Dog Breed Classification using Convolutional Neural Networks

Fall 2018 - Introduction to Visual Information Processing(CS-555)

By

Shriprasad Bhamare

Supervisor

Prof. Lijun Yin

Computer Science Department

Binghamton University

2018-19

Abstract

In this project, goal is to build a convolutional neural network (CNN) that can classify the breed of dog from any user-supplied image. If the image is of a human and not a dog, the algorithm will provide an estimate of the dog breed that is most resembling. I aim to use a convolutional neural network framework to train and categorize dog breeds. I approach this first using CNNs from scratch and CNNs using Inception,VGG-16, ResNet-50 models to refine the results of CNNs from scratch. Dog breed categorization is a very specific application of convolutional neural networks.This study investigates image classification and predictive models to accurately predict a dog's breed. Image classification is applicable to various fields such as bio-diversity studies, facilitating auto-tagging, and image search. Image recognition has a vast number of applicable industries ranging from national security to marketing. From a marketing perspective, it creates a vast new knowledge base about consumers' identities, brand preferences, shopping habits which marketers could use to serve their customers better. In our study, I perform image classification using three machine learning models on 133 different dog breeds and compare the accuracy of each model to predict any breed within the 133 given breeds. By using TensorFlow with Inception model in Python, I found that a Convolutional Neural Network (CNN) developed by Google researchers performed the best result with a predictive accuracy around 90%. I tried other popular machine learning algorithms and compared those to the Inception, VGG-16, ResNet-50 model.Based on the results,I believe that the inception model performs so well and compared to the other techniques I investigated in this work.

Introduction

Convolutional neural networks (CNN) have been used to great effect in applications such as object classification, scene recognition, and other applications. In many situations, we can imagine the features (both low-level and higher-level) that are learned by the CNNs in the process of training. However, when the objects the CNNs are trying to categorize share many similar features, such as the breeds of dogs, it becomes hard to imagine the specific features that CNNs must learn in order to categorize these dogs correctly. This is especially true if we take a look at sets of images such as Fig below, where the 3 dogs share almost all the same visible features, but belong to different classes. It is therefore interesting to see how well CNNs can perform on only dog breeds, compared to labels from all classes of objects in the regular ImageNet.

Convolutional Neural Networks (CNNs) are currently the state of the art in most computer vision tasks (majority of papers in CVPR). They were specifically designed to leverage certain properties about image data – mainly translation invariance, hierarchical composition, and autocorrelation among neighboring pixels – which allows them to efficiently and accurately disentangle factors of variation to solve the presented task. As long as the compute and memory requirements are satisfied within the problem scope (and obviously if one has access to either a pre-trained network or the available data to train or fine-tune a network), they are currently the sensible tool to use for extracting useful knowledge from visual data (2016 Goodfellow, 2016 Karpathy).

CNNs are used extensibility in industry, ranging from self-driving cars (many sources, will site in final paper) to cancer detection (2018 Serj, among many other sources). There has also been a lot of recent research in the area of real-time scene understanding. Low latency object detection architectures such as Yolo / SSD / Faster-R-CNN, and object instance segmentation architectures such as Mask R-CNN, are commonly used to make robots contextually aware of

their environments (2018 He, 2017 Huang). So long are the days when humans tediously hand crafted rule based programs for vision modules.

All of the models are based on Convolutional Neural network (CNN) which is an essentially mathematical model to solve optimization problems. Convolutional Neural Network has successfully been applied to analyzing sequential data such as image. It is made of neurons, the basic computation unit of neural networks, and usually results in lower RMSE for image classification problems.

In Dog Breed Classification Using Part Localization (Liu, 2012) Going Deeper with Convolutions (Christian, 2014), the performance of fine-grained classification can be improved by using part localization since dog breeds are similar in common parts but different in shape and appearance.In Dog Breed Identification (LaRow, 2016), the team picked the best model after comparing the accuracy of different machine learning models.

In Transfer Learning for Image Classification of Various Dog Breeds (Devikar, 2016), Image Classification in CNN has proven to be highly efficient, but it requires a large training data set and substantial time for training to achieve higher accuracy.In TensorFlow, A System for Large-scale Machine Learning (Abadi, 2016), using larger internal cluster with GPU can lead to fewer steps and high accuracy of the Inception model.



Husky Alaskan Malamute Eskimo

Dataset Description

- Dog breed images:There are 133 total dog categories and 8351 dog images having 6680 training dog images. 835, validation dog images, 836 test dog images. https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip
- Human Images:The data set contains more than 13,000 images of faces collected from the web.

  http://vis-www.cs.umass.edu/lfw/lfw.tgz

Technical Approach and Challenges

- Image data preprocessing was a real challenge due to the RAM size of the laptop I use. As, preprocessing so many images may give memory error. In order to tackle this problem, I have used AWS EC2 p1.xlarge instance
- The code is written in Python 3 and Keras with Tensorflow backend in Jupyter Notebook. Tensorflow is used as the back-end library in developing our model since it's widely used

for machine learning applications such as neural networks. It is designed for handling data that is kept in the tensor format. The color code in the RGB format for each pixel will be stored and used to create a depth in the data frame, which could be treated as a tensor.

- Comparison of accuracy is done based on results from CNNs from scratch, CNNs using VGG-16 model, CNNs using Inception model, CNNs using Resnet-50 model
- OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in images. OpenCV provides many pre-trained face detectors, stored as XML files
- Pre-trained VGG-16, Inception, and ResNet-50 model to detect dogs in images along with weights that have been trained on ImageNet, a very large, very popular dataset used for image classification and other vision tasks. ImageNet contains over 10 million URLs, each linking to an image containing an object from one of 1000 categories. Given an image, this pre-trained models returns a prediction (derived from the available categories in ImageNet) for the object that is contained in the image.

Method

This project is implemented in the following steps:

- Human and Dog face detector
- CNN from scratch
- CNN using models to improve the accuracy: VGG16, Resnet50,and Inception
- Predict Dog breed

Human face detection:

- Haar feature-based cascade classifiers
- OpenCV provides many pre-trained face detectors, stored as XML files.

- True if a human face is detected in an image and False otherwise

Dog face detection:

- pre-trained ResNet-50 model to detect dogs in images.
- weights trained on ImageNet
- Data Preprocessing for dog face detection
    - Keras CNNs require a 4D array
    - load the image and resize it to a square image that is 224×224 pixels
    - Image to an array, which is then resized to a 4D tensor.

Data Preprocessing for CNN:

- Rescale the images by dividing every pixel in every image by 255.

CNN from Scratch:

- Added 3x Convolutional Layers
- Added Max Pooling Layer after each Convolutional Layer
- Added Drop Out Layer to avoid overfitting
- Added Flatten Layer
- Compile with Optimiser 'rmsprop', ‘Adam’ , and ‘sgd’
- Trained with:
    - Epochs set to 5 as it was taking so long
    - model checkpointing to save the model that attains the best validation loss.
- Added Global Average Pooling layer for dimensionality reduction

CNN using model VGG16:

The model uses the the pre-trained VGG-16 model as a fixed feature extractor, where the last convolutional output of VGG-16 is fed as input to our model. We only add a global average pooling layer and a fully connected layer, where the latter contains one node for each dog category and is equipped with a softmax.

- Compile with Optimiser 'rmsprop', ‘Adam’ , and ‘sgd’
- Trained with:
    - Epochs and batch size set to 20.
    - model checkpointing to save the model that attains the best validation loss.

CNN using model Inception:

The model uses the the pre-trained Inception model as a fixed feature extractor, where the last convolutional output of Inception is fed as input to our model.

- Compile with Optimiser 'rmsprop', ‘Adam’ , and ‘sgd’
- Trained with:
    - Epochs and batch size set to 20.
    - model checkpointing to save the model that attains the best validation loss.

CNN using model Resnet 50:

The model uses the the pre-trained Resnet 50 model as a fixed feature extractor, where the last convolutional output of Resnet 50 is fed as input to our model. Features are feed into a Global Average Pooling (GAP) Layer to flatten the 3D input into a Vector.The Vector is feed into a Fully Connected Layer to output predictions for each of the 133 dog classifications.

- Compile with Optimiser rmsprop
- Trained with:

- Epochs and batch size set to 20
- model checkpointing to save the model that attains the best validation loss.

Results

Analysis of test accuracy of Convolutional Neural Network

Optimizer CNN	SGD	RMSPROP	ADAM
CNN from scratch	---	**~2%**	---
CNN using VGG16 model	50.48%	46.77%	**51.67%**
CNN using Inception model	**82.90%**	79.42%	78.58%
CNN using Resnet 50 model	---	**79%**	---

Lowest Accuracy : CNN from scratch with RMSPROP optimizer

Highest Accuracy: 82.90% for CNN using Inception model with sgd optimizer

```
: make_prediction('myImages/Golden_retriever_05258.jpg')
```

```
Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!
You look like a Golden retriever.
```

Golden Retriever

```
make_prediction('myImages/Bulldog_02845.jpg')
```

```
Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!
You look like a Bulldog.
```

Bull Dog

```
make_prediction('myImages/alaskan.jpeg')
```

```
Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Arf-arf!! Ruff-ruff!! Bow-wow!! Woof woof!!
You look like a Alaskan malamute.
```

Alaskan Malmute

Human face resemblance to dog breed

```
make_prediction('myImages/Screenshot from 2018-11-
```

```
Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Hello human!..Hello from the other side,I must hav
If you were a dog, you'd be a Chinese crested.
```

```
make_prediction('myImages/12748073_1100499126647020_3213429309359239526_o.jpg')
```

```
Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Hello human!..Hello from the other side,I must have called a thousand times..To tell you
If you were a dog, you'd be a American eskimo dog.
```

Loading image...
Extracting bottleneck features...
Feeding bottlenneck features into top model...
Predicting breed...
Could not detect dogs or humans in image.

Conclusion and Discussion

I successfully implemented Dogs breed classification using Convolutional Neural Network. For the CNN using inception model with sgd optimizer ,I got highest 82. 90% test Accuracy.
My Dog Breed classifier successfully predicts dog breed of input image and human resemblance to a dog breed if the image is of human face, provided the image is clear and face focused. If the image is not of human it states that no face is detected.It doesn't work well with complex images where dog face is hard to detect e.g. In the picture having elephant ,human, and a dog it won't be able to detect dog's face.So, in future scope of my project, I would generalize the classification of breedsl for variety of image types and increase the test accuracy. In this project I learnt about data preprocessing with image data , CNNs and different model to use along with CNN to improve the test accuracy of prediction.

Acknowledgements

I thank Dr. Lijun Yin for constant guidance on this project.

References

- Hsu, David, et. al,"Using Convolutional Neural Networks to Classify Dog Breeds"
- Wenting Shi,et. al,"Dog Breed Identification"
- Serge, et. al, "A Deep Convolutional Neural Network for Lung Cancer Diagnostic"
- Liu et al. "Dog Breed Classification Using Part Localization" Computer Vision – ECCV 2012 Lecture Notes in Computer Science - 2012
- Raduly, Zalan, et al. "Dog Breed Identification Using Deep Learning." *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)*, 2018, doi:10.1109/sisy.2018.8524715.