Dog Breed Identification

Ortal Ashkenazi 201272481
Dafna Schumacher 066434952

April 2019

1 Introduction

The project that we have selected was the 'dog-breed-identification' competition. The competition goal is determining the breed of a dog from an image. The data set comprises from 120 breeds of dogs, 10,223 train images and 10,357 test images. Figure 1 shows an histogram of the number of training samples per class. The number of images per class ranges from 65-126. Most classes have about 80 images, and there are few classes with over 100 images. The distribution of the samples between the classes is reasonable, most of the classes have between 70 to 90 samples.

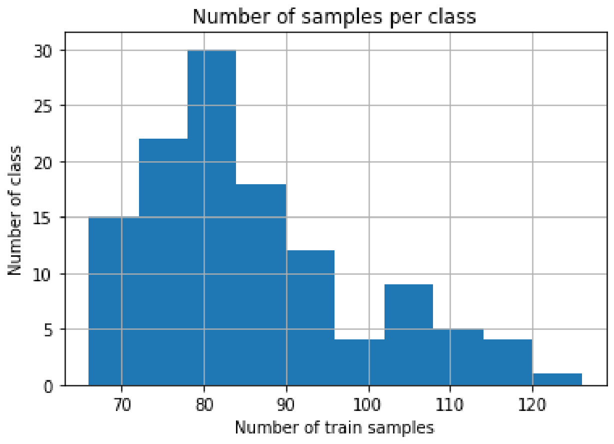

Figure 1: Number of samples per class.

After setting the sample classes we have divided the samples into training set and validation set in the 80:20 ratio. Figure 2 shows the train and the validation split per class. It can be seen that all classes are in both sets.

Our approach was using transfer learning on models that were pre-trained on ImageNet. We examined different network architectures and tuned the hyper-parameters such as optimizer, learning rate, batch size etc. After we found the best configuration per model we combined the models output using a linear classifier to get the final score.

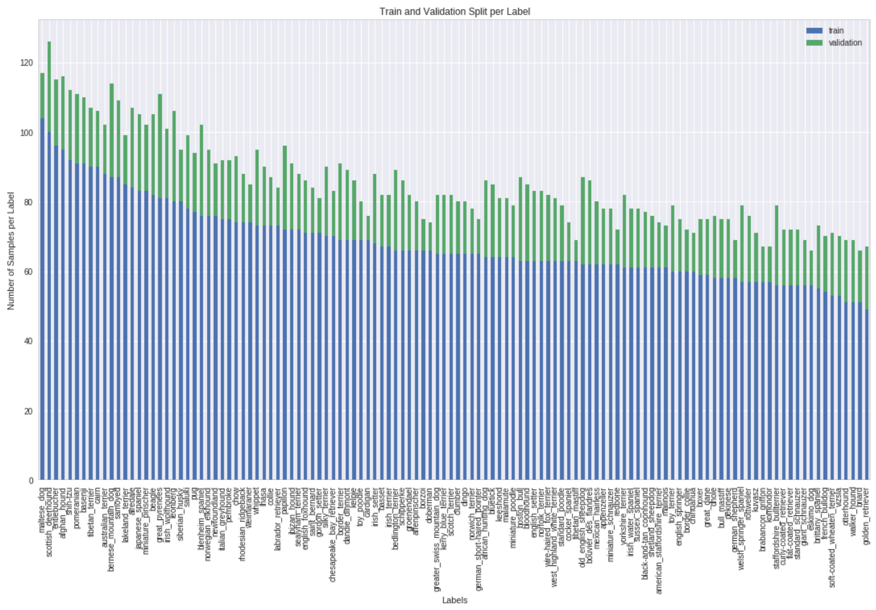

Figure 2: Number of samples per class.

2 Related work

According to [1], X Wang et al, subordinate-level categorization is the problem of determining categories of similar objects within the same basic-level class, such as different kinds of dogs. In recent years the dominant approach for this kind of problems is using neural networks. After reviewing the Kaggle dog-breed-identification discussion and kernel sections the preferred approach was transfer learning. This approach stems from the fact that the training set of size 10,223 is not enough to train a neural network from scratch. On the Pytorch website, there is a tutorial [2] regarding transfer learning for image processing using torchvision models. Pedro Lima wrote a kernel on using pre-trained Pythorch models for the dog-breed-identification task [3]. In [4], Yang Peiwen reported the following results on pre-trained models: batch size of 128 samples, Adam optimizer with a learning rate of 1e-4, weight decay 1e-5 and 50 epochs. Part of this table is displayed in table 1.

In [5], TreB1eN found after empirical experiments that in the pre-processing stage center cropping can gain a better result than resize. For this specific data set keeping the original image height and width ratio is more important than keeping the image margin. Additionally, he found that data augmentation does not improve model performance. Furthermore, he tried the following pre-trained models and found that their rank on the dog-breed-identification task is the same as on the image-net data set: nasnet, inception-v4, inception-v3, dpn107, xception, resnet152. In addition, he found that the combining the bottleneck features of different models together in a linear classifier can yield better results than a individual model.

In [6], Kirill Panarin conducted an analysis of model errors. He created a confusion matrix and draw images of the classes that the model had the most misclassification regarding them. He found that the network was confused between breeds with large resemblance. The resemblance is so large that even a human cannot differentiate between the breeds.

Table 1: Example event log

Model	Validation loss
inception-v3	0.2961
resnet152_v1	0.3994
resnet101_v1	0.4104
densenet161	0.4181
densenet201	0.4534
resnet50_v2	0.4844
resnet50_v1	0.4961
densenet169	0.5125
resnet34_v2	0.5367
vgg19_bn	0.5572

Vedder found that adding more layers at the end of Resnet, VGG, and Inception pre-trained network does not reduce the validation loss [7].

3 Implementation

Transfer learning from network architecture that was pre-trained on ImageNet data-set was the selected approach. All network parameters were frozen except for the last fully connected layer. This was the chosen approach due to the small size of the test data set 10k images compared to the ImageNet data-set size of more than 14 million images. Additionally, it is expected that transfer learning will work well for this task because it is very similar to the ImageNet classification task. More over, the dog breed data set is part of ImageNet data-set.

The test set was divided to test and validation 80:20. In each experiment, the selected model was the one that yields the highest accuracy on the validation set. Re-size, center crop and normalize using the average ImageNet image was done to the images as a pre-processing. Based on previous work [5],data augmentation was not used.

To start we had based our code on Pedro Lima's kernel [3]. We used resnet50 as the pre-trained model, with a learning rate of 0.001, cross-entropy as the loss function, 10 epochs and SGD with momentum 0.9 as the optimizer. The base score we received on the test set was 0.7625.

The first change we have made was to change the optimizer to Adam. It yield a score of 0.4988. Then we have tried various resnet models: resnet50, resnet101 and resnet152. The best score came from the most deep network, res-net152, with score of 0.3876.

Afterwards, we have examined additional architectures: inception-v3, squeezenet1_0, densenet121, vgg_19_bn and different learning rates: 0.001, 0.0001, decreasing the learning rate manually from 0.001 to 0.0001 when the loss reaches a stable number. The architecture that yield the best score was inception-v3 with a learning rate 0.001. The score was 0.504.

We have noticed during the experiments that regardless of the model or the learning rate, after the 6 epoch the validation loss stopped decreasing.

To summarize the experiments, the best two models where resnet152 (0.3876) and inception-v3 (0.504). Our next step was creating a linear combination of the best two models using logistic regression classifier. Each network produced a vector with a length of 120, per image. Each entry in the vector represented the probability that the label represents the image. We concatenated the two results so each image had 240 entries. Thereafter, logistic regression classifier was trained on the train and the validation data sets. It yield a score of 0.622 on the test set. This score is worst than the score of each network separately.

This result surprised us. In order to understand whether the models (resnet152 and inception-v3) add information to each other or misclassify the same labels we drew the confusion matrix and the top 10 misclassified pairs of breeds for each model. Figure 3 shows that both models are misclassified on similar pairs. The most common error is the same in both models. We hypothesize that if the models were wrong regarding very different pairs, the linear classifier would have been able to perform better.

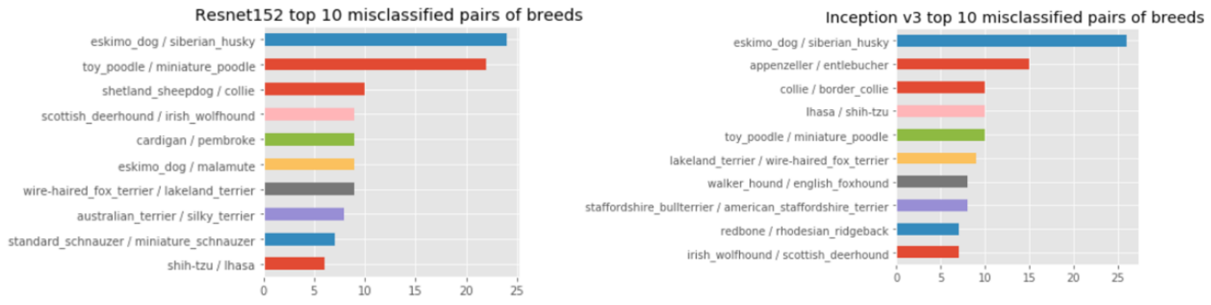

Figure 3: Comparison between the top 10 misclassified pairs of breeds resnet152 and inception v3.

4 Discussion

In previous work, Yang Peiwen reached a loss of 0.296 using transfer learning (see the best score in table 1). These results have not been reproduced in this work. It may have been due to the difference in the deep learning frameworks MXNet vs PyTorch and the stochastic nature of training.

In this work, we used transfer learning on different pre-trained models, we have explored different hyper-parameters and combined the best two configurations using a linear classifier which did not reduce the loss. The results of the combined models were less than the result of each model alone. It may have been necessary to do the combination of the models in a different way or to combined more models.

A technical limitation that we had was working on the Google colaboratory. It limited us in terms of the hardware we have had to our disposal. Training resnet152 for 10 epoch twice drained our daily GPU allocation. As a result, we avoided from conducting a full exhausting search on the network hyperparameters and focused on the search of the parameters that were discussed in previous works. Furthermore, for this reason, we have decided to freeze the weights of the entire pre-trained network except for the last fully connected layer. This is in contrast to the method that is trying to train the entire network with a very small learning rate and the last fully connected layer with a bigger learning rate. Additionally, the technical limitation were the deciding factor in the decision to combine the different models using an external linear classifier and not to combine the models using end to end network.

An important point that can be understood from the competition discussion [8] is that the train and test sets of this competition is comprised of the publicly available Stanford Dogs Dataset [9]. We felt that training the network on the test set is a deception. So we chose not to use this information. Nonetheless, it is inspired us to think of a solution for future work that would be moral and help us separate similar classes. It is possible to add training data with weak labels using Google Image search. This can be done by taking the first 30 images that Google returns for a specific breed query and label those images as related to this breed.

Another direction that was left unexplored is selecting the models to combine in the linear classifier according to the biggest difference in their confusion matrix.

5 Conclusions

The aim of this work was to classify dogs according to their breed. Transfer learning was the selected approach to solve the problem. Many pre-trained network architectures and hyper-parameter were tested. The best two models were combined using a linear classifier. To our surprise, the combination did not yield better results then the models separately. The best score was 0.3876. It was achieved using Resnet152 as a pre-trained model, Adam as the optimizer, a learning rate of 0.001 and batch size of 128.

We are aware that our solution is not perfect. This is due to the fact that the first place on the leader board reached multi-class log loss of 0. We have learned a lot and achieved the best result in our time and resources limitations.

References

[1] Xiaolong Wang, Vincent Ly, Scott Sorensen, and Chandra Kambhamettu. Dog breed classification via landmarks. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 5237–5241. IEEE, 2014.

[2] Nathan Inkawhich. Finetuning torchvision models. https://pytorch.org/tutorials, 2017.

[3] Pedro Lima. Transfer learning in kernels with pytorch. https://www.kaggle.com/pvlima/use-pretrained-pytorch-models.

[4] Yang Peiwen. Dogbreed$_g luon.https : //github.com/ypwhs/DogBreed_g luon.$

[5] TreB1eN. Dog-breed-competition-with-only-the-pretrained-models-from-imagenet. https://github.com/TreB1eN/Dog-breed-competition-with-only-the-pretrained-models-from-imagenet.

[6] Kirill Panarin. Dog breed classification using deep learning. https://towardsdatascience.com/dog-breed-classification-hands-on-approach-b5e4f88c333e.

[7] Vedder. How to improve pretrained models. https://www.kaggle.com/c/dog-breed-identification/discussion/44645.

[8] Yang Peiwen. I used inception v3, xception, resnet50, vgg16, vgg19 on the stanford dogs dataset. (https://www.kaggle.com/c/dog-breed-identification/discussion/4077.

[9] Li Fei-Fei Aditya Khosla, Nityananda Jayadevaprakash Bangpeng Yao. Stanford dogs dataset. http://vision.stanford.edu/aditya86/ImageNetDogs/main.html.