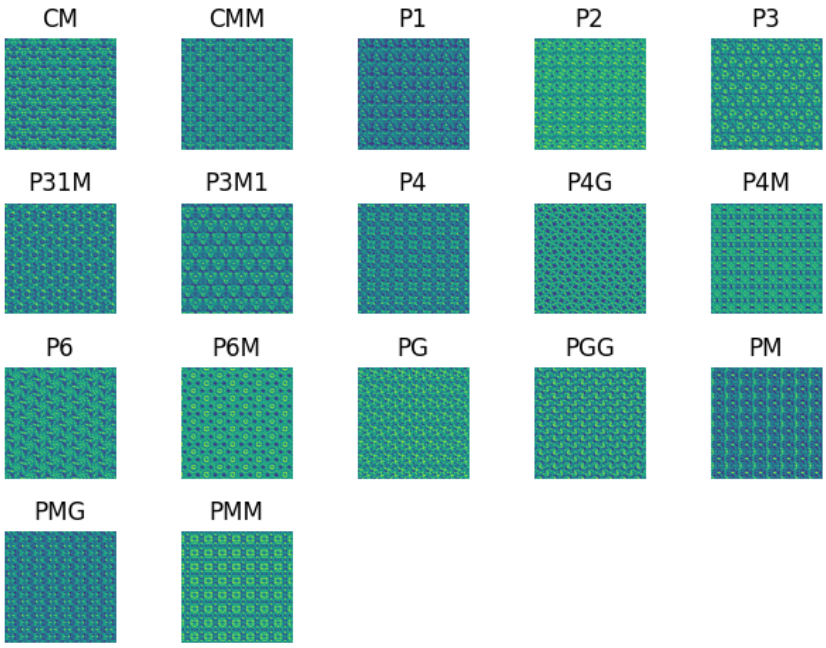

Figure 32: Feature maps of the first convolutional layer (first channel output) from 17 wallpaper groups

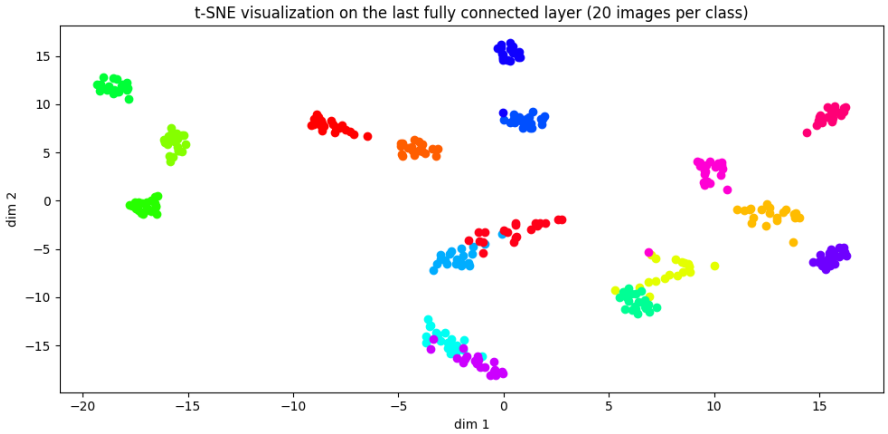

Figure 33: t-SNE visualization on the last fully connected layer