Cats and Dogs Identification Using Convolutional Neural Network

Phuc Nguyen
College of Computing and Informatics
Drexel University
Philadelphia, United States
phn26@drexel.edu

***Abstract*—The current paper presents a fine-grained multi-class object categorization problem, namely determining the breed of a cat or dog in a given image. The visual problem is very challenging as there can be very subtle differences between the breeds and some animals can be very deformable (particularly cats). The model in the Oxford paper, where the datasets came from, used a multi-class SVM and achieved an average accuracy of 59% [1], which is a very encouraging result at the time. The presented systems below employ innovative methods in deep learning, including three different convolutional neural network (CNN) architectures on the Oxford Cats and Dogs Breeds Dataset. The project concludes a better accuracy in all of the three CNNs using PyTorch framework with pre-trained weights for standard computer vision benchmark dataset, ImageNet, than the original Oxford SVM model.**

I. Introduction

The Convolutional Neural Networks (CNN) is a cutting edge area of deep learning and has been shown to achieve strong performance on different topics of deep learning: text classification tasks [2], traditional Natural Language Processing tasks [3], image recognition tasks [4], etc. CNNs are very similar to traditional Artificial Neural Networks, which have learnable weights and biases. However, CNNs become predominantly successful in object categorization because of its feature extractors [5] from image perk. While using a algorithm with pixel vector we can lose a lot of spatial interactions between pixels, a CNN effectively uses adjacent pixel information to downsample the image first by convolution and then uses a predictions layer at the end.

The current paper presents the methodology and the results of fine-tuning CNNs for three different types of architectures, using the Oxford Cats and Dogs Breeds Dataset: AlexNet [4] (ImageNet LSVRC-2012 winner), ResNet [6] (ILSVRC 2015 winner), and Efficientnet [7] (state-of-the-art 84.3% top-1 accuracy on ImageNet). There are multiple variants in each types of architectures, and the system implemented below uses three models: AlexNet, ResNet-50 and Efficientnet-b0 respectively. The choice of the models is limited by the GPU's capacity, and thus, may not capture of the Efficientnet's scaling capability (Fig. 1).

II. Related Works

The current section presents previous attempts at addressing the problems tackled by previous research. The authors of the Oxford dataset - Omkar, Andrea, Andrew and Jawahar [1] - approached the breed discrimination problem by first obtaining the pet family data. Two SVMs were created to distinguish cats and dogs based on shape and appearance scores and combined together and yielded a accuracy of approximately 95.37%. Then, they train a multi-class SVM by using the 1-Vs-rest decomposition [8] (this means learning 12 binary classifiers for cats and 25 for dogs). The best breed classification for cats and dogs are 63.48% and 55.68% respectively, and an average of 59.58% accuracy for the whole model.

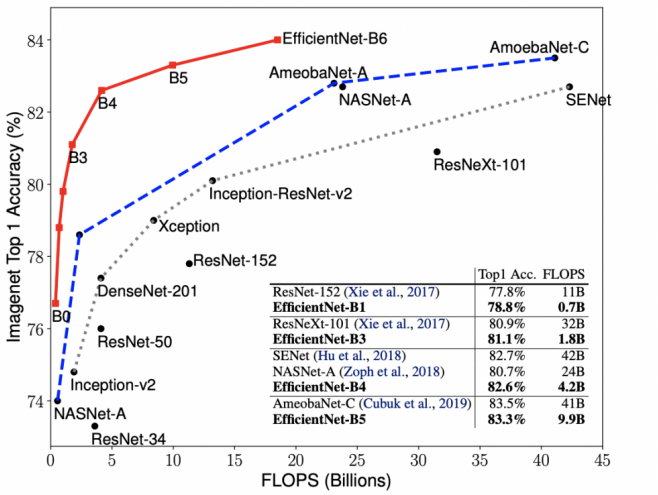

Fig. 1. Model Size vs. ImageNet Accuracy. The EfficientNets significantly outperform other ConvNets. In particular, EfficientNet-B7 achieves new state-of-the-art 84.3% top-1 accuracy but being 8.4x smaller. The model implemented EfficientNet-B0, however, only achieves somewhere the same accuracy as ResNet-50

III. Methods

A. Dataset and Preprocessing

The presented CNNs learning methodology revolves around the Oxford Cats and Dogs Breeds [1] Dataset. It is a 37 category pet dataset, including 25 dog breeds and 12 cat breeds, with a 7390 different resolution images in total or roughly 200 images for each class. The label for each image is in the image's file name in the form of "breed_number.jpg".

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	227x227x3	-	-	-
1	Convolution	96	55 x 55 x 96	11x11	4	relu
	Max Pooling	96	27 x 27 x 96	3x3	2	relu
2	Convolution	256	27 x 27 x 256	5x5	1	relu
	Max Pooling	256	13 x 13 x 256	3x3	2	relu
3	Convolution	384	13 x 13 x 384	3x3	1	relu
4	Convolution	384	13 x 13 x 384	3x3	1	relu
5	Convolution	256	13 x 13 x 256	3x3	1	relu
	Max Pooling	256	6 x 6 x 256	3x3	2	relu
6	FC	-	9216	-	-	relu
7	FC	-	4096	-	-	relu
8	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

Fig. 2. AlexNet architecture

layer name	output size	18-layer	50-layer
conv1	112×112	7×7, 64, stride 2	
conv2_x	56×56	3×3 max pool, stride 2	
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax	
FLOPs		1.8×10^9	3.8×10^9

Fig. 3. ResNet-50 architecture

The following steps were implemented as part of the pre-processing phase:

- Divide the dataset into 37 classes folders to easily label the data.
- Split the dataset into training and test sets with 70/30 ratio, resulting in 5172 images in training data set with roughly 150 per breed, and 2218 images in test set with roughly 50 per breed.
- Resize train and test images to 224x224 for all three models.
- Perform data augmentation on the training set to increase the diversity of the data available in the training models, including horizontally flip the given image randomly with a given probability, crop the image at the center, rotate the image by random angle, random affine transformation of the image keeping center invariant.
- Normalize training and test sets with ImageNet's mean and standard deviation.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Fig. 4. Efficientnet-b0 architecture

B. Fine-tuning architectures

By using transfer learning [9], we take advantage of the pre-trained model's weights to learn features from our problem with better performance than training the model from the scratch. This is because the pre-trained models from ImageNet are exposed and learned from millions of images and is taught the general reoccurring features and will have easier time converging in our problem. Three pre-trained models I used for my projects are:

- The AlexNet architecture: a classic architecture with total of 8 layers (5 convolutional layers and 3 fully connected layers) and 60 millions parameters (Fig. 2).
- The ResNet-50 architecture: 50-layer architecture with the use of the Residual Block [6] and total of 23 millions parameters (Fig. 3).
- The Efficientnet-b0 architecture: 237-layer architecture with the use of Inverted Residuals and Linear Bottlenecks [7] and total of only 4 millions parameters (Fig. 4).

During the training, the last fully-connected layer is un-frozen and fine-tuned to match with number of classes for this problems, while the other layers are unchanged and frozen.

C. Learning and Hyperparameters

The learning processes of the CNNs are performed on a personal computer with a GeForce GTX 1060 with Max-Q Design GPU, an Intel® Core™ i7-8750H CPU and 16 GB RAM.

During the learning, a Softmax Cross-Entropy Loss function and Stochastic Gradient Descent optimizer with a learning rate of 0.001 and momentum of 0.9 are chosen to train all of the three models.

The three models are trained with different parameters regarding to batch size and number of epochs. In particular, the AlexNet model uses a batch size of 64 and run for around 25 epochs, the ResNet-50 uses a batch size of 32 and converges after 10 epochs, while the Efficientnet-b0's hyperparameters are 16 and 20 respectively. The differences between batch size are due to number of layers of each model and fairly low RAM capacity. As for number of epochs, all models are experience with 50 epochs at first, then use early stopping to avoid overfitting (Fig. 5).

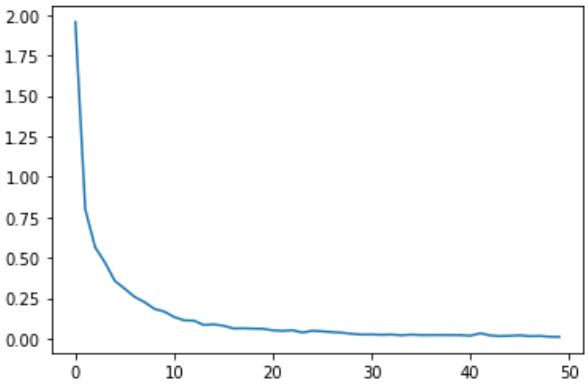

Fig. 5. AlexNet model's training loss. Early stopped at 25 epochs to avoid overfitting

Further experimented hyperparameters include: different learning rate (0.01, 0.005), difference momentum (0.5, 0.7), different epochs till convergence. Different optimizer and loss function like Adam was experimented but yields a worse accuracy then the chosen one. The reason are unknown.

IV. Result

In this section, several relevant metrics are evaluated to judge how well each model does: precision, recall, f1-score (table I), accuracy, and confusion matrix (Fig. 6).

The accuracy is the metric that tells the fraction of the prediction that got right, and in this project, is monitor on both the training and test set. The classic architecture AlexNet achieve 99.08% accuracy on training set and 79.67% accuracy on test set. The deeper CNNs like Resnet-50 and Efficientnet-b0 show better result: 92.19%, 87.68% accuracy on the former and 94.86%, 82.79% accuracy on the latter respectively on training and test datasets. These two models have close result to each other is expected, but with better hardware that can handle training more complex architecture, I expect the Efficientnet to outperform the Microsoft one.

The precision, recall and f1-score are also evaluated and shown in Table I. These metrics are obtained during making prediction on the test set.

TABLE I.

Model	*Precision*	*Recall*	*F1-score*
AlexNet	80.32%	80.93%	80.42%
ResNet-50	87.23%	88.41%	87.82%
Efficientnet-b0	82.89%	84.78%	83.77%

Further more, confusion matrices for breed discrimination are built to obtain the general mistake of the models. Although the three models can have different accuracy values, they all have major issues in distinguishing the following pair of breeds as the differences are very subtle:

- American Pit Bull Terrier vs Staffordshire Bull Terrier
- American Bulldog vs Boxer
- Birman vs Ragdoll
- Birman vs Siamese

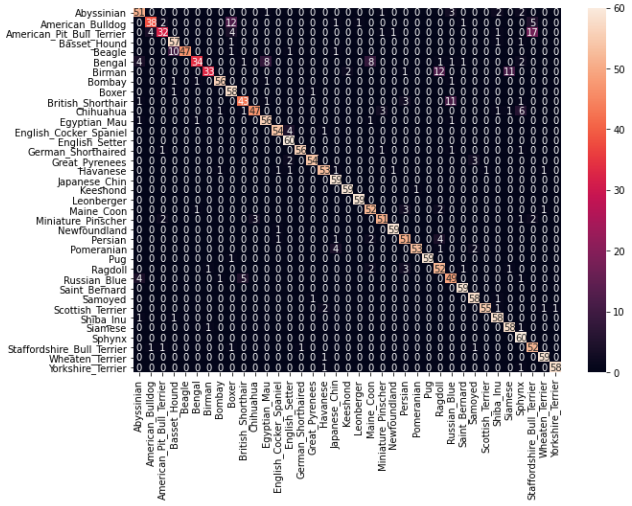

Fig. 6. Confusion matrix for ResNet-50 model

- British Short Hair vs Russian Blue
- Bengal vs Egyptian Mau
- Bengal vs Maine Coon

V. Conclusion

This paper attempts to break the old SVM's accuracy in the Oxford paper [1] using different CNNs, and has successfully done so with the differences of over 20% accuracy across three models.

VI. Future Work

In the future work, more techniques and experiments will be applied to achieve the true potential of these CNNs, which were not be demonstrated in this project due to lack of time. And time will be dedicated for studying the effects of each technique on this computer vision problem.

References

[1] Parkhi, V. (2012). Cats and dogs. 2012 IEEE Conference on Computer Vision and Pattern Recognition, 3498–3505. https://doi.org/10.1109/CVPR.2012.6248092

[2] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers, pages 655–665. The Association for Computer Linguistics.

[3] Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12:2493–2537.

[4] Krizhevsky, Sutskever. "ImageNet Classification with Deep Convolutional Neural Networks." Communications of the ACM, vol. 60, no. 6, ACM, May 2017, pp. 84–90, doi:10.1145/3065386.

[5] LeCun, Haffner. "Object Recognition with Gradient-Based Learning." Object Recognition with Gradient-Based Learning, Springer Berlin Heidelberg, 1999, doi:10.1007/3-540-46805-6_19.

[6] He, Zhang. "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pp. 770–78, doi:10.1109/CVPR.2016.90.

[7] Tan, Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. May 2019.

[8] B. Scholkopf and A. J. Smola. ¨ Learning with Kernels. MIT Press, 2002.
[9] Farzaneh Mahdisoltani, Guillaume Berger, Waseem Gharbieh, and Roland Memisevic. The more fine-grained, the better for transfer learning. 2018