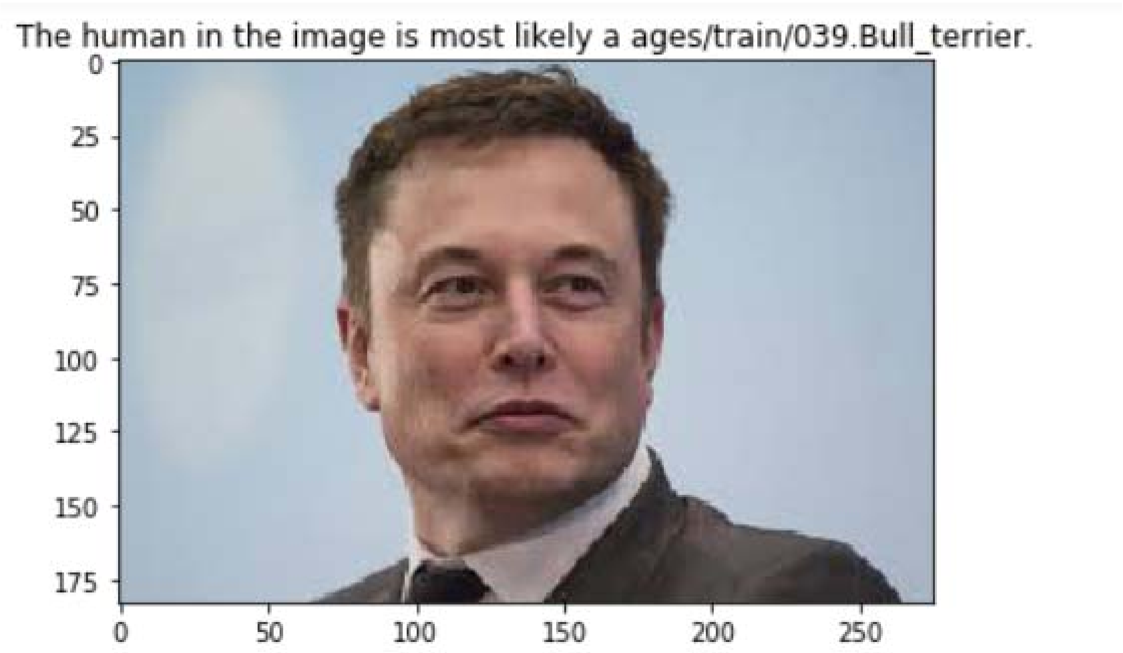
Only you can see this message

This story's distribution setting is on. Learn more

Dog Breed Classification Project

Lijuan Zhang
Feb 23 · 6 min read



```
The human in the image is most likely a ages/train/039.Bull_terrier.
```

This blog is a summary of the dog breed classification project. The goal is to classify images of dogs according to their breed and to develop an algorithm that could be used as part of a mobile or web app.

· · ·

Strategy

The problem we want to solve in the project is a multi-class classification.

In multi-class classification, the output layer of the neural network has the same as the number of classes, that is 133 dog breeds in our case.

Each output node belongs to a dog breed and outputs a score for that class. A Softmax layer converts the scores from the output into probability values.

We use 'categorical_crossentropy' loss for multi-class classification. and 'rmsprop' for optimizer. Keras library is used to build a CNN model for dog breed identification.

The dataset is splitted into training, validation and testing datasets. We use the training data to train our model and validate the performance using the validation dataset according to metrics, which is discussed in the next section. We then select the best performance model and save it for testing using the test dataset.

There are 6680 images for training the model, 835 for validation and 836 for testing.

Since we have a small dataset for training, we use the features from pretrained models to build our CNN. Transfer learning enable us to transfer knowledge from previous work to our classification.

If the model works well on the testing dataset, we deploy it in our

. . .

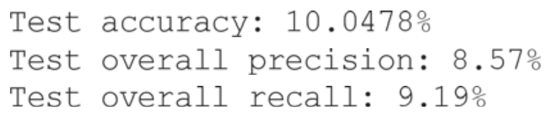
Processes

There are following steps in the project

- Step 0: Import Datasets
- Step 1: Detect Humans
- Step 2: Detect Dogs
- Step 3: Create a CNN to Classify Dog Breeds (from Scratch)
- Step 4: Use a CNN to Classify Dog Breeds (using Transfer Learning)
- Step 5: Create a CNN to Classify Dog Breeds (using Transfer Learning)
- Step 6: Write your Algorithm
- Step 7: Test Your Algorithm

As we see from the above steps, the algorithm has the ability to detect human and dogs. We can create a CNN from scratch or using transfer learning to do the job. We can use the features from the existing model to create a model and train our model.

The CNN model i trained from scratch reaches an accuracy of 10%, and precision and recall values are very low too:



```
Test accuracy: 10.0478%
Test overall precision: 8.57%
Test overall recall: 9.19%
```

Testings

An algorithm using the trained model is then developed to detect human and dog breeds. The algorithm should tell us:

- if a **dog** is detected in the image, return the predicted breed.
- if a **human** is detected in the image, return the resembling dog breed.
- if **neither** is detected in the image, provide output that indicates an error.

Following are some testing results:

Results and Discussions

Transfer Learning with CNN is very effective and efficient for image classification problems. The model could reach more than 83.78.% accuracy, as well as high precision and recall, in the prediction of dogs 83.78.% are correct, and 79.75% images with dogs are correctly predicted.

The final model is much better than the previous ones. The performance is good with limited training samples.

However, if we want to further increase the performance of the model, we need to train the model with more images from each dog breeds, especially when compare to the pre-trained models, such as VGG16 and InceptionV3.

Data balance is important too, as to be seen in the precision and recall values for each breed in the notebook, some classes have higher values than others, this could be improved with more images for the poor performance calsses.

In the project i expected higher accuracy than 81,46%. One reason is that i was working on the workspace from Udacity, i planned to use the Xception bottleneck features, however the file is too big i could not upload it to the workspace. after submitting the project i would like to take more time to try it out with my own environment.

I would use grid search method to find the optimal hyperparameters, such as drop rate, learning rate.

. . .

Reference:

1. https://keras.io/metrics/

2. https://medium.com/thalus-ai/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b

Dog Breed Classifier Deep Learning Transfer Learning