MACHINE LEARNING PROJECT: GARBAGE CLASSIFICATION COMPETITION

Yunyue Wei, Zeji Yi, Huahuan Zheng

January 12, 2021

1 Introduction

Since the promulgation of the strictest garbage classification policy in China on July 1 2020, how to carry out garbage classification has become a tricky task of residents' daily routine. With the knowledge learned in the machine learning class and the wish to beautify the world with technologies, we set up the garbage classification competition. The explosive improvement of AI level strengthened out confidence in the AI-Assisted garbage classification task. Therefore, the purpose of this garbage classification challenge cup is to build an image classification model based on deep learning technology, and realize the accurate recognition of garbage image categories. Also to bring more attention on rational use of resources among people. The competition refers to China's newest garbage classification standard, and classifies according to recyclables, kitchen waste, hazardous waste and other wastes.

In real life, because of the variance of garbage shape, angle, light, background and so on, it is difficult for AI training set data to identify the true face of garbage. Obviously, high generalization ability and anti-interference ability is required for the garbage classification model to ensure the accuracy of model recognition. Developers can not only use the existing tag images, but also label new images themselves. When you label more kinds of garbage pictures, it will improve the accuracy of model recognition and the final score.

The challenge cup is open to the whole society. Individuals, universities, scientific research institutions, enterprises, and other developers can participate. Whether you are an individual who is interested in environmental protection, a team of colleges and universities who are interested in AI, a member of scientific research institutions with unique skills, a team of makers with unlimited creativity, or a busy enterprise developer, we are waiting for you in the Tsinghua-AI garbage classification Challenge Cup.

1.1 Background

Manual garbage classification is the first link of garbage treatment, but the link that can deal with massive garbage is garbage treatment plant.However, at present, the domestic garbage treatment plants are basically using manual assembly line sorting method for garbage sorting, there are disadvantages such as poor working environment, high labor intensity, low sorting efficiency. Manual sorting can only sort out a very limited part of recyclable garbage and hazardous garbage, and the vast majority of garbage can only be landfill, which brings great resource waste and environmental pollution risk.

With the application and development of deep learning technology in the field of vision, we see the possibility of using AI to automatically classify garbage. By taking garbage pictures with cameras and detecting the categories of garbage in the pictures, the machine can automatically carry out garbage sorting and greatly improve the efficiency of garbage sorting.

Therefore, we held this garbage classification competition, hoping to jointly explore the AI technology of garbage classification, and contribute their wisdom to the national project of garbage classification, which benefits the country and the people.

file name	Explaination
data	Training set directory, including garbage pictures and corresponding tag files (. Txt)
classify _rule.json	In the dictionary of garbage classification rules, the key value is ID, and the value is "garbage type / specific item name". For example, the training data image tagged img_1.jpg The content of img_1.txt is "img_1.jpg, 0" for img_ 1.jpg meaning the garbage in this picture is "other garbage / disposable fast food box"

1.2 Requirement and Standard

This competition adopts Beijing garbage classification standard. The task of the competition is to classify the garbage pictures, that is, first identify the categories of the items in the garbage pictures (such as cans, peels, etc.), then query the garbage classification rules, and output which of the recyclables, kitchen waste, harmful garbage and other garbage in the garbage category.

The format of the output result required is:" result ": "recyclable/cans". i.e. first the broad category then the specific item.

Recyclable material refers to the waste suitable for recycling and resource utilization, including discarded glass, metal, plastic, paper, fabric, furniture, electrical and electronic products and annual flowers and oranges, etc.Kitchen waste refers to the perishable waste produced by families and individuals, including leftovers, vegetable leaves, peel, eggshells, tea leaves, soup residues, bones, waste food and kitchen scraps.Hazardous waste refers to the waste that causes direct or potential harm to human health or the natural environment and should be specially treated, including waste batteries, waste fluorescent lamps, etc.Other garbage refers to other household garbage other than the above three types of garbage, such as diapers, dust, cigarette butts, disposable fast food boxes, broken flower POTS and dishes, wallpaper, etc.

This competition uses recognition accuracy as the evaluation index. As shown in the example of model output format above, the item category predicted by the model is "can". If the real category of the picture is can, the picture prediction is correct, otherwise the prediction is wrong. The calculation method of evaluation index is as follows: Recognition accuracy = number of correct pictures / total number of pictures

However,due to the variety of image data and the potential imbalance among different categories, we encourage the participants to evlaluate the model with more delicate method and thoughts.

1.3 Backstage Organization

First part is the essential for the garbage classification task, which is how we collect the data. In fact, for most image related task, the collection of data-set is arguably the most important. Fortunately, for our task, we employ several methods to collect the data. The size of dataset reached 18000 images in the end. The source of the data are basically consists of the following aspects.

First is the huawei-data set. Through careful searching of internet we found this is the biggest among all the resoureces. Besides, it is classified into most specific categories. We believe that finer segmentation is the better the performance will be.

2 Dataset

We gather the data from Huawei garbage classification challenge task and Kaggle garbage classification competition. Source datasets:

1. Huawei:
   Size in total: 14683
   Number of classes: 40

Source: [Published page] or the direct download [link]
We use Huawei dataset as our base dataset.

2. Kaggle:
   Size in total: 2467
   Size included: 2371
   Source: [Published page]
   The Kaggle dataset oririnally contains 5 classes (cardboard (393), glass (491), metal (400), paper(584), plastic (472) and trash (127)). We manually classify the Kaggle images with the labels in Huawei dataset. For those can not be directly classified, we add extra labels if the amount of images is relative large and exclude others with few samples.

The sorted combined dataset:
Size in total: 17054
Number of classes: 42 (40 classes from Huawei dataset and 2 classes from Kaggle dataset)
Size of training set: 15522
Size of test set: 1532
Besides the Kaggle dataset, we have considered the images from ImageNet dataset. However, the images from ImageNet are all with messy backgrounds, which is quite different from Huawei and Kaggle data, thus we did not add these images.

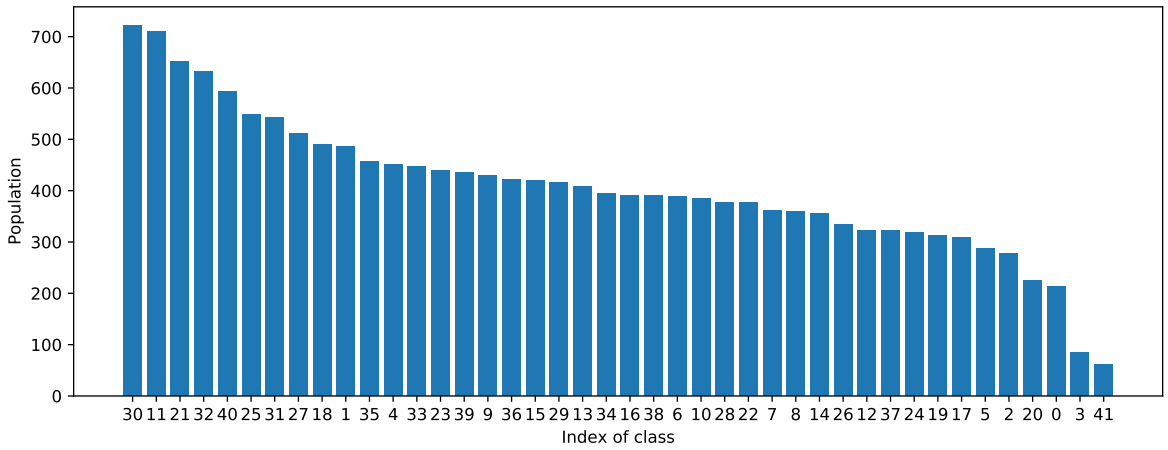


Figure 1: The population of all classes in our combined dataset.

3 Model Design

In this section, we will first introduce popular deep learning models used as baselines of our garbage dataset.

3.1 Baselines

We used three model architectures as our baseline of the dataset: VGGNet[9], ResNet[3] and DenseNet[4]. We also try the AlexNet[6] and SquezzeNet[5], but the training did not converge maybe due to the representation capacity. So we exclude these two models from our baseline list.

3.1.1 VGGNet

VGGNet is a Convolutional Neural Network architecture proposed by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014[9]. This model mainly focuses on the effect of the convolutional neural network depth on its accuracy. This model won the 2014 ImageNet Large Scale Visual Recognition Competition, achieving an top5 accuracy of 92.9% and top1 accuracy of 75.6%. The architecture of VGGNet is shown in Figure 2. VGGNet is composed of a stack of convolutional layers and 3 fully connected layers. Images directly passed through the model and the output is an 1000 dimensional vector to predict 1000 labels.

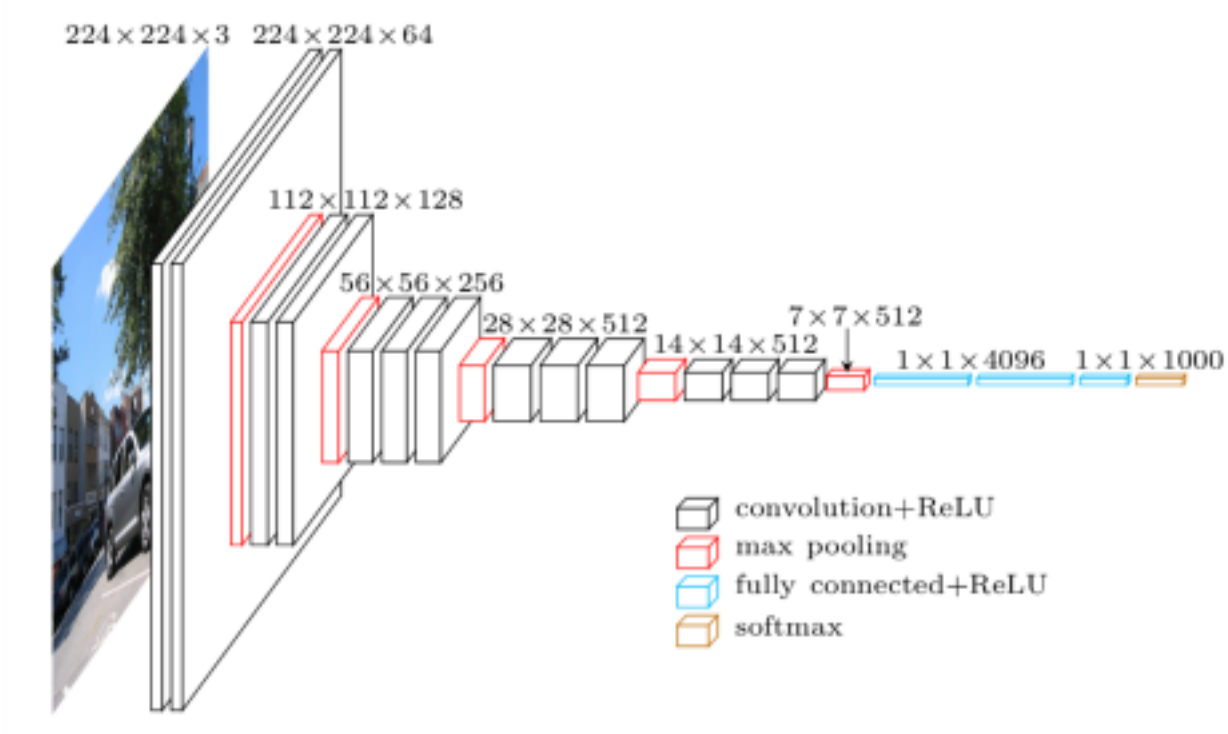


Figure 2: Architechture of VGG16[2]

The depth of VGGNet can vary by adjusting the number of convolutional layers. The model shown in Figure 2 is VGG16 composed of 13 convolutional layers and 3 fully connected layers, which add up to 16. Other popular VGG variants include VGG13 and VGG19. Batch normalization can also be added behind each convolutional layer to standardizes the inputs to a layer for each mini-batch. In the experiment, we test both classical and batch normalization version of VGG11, VGG13, VGG16 and VGG19. We find that only with batch normalization method can the VGGNet models learn the garbage dataset. Batch normalization has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks[1].

3.1.2 ResNet

ResNet is proposed by Kaiming He et al.[3] in 2016, which is was arguably the most groundbreaking work in the computer vision/deep learning community in the last few years. The most valuable contribution in this work is the residual learning, whose architecture is shown in Figure 3. This block create a identical mapping by adding a shortcut from input to the output. The identical mapping addressed problem of vanishing/exploding gradients. In this way, a large number of convolutional layers can be stacked, leading to a powerful representation capacity.

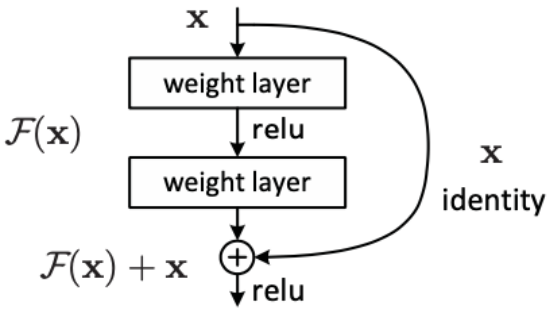


Figure 3: Architechture of Residual Learning[3]

Table 1: Training Condition

Condition	Setting
Batch Size	256
Epoch	90
Optimizer	SGD
Learning Rate	0.1
Momentum	0.9
Weight Decay	10^{-4}

By stacking different residual blocks and 1 fully connected layer, ResNet has many variants. In the experiment, we used ResNet18, ResNet50, ResNet101, ResNet152 to train and predict on our garbage dataset. The composition of these ResNet variants is show in Figure 4.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 4: Architechtures of Different ResNet[3]

3.1.3 DenseNet

DenseNet is proposed by Gao Huang et al. [4], and won the best paper in CVPR 2017 [4]. DenseNet extends the work of ResNet, building more identical shortcuts from one input to every output within the dense block. An example of a dense block with 5 convolutional layers is shown in Figure 5. With denser connections, DenseNet has better parameter efficiency and deeper supervision.

Like Resnet, the depth of DenseNet is controled by the number of stacked dense blocks. The composition of DenseNet variants is show in Figure 6. In the experiment, we use DenseNet121, DenseNet161, DenseNet169 and DenseNet201 to evaluate baseline performance of DenseNet in our garbage dataset.

3.1.4 Experiment Setting

We used the PyTorch version of models mentioned above, both pretrained and non-pretrained. The weights of the pretrained model have been trained on the 1000-class ImageNet dataset. We change the

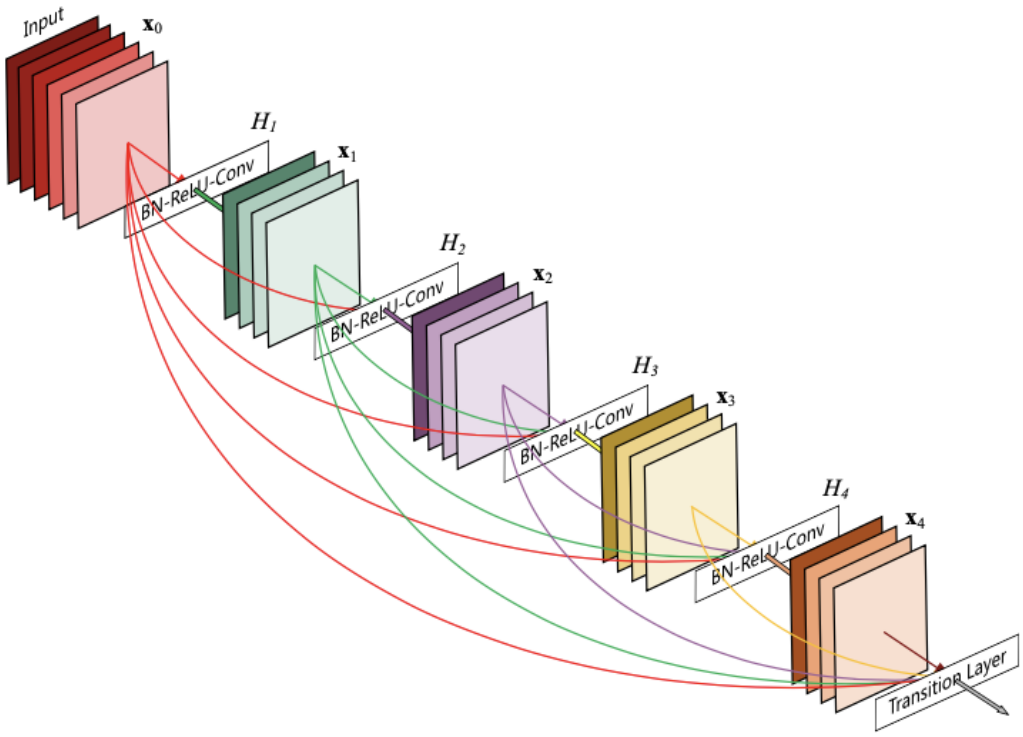


Figure 5: Architechtures of 5-layer Dense Block[4]

output channels of every model from 1000 to 42 to make the model fit to our garbage dataset. Models are trained using 10 Nvidia 2080ti GPU. Other training conditions are the same as the default settings of PyTorch official ImageNet example, which is concluded in Table 1.

4 Model Evaluation

4.1 Evaluation Metrics

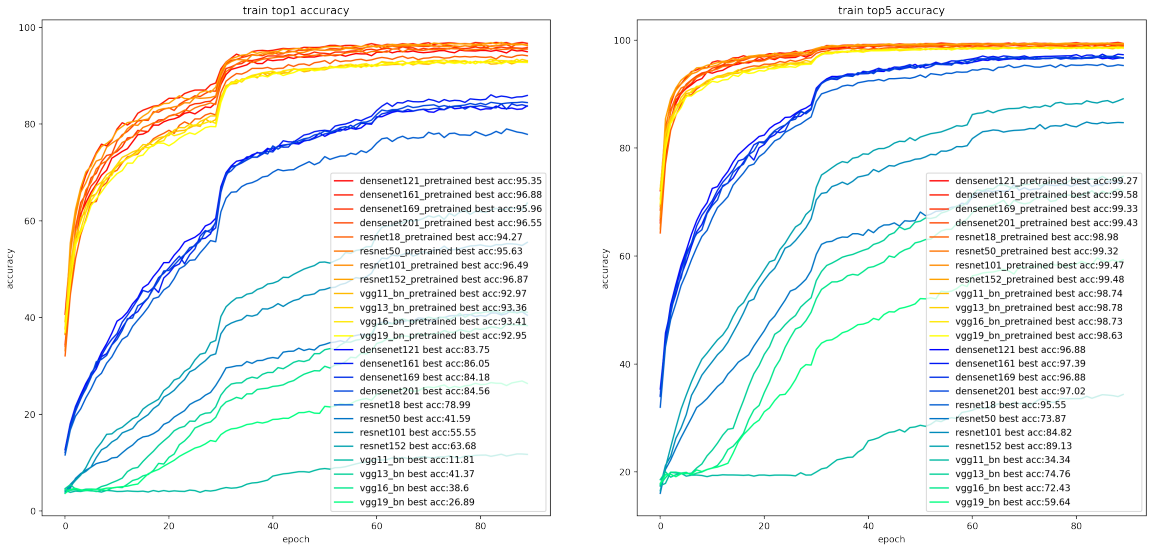
Multiple metrics are applied to evaluate the model performance on our garbage dataset, including top-k accuracy, confusion matrix, precision, recall and F1-score.

4.1.1 Top-k Accuracy

Top-k accuracy is when you measure how often your predicted class falls in the top k values of your softmax distribution[8]. we use top-1 and top-5 accuracy to measure how well the model predict the whole validation dataset. We draw both the training and validation accuracy curve during the training procedure, which is show in Figure 7a, Figure 7b, Figure 8a and Figure 8b. The best performance of baseline models on the garbage dataset is concluded in Table 2.

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

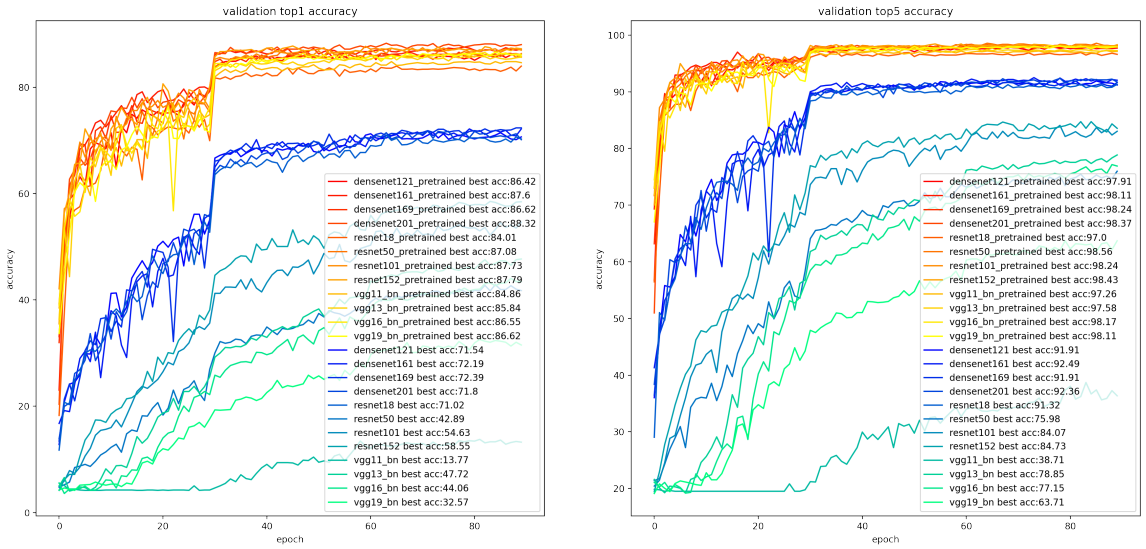
Figure 6: Architechtures of DenseNet with Different Depths[4]



(a) Top-1 accuracy

(b) Top-5 accuracy

Figure 7: Training accuracy of baseline models



(a) Top-1 accuracy

(b) Top-5 accuracy

Figure 8: Validation accuracy of baseline models

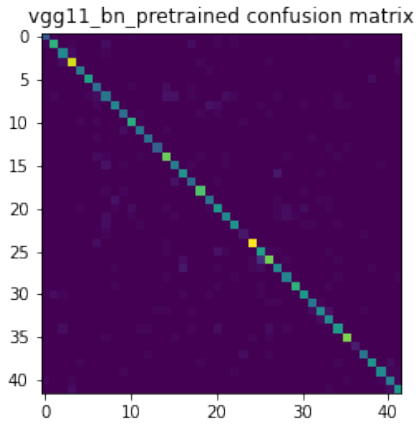
4.1.2 Confusion Matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making[7].

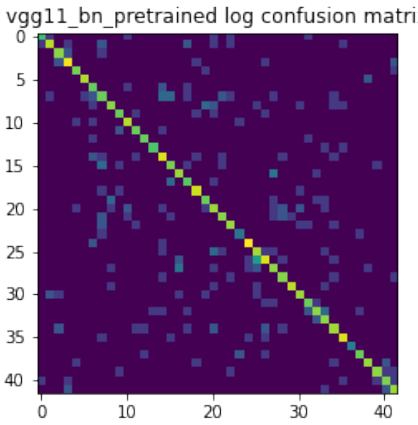
We draw the validation confusion matrix of all the models. To better observe the mismatched cases, we also add a log function to the matrix values, reducing the gap between maximum and minimum values. The results are shown from Figure 9 to Figure 20.

Table 2: Accuracy of baseline models

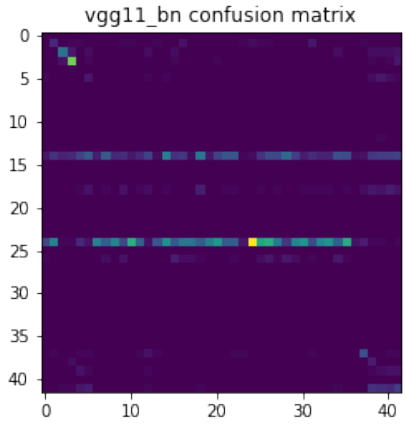
model	val top1	val top5	train top1	train top5
densenet201_pretrained	88.3	98.4	96.6	99.4
resnet152_pretrained	87.8	98.4	96.9	99.5
resnet101_pretrained	87.7	98.2	96.5	99.5
densenet161_pretrained	87.6	98.1	96.9	99.6
resnet50_pretrained	87.1	98.6	95.6	99.3
densenet169_pretrained	86.6	98.2	96.0	99.3
vgg19_bn_pretrained	86.6	98.1	93.0	98.6
vgg16_bn_pretrained	86.6	98.2	93.4	98.7
densenet121_pretrained	86.4	97.9	95.3	99.3
vgg13_bn_pretrained	85.8	97.6	93.4	98.8
vgg11_bn_pretrained	84.9	97.3	93.0	98.7
resnet18_pretrained	84.0	97.0	94.3	99.0
densenet169	72.4	91.9	84.2	96.9
densenet161	72.2	92.5	86.0	97.4
densenet201	71.8	92.4	84.6	97.0
densenet121	71.5	91.9	83.8	96.9
resnet18	71.0	91.3	79.0	95.6
resnet152	58.6	84.7	63.7	89.1
resnet101	54.6	84.1	55.5	84.8
vgg13_bn	47.7	78.9	41.4	74.8
vgg16_bn	44.1	77.2	38.6	72.4
resnet50	42.9	76.0	41.6	73.9
vgg19_bn	32.6	63.7	26.9	59.6
vgg11_bn	13.8	38.7	11.8	34.3



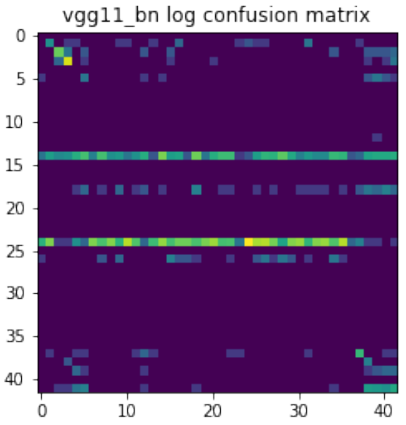
(a) Pretrained VGG11



(b) Log Pretrained VGG11

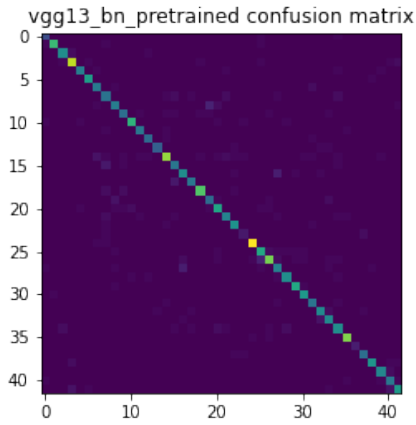


(c) Non-pretrained VGG11

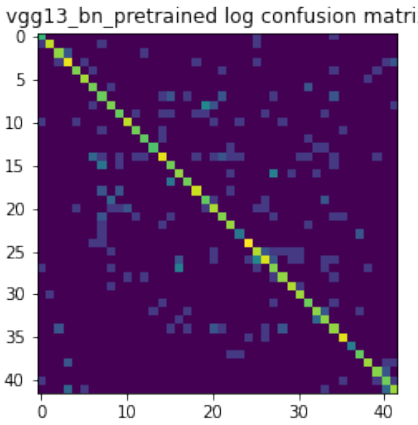


(d) Log Non-pretrained VGG11

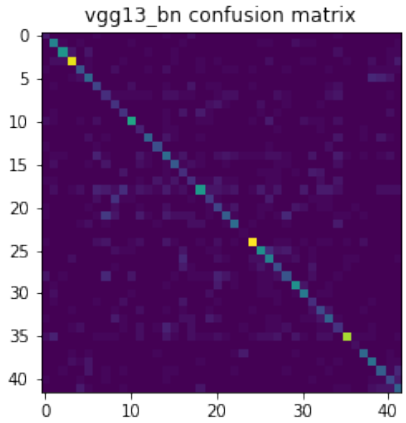
Figure 9: Confusion Matrix of VGG11



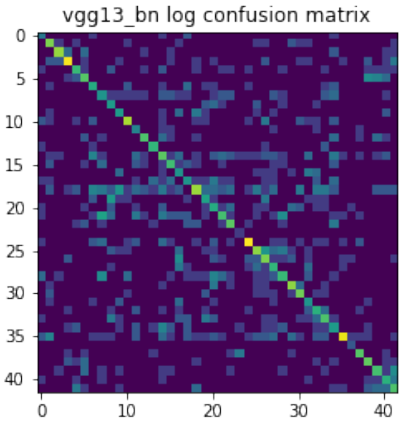
(a) Pretrained VGG13



(b) Log Pretrained VGG13

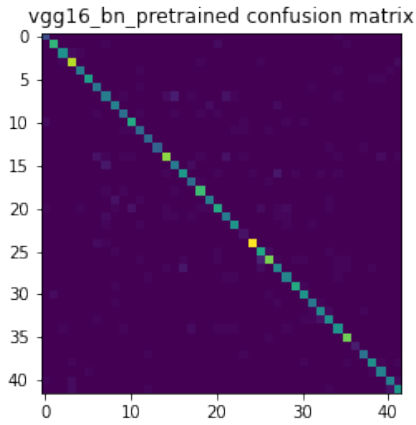


(c) Non-pretrained VGG13

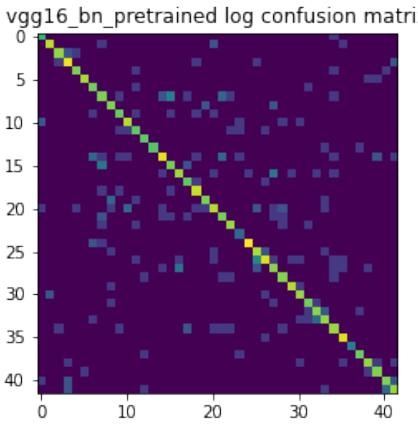


(d) Log Non-pretrained VGG13

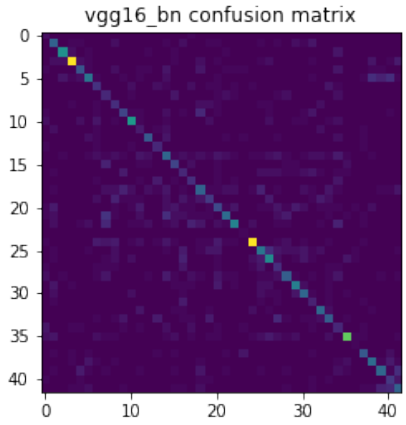
Figure 10: Confusion Matrix of VGG13



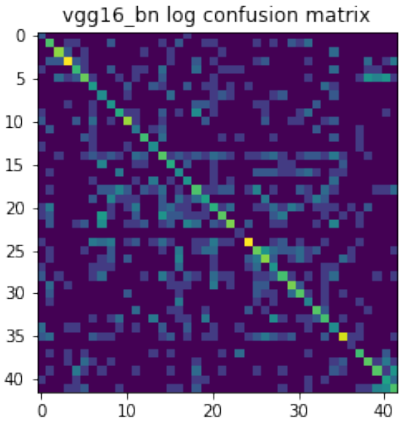
(a) Pretrained VGG16



(b) Log Pretrained VGG16

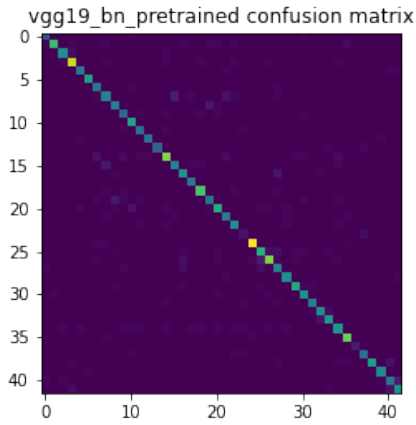


(c) Non-pretrained VGG16

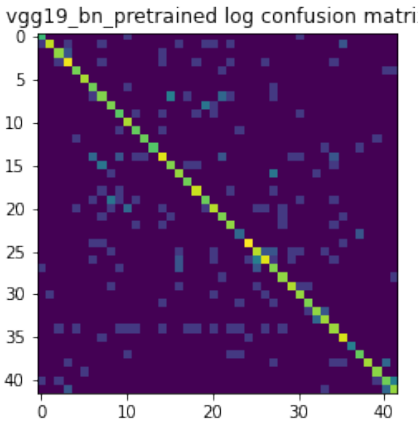


(d) Log Non-pretrained VGG16

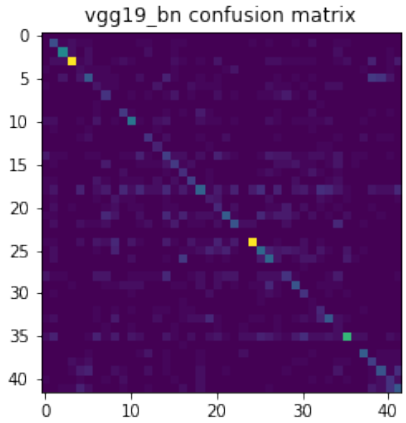
Figure 11: Confusion Matrix of VGG16



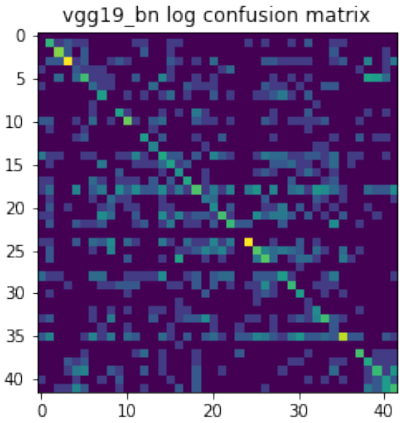
(a) Pretrained VGG19



(b) Log Pretrained VGG19

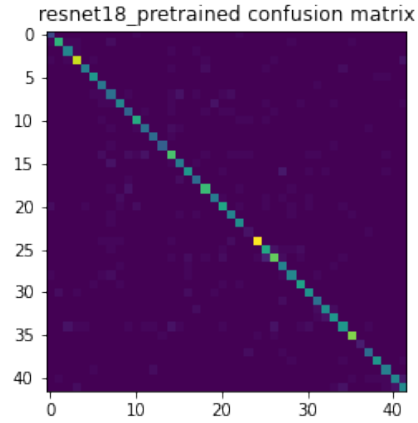


(c) Non-pretrained VGG19

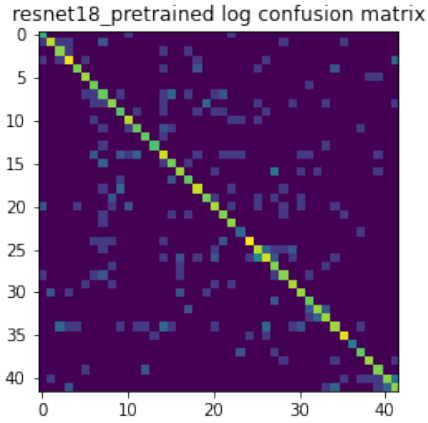


(d) Log Non-pretrained VGG19

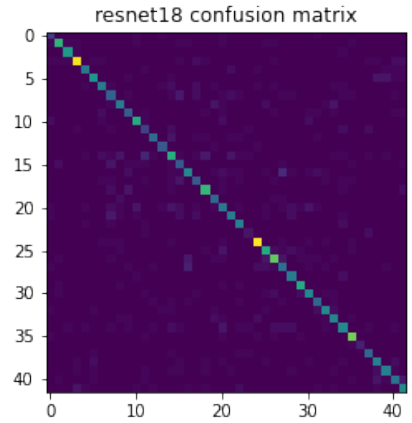
Figure 12: Confusion Matrix of VGG19



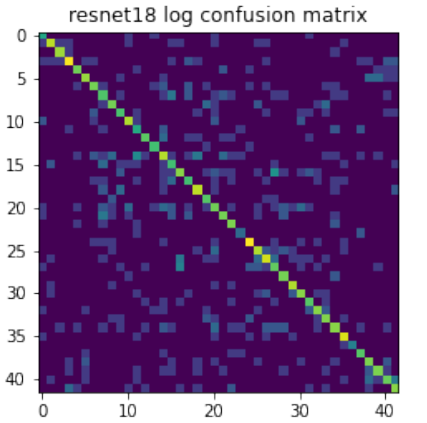
(a) Pretrained ResNet18



(b) Log Pretrained ResNet18

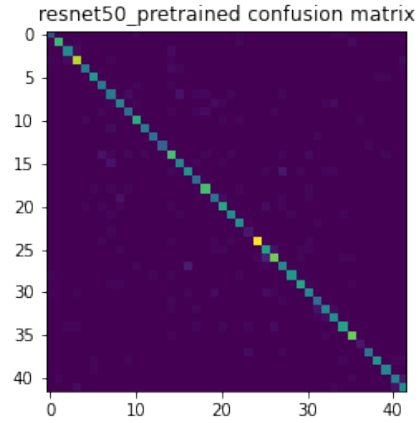


(c) Non-pretrained ResNet18

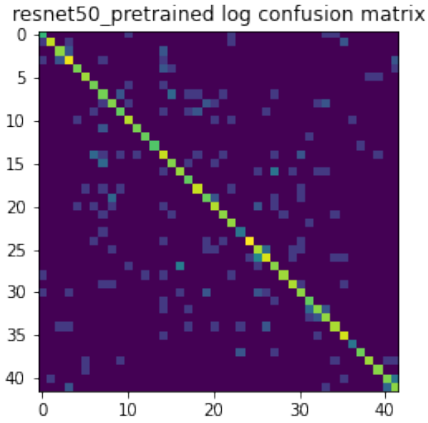


(d) Log Non-pretrained ResNet18

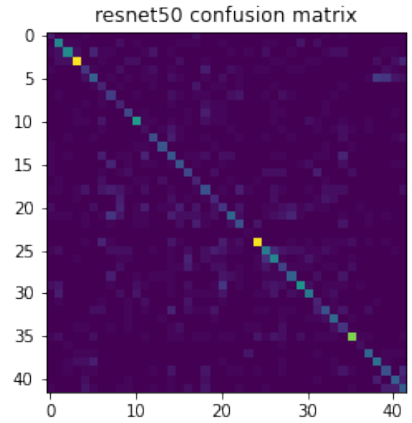
Figure 13: Confusion Matrix of ResNet18



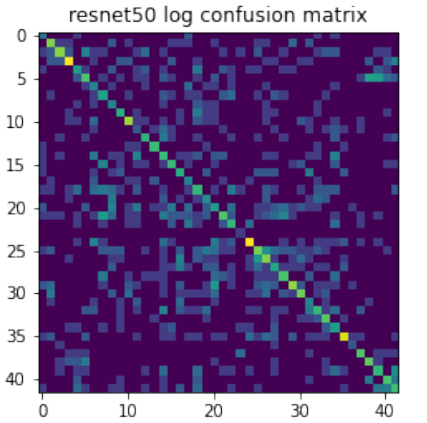
(a) Pretrained ResNet50



(b) Log Pretrained ResNet50

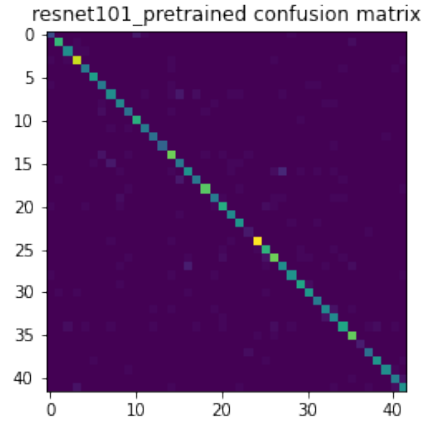


(c) Non-pretrained ResNet50

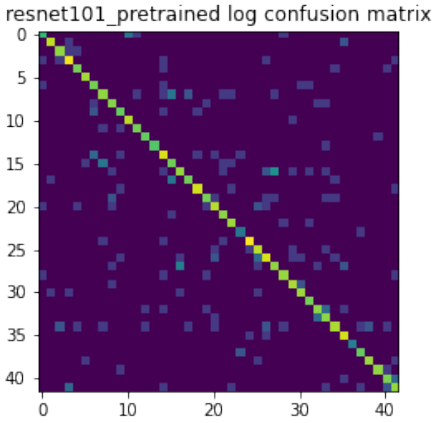


(d) Log Non-pretrained ResNet50

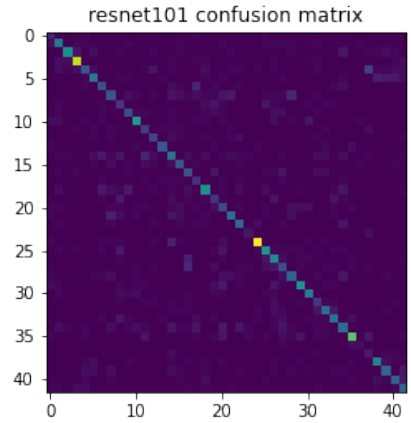
Figure 14: Confusion Matrix of ResNet50



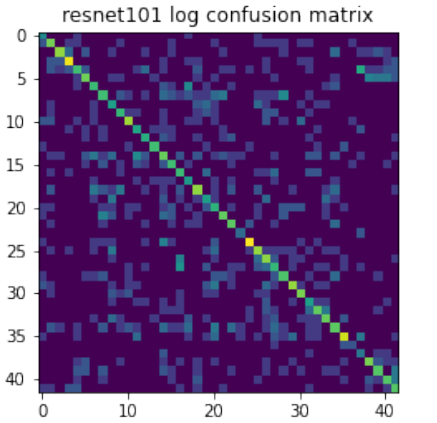
(a) Pretrained ResNet101



(b) Log Pretrained ResNet101

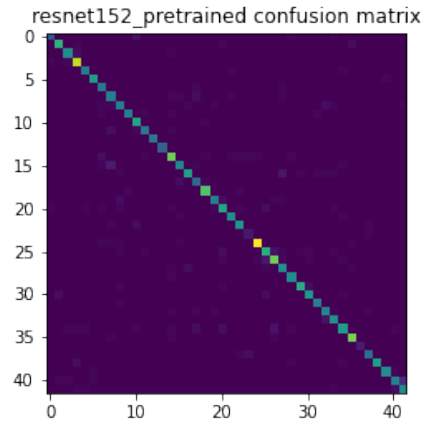


(c) Non-pretrained ResNet101

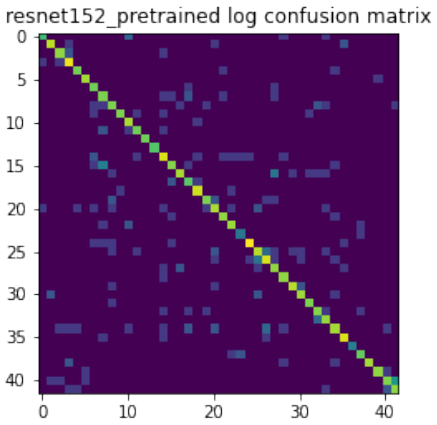


(d) Log Non-pretrained ResNet101

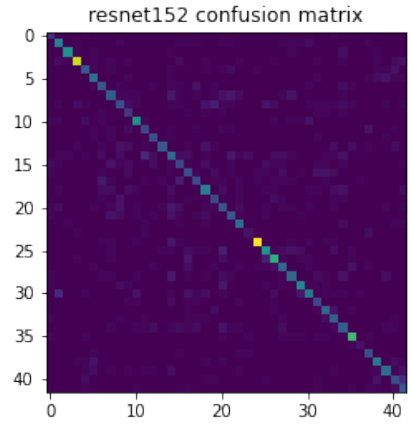
Figure 15: Confusion Matrix of ResNet101



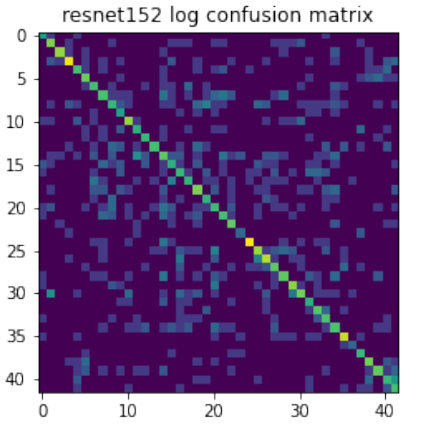
(a) Pretrained ResNet152



(b) Log Pretrained ResNet152

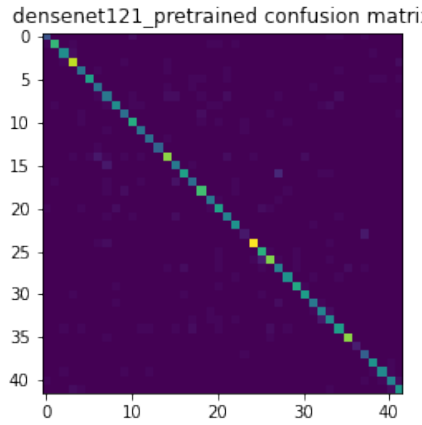


(c) Non-pretrained ResNet152

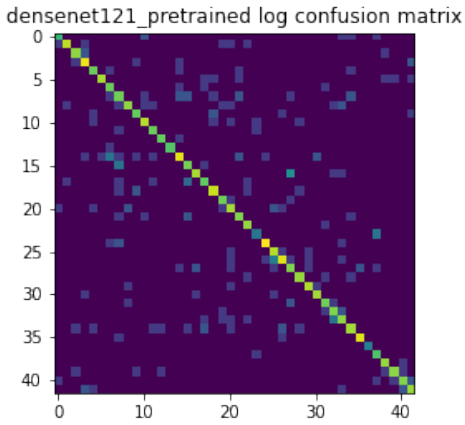


(d) Log Non-pretrained ResNet152

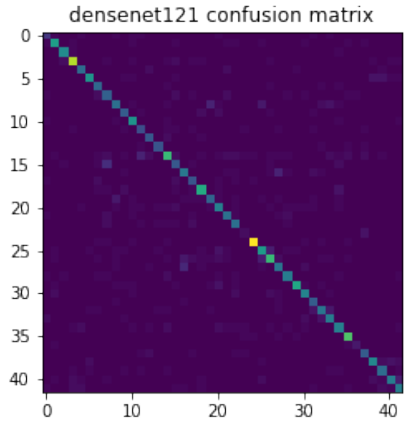
Figure 16: Confusion Matrix of ResNet152



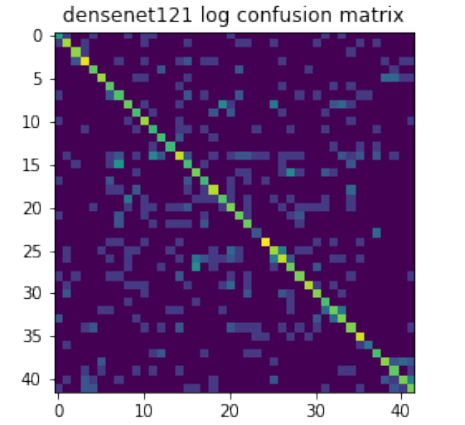
(a) Pretrained DenseNet121



(b) Log Pretrained DenseNet121

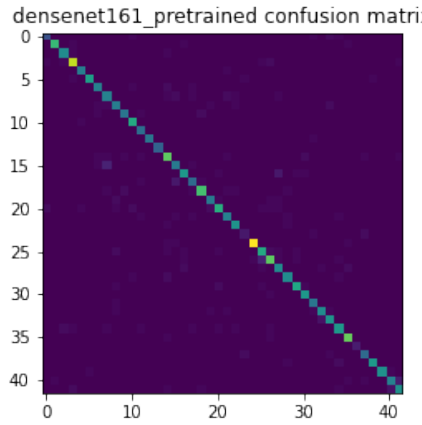


(c) Non-pretrained DenseNet121

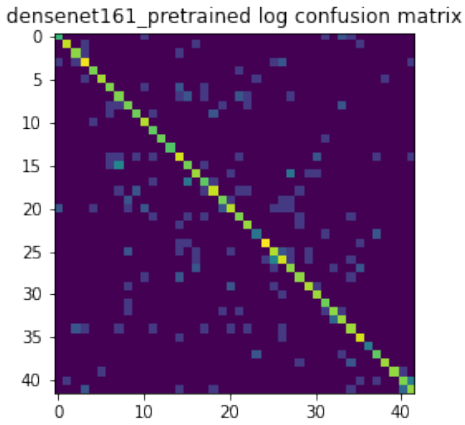


(d) Log Non-pretrained DenseNet121

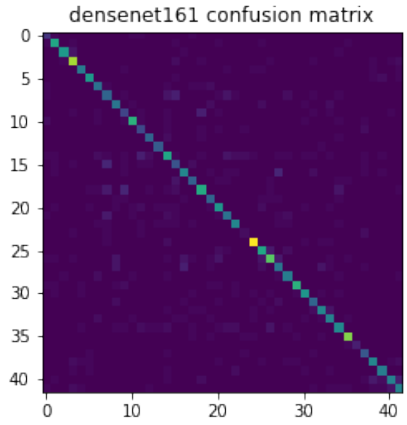
Figure 17: Confusion Matrix of DenseNet121



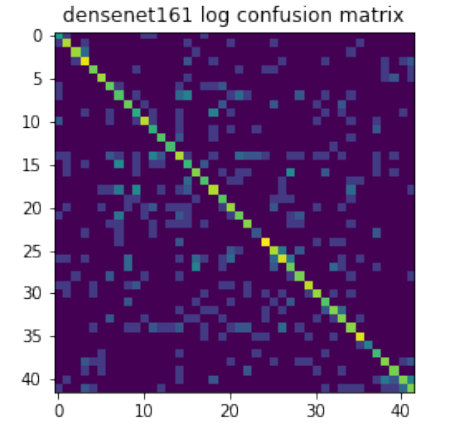
(a) Pretrained DenseNet161



(b) Log Pretrained DenseNet161

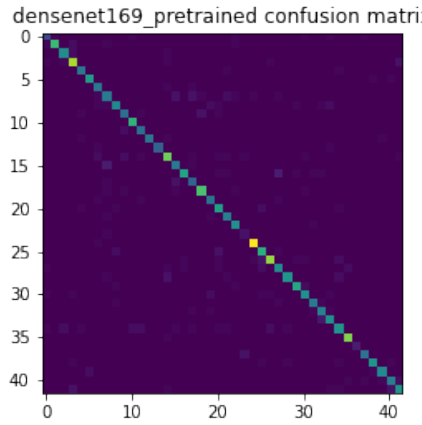


(c) Non-pretrained DenseNet161

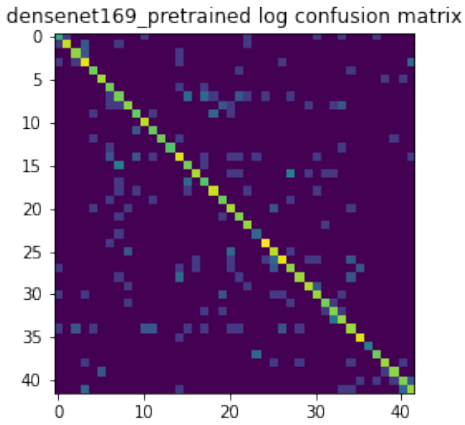


(d) Log Non-pretrained DenseNet161

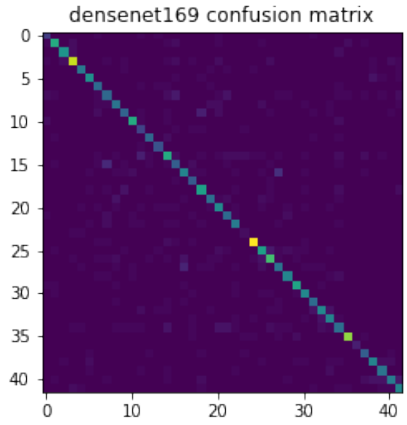
Figure 18: Confusion Matrix of DenseNet161



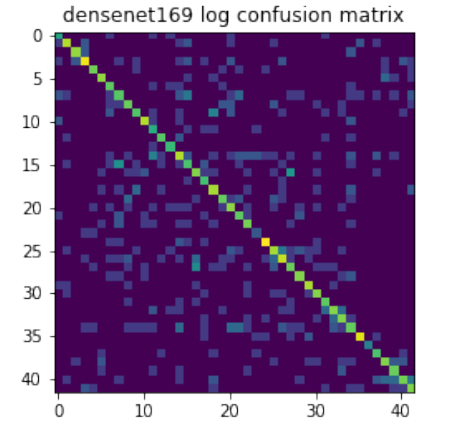
(a) Pretrained DenseNet169



(b) Log Pretrained DenseNet169

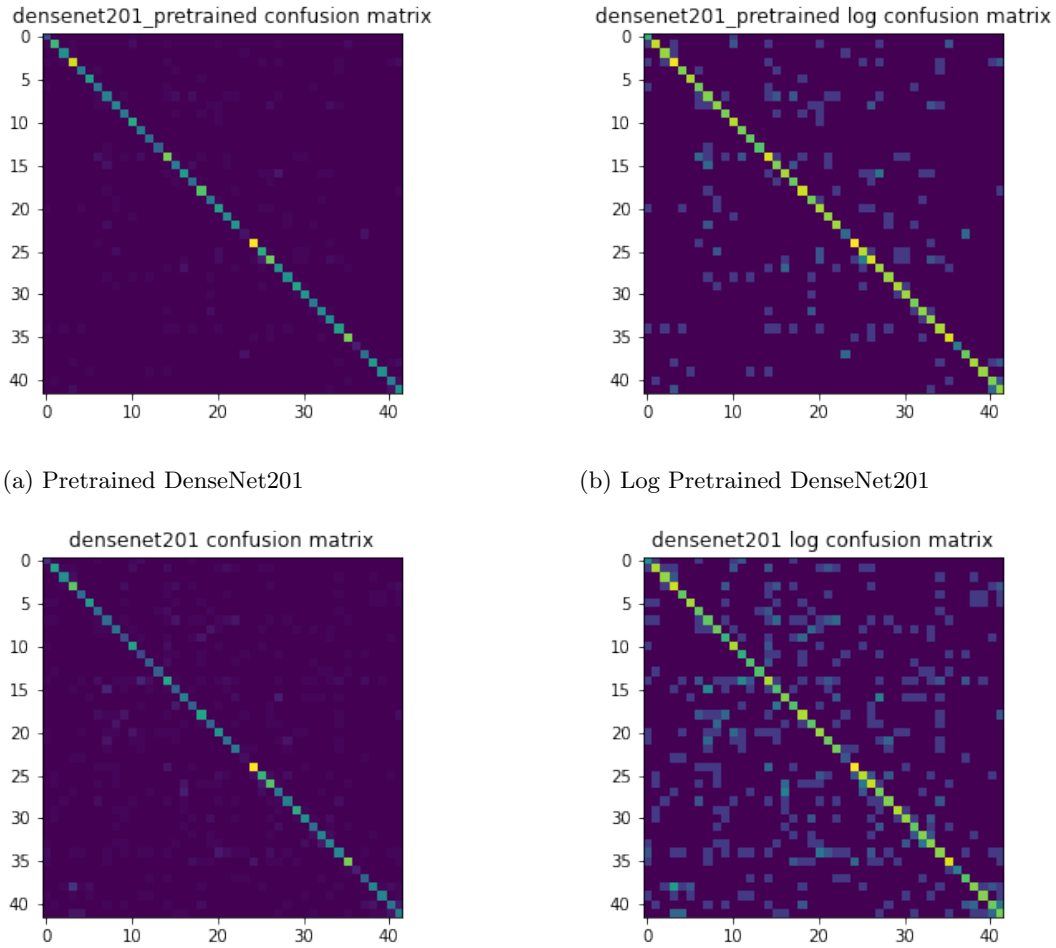


(c) Non-pretrained DenseNet169



(d) Log Non-pretrained DenseNet169

Figure 19: Confusion Matrix of DenseNet169



(a) Pretrained DenseNet201

(b) Log Pretrained DenseNet201

(c) Non-pretrained DenseNet201

(d) Log Non-pretrained DenseNet201

Figure 20: Confusion Matrix of DenseNet201

4.1.3 Precision, Recall and F1-score

The model prediction can be classified into four cases: true positive (TP), true negative (TN), false positive (FP), and false negative(FN). After counting the number of these four cases, the computation of precision and recall is as following equations:

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$

Considering a disease diagnosis scenario, precision is the measure of patients that the model correctly identify having a disease out of all the patients actually having it. Recall is how many we correctly identified as having a disease. Precision and recall are often in tension. That is, improving precision typically reduces recall. We need to balance the precision-recall tradeoff in real-world application.

F1-score is computed using the value of precision and recall:

Table 3: Precison, Recall and F1-score of Baseline Models

model	precision	recall	f1score
densenet201_pretrained	0.875	0.885	0.879
resnet152_pretrained	0.877	0.882	0.877
resnet101_pretrained	0.872	0.880	0.875
densenet161_pretrained	0.872	0.875	0.873
resnet50_pretrained	0.864	0.874	0.867
vgg16_bn_pretrained	0.856	0.876	0.862
vgg19_bn_pretrained	0.858	0.868	0.861
densenet169_pretrained	0.857	0.866	0.859
densenet121_pretrained	0.857	0.861	0.858
vgg13_bn_pretrained	0.852	0.857	0.853
vgg11_bn_pretrained	0.846	0.848	0.846
resnet18_pretrained	0.832	0.848	0.837
densenet169	0.712	0.733	0.718
densenet161	0.708	0.716	0.709
densenet201	0.706	0.713	0.706
resnet18	0.701	0.711	0.703
densenet121	0.696	0.710	0.700
resnet152	0.570	0.589	0.573
resnet101	0.523	0.534	0.520
vgg13_bn	0.450	0.500	0.454
vgg16_bn	0.409	0.463	0.406
resnet50	0.406	0.430	0.401
vgg19_bn	0.286	0.288	0.270
vgg11_bn	0.103	0.091	0.078

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

F1-score can be seen as a balance between precision and recall. Compared to accuracy, F1-score can better evaluate model generalization performance when the data distribution is unbalanced.

4.2 Baseline Summary

We compute all of the metrics on all the models, and concluded as Table 3. We also rank the model based on different evaluation metrics, as shown in Figure 21. From multi-dimensional results, we observed that models with identical shortcuts achives higher accuracy and F1-score than models without that. All of the pretrained models outperforms non-pretrained models at all the evaluation metrics, which have already built good generalization on the large ImageNet dataset. The accuracy gap between pretrained and non-pretrained models in VGGNet is lager than that in DenseNet and ResNet, indicating a harder training process in models without identical shortcut. Deeper networks often achieve better performance, indicating a powerful representation capacity.

Among all of 24 models, the pretrained DenseNet201 take the first place in both top1 validation accuracy and F1-score. The highest top1 validation accuracy is under 90%, which means there is still large space for us to improve the model performance.

4.3 Visualization of the activation feature map

The activation function ReLU is defined as

$$\mathrm{ReLU}(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases} \tag{1}$$

We can condiser the ReLU activation function as a filter of features, thus the positive element means more important part. By ploting the output of ReLU layer, we can visualize how the model processes the input images. Due to the output features map of latter layers are too small to view, we only draw the output of the first ReLU layer.

As the Fig. 22 shows, the model catch the knapsack in the image even though it is not on the center, showing the ability to extract features out of a images.

5 Conclusion

In this project, we collect the images from two open dataset, Huawei garbage classification and Kaggle garbage classification competition, and combine the two datasets into one. The dataset in our competition contains over 17000 images in total, which is sufficient large for garbage classification, and the classes of our collected images covered most of common garbages in our daily life.

Based on the collected data, we setup a competition, with the "garbage classification" theme, and the task is "a small granularity" task compared to the 4 classes urban classification guide[1], which includes 42 classes of garbages. Thus our competition is relative difficulty.

As for the solutions to our competition, we try using various of modern widely used models, including ResNet, DenseNet, eta. We use accuracy, precision, F1 score and confusion matrix as the performance metrics and compare different models.

In test set, the DenseNet201 pretrained on ImageNet obtains the best accuracy 88.32%. Compare with ResNet and VGGNet, DenseNet performs better on both training set and test set. For different architectures, the pretrained models consistently perform better. We tried add data augmentation methods, like randome rasing, though it seems help little.

We further analysis the how the images of garbages influence the results of classification by visualizing the activation maps. This helps us better understand which part of the input image matters and which does not.

6 Discussion

We hope the competition would encourage more people to join in the garbage classification task, proposing better models to handle it. The excellent models can also be applied in real-world garbage classification scenario, reducing the burden of dustmen. We also hope the competition can impress more people, making them paying more attention to garbage classification in daily life.

Due to the time limit, our current work is not perfect enough. There are some limitations to improve in the future. Although about 17,000 images are included in our garbage dataset, the data unbanlanced issue still exists, which makes it harder for our trained models to achieve competitive performance in real-world application as on the dataset. Some public image datasets may be potiential resources for us to extend current garbage dataset.

The baseline evaluation is done at one training using stochastic optimize methods, which is unstable in the final results. A better way to fix that is to train one model multiple times and average the results as the final result, but it may take a massive amount of time.

Garbage usually does not appear as single object, indicating the needs of multiple garbage classification in one image. Integrated with object tracking algorithms, the models would be more powerful and applicable in real-world complex garbage classification tasks.

[1] According to Beijing city garbage classification notification, the urban classification includes kitchen, harmful, recyclable and others garbages.

References

[1] Jason BROWNLEE. A gentle introduction to batch normalization for deep neural networks. *Machine Learning Mastery [online]*, 16, 2019.

[2] Davi Frossard. Vgg in tensorflow. `https://www.cs.toronto.edu/~frossard/post/vgg16/`.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[5] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[7] ANIRUDDHA BHANDARIRushabh Nagda. Everything you should know about confusion matrix for machine learning. `https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/`.

[8] Rushabh Nagda. Evaluating models using the top n accuracy metrics. `https://medium.com/nanonets/evaluating-models-using-the-top-n-accuracy-metrics-c0355b36f91b`.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

7 Appendix

The repository is available on GitHub: `https://github.com/wtxwyy/ML-Project`

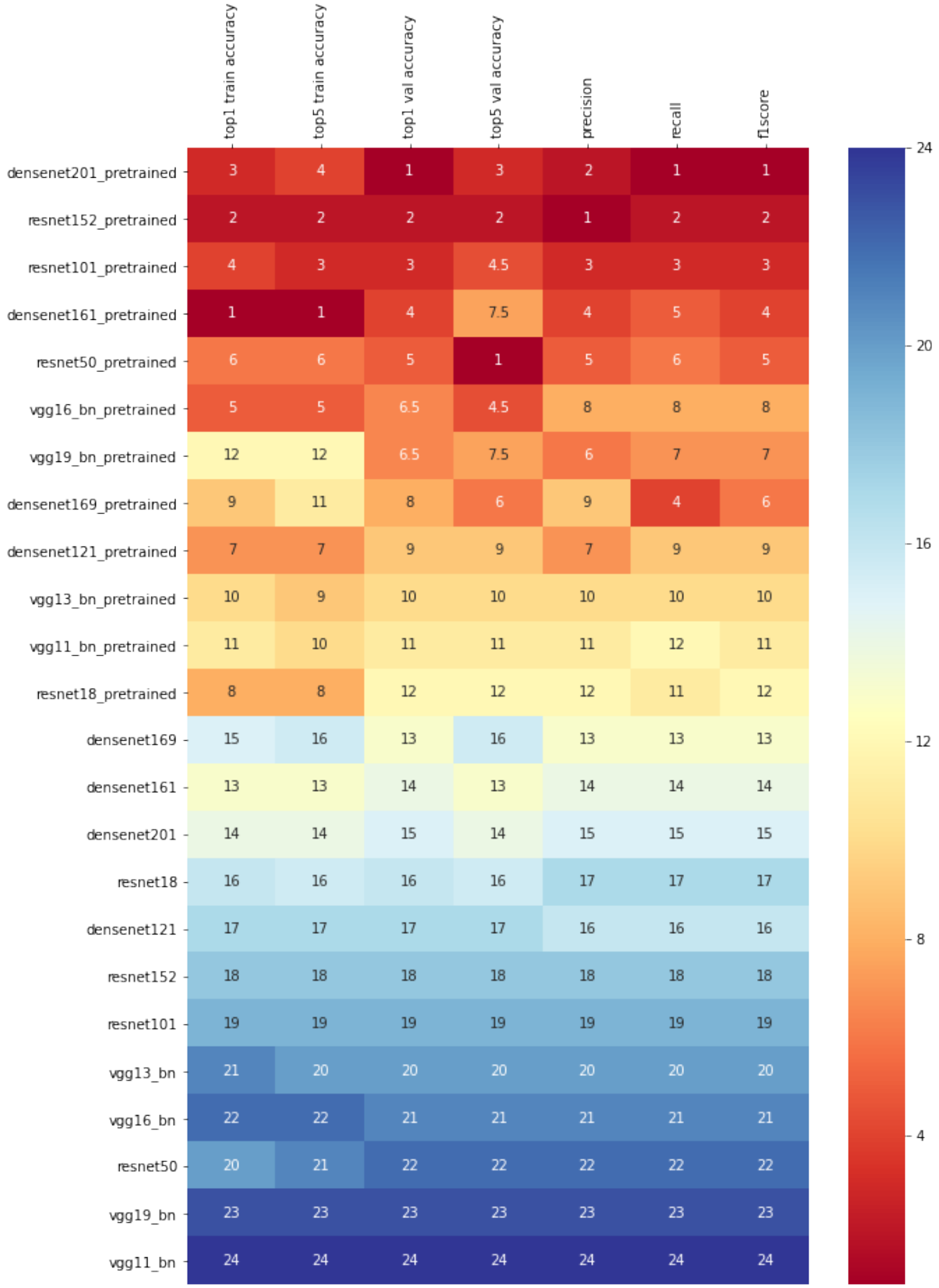
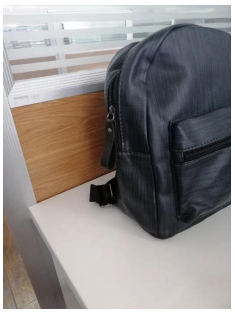
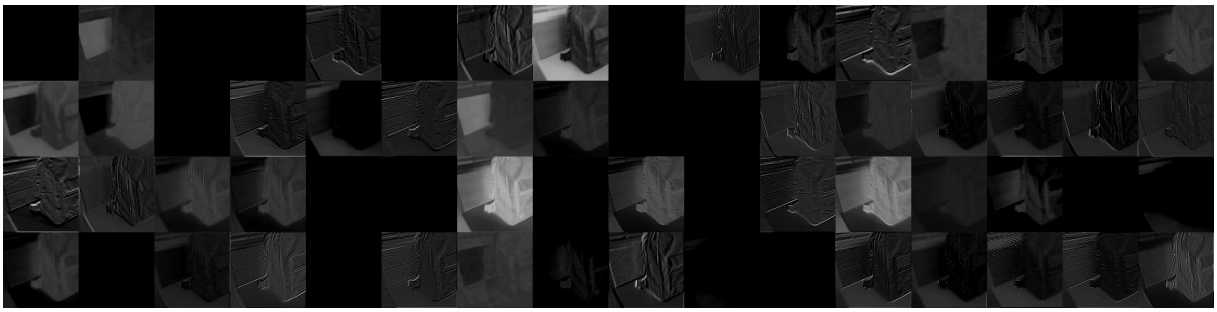


Figure 21: Performance Rank of Different Evaluation Metrics



(a) The original image img_7224



(b) The feature map of the first activation layer.

(c) The original image img_12875

(d) The feature map of the first activation layer.

Figure 22: Visualization of the activation map.