Deep Learning Image Recognition: Identifying Predominant Breeds within Mixed-Breed Dogs

Julia Goyco, Zoe Lambert, Jennifer Mince, Stephanie Schoch

December 12th, 2018

Abstract

This paper details our use of a convolutional neural network (CNN) model to predict the two predominant dog breeds from an image of a mixed-breed dog. Our first dataset consists of images of purebred dogs from Kaggle. Our second dataset consists of images of mixed dog breeds from 101 dogs. We selected a pre-trained CNN model, which was trained on ImageNet. Then we performed two steps of transfer learning. We re-trained the model with the first dataset and retrained again with the second dataset.

Introduction

The purpose of this paper and project was image recognition of dog breeds through deep learning methods. The guiding question that led to this research was: can we use a deep learning approach to determine the two predominant breeds in a mixed-breed dog?

The project began by looking at current dog breed image recognition projects for a starting point to work off of and change to fit the purpose of recognizing the two dog breeds that created the mixed-breed fed into the model.

Deep learning involves learning from data without a human explicitly programming the rules [12]. There are several types of deep learning networks that have been developed, each with different strengths and weaknesses in terms of applications. Two primary networks include convolutional neural networks and recurrent neural networks.

A convolutional neural network is a type of feedforward neural network with a convolutional layer at its core [12]. The convolutional layer has parameters in the form of learnable filters with small receptive fields extending through the full depth of the input volume, which are then connected to one or more fully connected layers [12]. Additionally, there is a fully convolutional network in which the fully connected layers are all replaced with convolutional layers [12]. In contrast, a recurrent neural network (RNN) can have activations that flow in a loop as connections form directed cycles [12]. RNNs are different from CNNs in that they can process arbitrary sequences of inputs by using internal memory [12]. Finally, a special type of RNN is an LSTM (long-short term memory) neural network which can remember a value for an arbitrary length of time. Convolutional neural networks have had remarkable success in the area of computer vision, outperforming other neural networks on similar image recognition and image classification tasks [3]. One specific type of image classification that remains a challenge, however, is fine-grained image classification, which is relevant to the challenge of dog breed identification. This can be a challenge as CNNs require a large amount of labeled data, and the number of labels needed for attributes in fine-grained image classification is expensive as subtle variances in local attributes must be used for classification [2].

The challenge of dog breed identification deals with fine-grained image classification, which aims to distinguish subordinate-level categories under some basic-level category [12]. This is a particularly challenging task for neural networks due to high intra-class and low interclass variance [12], in other words, there can be significant variations within individual classes and very slight variation between classes so classes can only be categorized based on subtle, local differences [11]. Further, the high intra-class variance can occur due to changes in the foreground or variation in pose, scale, or rotation, that make identifying discernable differences between classes difficult as the discriminative features are localized in object parts, such as in dog faces [11].

To solve this issue, a common approach to fine-grained image classification is to pipeline the process by first determining where the object is by determining what the foreground of the image is and then extracting discriminative features from the target object [11]. This pipeline helps to save processing by focusing on regions that have been identified in the proposal and selection process of candidate regions [8]. However, it is important to note that this process is imperfect and can lead to lost information during the pipelining process because the general process does not aggregate evidence from the different regions

so information is lost between candidate proposal and classification [8]. As well, in order to determine where the object is, most fine-grained image classification models rely on human-generated labels for a number of attributes in each image, as it is subtle differences in these local attributes that are responsible for classification of similar categories [2].

Current State of the Art

There have been several recent developments and projects on fine-grained image classification to address the challenges listed above. One prominent area of research has been in creating models that can perform the task of fine-grained image classification without explicit labels of fine-grained categories [2]. This is a particularly active area of research as labels beyond minimal class labels are very expensive in terms of time, particularly as datasets used for fine-grained image classification tend to train on images in the order of thousands [2].

In one example, rather than hiring people to complete the labeling process or crowdsourcing the labeling process, researchers used noisy data from publicly-available online sources to provide categorical data to aid in the classification process of publicly-available datasets [7]. Specifically, this data includes images obtained via a Google search for images of a given category that are then measured for cross-domain noise and cross-category noise, with cross-domain noise being the images that do not contain an image of the given category and cross-category noise being those with a wrong species label [2]. The researchers then applied a filtering method on the noise, and used the filtered results to a generic, domain-independent model that exceeded state of the art methods on 14,553 fine-grained categories, including 80.8% accuracy on the Stanford Dogs dataset [10].

In a different approach to the same issue of the expense of labeling, researchers wanted to explore the idea of zero-shot learning, which is learning that occurs on data for which there is no labeled data for some classes [2]. Specifically, they explored the use of a Structured Joint Embedding (SJE) framework for fine-grained image classification [2]. In this process, image embeddings (image features) and output embeddings (side information) are assigned values per a compatibility function that assigned high scores to matching embeddings and lower scores to mismatched [2]. They used the Animals with Attributes (AWA), Caltech-UCSD Birds (CUB), and Stanford Dogs (Dogs) datasets with the SJE and evaluated the use of five different types of output embeddings including those supervised embeddings (i.e. human annotation) and unsupervised embeddings (unlabeled text corpora, learned semantic hierarchies [2]. These were evaluated either singularly or as part of a combination [2]. Results showed improved state of the art performance on AWA (60.1%) CUB (29.9%) and Dog (35.1%) using unsupervised zero-shot learning with SJE [2]. The researchers concluded that unsupervised learned labels can improve fine-grained image classification performance, and combinations of output embeddings can further improve zero-shot performance as they can provide complementary information that can help in the classification task [2].

In a similar thread of research, researchers have begun to look at the use of attention in deep neural networks for fine-grained image classification with minimal labels, that is, labels at the class level only [12]. In one example of a recent research project dealing with attention for fine-grained image classification, researchers sought to address the challenge

of lost information in fine-grained image classification by training a recurrent neural network (RNN) end-to-end to aggregate information across observations [8]. Specifically, the researchers used a recurrent neural network (RNN) for fine-grained image classification on the Stanford Dogs dataset [8]. The Stanford Dogs dataset contains 8,580 images of 120 dog breeds, with the training set containing 100 images per class and the test images having 71 images per class) [8].

The researchers believed a RNN would help address lost information as well as the challenges of visual clutter, occlusion and variation of light and pose [8]. They chose to use minimal labels to address the challenge of discriminating difficult class boundaries, and addressed the challenge with an attention model in the hopes that it would learn to focus processing and discriminatory power on the parts of the image that are relevant for the task without requiring expensive hand-labeled bounding boxes [8]. In other words, they wanted the model to learn to focus processing power on the specific features of the objects that help to tell them apart [8]. They trained and tested the model using an 80/20 split and augmented the training set by reflecting the training images across the vertical axis [8]. Their results indicated increased performance using high, medium, and low resolution images with 3 glimpses at the images, specifically, their model reached 76.8% accuracy whereas the previous standard (GoogLeNet 224x224) has 75.5% accuracy on the same task.

In a different example of the application of attention to fine-grained image classification, in Xiao et al. (2015) researchers used a convolutional neural network (CNN) to apply visual attention to domain-specific fine-grained image classification by using three types of attention: bottom-up, object-level top-down, and part-level top-down, tested on two datasets: ILSVRC2012 dataset and CUB2002011 dataset [11]. These datasets have bird and dog categories, and thus deal with similar content to the Stanford Dogs dataset used by Sermanet et. al (2015). In detail, bottom-up proposes patches to attenuate on, object-level top-down selects patches of an object, and part-level top-down localizes features of the object [11]. They trained and tested on fine-grained image classification of dogs and birds in the datasets.

The use of both the bottom-up and top-down approaches helps to identify the where and the what of the image, respectively. First, the bottom-up approach helps to identify patches in an image that may contain parts of a specific object [11]. This leads to high recall but low precision, so the top-down approach supplements it and helps determine discriminative features that can then be used to classify an object [11], for example, a dogs tail or ears. In other words, the bottom-up and top-down help with object-level and part-level discrimination [11]. At the object-level, the goal is to remove noisy patches not relevant to the object, and at the part-level, the goal is to detect and cluster patches to determine local features that can be used for classification [11]. Further, with this approach, the researchers only used image-level labels because providing detailed labeling with bounded boxes is an expensive process in terms of time [11].

The architecture was a convolutional neural network with 5 convolutional layers and 3 fully connected layers, with the number of neurons on the output layer equal to the number of categories [11]. Since this model uses feature extraction, the first fully-connected layer of the CNN outputs features [11]. The researchers found increased performance on the CUB2002011 dataset under the weakest supervision setting (class labels only) [11]. They partially attribute this to the part-level attention that focuses on local patterns, as this

helps deal with shift/scale variances and helps achieve pose normalization for the object [11]. This points to a challenge of fine-grained image classification, the challenge of possibly small datasets. In applications where datasets are small, there is an increased need for pose normalization and accounting for variation between images, such as with backgrounds or age of live specimens. A potential way to compensate for this is through the use of transfer learning, in which a model that has been pre-trained on a large dataset is used for another application, as will be discussed next.

One of the prominent pre-trained model developments was that of the Inception architecture, introduced by Szegedy et al. in 2014 [3]. Prior to Inception, the primary architecture for fine-grained image classification was the VGG architecture which used a stack of simple convolution layers [3]. Convolution layers operate by learning filters for 3 dimensions using two spatial dimensions, specifically width and height [3]. Further, convolution kernels must simultaneously map cross-channel correlations and spatial correlations, which is one area in particular on which Inception improved. Inception introduced a new architecture by using stacked modules, rather than simple convolution layers, which allows the Inception architecture to learn richer representations with less parameters [3]. Specifically, these modules look at cross-channel correlations via a 1x1 convolution, then map input onto several smaller spaces and finally maps all correlations on the smaller 3D spaces via 3x3 or 5x5 convolutions [3]. Thus, the module architecture of Inception allows cross-channel correlation and spatial correlation to be decoupled so they do not have to be mapped jointly, which leads to increased performance over the VGG architecture [3].

However, in 2017 Francois Chollet of Google proposed a new architecture that would improve upon the Inception architecture and developed a model called Xception, or Extreme Inception [3]. Chollet proposed that a 1x1 convolution could be used to map cross-channel correlation, then the spatial-correlation of every output channel could be separately mapped, further decoupling spatial and cross-channel correlation [3]. This is similar to depthwise separable convolution, in which a spatial convolution is performed over each channel of an input, followed by a pointwise convolution, such as a 1x1 convolution, which projects output from the channels onto a new channel space [3]. This was introduced in 2014 and was included in the TensorFlow framework in 2016 [3]. The difference, however, between the depthwise separable convolution and the Extreme Inception proposed by Chollet, is that the depthwise separable convolution performs the channel-wise spatial convolution first, then the 1x1 convolution [3]. So, essentially, Chollet suggested replacing the modules of the Inception architecture with depth-wise separable convolutions by using the depthwise separable convolution implementation available in Tensorflow [3]. By using a convolutional neural network architecture based on depthwise-separable convolution layers, the mapping of cross-channel correlations and spatial correlations in feature maps could be entirely decoupled [3].

The Xception architecture uses 36 depth-wise separable convolution layers convolution layers structured as 14 modules for feature extraction, which forms the base of the network [3]. All layers except the first and last modules have linear residual connections, and the modules are followed by a logistic regression layer [3]. The architecture has similar parameter count as Inception V3, but does exhibits linearity in the linear stacks of layers, so it exhibits faster convergence and better final performance [3]. This shows that the architectures enhanced performance is a result of efficient use of parameters, rather than increased capacity [3]. To

demonstrate this, Chollet found small gains in classification performance on the ImageNet dataset over Inception V3 [3]. Due to the efficiency of Xception, we chose to use this as the basis for applying transfer learning to mixed-dog-breed classification [3].

Implementation

After looking at the work already established on dog breed image recognition, the group began examining and working with the Stanford Dog Breed Classification project [9]. The dataset consisted of 20,580 labeled images of dogs, each belonging to one of 120 dog breeds. As the team started to use the stanford dataset we noticed the labels seemed off. The labels for the dataset needed to be a certain format to predict the dog breeds accurately. The format required was a unique identifying number and a string of the dog breed name. Since there were so many images and over 100 dog breeds, we decided it would require too much time to change all of the labels appropriately.

The team decided to use a dataset from Kaggle. The dataset from Kaggle contains the same images as the Stanford dataset but it is split differently. It has a 50/50 split between test and train data. This dataset also had a labels.csv, which was very useful. This gave the team an example of how to create labels for the mixed breed images.

The mixed dog breed dataset was created by saving images from 101 dog breed. From there, it was imperative to find images of mixed-breed dogs. Importantly, the team noted that only choosing dogs with two distinct parent breeds would make it easiest for the model to learn and accurately identify. Another important thing to note is that the dog breeds had to be a mix of the dogs already included in the training data being used from the previously created Stanford data set. The team split up responsibilities to find dog breeds that fit this description and had images readily available for use. The site 101 Dog Breeds had a fairly good amount of mixed-breeds that came from two distinctly different breeds with features that related back to each of the parent breeds [5]. Each breed on 101 Dog Breeds includes details about the breed and a gallery of images with different dogs of various ages, sizes, and colors, per breed [1]. These images were saved in a similar way to the Stanford Dog Breed data set so they could be uploaded to the code and model easily. The specific breeds chosen to start with were: the Golden Saint (Golden Retriever and Saint Bernard), the Cheagle (Chihuahua and Beagle), the Peekapoo (Pekingese and Poodle), the Pitsky (Pitbull and Husky), the Siberian Retriever (Siberian Husky and Golden Retriever), the Puggle (Pug and Beagle), the Maltipom (Maltese and Pomeranian), the Schnoodle (Schnauzer and Poodle), the Chug (Pug and Chihuahua), the Meagle (Miniature Pinscher and Beagle), the Dorkie (Dachshund and Yorkshire Terrier), the Akita Chow (Akita and Chow Chow), the Bernedoodle (Bernese Mountain Dog and Poodle), the Labrahuahua (Labrador Retriever and Chihuahua), the Mal Shi (Maltese and Shih Tzu), the Basset Shepherd (Basset Hound and German Shepherd), the Beagle Bull (Beagle and Pitbull), the Border Beagle (Border Collie and Beagle), the Shepherd Pit (German Shepherd and Pitbull), the Chorkie (Chihuahua and Yorkshire Terrier), the Havapoo (Havanese and Poodle), and the Bull Pug (Pitbull and Pug).

The Kaggle data set contained train images along with a labels.csv. The labels.csv contained the dog breed labels for the train images. The labels.csv had contained 2 columns:

id and breed. The ID matched the name of the image and the breed was the pure breed dog that was shown in the image. The labels file and train images were passed into our Kaggle Dataset Preprocessing notebook file. The code then sent the images into a 4D tensor array along with the correct labels and saved as separate image and label files. These new files were then used in the model, split into 20% test and 80% train, and then set up to see if it could then determine the breed of the dog in a given image.

The mixed-breed data images were split 80-20 into mostly training so that the model would learn more from the images to better predict the ones it was tested on. The images were manually split into two folders: train and test. Then Excel label files were created for each folder with two columns: ID and Breed. The ID matched the name of the image and the breed was the mixed-breed that was shown in the images. The label files and images were passed into our Processing Mixed Breed Data notebook file. The code then sent the images into arrays along with the correct labels and saved as separate image and label files. These new files were then used in the model to see if it could then determine which two parent breeds made up the mixed-breed image.

The team chose a Convolutional Neural Network (CNN) for the preliminary model because research showed most projects of similar type chose this option. The option to change to a Recurrent Neural Network (RNN) was present in case the CNN could not fulfill what was needed from the model to correctly identify mixed-breeds. A RNN would have the same problem with the amount of time it would take to train. The team ultimately choose to use a transfer learning technique.

The team decided to implement transfer learning technique to reduce the amount of time to train a Convolutional Neural Network. Transfer learning is a technique where a pre-trained model is re-purposed for a different task. The team chose to use the Keras Xception model. This model is for image classification with weights trained on ImageNet. On ImageNet this model gets an accuracy of 94.5% [10].

On top of the pre-trained Xception model some addition layers were added to increase performance. The additional layers include a global average pooling layer, a fully connected layer and a dropout layer [6]. To fine-tune the model all convolutional Xception layers are frozen [6]. The model is compiled and trained on the kaggle dataset. Then the transfer learning technique is applied again to train the model on mixed dog breed dataset.

To predict parent dog breeds with the model, the bottleneck features from the Xception model and the best weight file are compiled and used to make a predicted vector for an image. The top two numbers of that vector are used to determine the dog breed by using the numbers and going to that index in the list of unique labels (contains: dog breeds) [9].

The team decided to get a model correctly predicting the pure breed dogs first before moving on to mixed breeds. To accomplish this the team re-trained the Xception model on the dataset from Kaggle that contained 120 different dog breeds. The transfer learning model was able to accurately predict purebred dog breeds. Then the model was tested with mixed dog breed images and was able to accurately predict one parent.

To predict both parents the team tried the same technique with the mixed dog breed dataset. The team tried re-training layers of the Xception model and training for more epochs because the mixed dog breed dataset has less data. When tested with the mixed breed images this model was unable to correctly predict parent breeds.

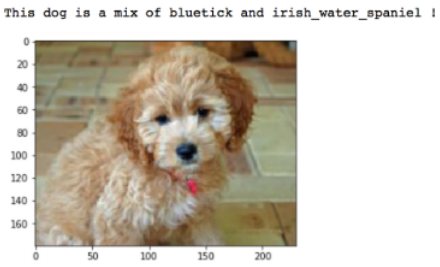

Figure 1: Golden Retriever and Poodle mix- not Bluetick and Irish Water Spaniel

Results

The results with the CNN from scratch were about 3% accuracy. This model was not able to correctly predict pure dog breeds. The accuracy increased to 83% by using transfer learning. The accuracy was increased further by increasing epochs to 4.This resulted in a 7% increase. The accuracy at predicting purebred dogs is 90%. The model was able to correctly predict pure breeds. When this model was tested on mixed dog breeds it was able to correctly predict at least one parent correctly. So the team added training with the mixed breed dog dataset to see if it would help in predicting the parent breeds. Half the convolutional layers of the Xception model we trained with the mixed breed dataset. This was trained with 1 epoch and a batch size of 32. After training with the mixed breed dataset the model was not able to predict either parent.

Then the team tried training only the top layers of the Xception model with the mixed breeds with the same number of epochs and batch size. This change caused the model to sometimes predict one parent breed correctly but was mostly incorrect. It was able to predict that a Gerberian Shepsky is part German Shepherd.

In the end the team was only able to accurately predict one parent. This model only trains on the purebred dataset and is tested on mixed breed images. The model trained with epochs to 4 and batch size of 32 and had 90% accuracy at predicting purebred dogs. When testing with mixed dog images the model is able to predict one parent. For example the model predicts this Gerberian Shepsky is an mix of a German Shepherd and a Cardigan.

This dog does share a similar coloring and ear shape but other than that these dogs look dissimilar.

Another example is of a Pomchi (Pomeranian and Chihuahua). The model gets one parent correct but this is a tough combination to predict because a Paperanian (Papillon and Pomeranian) looks very similar to a Pomchi.

Future Work

Since the training with the mixed breed dataset did not seem to help, the team could try training more layers of the Xception model with the Kaggle dataset since that seemed to work best. This means freezing less layers. The Xception model has 232 convolutional layers.

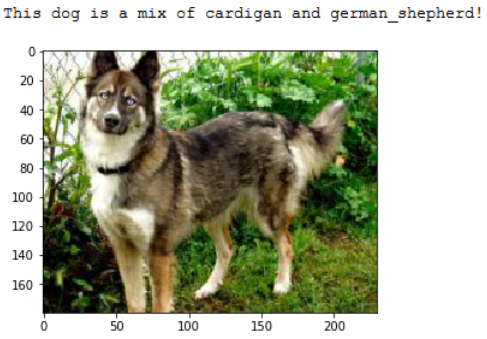

Figure 2: Gerberian Shepsky

Figure 3: Cardigan

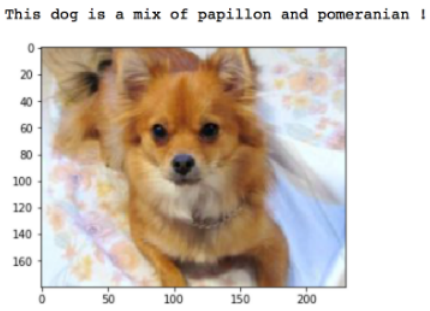

Figure 4: Pomchi

Figure 5: Paperanian

Training half the layers with the Kaggle dataset would be enough. Adding more images into the training dataset would also increase the performance. Another thing to try is augmenting the images. This means shifting, rotating and flipping the images. Data augmentation is used to reduce the amount of irrelevant features like the direction a dog is facing. Other pretrained models could be used to see if those models had more success with mixed breed images. Another possible improvement would be to use labels in the mixed dog breed dataset that correspond to the two parent dog breeds, instead of the combined name.

Conclusion

This implementation was successful in predicting one parent dog breed. The model was always able to predict at least one parent. Training on the pure breed dog dataset from Kaggle was also successful. Our initial selection of the Stanford dog dataset was unsuccessful. The team did not use it because of the labels it used. The implementation was unable to predict the the second parent. Often, the model would predict the second parent to be a similar breed to the actual parent. Some mixed dog breeds would be difficult to discern, even for a person.

Bibliography

[1] 101 Dog Breeds, *101 Dog Breeds* (2018). Web. 20 Nov. 2018.

[2] Zeynep Akata, Scott Reed, Daniel Walter, LEe Honglak, and Bernt Schiele, *Evaluation of output embeddings for fine-grained image classification.* CVPR2015. 2927-2936.

[3] Francois Chollet, *Xception: Deep learning with depthwise separable convolutions.* 2017.

[4] Depends on the Definition, *How To Build A Smart Product: Transfer Learning For Dog Breed Identification With Keras* (2018). Web. 29 Nov. 2018.

[5] Kaggle Inc, *Dog Breed Identification* (Mar. 2018). Web. 13 Nov. 2018.

[6] Keras Documentation, *Xception.* Web. 24 Nov. 2018.

[7] Jonathan Krause, Benjamin Sapp, Andrew Howard, Zhou Howard, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei, *The unreasonable effectiveness of noisy data for fine-grained recognition* (2016).

[8] Pierre Sermanet, Andrea Frome, and Esteban Real, *Attention for fine-grained categorization* (2015). ICLR 2015.

[9] srikantrao, *Dog-Breed-Classifier/README.md* (Feb. 2018). Web. 29 Nov. 2018.

[10] Stanford University, Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei (2011). Web. 13 Nov. 2018.

[11] Tianjun Xiao, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, and Zheng Zhang, *The application of two-level attention models in deep convolutional neural network for fine-grained image classification.* CVPR2015. 842-850.

[12] Bo Zhao, Jiashi Feng, Xiao Wu, and Shuicheng Yan, *A survey on deep learning-based fine-grained object classification and semantic segmentation. International Journal of Automation and Computing.* (2017). no. 2, 119-135. DOI 10.1007/s11633-017-1053-3.