Proposal

August 20, 2020

1 Machine Learning Engineer Nanodegree

1.1 Capstone Proposal

Artem Shramko August 19th, 2020

1.2 Proposal

Convolutional Neural Networks (CNN) project: Dog Breed Classifier

1.2.1 Domain Background

This project is devoted to building a dog breed classification app based on the Convolutional Neural Networks (CNN). The core purpose of the project is to explore the procedure of developing an end to end Machine Learning pipeline, which takes user input in form of a graphical image and returns a prediction of the dog breed.

For the intertainment purposes, the model is also intended to be able to accept pictures of humans as an input and return the name of the dog breed the submited person on a picture resembles the most.

The project comprises of several problems that are expected to be solved via application of Machine Learning techniques. Amonth them are: object recogniton (identify human and/or dog in a picture), ETL pipeline (preprocessing raw images as a model input), object classification with high interclass homogeneity (dataset comprises of 133 different dog breeds). s

1.2.2 Problem Statement

Given an image, identify whether a dog or a human is present on it. Based on the input, classify the dog's breed in the first case, or predict, which dog the person on the photo resembles in the letter. The predicted breed must be within 133 classes present in the dataset. If neither a dog nor human is present on the submitted picture, return an error. The algorithm must be able to accept images of different formats, sizes and resolutions and further transform them into the required by the model format.

1.2.3 Datasets and Inputs

The data used in this project has been provided by Udacity. It consists of two datasets: dog images and human images.

Human images dataset is not divided into train and test data. Images are sorted into folders with several pictures of a particular person in each folder. Overall there are 13233 human pictures in the dataset. Pictures are primarily face shots with face almost always being in focus of the picture. Shape and size differs.

Dog images are split into train, validation and test data. Dataset consists of 133 classes (dog breeds) with total of 8351 dog images. Pictures are of different sizes, shapes and resolutions. Dataset can be described as one containing high intraclass (f.e. golden, brown and balck labradors are considered to be the same Labrador retriever breed) and interclass (different subfamilies of spaniels or retrievers are represented as different classes despite their similarity) homogeneity. Thus the challange for the model is to learn how to differentiate dog breeds based on particular dog features, such as face shape, fur etc., while ignoring such characteristics as f.e. color or height.

1.2.4 Solution Statement

The solution to the described problem is presented as an application that accepts image as an input and returns predicted dog breed (if applicable) as an output.

The proposed solution consists of the following steps:

1. Identify whether dog or human is present in the picture.
2. Predict resembling dog breed with CNN model.
3. Return predicted dog breed with a sample picture.

In the first step two separate models are applied. Both models return binary output: True if dog (human) is present in the picture, False otherwise. If both models return False, further steps are not performed and exception is raised. Otherwise, predicted class (dog or human) is passed further.

At step two the main model is applied. A Convolutional Neutal Network is trained solely on dog data and returns one of 133 dog breeds as an output. However, it can also accept human image as an input. In this case the neural network returns the dog breed that human facial features resemble the most.

Third step describes UX. In this particualar problem setting, a simple printed image with predicted class is foreseen as an output.

The described solution can further be embedded in a web and/or mobile application, which is, hovewer, out of scope of this project.

1.2.5 Benchmark Model

The solution to the stated problem is developed according to the guidelines, provided by Udacity. The expected solution does not foresee comparison of the model performance against an explicit benchmark. However, throughout the completion of solution steps several approaches are compared.

For the first step (human/dog identification) a pre-trained OpenCV model is applied for human recognition. For the dog recognition a pretrained pytorch model VGG16 is suggested as a benchmark. Further models, such as SqueezeNet and ResNet151 are used. Models are compared based on classification accuracy as well as precision and recall. Models are tested on both human dataset (minimize False Positives) and dog dataset (minimize False Negatives).

For the 2nd step - dog breed cassification - a built from scratch CNN model is compared against a transfer learning based model. The selected metric for multiclass classification problem is a Cross-Etnropy Loss function.

1.2.6 Evaluation Metrics

As described in the previous section, following evaluation metrics are applied:

- Accuracy: `(TP + TN)/(TP + FP + TN + FN)`

  Accuracy defines the fraction of predictions that the model has classified correctly.

- Cross-Entropy Loss: `-1/N * SUM( y * log(y_hat) )`

The cross entropy metric comes from the Information Theory and calculates the dirrence between the true and estimated distribution. In out multiclass discrete problem `y` and `y_hat` represent the true and predicted classes respectively. This is a classic metric for multiclass classification problems due to its ability to accept softmax function outputs as an input.

For example, our true values vector is `y=[0, 1, 0, 0]` and predicted vector `y_hat = [0.2, 0.9, 0.1, 0.3]` is. Then the Cross-Entropy Loss can be calculated as `L = -(0*log(0.2) + 1*log(0.9) + ...)`. It can be seen, that when only 1 true class is present, the Cross-Entropy Loss will only take into account how close the predicted value to True class is, since other `0` classes will be ignored.

1.2.7 Project Design

This project is implemented in form of a Jupyter Notebook, which contains detailled analysis for realization of each of the steps described in Solution Statement section, as well as the source code of the final solution. The final solution is an app that accepts a path to where the image is stored and returns a predicted dog breed to a user. It is executed interactively in the same Jupyter Notebook.

The introductory section of the notebook presents the datasets and defines data loaders. For reading and visualizing image data, such packages as matplotlib, OpenCV and PIL are used. It presents data samples as well as provides summary statistics on the datasets.

The first section presents implementiation of the dog and human recognizer models. The letter is performed using the pretrained Haar-cascades frontal face detection model from OpenCV package. The performance of the model is then tested both on the human and dog datasets.

For the dog detection algorithm, a pretrained VGG16 implementation in pytorch is used. It is a general object recognition model, trained on the ImageNet dataset with 1.2M samples and 1000 classes, around 100 of which resemble dog breeds. Thus if predicted object class falls into one of the breed categories, picture is classified as containing dog image. The VGG16 model is further comaperd against SqueezeNet and ResNet152. Model performance is assessef based on the accuracy metric.

The second and main section is devoted to building a dog breed classification model. First, a custom CNN is build from scratch via pytorch library to serve as a performance baseline. The final architecture of the model was defined during an iterative procedure of performance optimisation. It consists on 5 Convolutional layers with Max Pooling and 3 Fully Connected layers. Such techniques as Dropout and Batch Normalization have been applied to avoid overfitting. The train data has been augmented using several techniques to improve dataset diversity.

The main dog breed classification model was built by applying transfer learning technique. A pretrained ResNet18 with fixed parameters has been selected as core. An additional Linear layer with 133 outut features has been added to serve as a dog breed classifier.

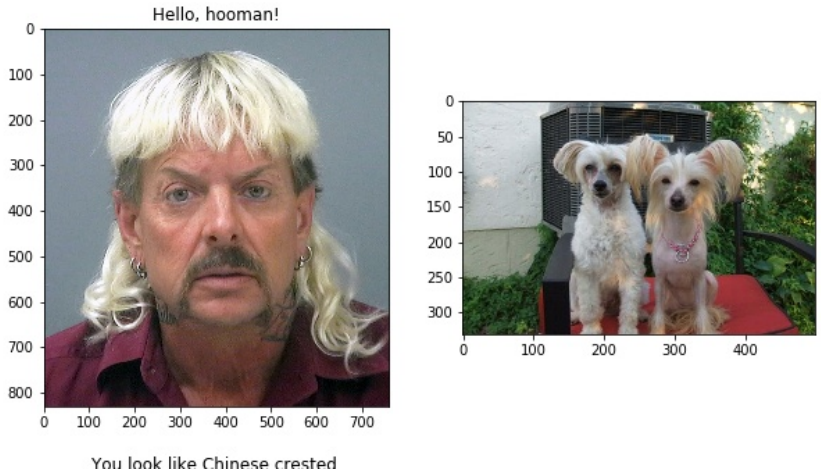
Basically there are two options for transfer learning: fine-tune all layers of existing pretrained model given new data, or "freeze" params and add one Linear layer to predict classes in your data. Since we have a relatively small dataset, and the original ImageNet dataset contains a lot of different dog breeds already, picking the first technique could lead to overfitting.

In terms of the model selection, the ResNet18 was selected due to its lower training time compared to more advanced models like for example ResNeXt101.

The model performance for this multiclass classification problem was assessed with Cross-Entropy Loss metric as described in the previous section.

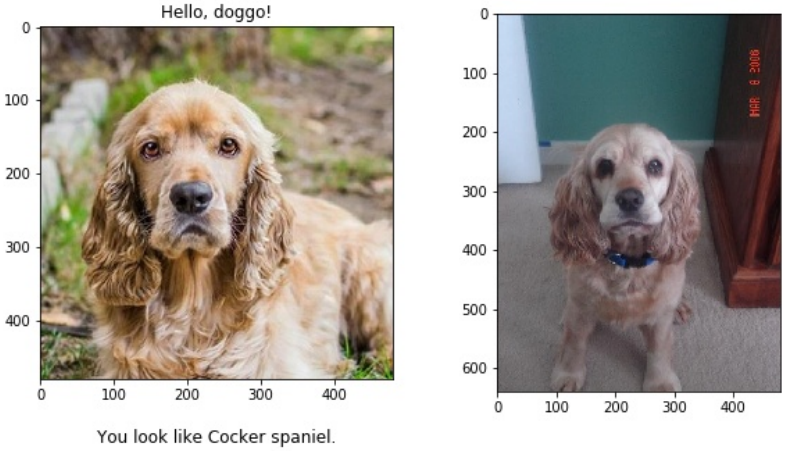
The final section implements the UX part of the application. It accepts path to the image and returns output of a form:

- Human picture input



human input example

- Dog picture input



dog input example

If the input image is not classified as a dog or human, exception is raised notifying user that an invalid image has been submitted.

So far the model, trained on 10 epochs due to computational power constraints, gives 80% accuracy on the test set. It can be further improved by varying core model for transfer learning, increasing train set diversity as well as number of epochs. In case of significant improvement of model performance, further serialization as an endpoint on AWS or embedding in a mobile app will be considered.

[ ]: