Deep Learning

Transfer Learning

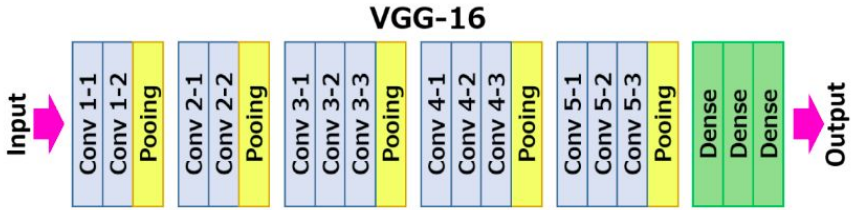
Problem to solve

MAMe: Museum Art Medium dataset

Image classification with 29 classes.

Expected final accuracy be higher than 71%

Pretrained model



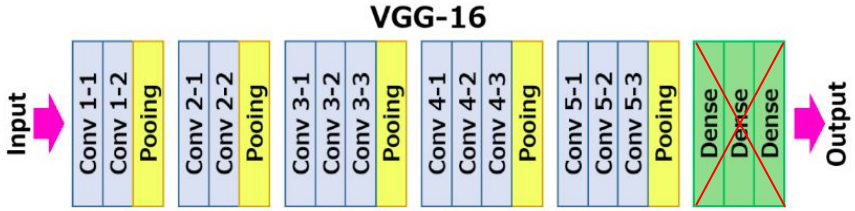
For the pretraining the VGG16 model was used.

Early stopping: val_loss with patience=20

Saved weights: the best val_accuracy

Fine tuning

Initial architectures



Weights for first 13 layers were loaded.

3 different Dense layers architecture were tried.

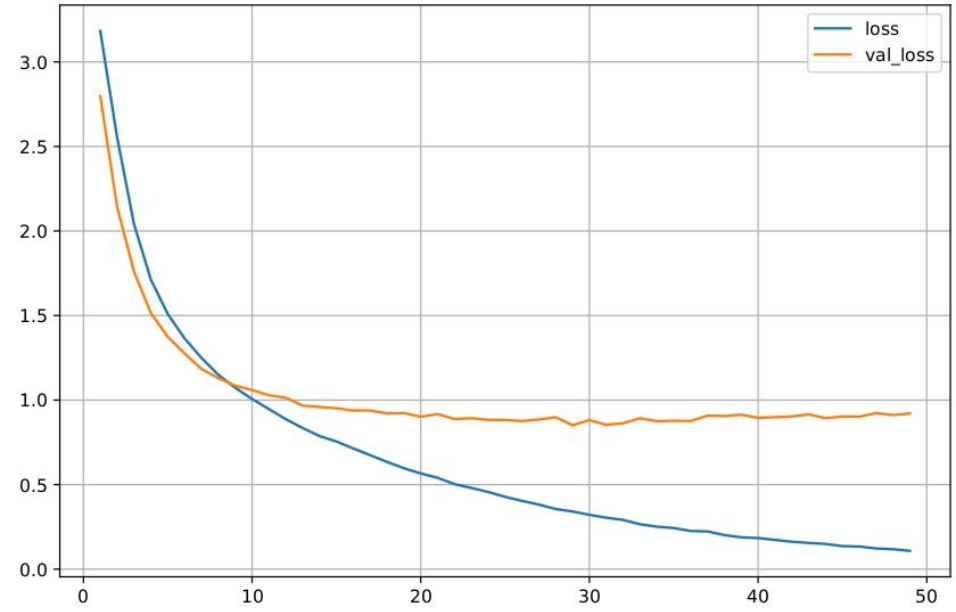
i. Dense(512) -> Dropout(0.2) -> Dense(512) -> Dense(29)
ii. BN -> Dense(512) + L2 -> Dropout(0.3) -> Dense(512) + L2 -> Dense(29)
iii. BN -> Dense(256) + L2 -> Dropout(0.4) -> Dense(256) + L2 -> Dense(256) + L2 -> Dense(29)

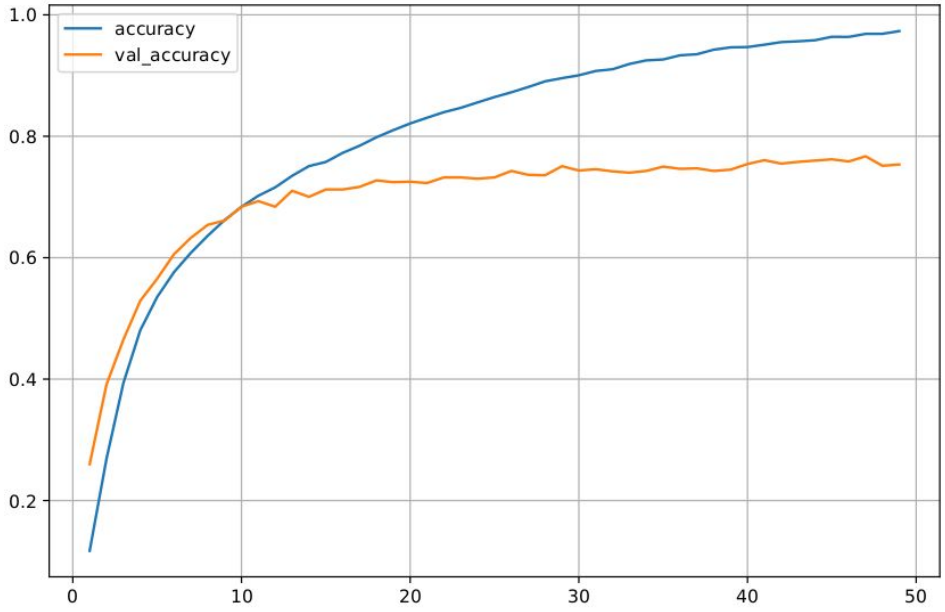
Results

i [Dense(512) -> Dropout(0.2) -> Dense(512) -> Dense(29)]

Loss: 0.1084; Accuracy: 97.33%;

Validation loss: 0.9205; Validation accuracy: 75.36%



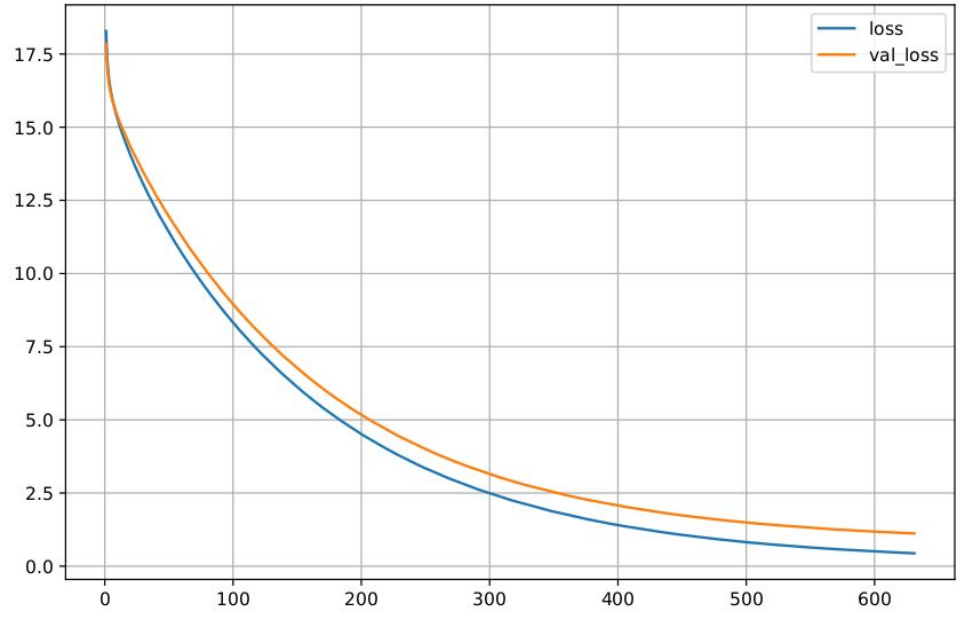


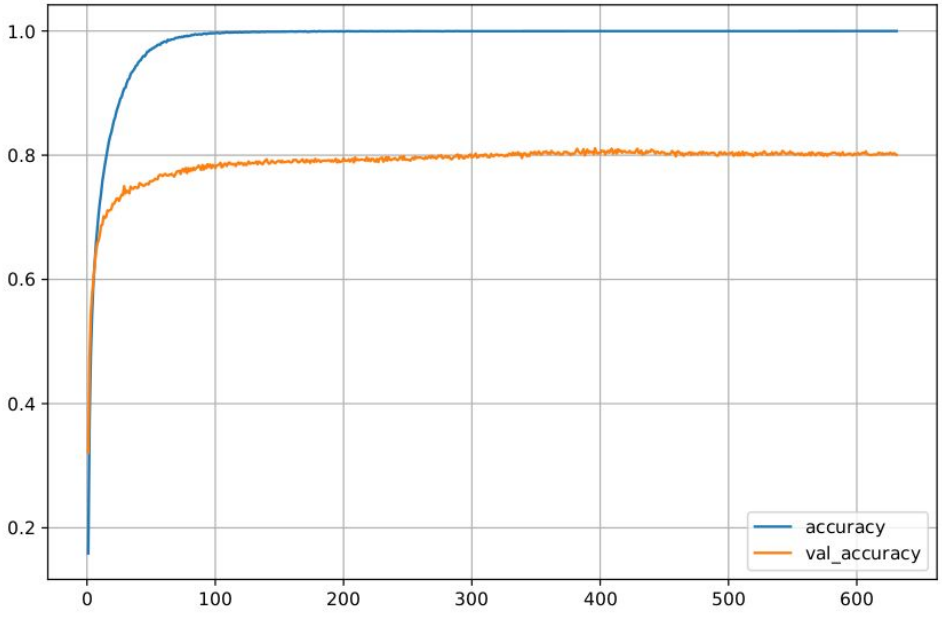
Results

ii [BN -> Dense(512) + L2 -> Dropout(0.3) -> Dense(512) + L2 -> Dense(29)]

Loss: 0.4393; Accuracy: 99.99%;

Validation loss: 1.1183; Validation accuracy: 80.04%



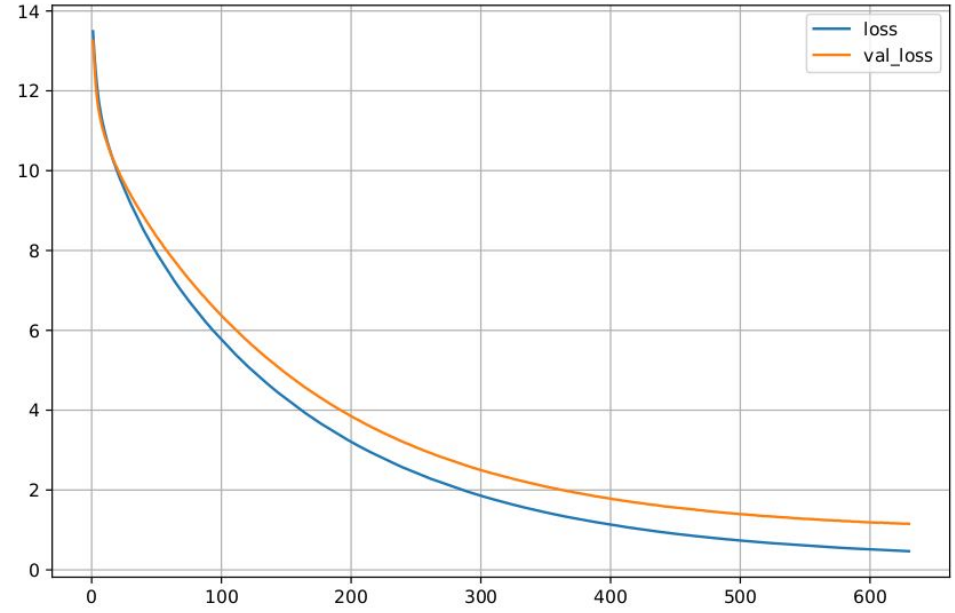


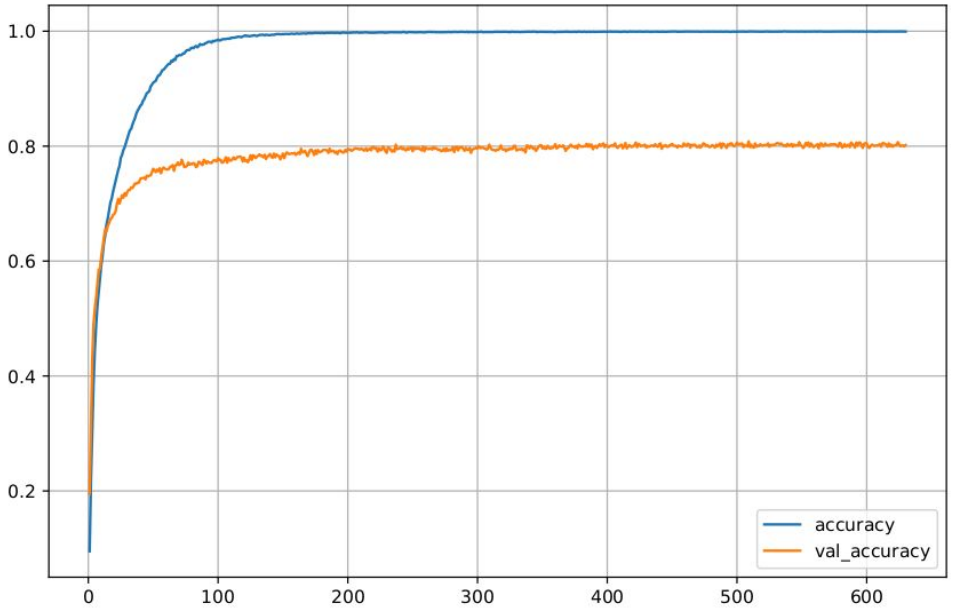
Results

iii [BN -> Dense(256) + L2 -> Dropout(0.4) -> Dense(256) + L2 -> Dense(256) + L2 -> Dense(29)]

Loss: 0.4676; Accuracy: 99.94%;

Validation loss: 1.1532; Validation accuracy: 80.18%





The third architecture was chosen

i. Experiment was repeated with longer time on cluster to reach the early stopping moment.

ii. The same experiment was runned with data augmentation:

rotation_range=20,

width_shift_range=0.2,

height_shift_range=0.2,

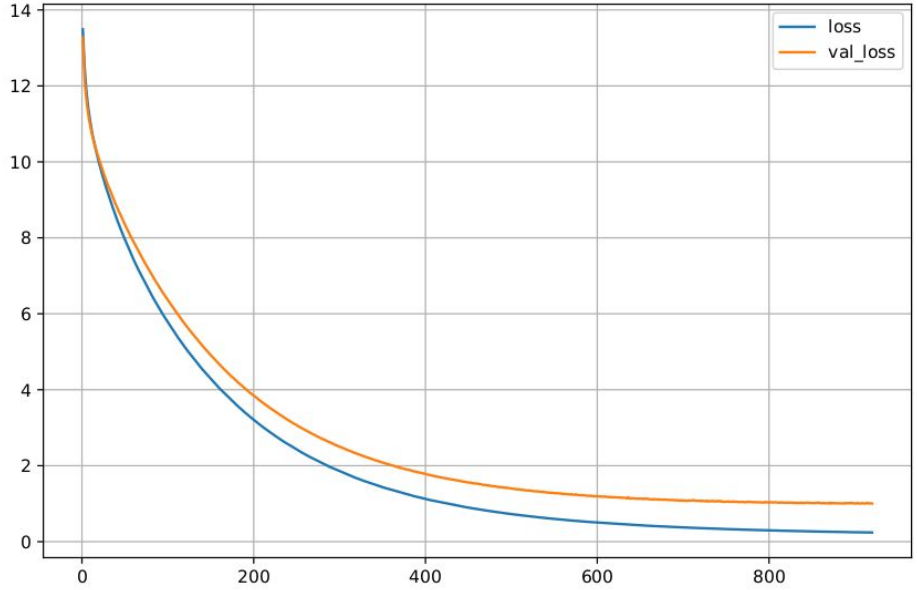
horizontal_flip=True,

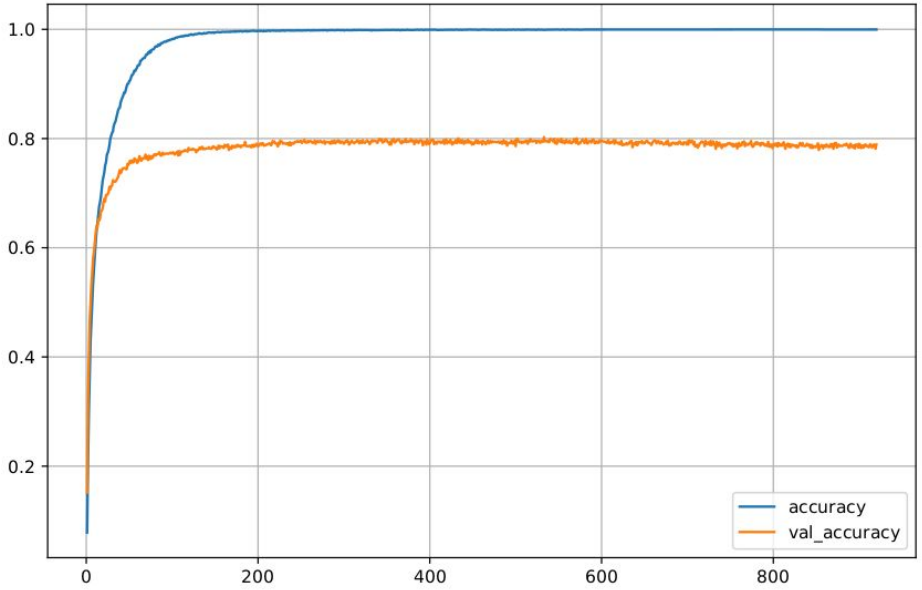
zoom_range=0.2

Results

i Loss: 0.6381; Accuracy: 99.92%;

Validation loss: 1.3175; Validation accuracy: 80.03%

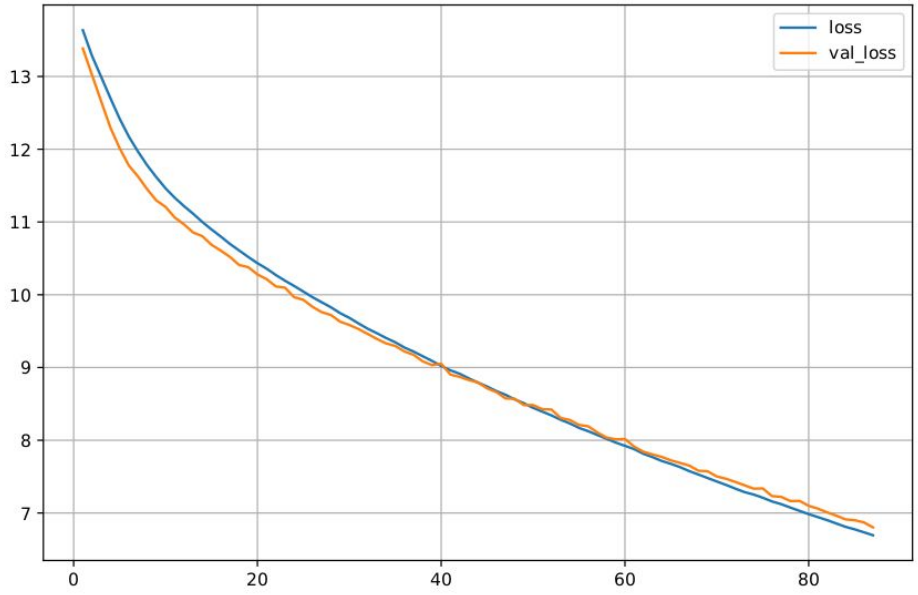


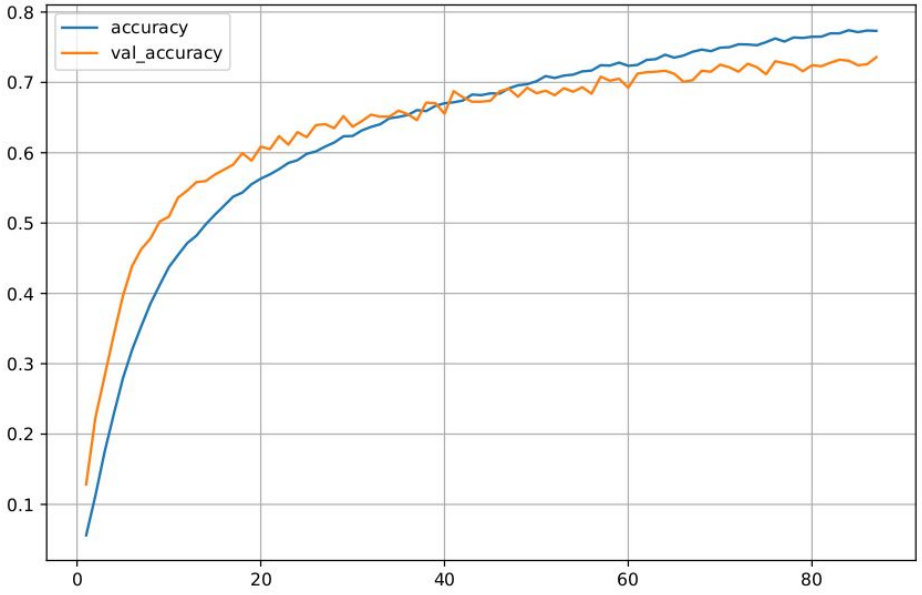


Results

ii Loss: 6.6944; Accuracy: 77.33%;

Validation loss: 6.8026; Validation accuracy: 73.58%

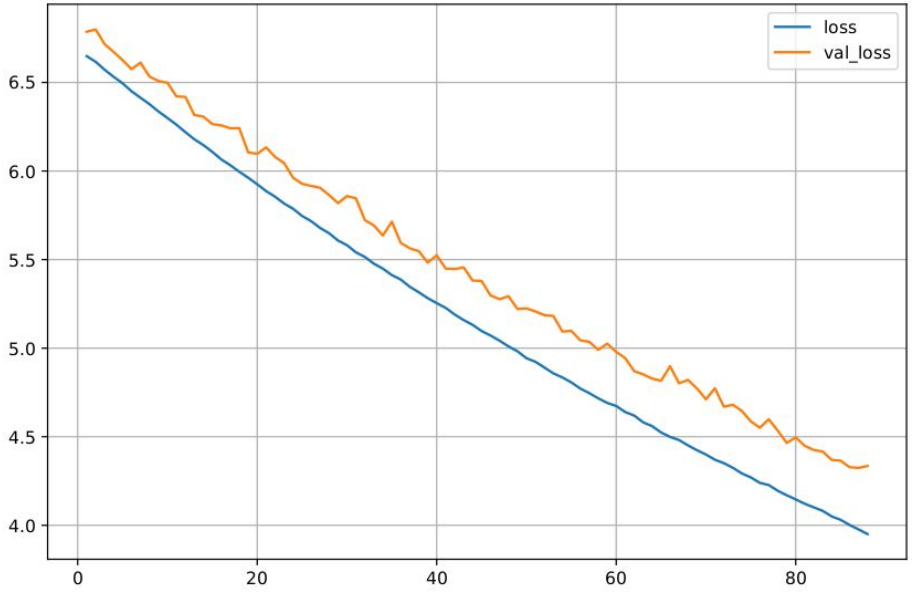


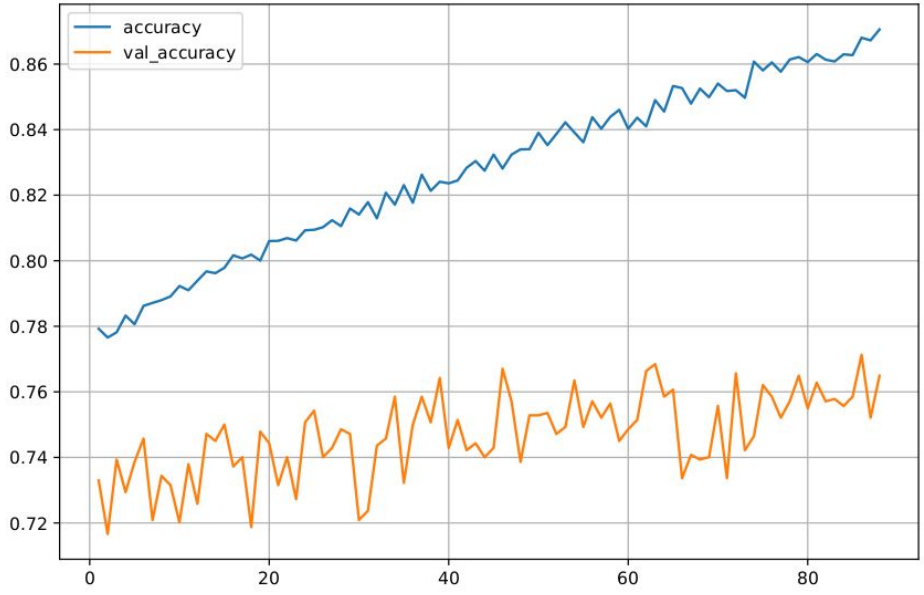


Results

ii Loss: 4.0019; Accuracy: 86.80%;

Validation loss: 4.3281; Validation accuracy: 77.13%





I’ve decided to continue learning from the augmentation part.

The best in my opinion options was to:

- keep using data augmentation
- unfroze layers layer by layer

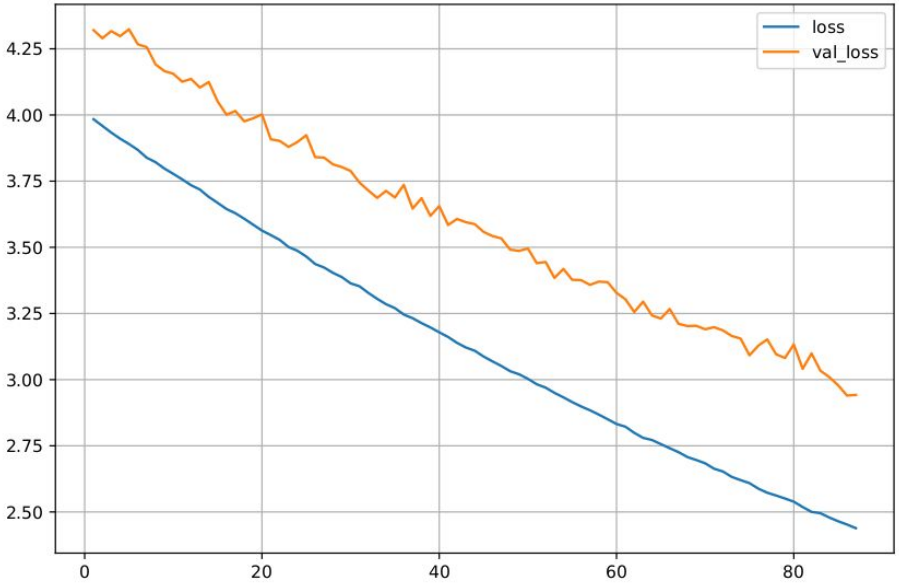
Risks: not enough time to do both at the same time.

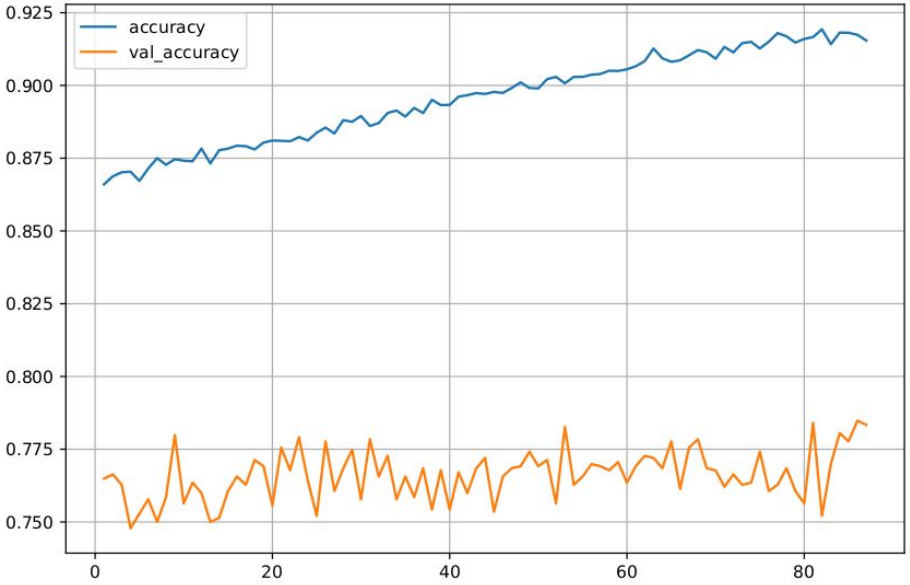
Solution:

I’ve decided to went in both directions (possibility to run two experiments in the same time):

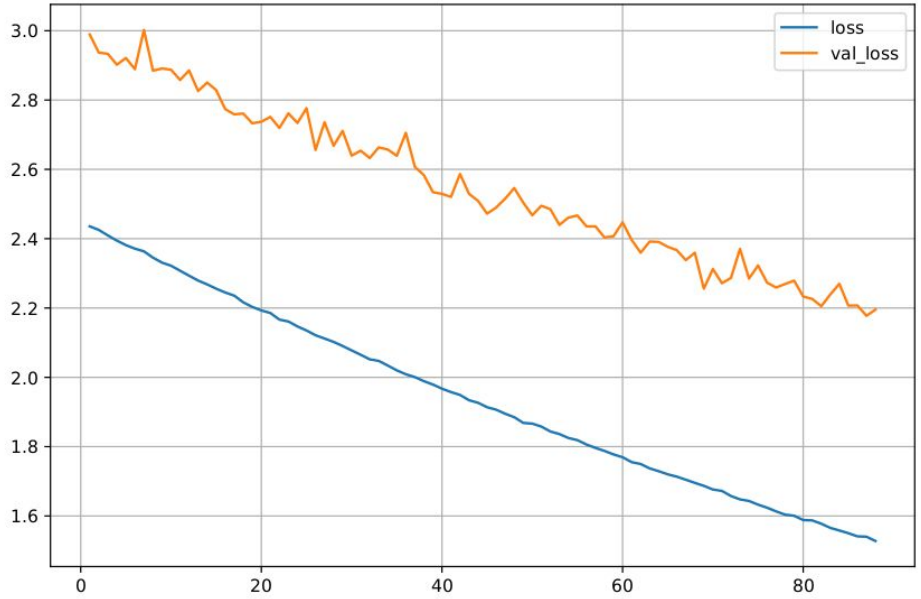
1. Stay with data augmentation but unfroze all layers
2. Resign from data augmentation but unfroze layers in two steps.

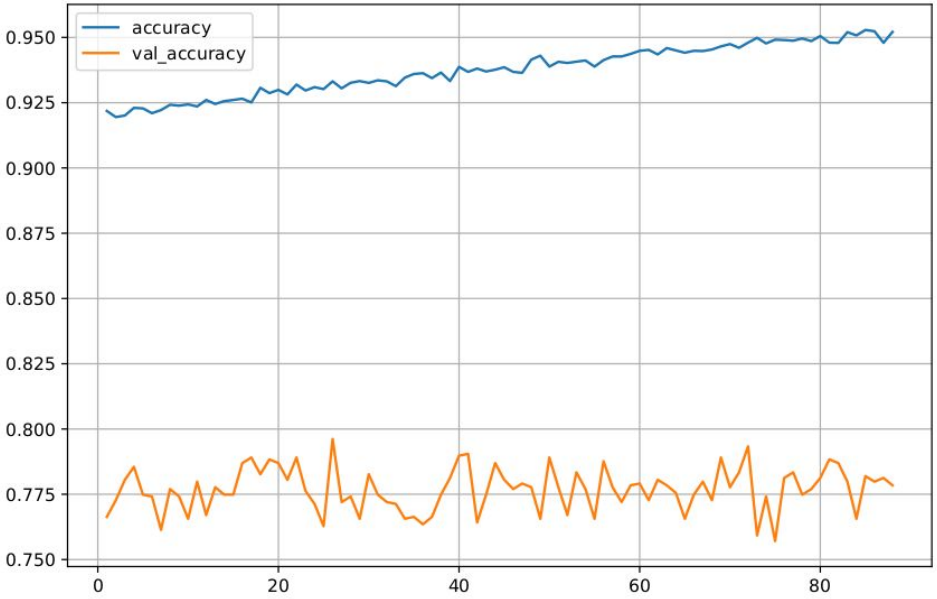
Stay with data augmentation but unfroze all layers (1)





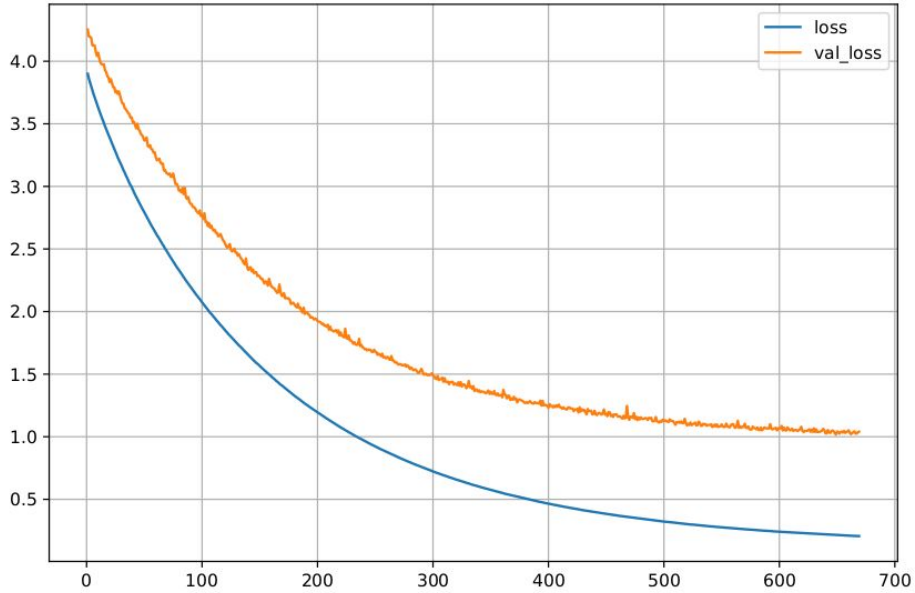
Stay with data augmentation but unfroze all layers (2)

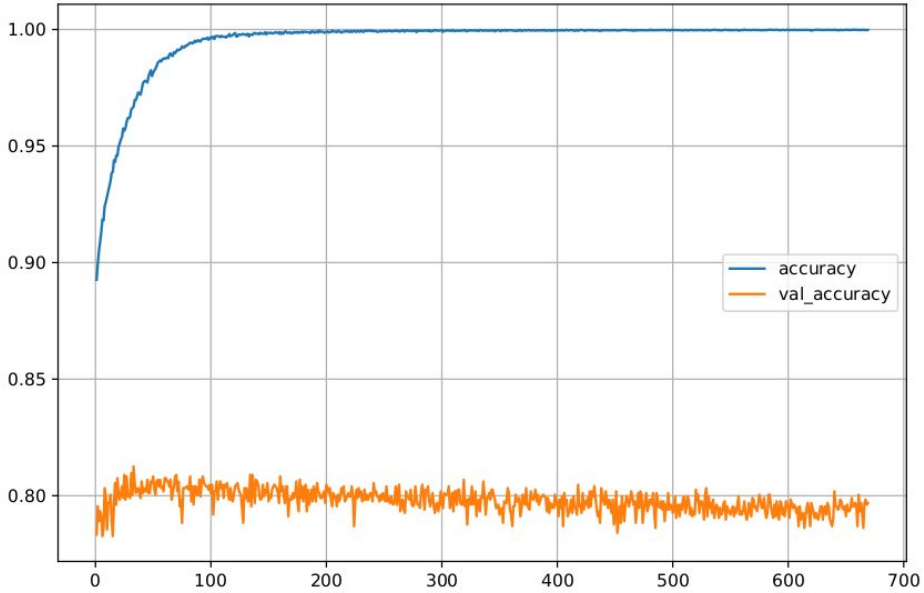




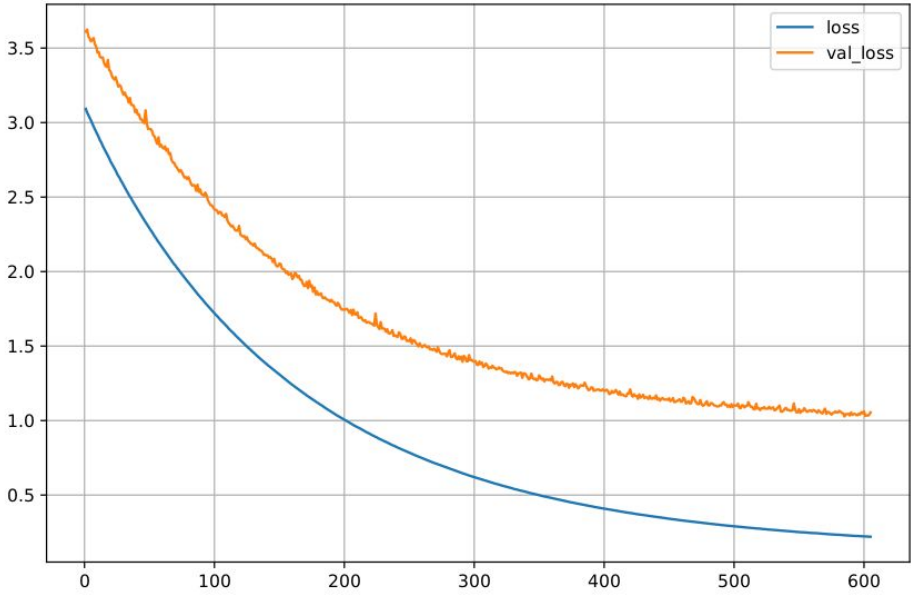
Loss: 2.1214; Accuracy: 93.32%; Validation loss: 2.6561; Validation accuracy: 79.62%

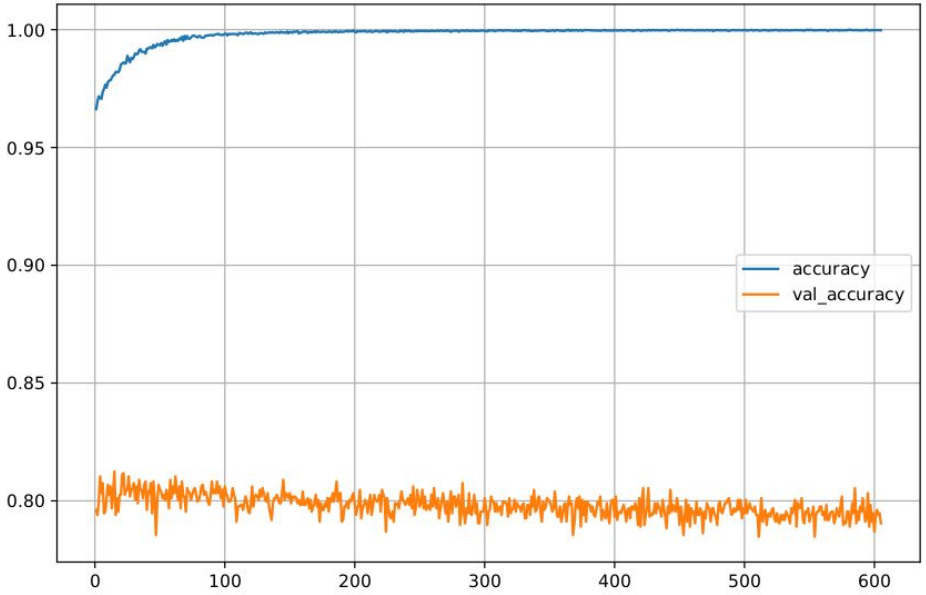
Resign from data augmentation but unfroze layers in two steps. (1)





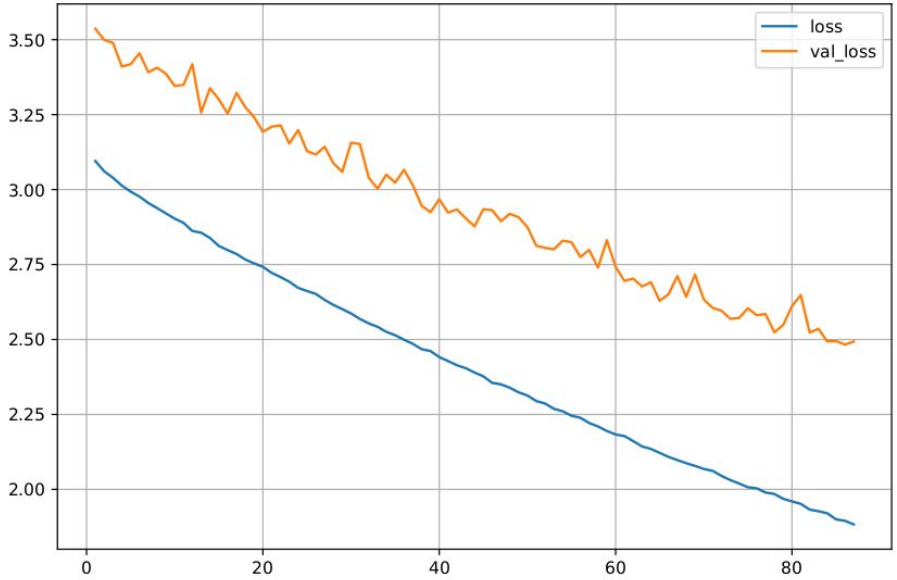
Resign from data augmentation but unfroze layers in two steps. (2)

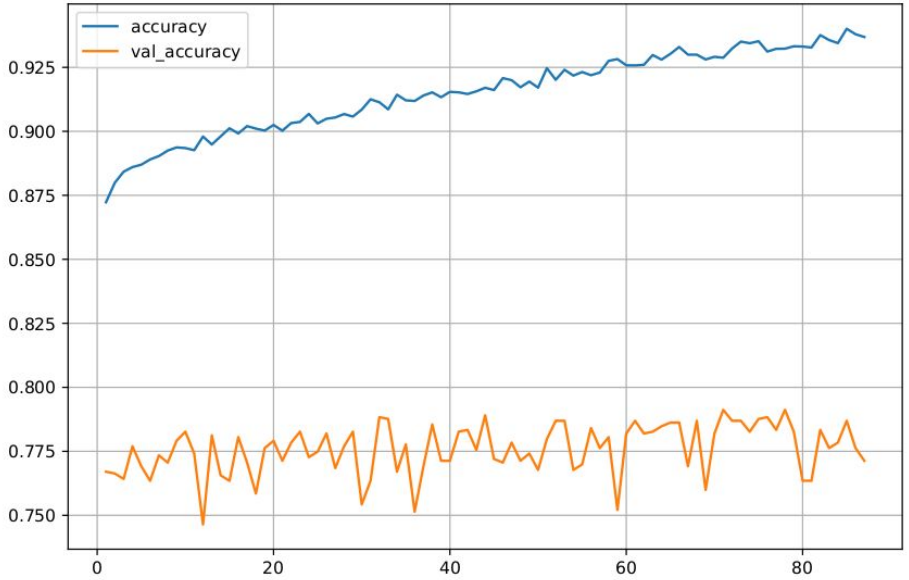




Loss: 2.8288; Accuracy: 98.09%; Validation loss: 3.4019; Validation accuracy: 81.25%

Resign from data augmentation but unfroze layers in two steps. (* After that, I wanted to see if training one time more but with augmentation will help)





Loss: 2.0618; Accuracy: 92.88%; Validation loss: 2.6053; Validation accuracy: 79.12%

Comparison of results

	Loss	Accuracy	Val_loss	Val_accuracy
only dense layers no augmentation	0.6381	99.92	1.3175	80.03
only dense layers augmentation	4.0019	86.80	4.3281	77.13
augmentation all layers unfrozen	2.1214	93.32	2.6561	79.62
no augmentation layers unfrozen steppedly	2.8288	98.09	3.4019	81.25
no augmentation layers unfrozen steppedly + one training with augmentation	2.0618	92.88	2.6053	79.12

Feature extraction

Classificator choosing

Because of huge training time of SVM, Random Forest Classifier was chosen.

Layers to extract features from

'block1_conv1' 'block1_conv2'

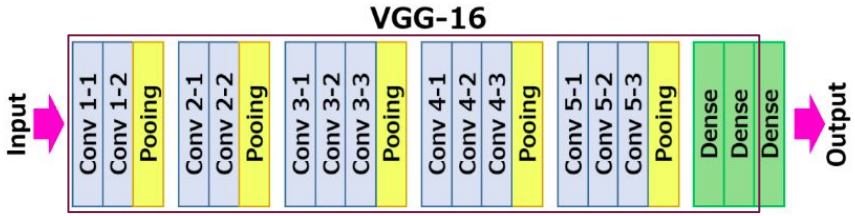
'block2_conv1' 'block2_conv2'

'block3_conv1' 'block3_conv2', 'block3_conv3'

'block4_conv1' 'block4_conv2' 'block4_conv3'

'block5_conv1' 'block5_conv2' 'block5_conv3'

'fc1' 'fc2'



Number of features: 12416

Checked RFC paramthers

DEFAULT VALUES:

- n_estimators = 100
- min_samples_leaf = 2

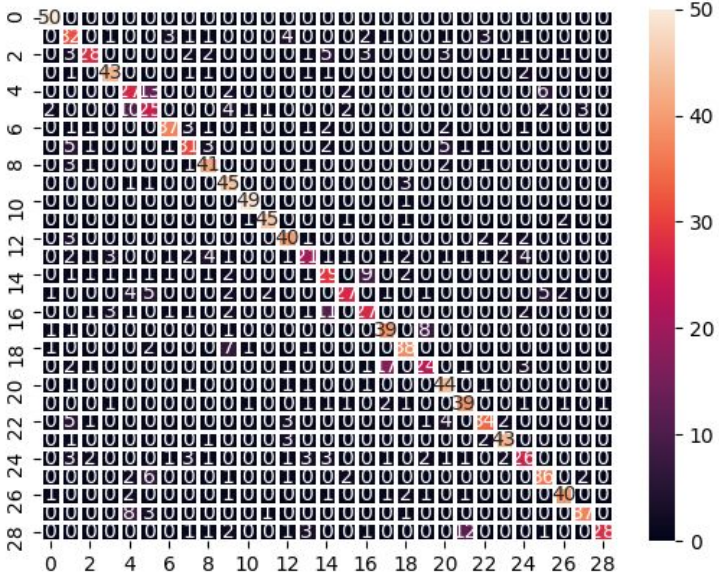
- RFC: min_samples_leaf = 20; n_estimators = 100
- RFC: min_samples_leaf = 7; n_estimators = 100
- RFC: min_samples_leaf = 2; n_estimators = 12
- RFC: min_samples_leaf = 2; n_estimators = 50

➢ with or without discretization

Results

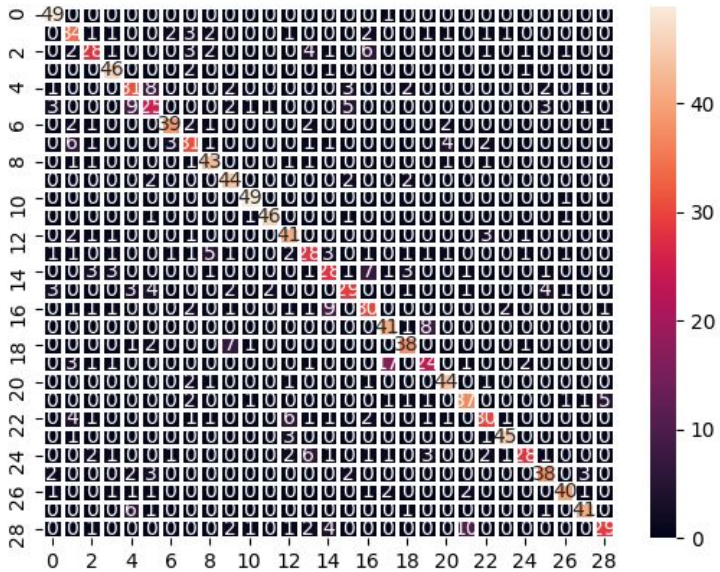
min_samples_leaf = 2; n_estimators = 50

DISCRETIZATION



Accuracy: 71%

NO DISCRETIZATION

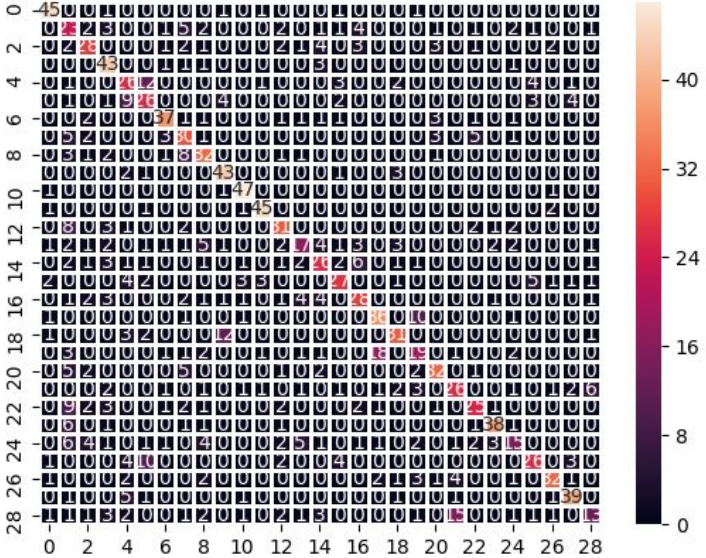


Accuracy: 73%

Results

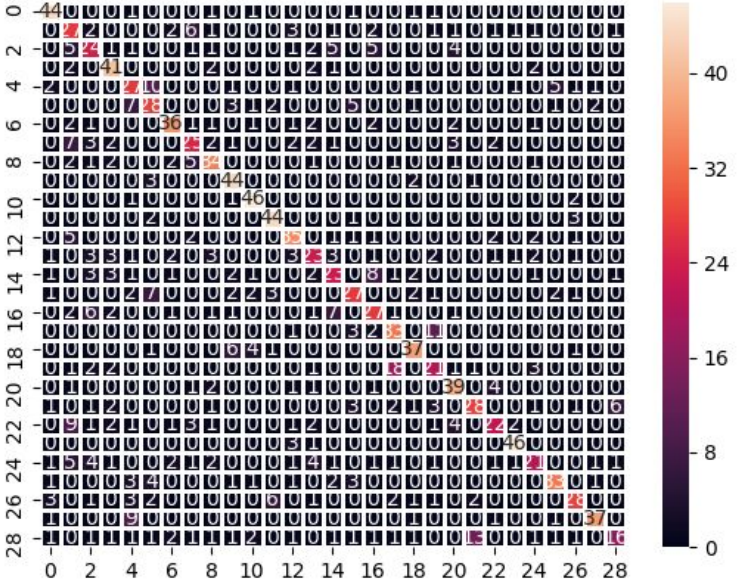
min_samples_leaf = 2; n_estimators = 12

DISCRETIZATION



Accuracy: 61%

NO DISCRETIZATION

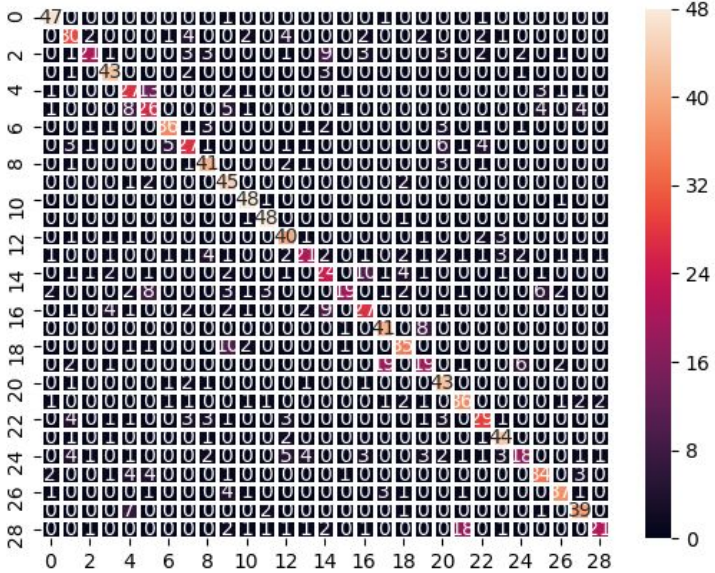


Accuracy: 63%

Results

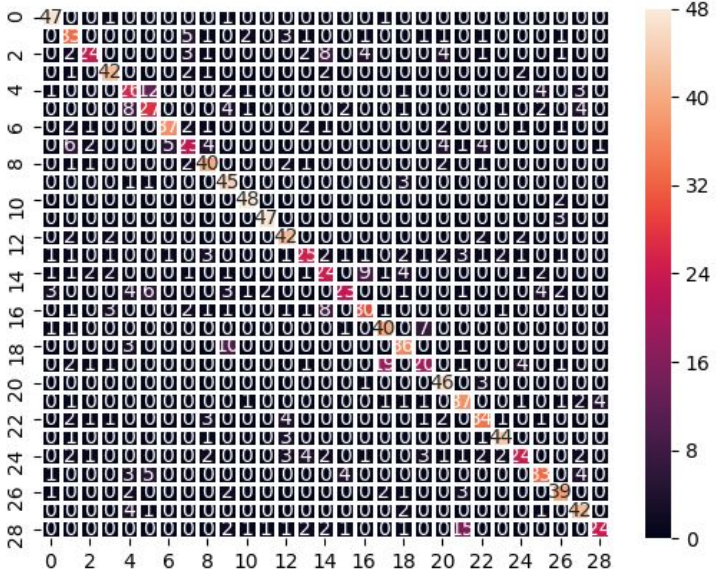
min_samples_leaf = 20; n_estimators = 100

DISCRETIZATION



Accuracy: 67%

NO DISCRETIZATION

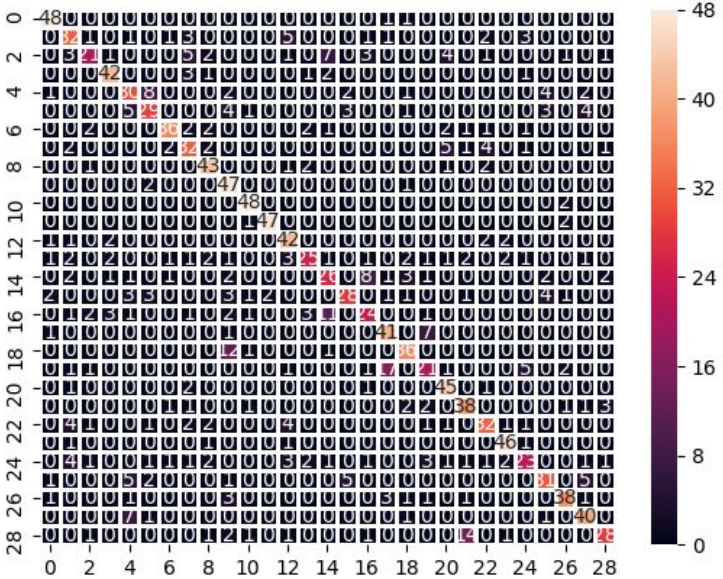


Accuracy: 69%

Results

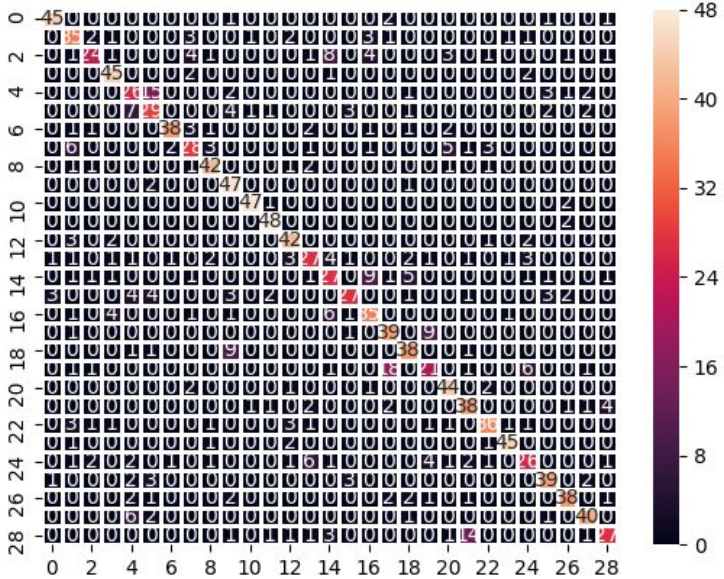
min_samples_leaf = 7; n_estimators = 100

DISCRETIZATION



Accuracy: 70%

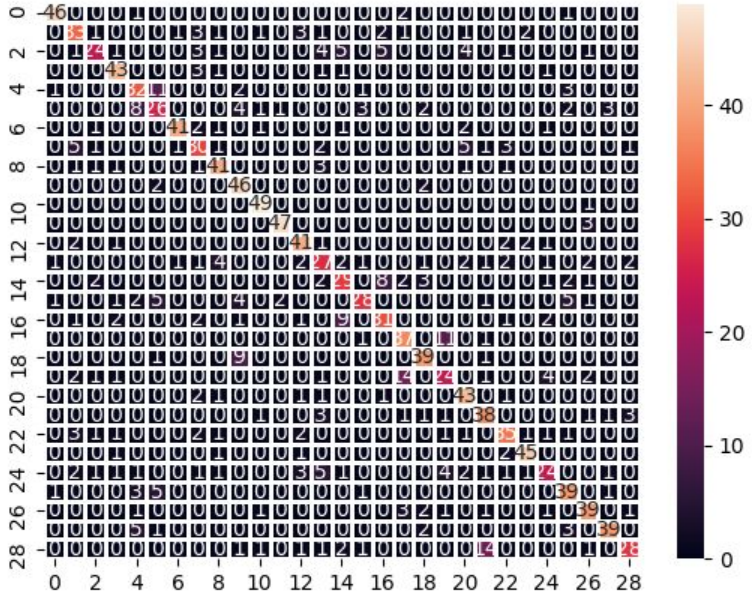
NO DISCRETIZATION



Accuracy: 72%

Combined experiment:

- no discretization
- n_estimators = 50
- min_samples_leaf = 7



Accuracy: 72% (not the best one)

CONCLUSION:

The best model (acc = 73%):

- no discretization
- n_estimators = 50
- min_samples_leaf = 2

Layers to extract features from

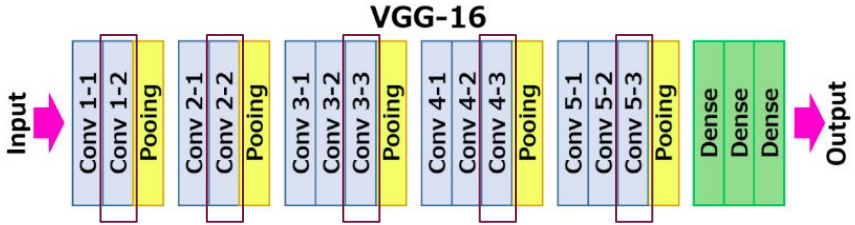
'block1_conv2'

'block2_conv2'

'block3_conv3'

'block4_conv3'

'block5_conv3'



Number of features: 1472

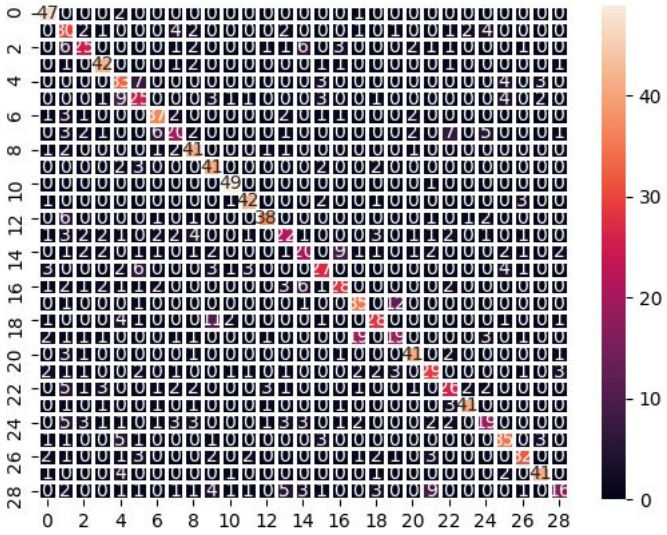
Checked RFC paramthers

ALL EXPERIMENTS
WITHOUT DISCRETIZATION

- RFC: min_samples_leaf = 20; n_estimators = 100
- RFC: min_samples_leaf = 7; n_estimators = 100
- RFC: min_samples_leaf = 2; n_estimators = 12
- RFC: min_samples_leaf = 2; n_estimators = 50

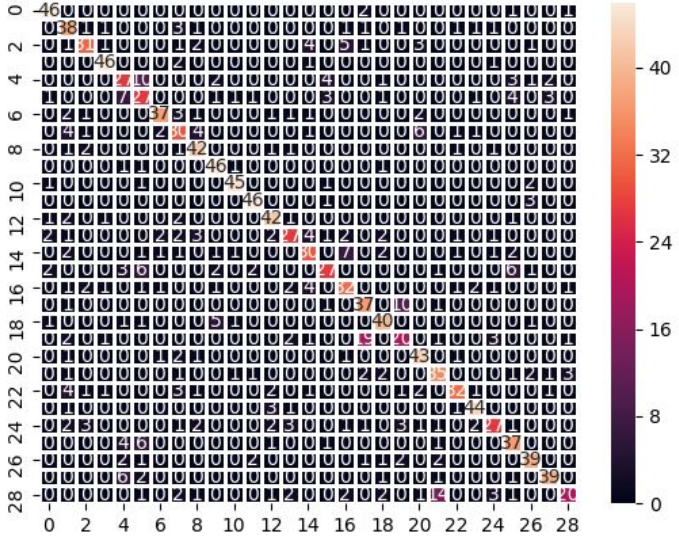
Results

min_samples_leaf = 2;
n_estimators = 12



Accuracy: 64%

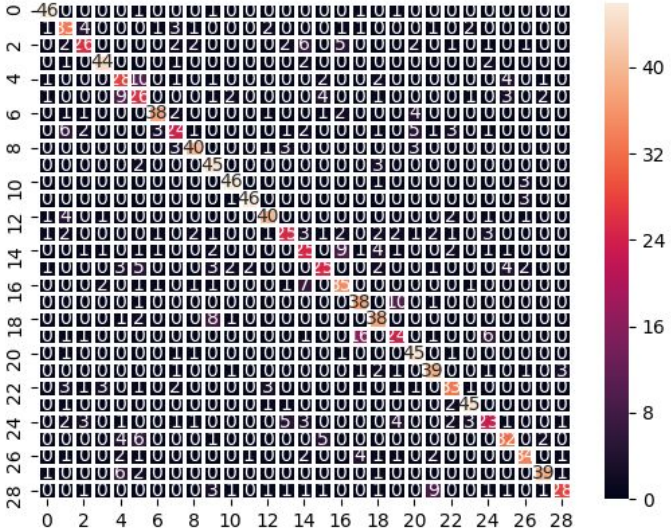
min_samples_leaf = 2;
n_estimators = 50



Accuracy: 71%

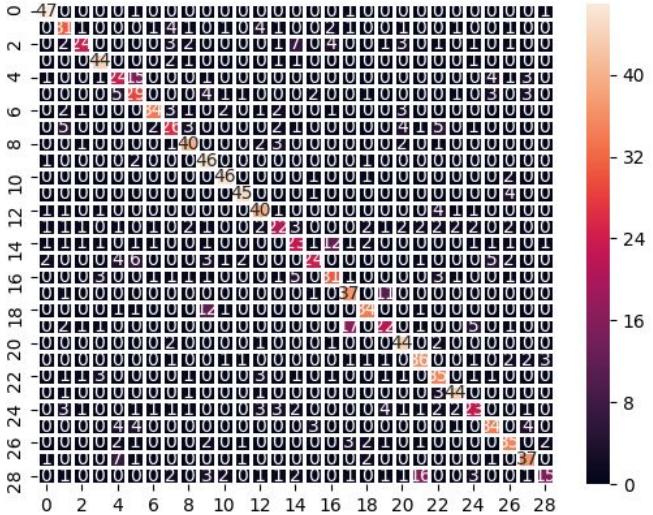
Results

min_samples_leaf = 7;
n_estimators = 100



Accuracy: 70%

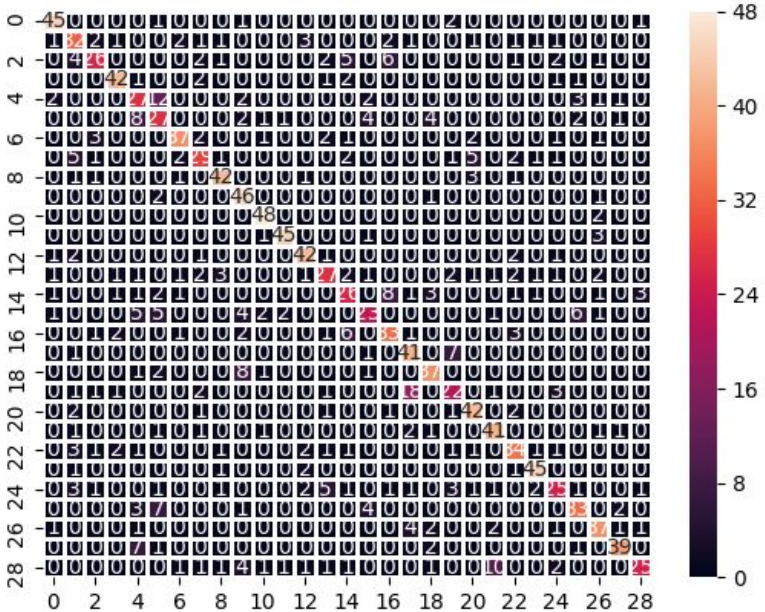
min_samples_leaf = 20;
n_estimators = 100



Accuracy: 67%

Combined experiment:

- n_estimators = 50
- min_samples_leaf = 7



Accuracy: 70% (not the best one)

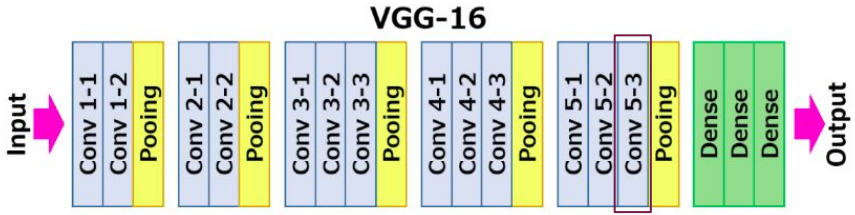
CONCLUSION:

The best model so far (acc = 73%):

- features extracted from each layer
- no discretization
- n_estimators = 50
- min_samples_leaf = 2

Single layer to extract features from

'block5_conv3'



Number of features: 512

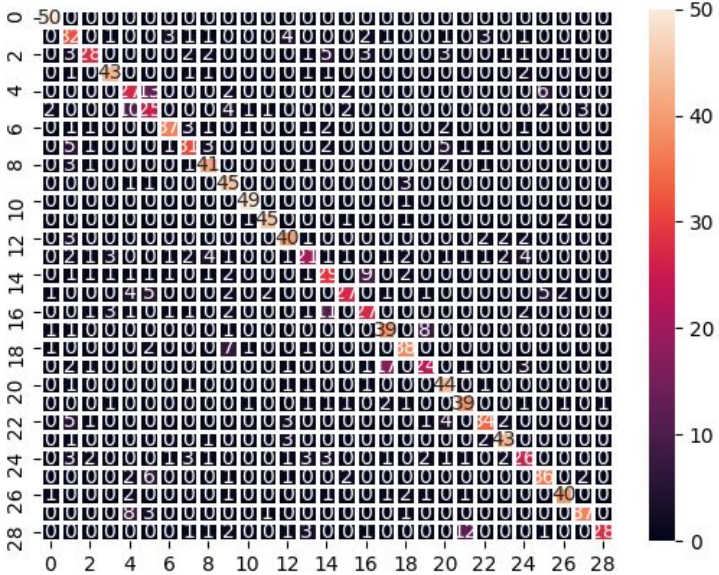
Checked RFC paramthers

ALL EXPERIMENTS
WITHOUT DISCRETIZATION

- RFC: min_samples_leaf = 20; n_estimators = 100
- RFC: min_samples_leaf = 7; n_estimators = 100
- RFC: min_samples_leaf = 2; n_estimators = 12
- RFC: min_samples_leaf = 2; n_estimators = 50

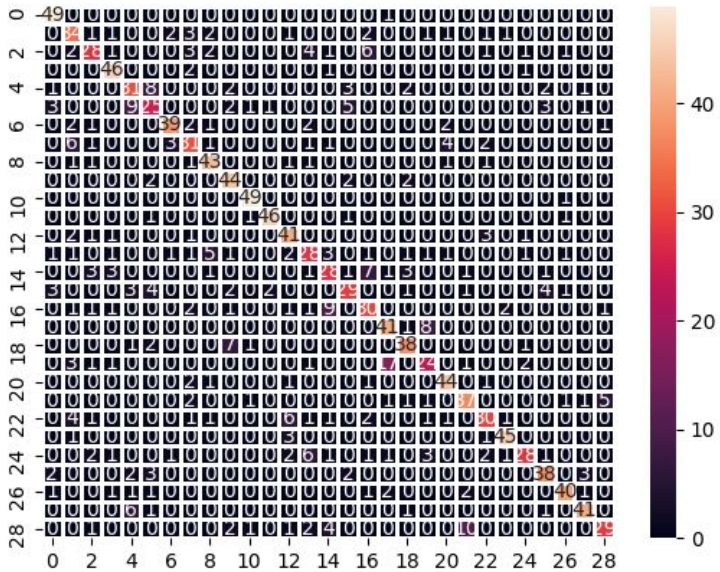
Results

min_samples_leaf = 2;
n_estimators = 12



Accuracy: 50%

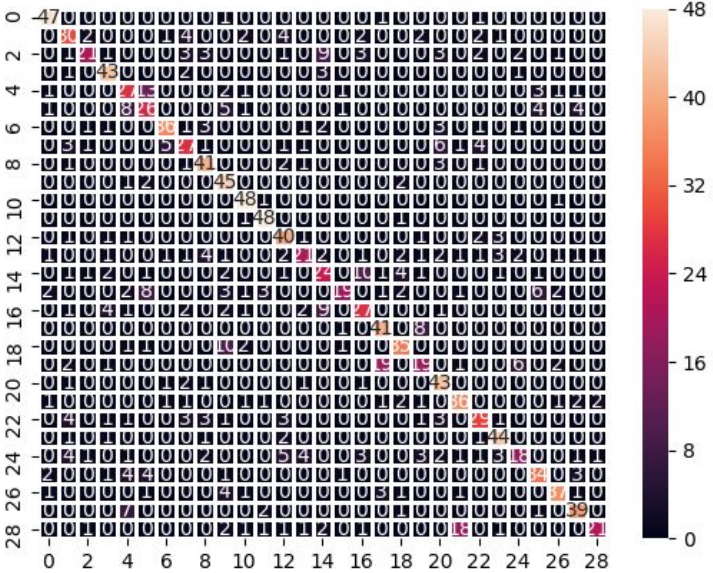
min_samples_leaf = 2;
n_estimators = 50



Accuracy: 61%

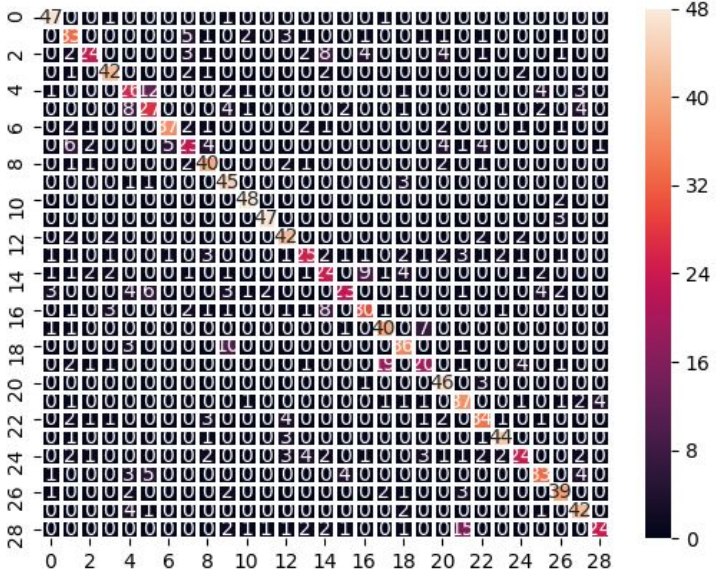
Results

min_samples_leaf = 7;
n_estimators = 100



Accuracy: 61%

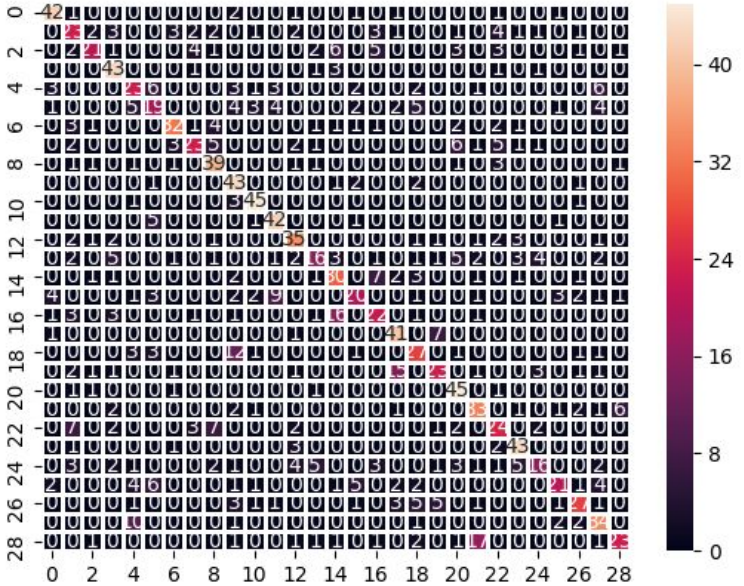
min_samples_leaf = 20;
n_estimators = 100



Accuracy: 59%

Combined experiment:

- n_estimators = 50
- min_samples_leaf = 7



val_ccuracy: 60% (not the best one)

CONCLUSION:

The best model so far (val_acc = 73%):

- features extracted from each layer
- no discretization
- n_estimators = 50
- min_samples_leaf = 2

Feature extraction from fine tuned model

Fine tuned model

	Loss	Accuracy	Val_loss	Val_accuracy
only dense layers no augmentation	0.6381	99.92	1.3175	80.03
only dense layers augmentation	4.0019	86.80	4.3281	77.13
augmentation all layers unfrozen	2.1214	93.32	2.6561	79.62
no augmentation layers unfrozen steppedly	2.8288	98.09	3.4019	81.25
no augmentation layers unfrozen steppedly + one training with augmentation	2.0618	92.88	2.6053	79.12

Layers to extract features from

'block1_conv1' 'block1_conv2'

'block2_conv1' 'block2_conv2'

'block3_conv1' 'block3_conv2', 'block3_conv3'

'block4_conv1' 'block4_conv2' 'block4_conv3'

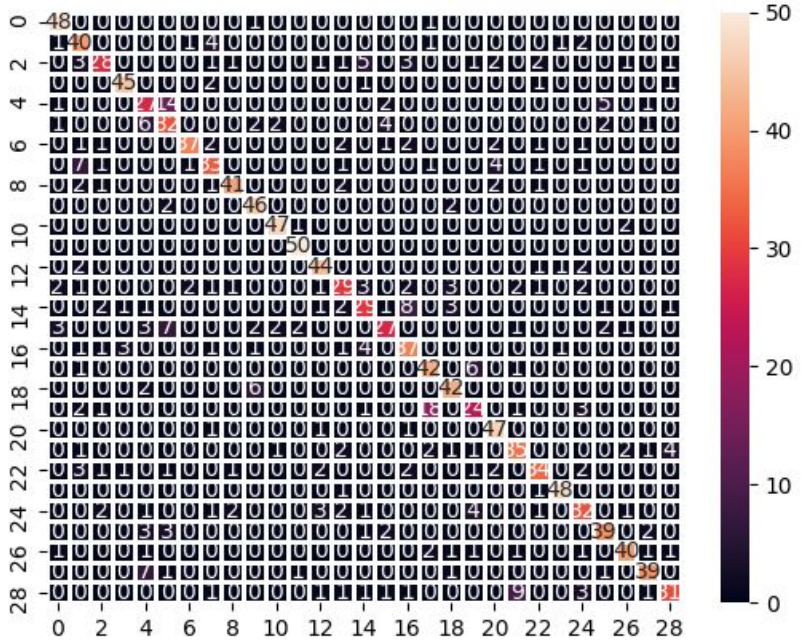
'block5_conv1' 'block5_conv2' 'block5_conv3'

'fc1' 'fc2' ‘fc3’

Number of features: 4224

Better than on original VGG16

- n_estimators = 50
- min_samples_leaf = 2



Accuracy: 74%

better than the best one with features from original VGG16

(even with much smaller number of features)

Fine tuned model

	Loss	Accuracy	Val_loss	Val_accuracy
only dense layers no augmentation	0.6381	99.92	1.3175	80.03
only dense layers augmentation	4.0019	86.80	4.3281	77.13
augmentation all layers unfrozen	2.1214	93.32	2.6561	79.62
no augmentation layers unfrozen steppedly	2.8288	98.09	3.4019	81.25
no augmentation layers unfrozen steppedly + one training with augmentation	2.0618	92.88	2.6053	79.12

Layers to extract features from

'block1_conv1' 'block1_conv2'

'block2_conv1' 'block2_conv2'

'block3_conv1' 'block3_conv2', 'block3_conv3'

'block4_conv1' 'block4_conv2' 'block4_conv3'

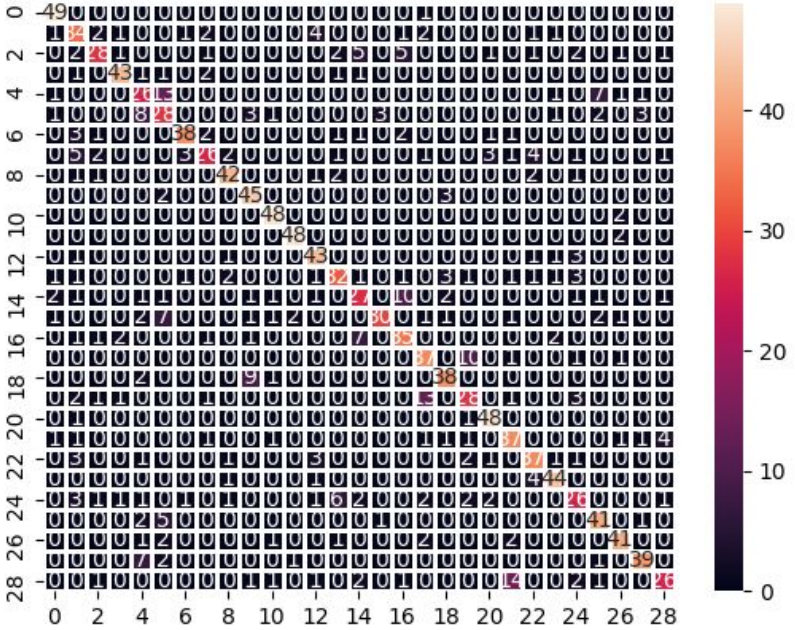
'block5_conv1' 'block5_conv2' 'block5_conv3'

'fc1' 'fc2' ‘fc3’

Number of features: 4224

Combined experiment:

- n_estimators = 50
- min_samples_leaf = 2



Accuracy: 73%

CONCLUSION:

The best model so far (acc = 74%):

- features extracted from each layer
- no discretization
- n_estimators = 50
- min_samples_leaf = 2
- **based on fine tuned model**

TESTING

The model with the best accuracy on validation dataset was chosen

Method	Description	Augmentation	Val_accuracy
fine tuning	trained in two steps: - only dense layers (BN -> Dense(256) + L2 -> Dropout(0.4) -> Dense(256) + L2 -> Dense(256) + L2 -> Dense(29)) - all neural network	YES	79.62%
fine tuning	trained in three steps: - only dense layers (BN -> Dense(256) + L2 -> Dropout(0.4) -> Dense(256) + L2 -> Dense(256) + L2 -> Dense(29)) - dense layers + convolutional from block4 and block5 - all neural network	NO	81.25%
feature extraction	Features extracted from original VGG16 network, form all layers. Random forest model applied on these features without discretization. Random forest parameters: - n_estimators = 50 - min_samples_leaf = 2	NO	73%
feature extraction + fine tuning	Features extracted from fine tuned with augmentation VGG16 network, form all layers. Random forest model applied on these features without discretization. Random forest parameters: - n_estimators = 50 - min_samples_leaf = 2	YES	74%

Performance of chosen model

Analysis of results

	precision	recall	f1-score	support
0	0.96	0.97	0.97	700
1	0.84	0.74	0.79	700
2	0.84	0.78	0.81	700
3	0.79	0.85	0.82	313
4	0.80	0.70	0.75	700
5	0.69	0.75	0.72	700
6	0.91	0.78	0.84	700
7	0.80	0.78	0.79	700
8	0.90	0.89	0.90	700
9	0.76	0.90	0.82	188
10	0.98	0.98	0.98	328
11	0.95	0.97	0.96	584
12	0.73	0.81	0.77	265
13	0.75	0.72	0.73	572
14	0.77	0.80	0.79	700
15	0.83	0.75	0.79	700
16	0.54	0.79	0.64	257
17	0.82	0.84	0.83	700
18	0.77	0.89	0.82	286
19	0.59	0.61	0.60	375
20	0.86	0.88	0.87	700
21	0.26	0.69	0.37	95
22	0.81	0.81	0.81	700
23	0.61	0.92	0.73	133
24	0.80	0.71	0.76	700
25	0.70	0.83	0.76	361
26	0.90	0.93	0.91	700
27	0.92	0.87	0.90	700
28	0.88	0.66	0.76	700
accuracy			0.81	15657
macro avg	0.79	0.81	0.79	15657
weighted avg	0.82	0.81	0.81	15657