Dog Breed Identification

Wenting Shi, Jiaquan Chen, Muyun Liu,Fangyu Liu
Electrical and Computer Engineering
University of California, San Diego
California, CA 92037
{$wes053, jic496, mul046, fal053$}@eng.ucsd.edu

***Abstract*— Pattern recognition(PR) is realized as a human recognition process which can be completed by computer technology. We should first enter useful information of identifying the object into the computer. For this reason, we must abstract the recognition object and establish its mathematical model to describe it and replace the recognition object for what the machine can process[1]. The description of this object is the pattern. Simply speaking, the pattern recognition is to identify the category to which the object belongs, such as the face in face recognition. Our project is based on PR which is to identify the dog's breed. In our project, based on 10,000+ images of 120 breeds of dogs, we use 4 methods to do the identification. Each method has a different training model. The four models are ResNet18, VGG16, DenseNet161, and AlexNet. Based on our models, we also make some improvements on the optimization methods to increase our identification accuracy. After our comparisons, we find that the DenseNet model is the best, and we take it as our prime model. Our best accuracy can be up to 85.14%.**

I. INTRODUCTION

Classification is one of the most important parts of machine learning. Our project is to identify different dogs from different breeds. Our dataset is downloaded from Kaggle: Dog Breed Identification. There are totally 10,000+ images of dogs in our dataset. We separate our dataset into two parts for training and testing. Since our data is not very big enough, we use transfer learning to augment our dataset to prevent overfitting. Also, we have a label document, which includes the dog's id and its breed. We have tried 4 methods to train our data, and make some improvements on some of them to increase our identification accuracy. The models we use in our project are ResNet18, VGG16, DenseNet161, and AlexNet. They are all popular methods to do classification. ResNet was proposed in 2015 and won the first place in the ImageNet competition classification task because it was simple and practical. Afterward, many methods were built on the basis of ResNet50 or ResNet101, and detection, segmentation, and identification were performed. Alpha zero also uses ResNet, so it can be seen that ResNet really works well. The number following the ResNet is the number of layers. We first use one of the four models as the pre-training model and then we add two full-connected layers at the end of the pre-training model. In order to speed up the coverage, we tried to implement SGD and Adam in the project. The parameters we use are after our cross-validation to get the best solutions. In this report, we will first discuss the dataset we use and how we clean it, and then we will discuss how we use the models and what improvement we make to increase our accuracy.

II. DATASET AND FEATURES

ImageNet is a large visual database designed for use in visual object recognition software research. Over 14 million images have been hand-annotated by ImageNet to indicate what objects are pictured. The dataset we used is from Kaggle: Dog Breed Identification[2]. This dataset contains more than 10000 images and totally 120 breeds of dogs, and all the images are from ImageNet. Every image in the dataset has its own ID and corresponding dog breed. We divide the dataset into two parts: train set and test set. Train set occupies about 90% of the whole dataset, and the rest is test set.

All the data are RGB images, and in the center of each image, there is the body of the dog. The features we use are color, shape and texture. As we can distinguish different dog breeds from colors, shape of organs, and texture of the body and so on. CNN uses convolution layers to extract those features from the images.

As the number of images in the dataset is not big enough, we also did some pre-processing to enlarge the dataset. We resize all the images to a fixed resolution of 224x224. And also did random scale, random crop, random horizontal flip to augment the data. Fig.1 shows an overview of our dataset. Fig.2 shows the distribution of breeds of dogs in the dataset.

Fig. 1. Overview of dataset

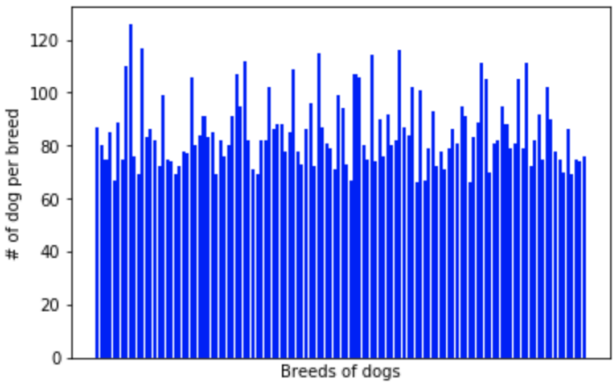

Fig. 2. Distribution of images in dataset

III. Related Work

Various models are used in image classification. On approach is based on Support Vector Machine (SVM) which is a supervised learning model with associated learning algorithms that analyze data used for classification and regression analysis[3]. In 2012, Liu et. al proposed a model based on SVM regressor [4]. In their paper, they first use an SVM regressor using greyscale SIFT descriptors as features to isolate the face of the dog. As for the result, their accuracy can reach to 90%.

In 2012, Liu J., Kanazawa A., Jacobs D. and Belhumeur P. proposed a novel approach to fine-grained image classification by using part localization [5]. They take dog breed identification as a test case. In their paper, they extract corresponding parts to make classification. Their approach features a hierarchy of parts and breed-specific part location. The recognition rate can reach to 67%.

Another widely used approach in species classification is based on Convolutional neural networks (CNNs). Hsu, David [6] approach the dog breed classification by using CNNs based on LeNet and GoogLeNet architectures. The architectures for LeNet include (number of CONV-RELU-POOL) x N + (number of FC) x M + FC-120 and this architecture performs pretty well on many classification problems like as ImageNet. As for the GoogLeNet, it can reduce overfitting while maintaining good performance on classification.

Similar to Hsu's approach, we use CNNs to identify 120 breeds of dogs. The difference is, instead of LeNet and GoogLeNet, we used ResNet, VGG, DenseNet and AlexNet. We also introduce transfer learning to improve accuracy. Compare to Hsu's model, Our model is less complex and run time is much shorter.

IV. Methods

As the dataset contains only about 10000 images, which is a little small for us to train a good neural network. After reading papers and discussing, we decide to use transfer learning[7]to train our own model. Transfer learning is a research skill in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. Transfer learning saves time, and it can also solve some problems that may be caused by a small dataset. To implement transfer learning method, firstly we need to choose a good pre-trained model. In this paper, we tested four different pre-trained models, and finally selected the one with the best performance on our dataset, as our based model, to do transfer leaning. The four models we tried are ResNet18, VGG16, AlexNet and DenseNet161. As there are totally 120 breeds of dogs, the final amount of labels after classification should be 120. To fit the model mentioned above to our dataset, we need to add some fully connected layers at the end of the model as our classifier layer. Fig.3 shows the whole process of our dog breed identification.

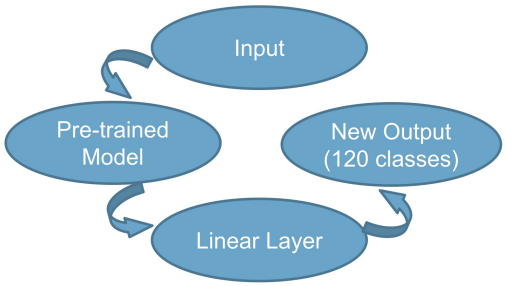

Fig. 3. Process of Classification

A. ResNet18

ResNet[8] means residual neural network. Residual learning framework can ease the training of networks that are substantially deeper than those used previously. Residual block is that adding the output of one layer and the input of the previous layer together as the input of the next layer. Here we used ResNet18, which is trained using ImageNet.

$$y = F(x, W_i) + x \tag{1}$$

F(x,Wi) is represented as the residual mapping function that we need to train. And through the previous figure we can find the same stack layer and residual module, just add an x. The architecture of ResNet we use is shown in Fig.4.

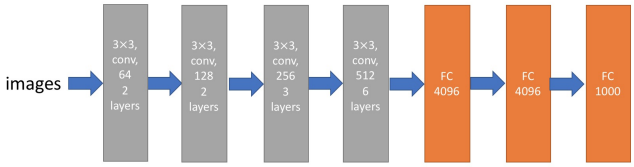

Fig. 4. Block of ResNet

B. VGG16

VGG is a convolution neural network model proposed by K. Simonyan and A. Zisserman[9]. They steadily increase the depth of the network by adding more convolution layers, which is feasible due to the use of very small 3x3 convolution

filters in all layers and 2x2 max pooling and down-sampling with CNN layers with stride 2. Also, it consists Global average pooling layer and a 1000-way fully-connected layer with Softmax in the end. And this network consists 16 layers in total. As this algorithm first proposed, it was trained on 4 GPUs for 2 to 3 weeks, and currently the most preferred choice in the community for extracting features from images. Here we used VGG16 which is pre-trained on ImageNet. The architecture of VGG16 is shown in Fig.5.

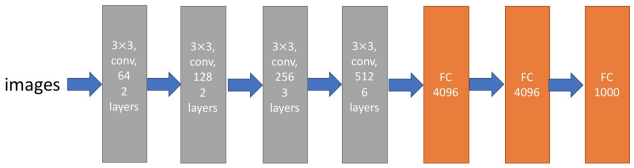

Fig. 5. Block of VGG

C. AlexNet

AlexNet[10], which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, AlexNet is used non-saturating neurons and a very efficient GPU implementation of the convolution operation, and it splits the channels into two stages in order to be trained simultaneously on two Nvidia Geforce GTX 580 GPUs. To reduce overfitting in the fully-connected layers, dropout was used in the network. Here we used AlexNet which was trained on ImageNet. Its architecture is shown in Fig.6

Using traditional saturating activation functions such as Sigmoid and tanh, we often do normalization to avoid situations where neurons are insensitive to large input values (the total output value after Sigmoid and tanh Is a value close to 1). In fact, non-saturating functions such as ReLU do not do normalization, but the author still gives a method called Local Response Normalization, but this normalization operation is after ReLU. The network can be described in the following formula:

$$b^i_{x,y} = a^i_{x,y}/(k + a \sum_{j=max(0,i-n/2)}^{min(N-1,I+N/2))} (a^j_{x,y})^2)^\beta \qquad (2)$$

$a^i_{x,y}$ is the output of ReLU of the ith channel's at (x,y), k, n, a, β are constants.

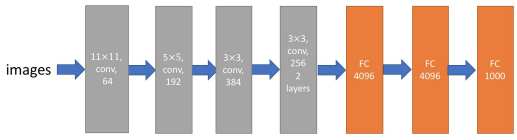

Fig. 6. Block of AlexNet

D. DenseNet161

Dense Convolutional Network (DenseNet)[11], which connects each layer to every other layer in a feed-forward fashion. For each layer, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers. DenseNets have several advantages: they reduce the influence on vanishing gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

One advantage of DenseNet is that the network is narrower and has fewer parameters. This is due in large part to the design of the dense block. It is mentioned later that the number of feature maps per convolution layer in the dense block is small, and not as wide as hundreds of thousands of other networks. At the same time, this type of connection makes the transmission of features and gradients more efficient and the network is easier to train. The reason is that the input information and gradient information are transmitted between layers, and now the sense connection is equivalent to that each layer is directly connected to the input and loss, so it can reduce the gradient disappearance, so deeper networks are not a problem. In addition, the author also observed that this sense connection has a regularization effect, so it has a certain inhibitory effect on overfitting[12]. The architecture of DenseNet16 we used in the project is shown in Fig.7.

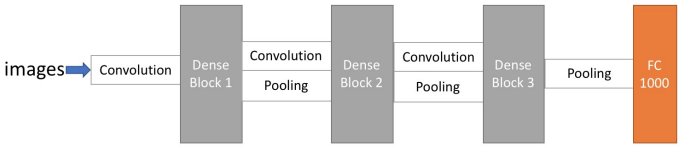

Fig. 7. Block of DenseNet

All the models mentioned above were trained on ImageNet. As our dataset is only a small part of ImageNet, and the final label of classification is 120.

V. Experiment

Some of the models we are using are too complex to fit our training dataset. Especially, there are more than 20,000 parameters in VGG16. So overfitting is a general problem in our experiment. In this section, we plan to discuss some methods we used to preprocessing our data, improve the effect of model training and alleviate the effect of overfitting.

A. Data Augmentation

In our experiment, We apply data augmentation to preprocessing dataset in order to increase the number of training data. Because there are 120 breeds of dogs and more than 10,000 images, which is not big enough for models. We preserve label of each original image in transformation. our exact methods to transform each image are as follows:

1) Flip: Each image can be flipped horizontally and vertically. In our task, we just flip images horizontally.

2) Crop: Random Cropping is a common method in augmentation, which is to randomly sample a section from the original image and resize it to its original image size. In our experiment, we randomly extract a 224×224 pixels section from 256×256 pixels.

3) Scale: Each image can be scaled outward or inward. After scaling, we also need to resize it. Because image size will be different from the original size. We set our parameter of scaling as 0.8-1.0 in our code.

B. Transfer Learning

Transfer learning is to get knowledge from an existing task that has already been trained and apply it on a different but related task[13]. The pre-trained model that we download includes existing ConvNets and the associated trained network parameters. In our task, we need to classify 120 breeds of dogs from only 10,000 images. We do not have sufficient training data to train a new and complex model that can perform well on testing dataset. Thus, we directly download different models ,such as ResNet18, VGG16, AlexNet, and DenseNet161 from Pytorch which we use as our framework. But the weights on the last layer of each model are exactly random and the number of categories in outputs of each model is different from 120 that we need. In order to fix it, we freeze all layer but the last. Freezing is to make the layer weights invariant during training. We also add two fully-connected layers to replace the last layer of each pre-trained model.

C. Stochastic Gradient Descent(SGD)

Stochastic Gradient Descent is one of the famous optimization algorithms. SGD algorithm[14] is a drastic simplification. Because it does not compute the gradient of $En(fw)$ exactly. In each iteration, it estimates this gradient on the basis of a single randomly picked example z_t :

$$w_{t+1} = w_t - \gamma_t \nabla_w Q(z_t, w_t) \tag{3}$$

The stochastic process $w_t, t = 1, ...$depends on the examples randomly picked at each iteration. Compared with original gradient descent, SGD is able to converge under technical assumptions with high probability, which result in saving time on computation. In our task, we set SGD with mini-batch size, learning rate schedule, momentum, and weight decay. And we adjust these parameters to find the best performance.

D. Adaptive Moment Estimation(Adam)

In our previous experiment, there is a big problem for us to choose a proper learning rate. When a learning rate is too small, convergence is painfully slow. However, a learning rate is too large, which leads to hinder convergence and causes the loss function fluctuate around the minimum or even to diverge. Thus, we use Adam in our experiment. Adaptive Moment Estimation[15] called Adam is an extension to stochastic gradient descent. Adam computes adaptive learning rates for each parameter and keeps an exponentially decaying average of past gradients.

E. Parameter tuning

In our experiment, we set SGD with mini-batch size as 128, learning rate schedule as 0.05, momentum as 0.9, and weight decay as 1e-4. For mini-batch size, setting it as 64 need to spend more time on training. When we choose 256 as mini-batch size, the memory usage is not sufficient. Moreover, before we choose 0.05 as our learning rate, we try different values of learning rate. When learning rate is too small, it spend much more time for models on converging. It is easier for a too large learning rate to miss optimal value or diverge.

VI. RESULTS

A. Loss Analysis

In our project, we compare 4 different pre-trained model: ResNet18, AlexNet, DenseNet161 and VGG16. We use softmax and cross-entropy to evaluate their performance in terms of the training and testing loss. The softmax function is a generalization of the logistic function that "squashes" a K-dimension vector z of arbitrary real values to a K-dimensional vector $\sigma(z)$ of real values, where each entry is in the range (1,0), and all the entries add up to 1. The function is given by[6]:

$\sigma : R^K \to \left\{ \sigma \in R^K \mid \sigma_i > 0, \sum_{i=1}^{K} \sigma_i = 1 \right\}$

$\sigma(Z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$

Cross-Entropy loss measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increase as the predicted probability diverges from the actual label. In multiclass classification, cross-entropy can be calculated as[6]:

$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$

The loss comparison results after 50 epochs are shown in Fig.8 and Fig.9.

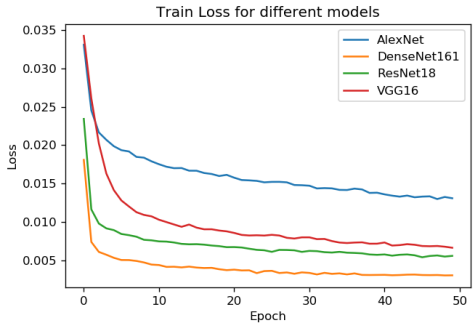

Fig. 8. Comparison of Train Loss

B. Accuracy Analysis

To better evaluate the performance of these four models, we also compare their accuracy. The accuracy comparison results after 50 epochs are shown in Fig.10 and Fig.11.

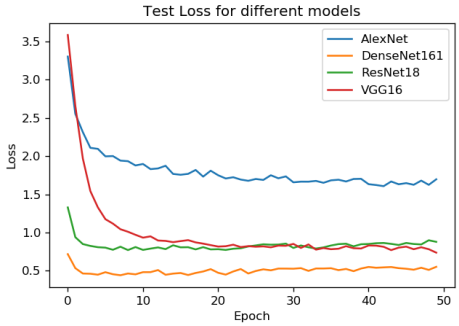

Fig. 9. Comparison of Test Loss

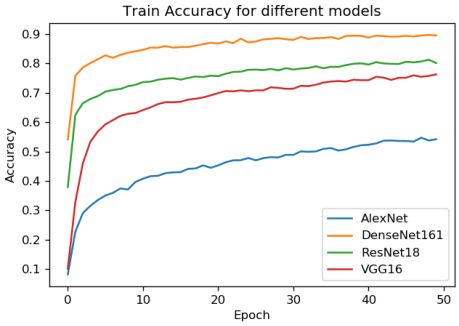

Fig. 10. Comparison of Train Accuracy

C. Sample of Output

The accuracy of our model can reach to 85.14% Fig.12 shows one of our sample output.

D. Overall results analysis

From the experiment,we can compare the performance of the different models. Table 1 shows the conclusion of the comparison.

TABLE I
COMPARISON OF DIFFERENT MODELS(50 EPOCHS)

Model	Loss (Train/Test)	Accuracy (Train/Test)
ResNet18	0.0056/0.8782	0.8013/0.7542
DenseNet161	0.0031/0.5522	0.8954/0.8514
AlexNet	0.0131/1.6958	0.5419/0.5556
VGG16	0.0067/0.7371	0.7630/0.2667

Based on previous results, we can rank performance of the four models.

Train Loss (small to large):
DenseNet161→Resnet18→VGG16→Alexnet
Test Loss (small to large):
DenseNet161→VGG16→Resnet18→Alexnet
Train Accuracy:
DenseNet161>Resnet18>VGG16>Alexnet
Test Accuracy:
DenseNet161>Resnet18>Alexnet>VGG16
Therefore, DenseNet161 is the best choice for our model.

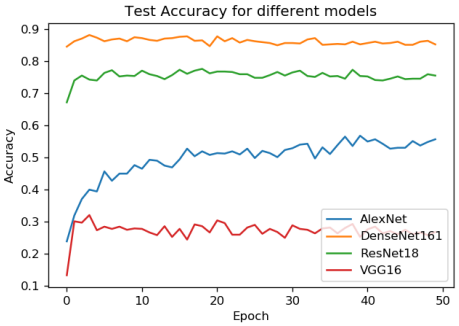

Fig. 11. Comparison of Train Accuracy

Breed	Probability
maltese_dog	87.82%
sealyham_terrier	11.12%
west_highland_white_terrier	0.214%
shih-tzu	0.153%
malinois	0.055%

Fig. 12. Sample of output

VII. CONCLUSION AND FUTURE WORK

A. Conclusion

In our project, we implement the CNNs to identify 120 breeds of dogs. We first pre-processed our data set by: 1. Putting the same breed of dog images into the same folder; 2. Augmenting the dataset. Second, we use transfer learning to moderate the problem that our dataset is not big enough and improve accuracy. Third, we train our model and tune the parameters to find the optimization. We also use adaptive learning rate to improve our model. Then, we use multi-methods (dropout, weight decay) to avoid overfitting. At last, we compare different pre-trained models (ResNet18, DesenNet161, AlexNet and VGG16) and different optimizers (Adam and SGD). Based on our experiments and analysis, we decided to choose: 1. DenseNet161 and 2 fully-connected layers as our pre-trained model; 2. SGD with mini-batch size 128 as our optimizer. The block of our models is shown in Fig.13.

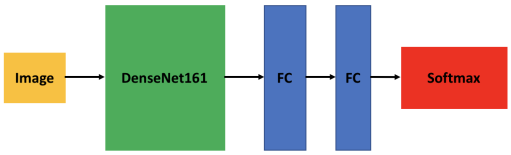

Fig. 13. Block of Models

B. Future Work

In the future, there are three directions we can go. First, in our project, we only use CNN to implement classification. To make a better choice, we can compare it with other train models like SVM and PCA. Second, in our model, we only add two FC layers after pre-trained model. To improve the accuracy, we can try to add more layers. Third, our accuracy is still not large enough. To improve accuracy, we can try other methods to avoid overfitting (like stop early) and tune the parameters (like learning rate, momentum and batch-size).

REFERENCES

[1] Pavel, M. (1992). Mathematics in pattern recognition. Pattern Recognition Letters, 13(4), 305. doi:10.1016/0167-8655(92)90080-j
[2] https://www.kaggle.com/c/dog-breed-identification/data
[3] Cortes, Corinna; Vapnik, Vladimir N. (1995). "Support-vector networks". Machine Learning. 20 (3): 273297.
[4] J. L. et al. Dog breed classification using part localization. Computer Vision: ECCV 2012, pages 172185, 2012.
[5] Liu J., Kanazawa A., Jacobs D., Belhumeur P. (2012) Dog Breed Classification Using Part Localization. In: Fitzgibbon A., Lazebnik S., Perona P., Sato Y., Schmid C. (eds) Computer Vision ECCV 2012. ECCV 2012. Lecture Notes in Computer Science, vol 7572. Springer, Berlin, Heidelberg
[6] Hsu, David."Using Convolutional Neural Networks to Classify Dog Breeds".
[7] Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." IEEE Transactions on knowledge and data engineering 22.10 (2010): 1345-1359.
[8] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[9] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
[10] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
[11] Huang, G., Liu, Z., Weinberger, K. Q., van der Maaten, L. (2017, July). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (Vol. 1, No. 2, p. 3).
[12] Huang, G., Liu, Z., Maaten, L. V., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2017.243
[13] Torrey, Lisa, and Jude Shavlik. "Transfer learning." Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. IGI Global, 2010. 242-264.
[14] Bottou, Lon. "Large-scale machine learning with stochastic gradient descent." Proceedings of COMPSTAT'2010. Physica-Verlag HD, 2010. 177-186.
[15] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).