# TBMI26 – Computer Assignment Reports Deep Learning

Deadline – March 14 2021

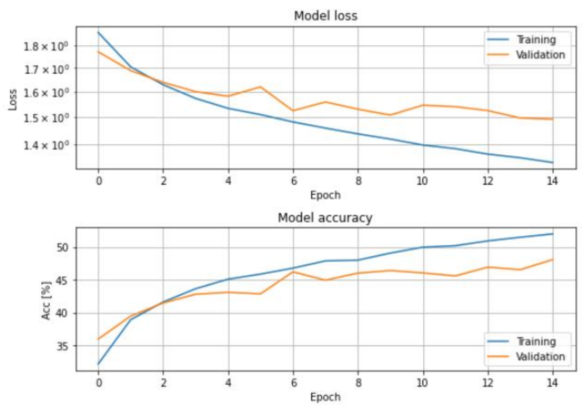
Author/-s:
Abhijeet Anand(abhan872)
Jaskirat Marar(Jasma356)

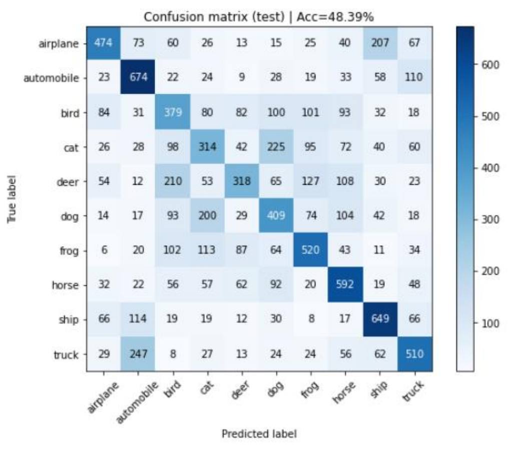
In order to pass the assignment, you will need to answer the following questions and upload the document to LISAM. Please upload the document in PDF format. **You will also need to upload the Jupyter notebook as an HTML-file (using the notebook menu: File -> Export Notebook As…)**. We will correct the reports continuously so feel free to send them as soon as possible. If you meet the deadline you will have the lab part of the course reported in LADOK together with the exam. If not, you'll get the lab part reported during the re-exam period.

1. **The shape of X_train and X_test has 4 values. What do each of these represent?**
   Shapes of data:
   X_Train: (50000, 32, 32, 3)
   X_Test: (10000, 32, 32, 3)

   There are 10 classes/objects that needs to be classified.
   The 4 values in the data signifies no of images, no of pixels in terms of height, no of pixels in terms of width, RGB(color channel) respectively.

2. **Train a Fully Connected model that achieves above 45% accuracy on the test data. Provide a short description of your model and show the evaluation image.**
   To train a fully connected model, we used 4 dense layers. Each dense layer has 128 hidden units. These layers use 'relu' as activation function. In the final layer or output layer we use a dense layer which has 10 hidden units and 'softmax' activation function. We received an accuracy of 48.4% on the test data. The result is shown below.

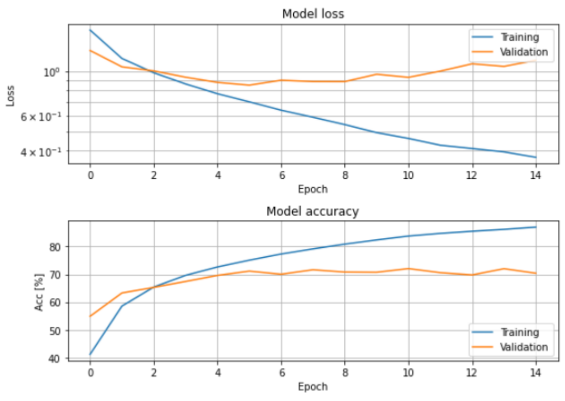


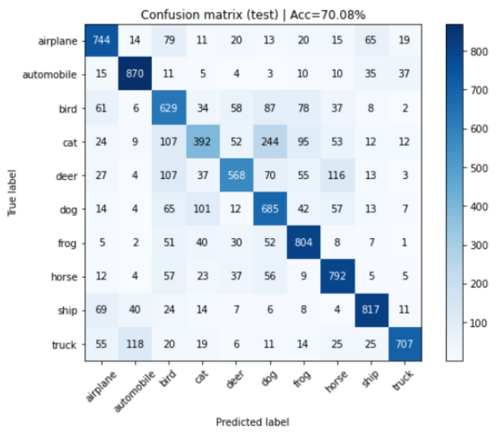


| | airplane | automobile | bird | cat | deer | dog | frog | horse | ship | truck |
|---|---|---|---|---|---|---|---|---|---|---|
| airplane | 474 | 73 | 60 | 26 | 13 | 15 | 25 | 40 | 207 | 67 |
| automobile | 23 | 674 | 22 | 24 | 9 | 28 | 19 | 33 | 58 | 110 |
| bird | 84 | 31 | 379 | 80 | 82 | 100 | 101 | 93 | 32 | 18 |
| cat | 26 | 28 | 98 | 314 | 42 | 225 | 95 | 72 | 40 | 60 |
| deer | 54 | 12 | 210 | 53 | 318 | 65 | 127 | 108 | 30 | 23 |
| dog | 14 | 17 | 93 | 200 | 29 | 409 | 74 | 104 | 42 | 18 |
| frog | 6 | 20 | 102 | 113 | 87 | 64 | 520 | 43 | 11 | 34 |
| horse | 32 | 22 | 56 | 57 | 62 | 92 | 20 | 592 | 19 | 48 |
| ship | 66 | 114 | 19 | 19 | 12 | 30 | 8 | 17 | 649 | 66 |
| truck | 29 | 247 | 8 | 27 | 13 | 24 | 24 | 56 | 62 | 510 |

3. **Compare the model from Q2 to the one you used for the MNIST dataset in the first assignment, in terms of size and test accuracy. Why do you think this dataset is much harder to classify than the MNIST handwritten digits?**
   Even though both the data looks very similar, CIFAR10 is much more complex to classify. We observed around 96% of accuracy in the MNIST dataset with just one dense fully connected layer. However, in case of CIFAR10, we could get only 48.4% with 4 dense fully connected layers. One of the reasons is size of the image.
   MNIST (each image): 8 x 8 x 1
   CIFAR10 (each image): 32 x 32 x 3
   In CIFAR10, we have 3 different color channels, where as MNIST is grayscale. The classes are very inconsistent in CIFAR10, which makes it very harder to classify. The same object can be found far, close, turned at some angle or more.

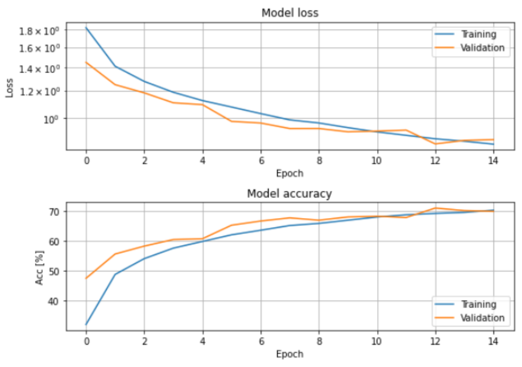
4. **Train a CNN model that achieves at least 62% test accuracy. Provide a short description of your model and show the evaluation image.**
   To a CNN model, we utilized conv2D and maxpooling2D from keras.layers. We successfully achieved an accuracy of 70.1% in test data. To achieve this, 3 block of [convolution-activation-pooling] layers stacked over one another. We used 64 filters in all the (3, 3) convolution network. The activation function used was 'relu'. We added (2, 2) maxpooling in all the blocks after each convolution. At the end, the results were flattened, and a dense layer of 64 hidden units with 'relu' activation function was added, followed by the output dense layer with 10 hidden layers and 'softmax' activation function.

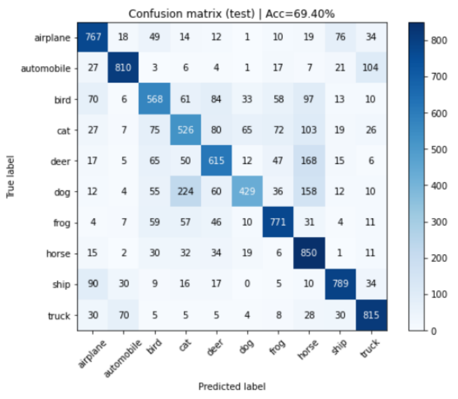




5. **Compare the CNN model with the previous Fully Connected model. You should find that the CNN is much more efficient, i.e. achieves higher accuracy with fewer parameters. Explain in your own words how this is possible.**
   With compared to previous Fully connected model, we observed CNN gives us higher accuracy. We had 444,170 parameters in the Fully connected model giving us 48% of accuracy. However, we noticed 141, 898 parameters giving us 70% of accuracy. This magic was possible due to adding convolution layers. In CNN model, each pixel is not treated as a feature unlike to Fully Connected model. We used (3, 3) kernel which worked as sliding window from left to right, going over an entire image and storing data in fewer dimension. We used big number of filters to extract significant features. Finally, we used a maxpooling of (2,2), which signified that output will a max of (2,2) squares.

6. **Train the CNN-model with added Dropout layers. Describe your changes and show the evaluation image.**
   We added a dropout layer with a drop rate of 0.4 at the end. The accuracy achieved after adding dropout layer was 69.4%

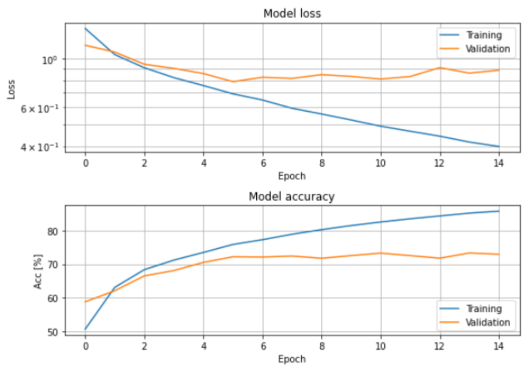


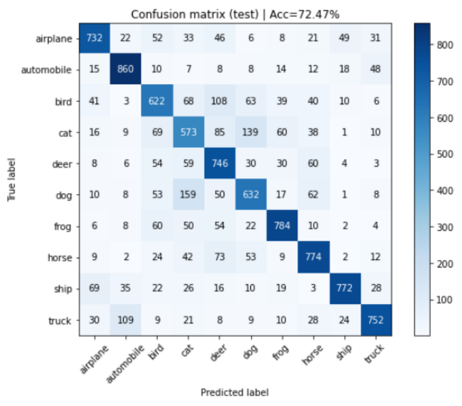

Confusion matrix (test) | Acc=69.40%

| True label \ Predicted label | airplane | automobile | bird | cat | deer | dog | frog | horse | ship | truck |
|---|---|---|---|---|---|---|---|---|---|---|
| airplane | 767 | 18 | 49 | 14 | 12 | 1 | 10 | 19 | 76 | 34 |
| automobile | 27 | 810 | 3 | 6 | 4 | 1 | 17 | 7 | 21 | 104 |
| bird | 70 | 6 | 568 | 61 | 84 | 33 | 58 | 97 | 13 | 10 |
| cat | 27 | 7 | 75 | 526 | 80 | 65 | 72 | 103 | 19 | 26 |
| deer | 17 | 5 | 65 | 50 | 615 | 12 | 47 | 168 | 15 | 6 |
| dog | 12 | 4 | 55 | 224 | 60 | 429 | 36 | 158 | 12 | 10 |
| frog | 4 | 7 | 59 | 57 | 46 | 10 | 771 | 31 | 4 | 11 |
| horse | 15 | 2 | 30 | 32 | 34 | 19 | 6 | 850 | 1 | 11 |
| ship | 90 | 30 | 9 | 16 | 17 | 0 | 5 | 10 | 789 | 34 |
| truck | 30 | 70 | 5 | 5 | 5 | 4 | 8 | 28 | 30 | 815 |

7. **Compare the models from Q4 and Q6 in terms of the training accuracy, validation accuracy, and test accuracy. Explain the similarities and differences (remember that the only difference between the models should be the addition of Dropout layers).**
   **Hint: what does the dropout layer do at test time?**
   In Q4, we can see,
   Training accuracy: ~ 88%
   Validation accuracy: ~ 70%
   Test accuracy: ~ 70%
   Since, we observed overfitting in the CNN model, we added a dropout layer with drop rate 0.4 at the end. The drop rate indicates that 40% of the nodes will have 0 output. After adding dropout layer, the overfitting issue was resolved. However, the accuracy remained almost same. The total number of parameters was reduced to 73, 418, because the first conv layer has 32 filters instead of 64 this time.
   The accuracies are:
   Training accuracy: ~ 70%
   Validation accuracy: ~ 70%
   Test accuracy: ~ 70%
   This model looks more efficient and reliable.
   Dropout is a regularizing technique; it regularizes the parameters to reduce dependency over specific parameter values. It uses methods like L1 and L2 which reduce overfitting by modifying the cost function. The dropout is not implemented on test/unseen data.

8. **Train the CNN model with added BatchNorm layers and show the evaluation image.**
   added the Batch normalization layers and trained the model. We achieved an accuracy of 72.4%. We The results are shown below:



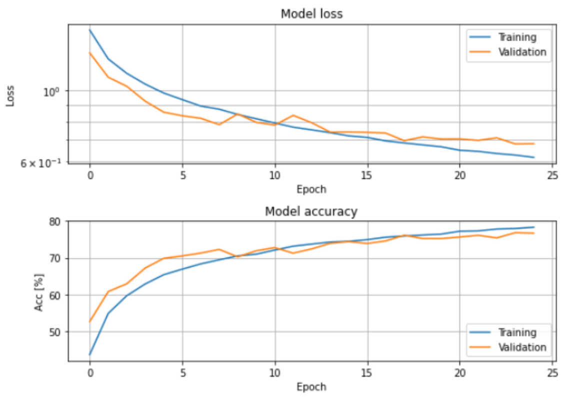

Confusion matrix (test) | Acc=72.47%

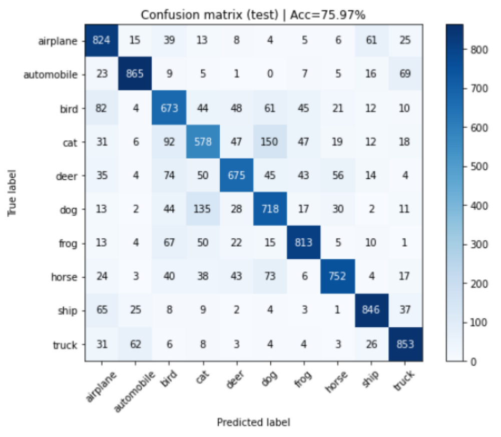
| True label \ Predicted label | airplane | automobile | bird | cat | deer | dog | frog | horse | ship | truck |
|---|---|---|---|---|---|---|---|---|---|---|
| airplane | 732 | 22 | 52 | 33 | 46 | 6 | 8 | 21 | 49 | 31 |
| automobile | 15 | 860 | 10 | 7 | 8 | 8 | 14 | 12 | 18 | 48 |
| bird | 41 | 3 | 622 | 68 | 108 | 63 | 39 | 40 | 10 | 6 |
| cat | 16 | 9 | 69 | 573 | 85 | 139 | 60 | 38 | 1 | 10 |
| deer | 8 | 6 | 54 | 59 | 746 | 30 | 30 | 60 | 4 | 3 |
| dog | 10 | 8 | 53 | 159 | 50 | 632 | 17 | 62 | 1 | 8 |
| frog | 6 | 8 | 60 | 50 | 54 | 22 | 784 | 10 | 2 | 4 |
| horse | 9 | 2 | 24 | 42 | 73 | 53 | 9 | 774 | 2 | 12 |
| ship | 69 | 35 | 22 | 26 | 16 | 10 | 19 | 3 | 772 | 28 |
| truck | 30 | 109 | 9 | 21 | 8 | 9 | 10 | 28 | 24 | 752 |

9. **When using BatchNorm one must take care to select a good minibatch size. Describe what problems might arise if the wrong minibatch size is used.**
   **You can reason about this given the description of BatchNorm in the Notebook, or you can search for the information in other sources. Do not forget to provide links to the sources if you do!**
   In Batch normalization, our intention is to put the minibatch's mean to 0 and standard deviation to one. The selection of minibatch size is very crucial. The estimation will be very accurate if the batch size is fairly large. The accuracy keeps decreasing as batch size decreases. So, selecting small batch size will reduce accuracy. The downside of using a smaller batch size is that the model is not guaranteed to converge to the global optima. Why not select large then? Larger batch size allows faster computation, but it leads to poor generalization. Larger batch size is suitable for a single update step, but for an epoch it is not good, as the number of updates is reduced. We can't select a batch size of 1, because it will be mean of itself. Thus, the selection is very crucial and can be very unstable.

10. **Design and train a model that achieves at least 75% test accuracy in at most 25 epochs. Explain your model and motivate the design choices you have made and show the evaluation image.**

    We are using three blocks of [convolution-batch-activation-pooling]. The activation function of 'relu' is used throughout other than the output layer which has 'softmax' activation function. The dropout with drop rate of 0.4 is implemented once after the above blocks. In the first layer the filter size is 32, whereas 64 in the rest. This method is carried out in order to reduce the number of parameters. The kernel size was set to (3,3) in all the convolution layers. We achieved an accuracy of 75.9%.


Model loss
Training
Validation
Loss
$10^0$
$6 \times 10^{-1}$
Epoch
Model accuracy
Acc [%]
Training
Validation
Epoch



Confusion matrix (test) | Acc=75.97%
airplane 824 15 39 13 8 4 5 6 61 25
automobile 23 865 9 5 1 0 7 5 16 69
bird 82 4 673 44 48 61 45 21 12 10
cat 31 6 92 578 47 150 47 19 12 18
deer 35 4 74 50 675 45 43 56 14 4
dog 13 2 44 135 28 718 17 30 2 11
frog 13 4 67 50 22 15 813 5 10 1
horse 24 3 40 38 43 73 6 752 4 17
ship 65 25 8 9 2 4 3 1 846 37
truck 31 62 6 8 3 4 4 3 26 853
True label
Predicted label
airplane automobile bird cat deer dog frog horse ship truck