Household Animals Classification Using Deep Learning

Lei Lin*
Department of Computer Science
Stanford University
lli978ru@stanford.edu

Abstract

Using deep learning to study household animals' demeanor and body language, we can find out if they are sick or not and provide necessary help in time. In order to achieve this goal, we need to start with animal species classification. In this project I will train one of the deep learning models, VGG, to distinguish between images of cats and dogs. After using the Transfer Learning from VGG-16, the accuracy can increase from 80% to over 95%. Then two ways of visualizing the model outputs have been successfully demonstrated for more insight of the VGG model.

1 Introduction

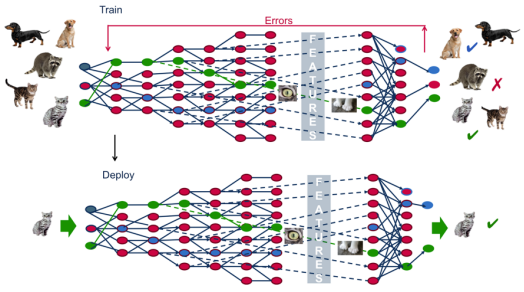

Figure 1: the demonstration of deep learning on animal classification[1].

Recently, animal detection for wildlife has been an area of great interest among biologists. Since there are many species, manually identifying them can be a daunting task. So, a deep learning algorithm that classifies animals based on their images can help monitor them more efficiently. A further possible application of this technology can be used to identify certain species' behaviors. Using deep learning to study animals, especially household animals' demeanors and body language, we may know if they are sick or not and provide necessary help and treatment in time. All these challenges necessitate an efficient algorithm for classification. In this project, Dogs vs. Cats, the data set from Kaggle will be applied on the VGG models that I made from scratch. In order to further improve the accuracy, the transfer learning will be used in this project. To gain more insight of the VGG models, I will demonstrate the methods of visualizing the model outputs.

2 Related work

Related works about animal classification were published in regard of wild animal monitoring [2-5]. The authors used convolutional neural networks to create models for animal species identification. Similarly, convolutional neural networks were used to recognize 20 species common in North America [6]. Another approach was taken for classifying different animals in automating species recognition[7]. The authors enforced ScSPM (Sparse coding Spatial Pyramid Matching) to extract

*SCPD student; https://www.linkedin.com/in/leilin/

and classify animal species over a 7 thousand image data set. After that multi-class pre-trained SVMs were applied to classify global features of animal species. All mentioned authors follow a similar pattern when using machine learning for image classification, but none of them concern household animals. Also, most of them lack the application of visualization technique. This technique gives insight into what types of features deep neural networks are learning at specific layers in the model and provides a debugging mechanism for improving a model.

3 Dataset and Features

In this project, the data set is from Kaggle [8], which provides 25,000 labeled photos: 12,500 dogs and the same number of cats. Out of all the photos, 16% are for validation, 4% for testing, and the remainder for training. Kaggle also provides unlabeled photos for testing in another folder. The images (Fig.2) are of non-uniform size (averaged around 350×500) and varying image quality. Some of them are grayscale, and some files are corrupted, making them unreadable. Additionally, the perspective can differ from faces to full-body. In some images, part of the animal is obstructed from view, and others contain more than one of the same animals.

Figure 2: Dogs vs Cats from Kaggle[8].

4 From VGG-6 to VGG-16

4.1 Develop a simple CNN model

The baseline CNN model is from the general architectural principles of the VGG [9] models. The architecture involves stacking convolutional layers with small 3×3 filters followed by a max pooling layer. Together, these layers form a block, and these blocks can be repeated where the number of filters in each block is increased with the depth of the network such as 32, 64, 128, 128 for the first four blocks of the model. Padding is used on the convolutional layers to ensure the height and width shapes of the output feature maps matches the inputs. Each layer uses the ReLU activation function. The model (Fig.3) has been fit with RMSprop optimizer which is similar to the gradient descent algorithm with momentum.

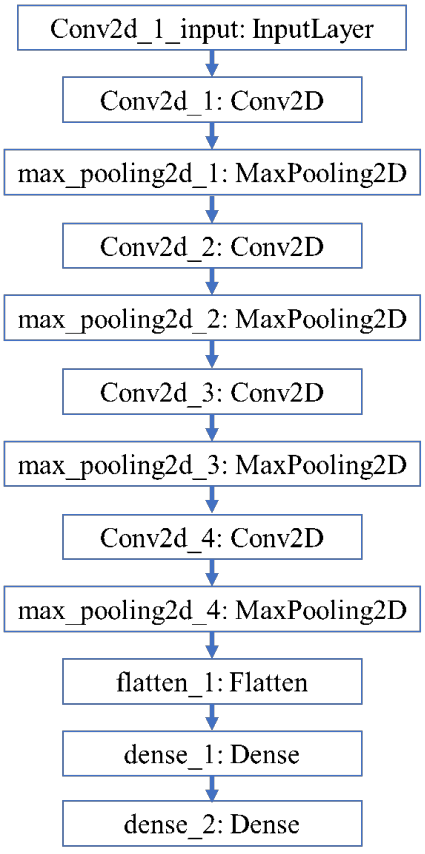

Figure 3: The structure of VGG-6

4.2 Transfer Learning from VGG-16

A more refined method would be to utilize a network which is pre-trained on a large dataset. This network would have already learned features that are useful for solving various problems such as Image Classification and Object Detection. This method would allow us to obtain better accuracy. Transfer learning involves using all or parts of a model trained on a related task. A useful model for transfer learning is one of the VGG models, such as VGG-16 [10] with 16 layers. I use the feature extraction part of the model and add a new classifier part that is tailored to

the Dogs vs. Cats dataset. Specifically, I held the weights of all the convolutional layers fixed during training, and only train new fully connected layers that will learn to interpret the features extracted from the model to make a binary classification.

4.3 Accuracy and Loss

The overfit is observed from my initial VGG-6 model. Therefore, I modify the baseline model through Dropoff and Data augmentation. The training accuracy is above 80% and validation accuracy is around 86% with the loss lower than 0.4 (red curves in Fig. 4). We could keep tuning the network to improve training accuracy by lowering the Dropout rate a bit and to train more. But I guess we're not going to be able to reach 95% on this dataset.
The VGG-16 base model (black curves in Fig. 4) with data augmentation can significantly improve the training performance. Reviewing the learning curves, we can see that the model fits the dataset quickly at the first 20 epochs. The training accuracy comes out to be around 90% and the validation accuracy being 91%. To further improve the accuracy, we can fine-tune the weights of some layers in the feature detector part of the model. In this project, we unfreeze from the layer 'block5_conv1' along with our Dense layers, resulting in more than 95% validation accuracy (blue curves in Fig.4). Applying fine-tuning allows us to utilize pre-trained networks to recognize classes they were not originally trained on, reducing the loss to 0.1.

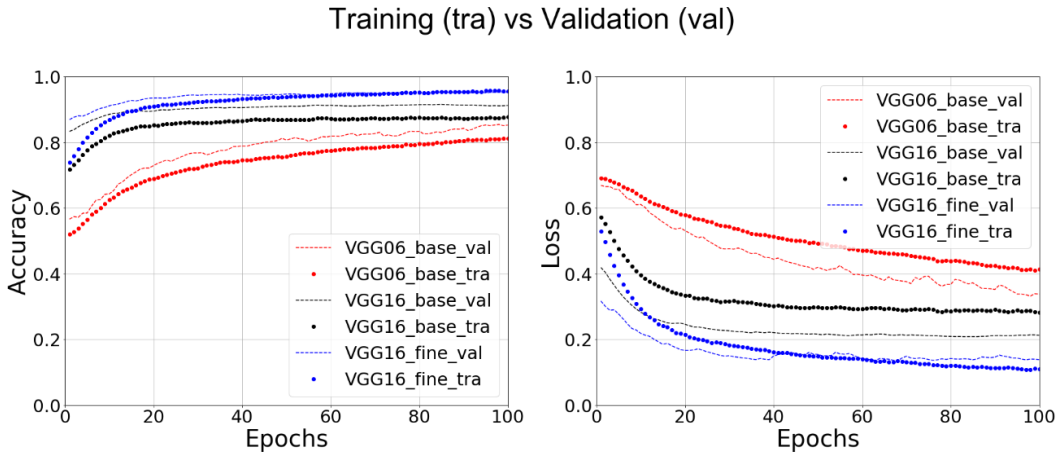

Figure 4: The result of training and validation from VGG model

5 Understanding the Convolution Network with Visualizations

Visualizing the output of the model is a great way to see how its progressing. While training deep networks, most people are only concerned with the training error(accuracy) and validation error(accuracy). Judging these two factors does give us an idea of how our network is performing at each epoch. When it comes to deep CNN networks like VGG-16 there is so much more that we can visualize, thus allowing us to learn about network architecture.[11] In this project, I will demonstrate two ways of visualization the model outputs (intermediate as well as final layers), which can help us gain more insight into working of the model.

5.1 Visualizing Intermediate Layer Activations

For understanding how the deep CNN model is able to classify the input image, we need to understand how my model sees the input image through studying the output of its intermediate layers. By doing so, we are able to learn more about the workings of these layers. On the following part, I will pick an image of a dog and to try to see what will be the visualized outputs from some of the intermediate convolution of the trained VGG-6 model.
If we take a look at the different images from Convolution layers filters, it is pretty clear to see how different filters in different layers are highlighting or activating different parts of the image (see Fig.5).

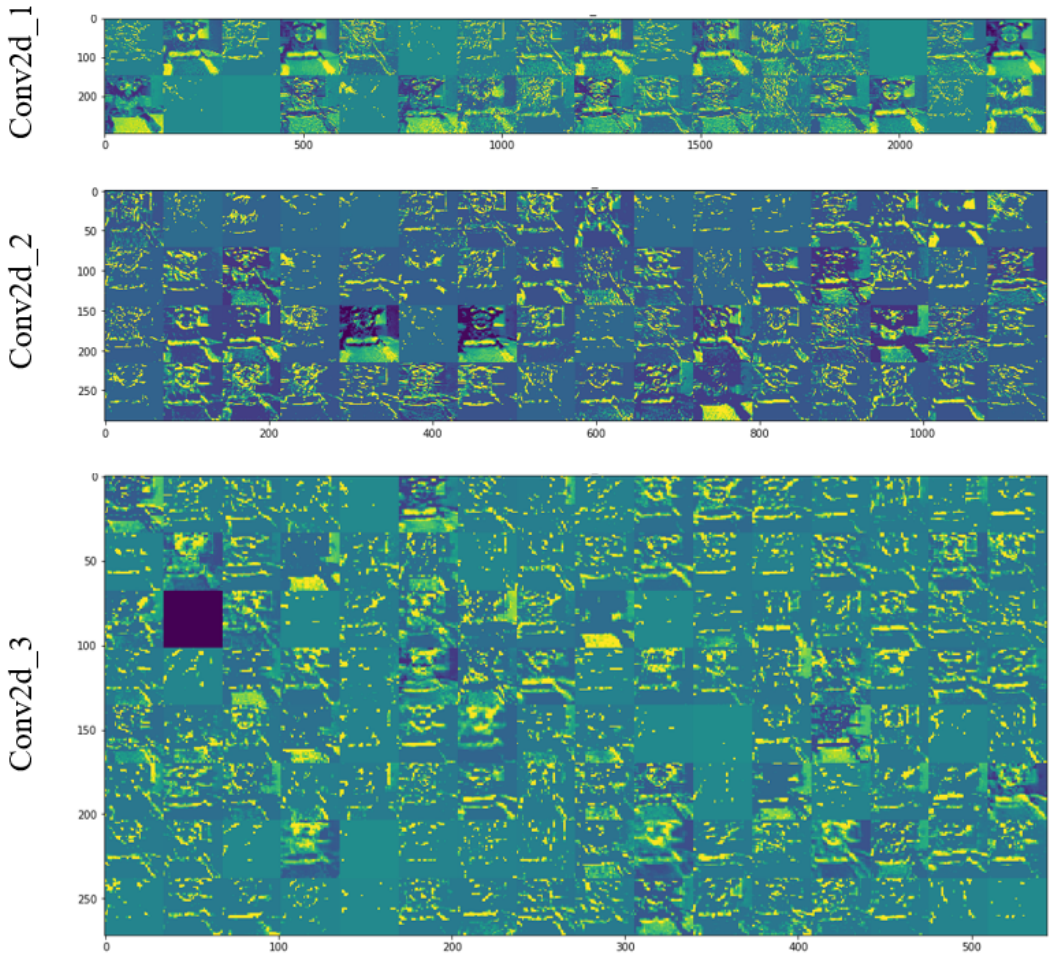

Figure 5: The filters from conv2d_1, conv2d_2 and conv2d_3 by running the trained VGG like model on one of the test images.

The first layer acts as a collection of various edge detectors. At that stage, the activations are still retaining most of the information present in the initial picture. As we go higher-up, the activations become increasingly abstract and less visually interpretable. They start encoding higher-level concepts such as "dog ear" or "dog eye". Higher-up presentations carry increasingly less information about the visual contents of the image, but adding more information related to the class of the image to the model. The sparsity of the activations increases with the depth of the layer: in the first layer, all filters are activated by the input image, but in the following layers more and more filters become blank, meaning that the pattern encoded by the filter is not found in the input image.

By visualizing the output from different convolution layers in this manner, the most crucial thing is that the layers deeper in the network visualize more training data specific features, while the earlier layers tend to visualize general patterns like edges, texture, background, etc. This knowledge is very important when we use Transfer Learning whereby training some part of a pre-trained network on a completely different dataset. The general idea is to freeze the weights of earlier layers because they will anyways learn the general features. This is to only train the weights of deeper layers which are actually recognizing our objects.

5.2 Visualizing Heatmaps of Class Activations

While predicting the class labels for images, sometimes the model will predict the wrong label for the class, i.e. the probability of the right label will not be maximum. In such cases, it will be helpful if we could visualize the parts of the image the convnet looking at to deduce the class labels. The

general category of such techniques is called Class Activation Map (CAM) visualization. CAM can produce heatmaps of class activations over input images [12]. A class activation heatmap is a 2D grid of scores associated with a particular output class. It is computed for every location of an input image, indicating how important each location is with respect to that output class. The Global Average Pooling layer (GAP) is preferred over the Global MaxPooling Layer (GMP) which is used in VGG-16 model because GAP layers help to identify the complete extent of the object as compared to GMP layer which only identifies the discriminative part. This is because in GAP we take an average across all the activation that helps to find all the discriminative regions while the GMP layer only considers the most discriminative one.

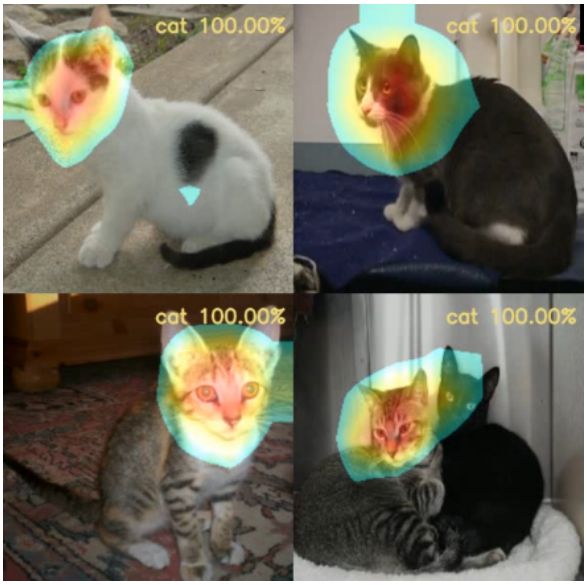

Figure 6: The demo of CAM visualization

In Fig.6, you can see how this technique works. Start from top two plots. At the early training phase, the model seems to randomly pick up interest locations, plotting heatmap points by chance. As the training moves forward, the model slowly converges on the cat face and paws with the accuracy of 99.99%. At the end, the heatmap finally captures the entirety of the cat face with the output of 100% accuracy. So basically, what the heatmap is trying to tell us is the important locations in the image for that particular layer to classify it as the target class, which is cat in this case. I have uploaded one small video onto YouTube*, which demonstrates how the CAM works from the beginning to the end.

6 Conclusion and Future Work

In this report, I discussed how to train a CNN model to classify Dogs vs. Cats images. I trained the models from scratch at first and then using Transfer Learning, I got over 95% accuracy. Meanwhile, two ways of the model outputs visualization have been demonstrated, which can help us gain more insight into how the model works.

One of the original tasks for this project is applying the CNN model to detect whether animals or pets are sick or not? In real life, it may be difficult to know if subtle changes in the animals indicate a health problem. Using the CNN model, I may be able to keep track of animals' behavior in order to prevent extreme illnesses. The discharge from eyes or nose may indicate a possible upper respiratory infection; or skin irritation or hair loss may be a sign of allergies, external parasites, or another skin condition. Because of the time limitations, I did not collect enough dataset for this kind of study. For the future works, I will keep collecting datasets for a month or a year in order to find the correlation between illnesses and animal behavior. Through the deep learning model, I will be able to predict the probability of a symptom for every animal picture or video. Future work might also include robotic systems that monitor the state of the household animals, adjust food distribution depending on image readings or alert when the animals from any kind of illness.

Moreover, different types of models can be employed to see, which one fits the needs the most. The project shouldn't be limited to VGG models.

Heatmap Demo on YouTube Links*

`https://youtu.be/tvI4lSGjIBM`

References

[1] McDonald, C. (2017) Demystifying AI, Machine Learning and Deep Learning.
https://mapr.com/blog/demystifying-ai-ml-dl/

[2] Gomez, A. Salazar, A.& Vargas, F. (2016) Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks.
https://arxiv.org/pdf/1603.06169.pdf

[3] Favorskaya, M. & Pakhirka, A. (2019) Animal species recognition in the wildlife based on muzzle and shape feature using joint CNN. *Procedia Computer Science* **159**:933–942.

[4] Zaitoon, A. (2018) Deep learning for Animal Identification.
https://github.com/A7med01/Deep-learning-for-Animal-Identification

[5] Mitrev, D. (2018) Automated Animal Identification Using Deep Learning Techniques.
https://medium.com/coinmonks/
automated-animal-identification-using-deep-learning-techniques-41039f2a994d

[6] Chen, G. Han, T. X. He, Z. Kays, R. & Forrester, T. (2014) Deep convolutional neural network-based species recognition for wild animal monitoring. *International Conference on Image Processing (ICIP) IEEE* **2014**:858-862.

[7] Yu, X. Wang, J. Kays, R. Jansen, P. A. Wang, T. & Huang, T. Huang (2013) Automated identification of animal species in camera trap images. *EURASIP Journal of Image and Video Processing* **2013**:52

[8] Dogs vs. Cats. https://www.kaggle.com/c/dogs-vs-cats/data

[9] Simonyan, k. & Zisserman, A. (2014) Very Deep Convolutional Networks for Large-Scale Image Recognition. https://arxiv.org/abs/1409.1556

[10] Tewari, S. (2019) CNN Architecture Series — VGG-16 with implementation
https://medium.com/datadriveninvestor/
cnn-architecture-series-vgg-16-with-implementation-part-i-bca79e7db415

[11]Paliwal, A. (2018) Understanding your Convolution network with Visualizations
https://towardsdatascience.com/
understanding-your-convolution-network-with-visualizations-a4883441533b

[12] Mishra, D. (2019) Demystifying Convolutional Neural Networks Using Class Activation Maps
https://towardsdatascience.com/
demystifying-convolutional-neural-networks-using-class-activation-maps-fe94eda4cef1