PRASANNA, LAKSITA
2/17/25

DEEP LEARNING HW 2

In this homework assignment, categorical classification was implemented on the Intel Image Scene Classification data using Convolutional Neural Networks (CNNs). Since the dataset provided was too huge, only a subset of 350 images or less were used.

After hyper tuning the parameters, it was evident that using Adam optimizer significantly increased both training and validation accuracy than RMSProp. Moreover, using more epochs, batch size, and more images increased the overall training and validation accuracy.

Problem 1

For both parts of Problem 1, some hyperparameters were kept constant: Adam optimizer and learning rate of 1e-4. Num of images (350), epochs (30), and batch size (35). Dropout layers were specifically not used in both parts of this problem to serve as some baseline to the sub parts in Problem 2. Because these techniques were avoided in Problem 1, overfitting and inefficient generalizability can be expected.

(P1 Simple network)

3 Conv layers with these # of filters: 32, 64, 128; filter size (3 x 3)

Dense: 256, 6 nodes

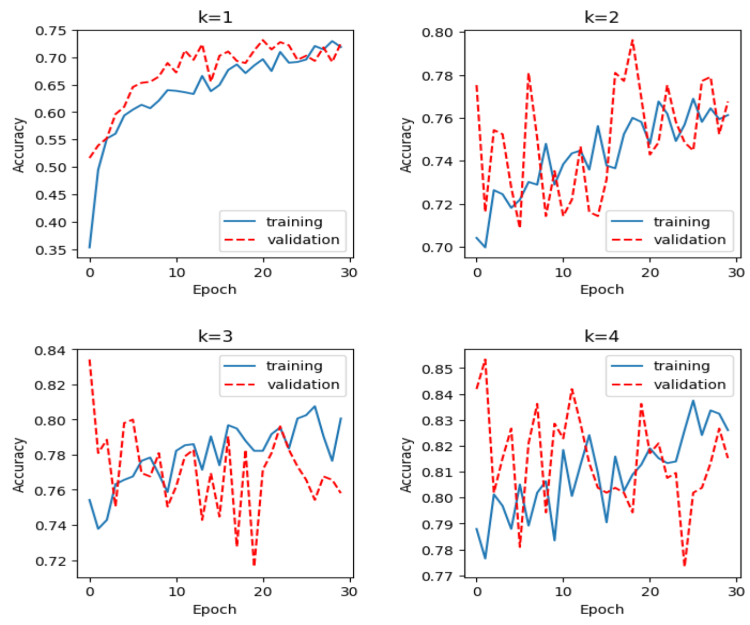

Figure 1a: Train and Val acc for 4-folds (Problem 1 Simple Network)

(P1 Complex Network)

7 Conv layers with these # of filters: 32, 64, 128, 256, 256, 512, 512; filter size (3 x 3)

Dense layers: 512, 6 nodes

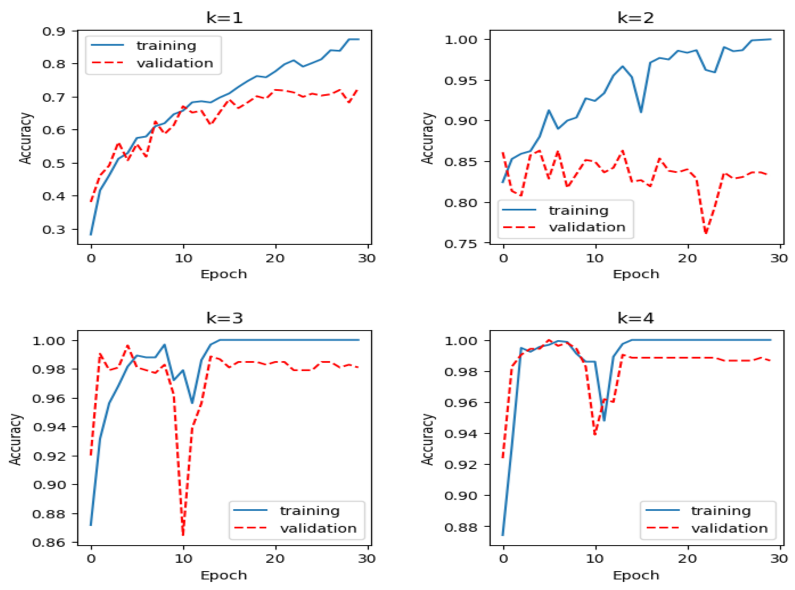

Figure 1b: Train and Val acc for 4-folds (Problem 1 Complex Network)

Table 1: Training, Validation, and Test Accuracy for Problem 1

Model	Training Accuracy	Validation Accuracy	Test accuracy
Problem 1 Simple	0.8292	0.8152	0.7683
Problem 1 Complex	1.0	0.9867	0.7588

Underfitting: The simple and complex network are not underfitting as all training and validation accuracies are above 0.5 as seen in Table 1.

Overfitting: The test accuracy of 0.7683 is a drop from the training and validation accuracies which suggests there may be some level of overfitting. The complex network is severely overfitting as the Training accuracy is 1.0. This is evident after epoch 3 in both the 3rd and 4th folds of the complex model in Figure 1b.

Generalization: Both test and validation accuracies are less than the Training accuracy, therefore the complex model poorly generalizes. In comparison of the test accuracies in Table 1, the simple model is more generalizable than the complex model.

Problem 2

In this problem, the complex model from Problem 1 was used to experiment with data augmentation, dropout layer, weight decay, and a combination of all these options. Some parameters were kept constant for all sub parts in Problem 2: Adam optimizer, Epochs (30), Batch size (35), Num of images (350), and number of filters in convolutional layers: 32, 64, 128, 256, 256, 512, 512. And 2 Dense layers with 512 and 6 nodes respectively.

a) Data Augment

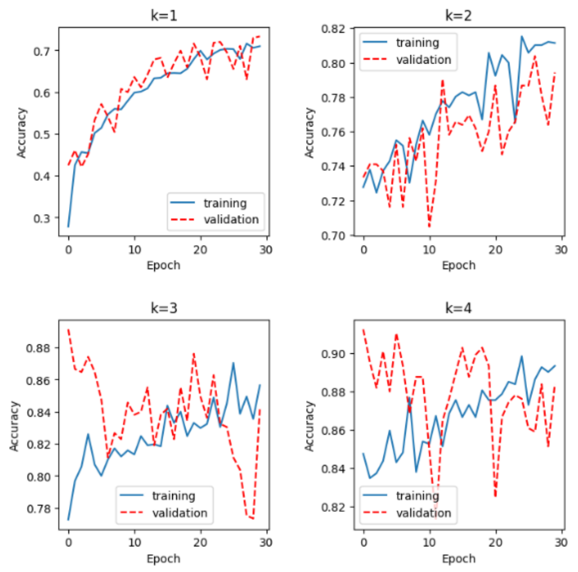

Figure 2a: Train and Val acc for 4-folds (Problem 2 Data Augment only)

b) Dropout

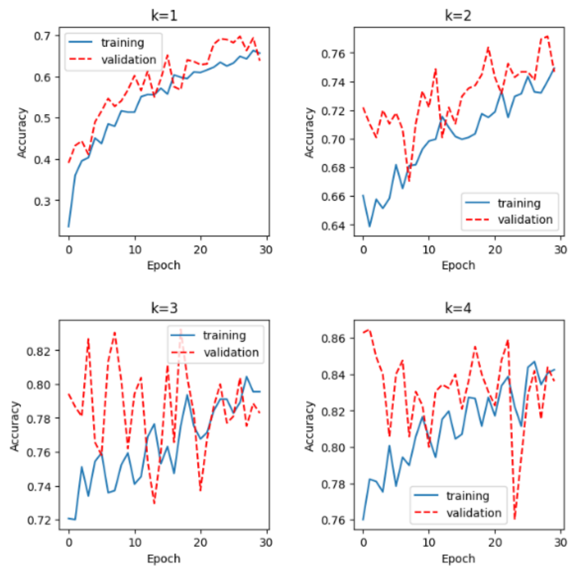

Figure 2b: Training and Val Acc for 4-Folds (Problem 2 Dropout)

c) L2 regularize

L2 Regularizer = 0.01

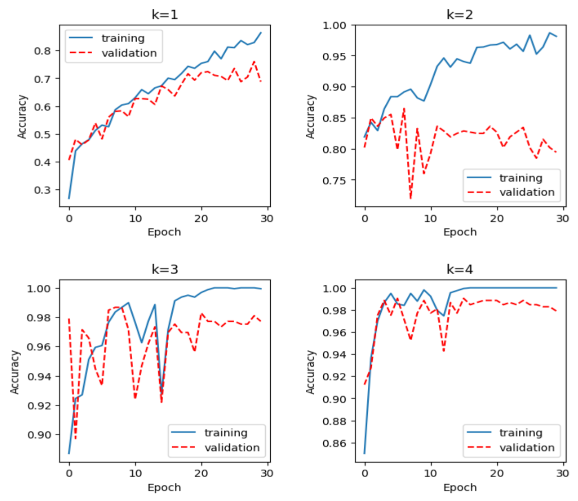

Figure 2c: Training and Val acc for 4-folds (Problem 2 L2 Regularizer)

d) Combine all

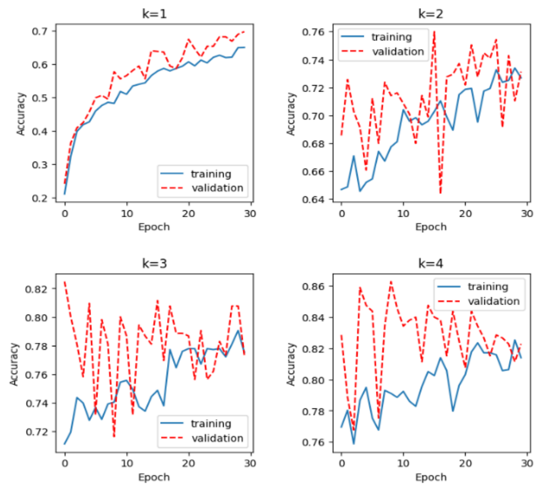

Figure 2d: Training and Val acc for 4-folds (Problem 2 Combine all)

Table 2: Training, Validation, and Test Accuracy for Problem 2

Model	Training Accuracy	Validation Accuracy	Test Accuracy
Data Augment	0.8951	0.8838	0.7963
Dropout Layer	0.8470	0.8362	0.7916
L2 Regularizer	1.0	0.9886	0.7304
Combine all	0.8039	0.8229	0.7845

Underfitting: Table 2 indicates that all accuracies are higher than 0.5. Since they are higher in chance, it can be concluded that there is no underfitting in these 4 options of Problem 2.

Overfitting: In all four of these options, there is only a max of 0.02 difference between Training and Validation accuracy, therefore overfitting has been heavily minimized. In Figure 2a (Data augment) and 2b (Dropout Layer), towards the end of epoch 30 in the 4th fold, the validation and training accuracy are extremely close to each other so not much overfitting. After 6th epoch, the model L2 Regularizer can be seen overfitting as seen in Figure 2c and Table 2 shows training accuracy of 1 indicates severe overfitting.

Generalization: As seen in Table 2, all models are generalizable as they performed more than 70% on unseen test data. Moreover, test accuracy of all models in Problem 2 increased significantly when compared to the Test accuracy of both the Simple and Complex model of Problem 1 as seen in Table 1. Data augment and Dropout layer models performed the best on unseen data, where Test Accuracy of Data Augment is 0.7963 as seen in Table 2.

The ‘Data Augment’ model of Problem 2 will be considered the best model as it scored the highest in Test accuracy (0.7963) as seen in Table 2.

Problem 3 Num images used = 200

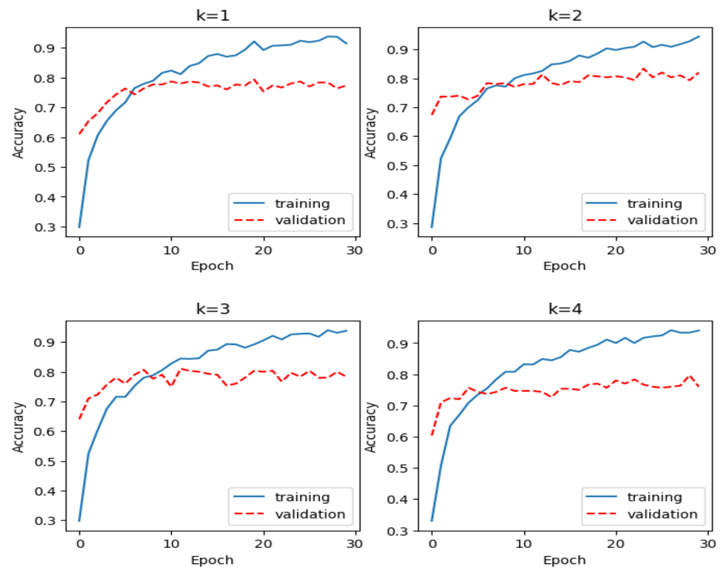

Figure 3: Train and Val acc for 4-folds (Problem 3)

Table 3: Training, Validation, and Test Accuracy for Problem 3

Model	Training Acc	Validation Acc	Test Acc
Problem 3	0.9318	0.7333	0.7959

Underfitting: As can be seen in Table 3, the training and validation accuracies of Problem 3 are both above 0.5 therefore this model is not underfitting.

Overfitting: Figure 3 indicates that somewhere after the 8th epoch, overfitting takes place in all the 4-folds. To avoid overfitting, this model should only run for about 10 epochs just to be safe.

Generalization: It can be seen in Table 3 that both Validation and Test accuracies are almost close to each other and lower than the Training accuracy. Since the model performed better on unseen data than validation accuracy, it can be concluded that Problem 3 Model is generalizable. Since fewer images (200) was used to train the Densely Connected classifier in Problem 3 compared to the images (350) used in all other models, the Test accuracy may be lower than what it can achieve.

Problem 4

Froze first 4 blocks

Dense layers: 256, 6 nodes

Adam optimizer; Epochs = 30; Batch size = 20; Num Images = 350

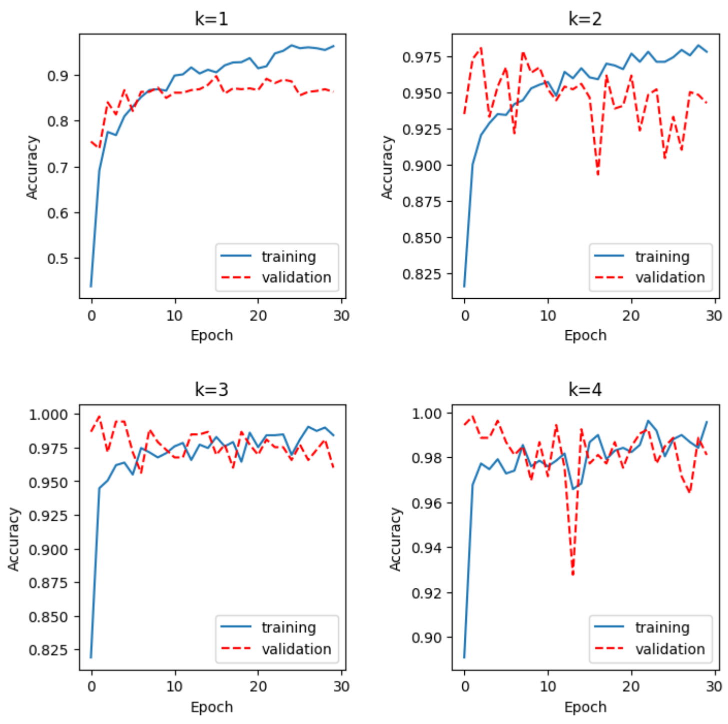

Figure 4: Train and Val acc for 4-folds (Problem 4)

Underfitting: The model in Problem 4 is not underfitting as there's only a 0.03 difference between the Training and Validation Accuracy as can be seen in Table 4.

Overfitting: The model is overfitting after epoch 3 as presented in Figure 4. A solution would be to use fewer epochs/ early stopping.

Generalization: This model performed well on Test data yielding Test accuracy of 0.8511, therefore it's generalizable.

Table 4: Comparison of Accuracy in Problem 2 (Combine All), 3, 4

Model	Training Acc	Validation Acc	Test Acc
Problem 2 (Data Augment)	0.8951	0.8838	0.7963
Problem 3	0.9318	0.7333	0.7959
Problem 4	0.9866	0.9581	0.8511

It can be concluded that the model used in Problem 4 (VGG16) along with the Densely connected classifier extended on top of the convolution base performed the best compared to the other models in Problem 2 (Data Augment) and Problem 3. Even though the first four blocks of the conv base were frozen, the 5th block along with the densely connected layers were trained. Two dropout layers and data augmentation were used in Problem 4 to minimize overfitting.

Examples of misclassified samples in Problems (2, 3, 4):

Problem 2 (Data augment)

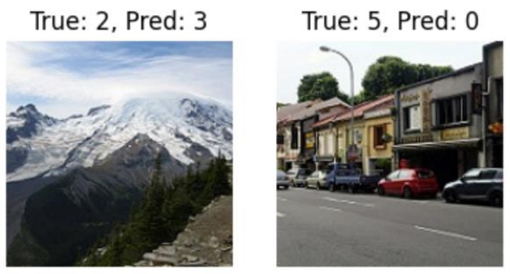

Figure 5a: 2 examples of misclassification in Problem 2 (data augment)

It seems that the model in Problem 2 incorrectly classified the 1st image in Figure 5a as 'Mountain' when it's actually a 'Glacier'.

Problem 3

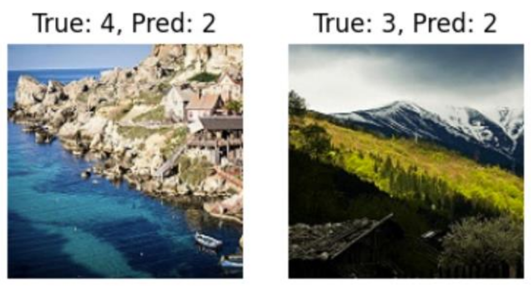

Figure 5b: 2 examples of misclassification in Problem 3

It seems that the model in Problem 3 incorrectly classified the 1st image in Figure 5b as 'Glacier' when it is a 'Sea'.

Problem 4

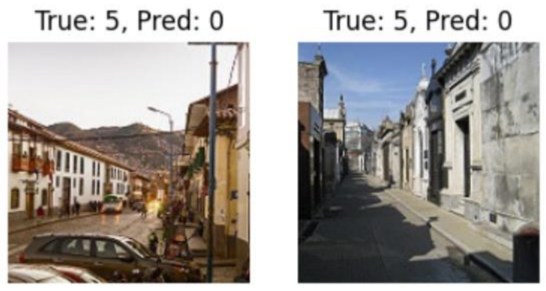

Figure 5c: 2 examples of misclassification in Problem 4

It can be seen in Figure 5c, that the 2nd and 3rd image were incorrectly predicted as 'Building' (0) when they 'Street' (5).

Bonus

Used image 20060 in the buildings directory inside the test directory.

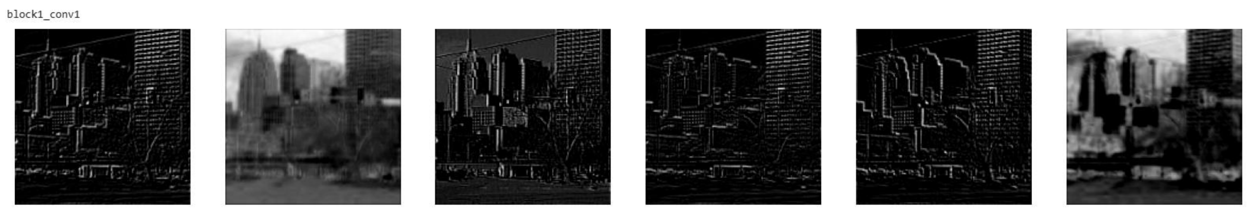

Figure 6a: convolution filter at Block1Conv1

The convolution filter at block 1 shows lots of details like the edges and texture of the image in Figure 6a. For example, each image neatly outlines the shape of the building. This can only be noticed in the earlier convolution filters.

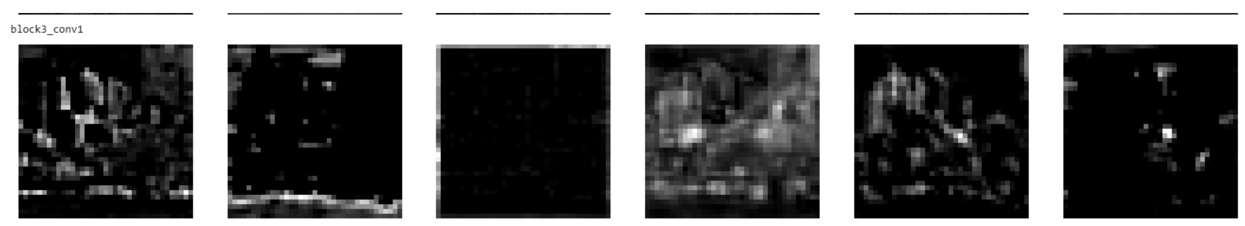

Figure 6b: convolution filter at Block3Conv1

The convolution filter at block 3 shows abstract features of the image in Figure 6b. For example, parts of the image or its shape can be seen at this level.

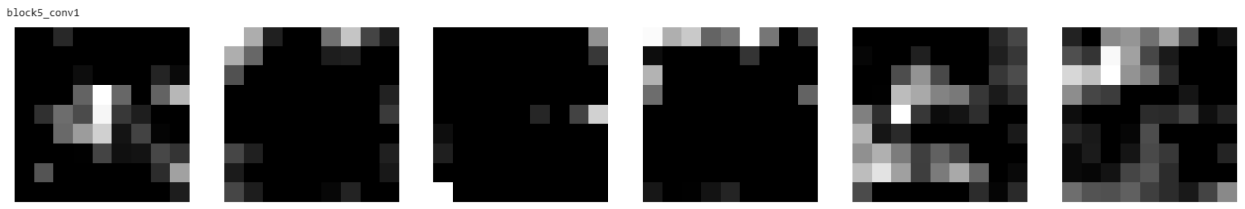

Figure 6c: convolution filter at Block5Conv1

The convolution filter at block extracts abstract features of the image in Figure 6c. Very few details of the image can be visualized. It all seems like a bunch of pixels of white and black blocks.

The earlier convolutional filters should be used to view the details of the image whereas the later layers only show the abstract versions of the image.