

Machine Learning Engineer Nanodegree

Capstone Project

Gender Classification

I. Definition

Project Overview

Gender classification is to determine a person's gender, e.g., male or female, based on his or her biometric cues. Usually facial images are used to extract features and then a classifier is applied to the extracted features to learn a gender recognizer. It is an active research topic in Computer Vision and Biometrics fields. The gender classification result is often a binary value, e.g., 1 or 0, representing either male or female. Gender recognition is essentially a two-class classification problem. Although other biometric traits could also be used for gender classification, such as gait, face-based approaches are still the most popular for gender discrimination [[ref](#)].

Problem Statement

The goal is to create a gender classification based on person face image so it's a binary classification problem; the tasks involved are the following:

1. Download and preprocess the [Adience Benchmark](#) dataset
2. Explore the dataset distribution and make sure the data is balanced
3. Use a pre-trained CNN model to extract features from the images
4. Build a neural network to classify gender given image features
5. Train the network and try different network architecture and configuration
6. Evaluate each network and choose the best one

Evaluation Metrics

I'm working with the [accuracy metric provided by keras](#) for binary classification problems, it's a good metric as I'm ensuring my data is balanced.

II. Analysis

Data Exploration

The dataset used for training and testing for this project is the [Adience Benchmark - collection of unfiltered face images](#). It contains total 26,580 images of 2,284 unique subjects that are collected from Flickr. There are 2 possible gender labels: M, F and 8 possible age ranges: 0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60+. Each image is labelled with the person's gender and age-range (out of 8 possible ranges mentioned above).

I'm only interested in gender labels, there are 9372 images with label f and 8120 with label m.

The images are Face images, cropped and aligned, and are of different sizes so I handle size when loading images to models.

I'm only going to work on a subset of this data, since it's a binary classification problem and I'm using a pre-trained model, I will use 6000 images for training, 2000 for validation and 2000 for testing, where I will sample each subset such that half of it with label f and the other half with label m.

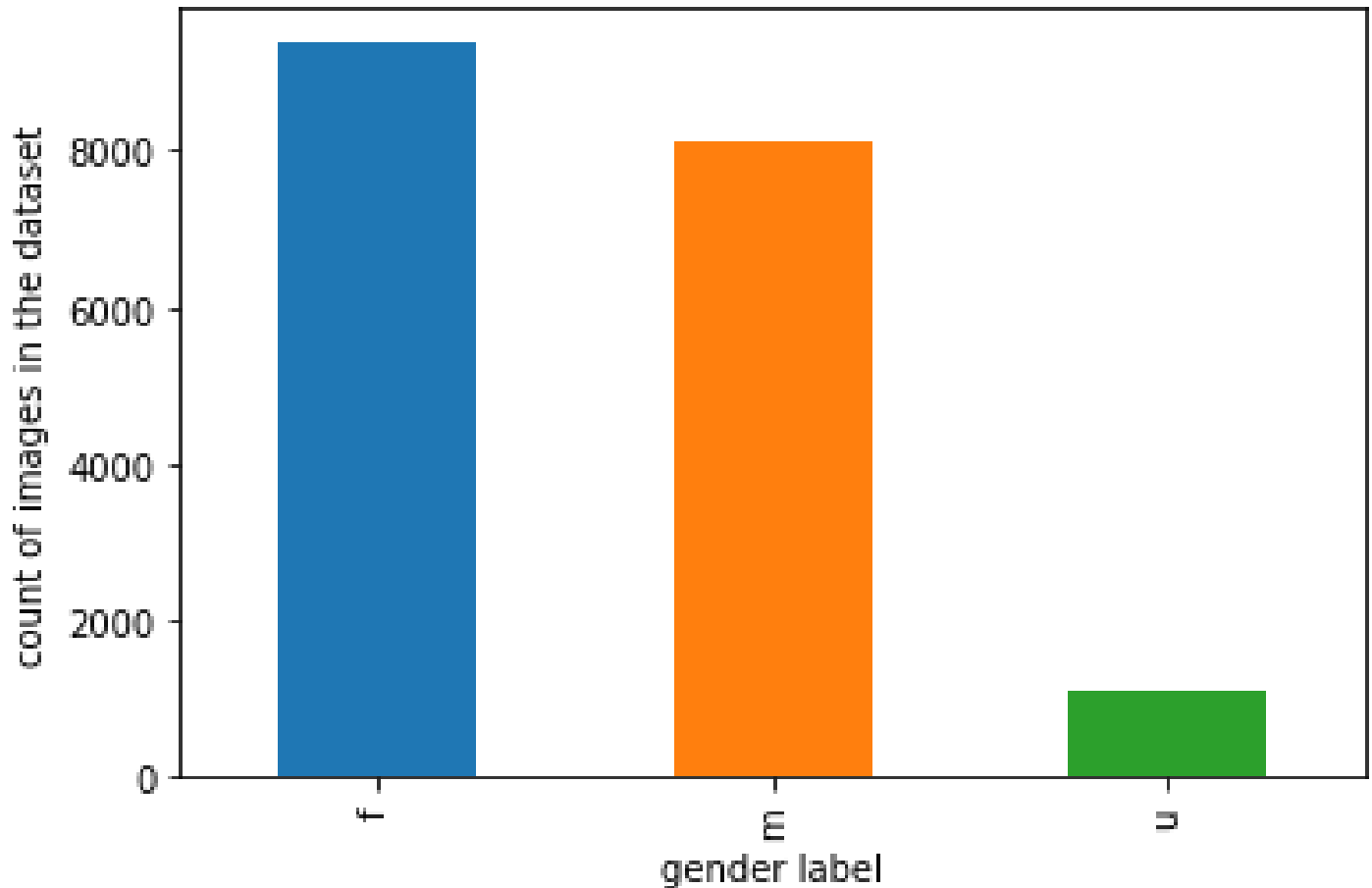


Figure 1 Images from the Adience Benchmark dataset

Exploratory Visualization

I'm only working to classify gender whether it's male or female, so I'm only interested in gender distribution of the data.

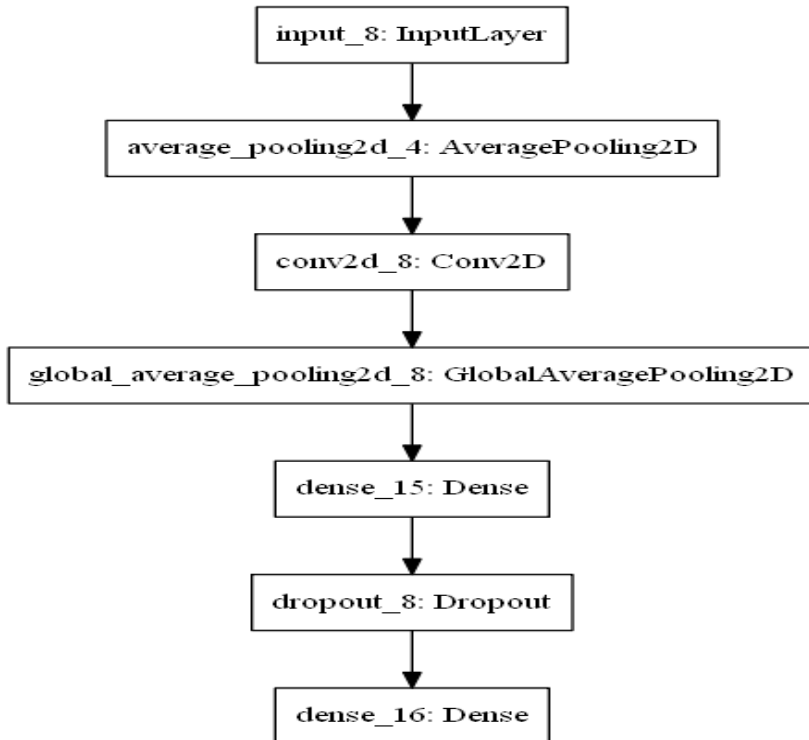
This bar plot shows how many of the images are labeled m and which are labeled f and some are unknown.



Algorithms and Technique

As this problem works with people face images I'm going to use a pre-trained CNN model for image classification ([InceptionResNetV2](#)) trained on imagenet dataset to extract useful features from images and then build a neural network that given the image features extracted from InceptionV3 can classify whether the person in this image is male or female, and will try to add Dropout layer with different rates to avoid over-fitting to the data and will be checking different rates for Dropout and different learning rates and optimizers based on the validation accuracy .

Benchmark Model



I used a simple CNN model with input (224,224,3) then decreasing size by using average pooling layer then using a convolution layer with 64 filters then a global average pooling layer then a dense layer with 32 neurons and a dropout layer then a dense layer with sigmoid activation.

I got accuracy 0.54 and val_acc 0.57 after 16 epoch when training on 5984 image and validation on 1984 with batch size 32

III. Methodology

Data Preprocessing

Data labels comes in 5 txt files (fold_0_data.txt - fold_4_data.txt), I merged all of them into one txt file called data.txt

Then the preprocessing as done in “pre_data” notebook consist of the following:

1. Loading the data and visualizing distribution of the labels of gender
2. There are some images with label ‘unknown’ we ignore them
3. Define the path image from the data columns and explore some images
4. Sample 10000 images of the 17492 images such that 5000 are with label f and 5000 are label m and use 6000 as training set and 2000 as validation set and 2000 as test set
5. Save this cleaned data to try on different pretrained models

Implementation

After preparing the data and dividing it into training set, validation set and testing set.

Now we will choose a pretrained model offered by keras trained on imagenet dataset for image classification, we use it to extract features given the initial images.

Then using the extracted features we train a simple dense layer to classify the gender, and also try to add some Dropout layer to avoid over-fitting.

Here are the process for the implementation:

1. Choose a model from [the pretrained models offered by keras](#)
2. Load the model from keras and remove the predication layer
3. Load the data we saved earlier
4. The images of the dataset are of different size, so we need to resize them to the default pretrained model input shape, keras offers this with the `load_image` function.
5. Also the images needs to normalized, but luckily keras also provide function called `preprocess_input` that take care of it.
6. After `preprocess_input` we just use the loaded model to get the features of the image
7. Now build a dense layer with dropout layer and train it to predict gender and choose a threshold to differentiate the two classes given the model predication probability.
8. Try different values of number of neurons on dense layer and dropout layer rate and choose best one based on the validation accuracy
9. Test the chosen best model on test data
10. Develop a function that given any image can extract it's feature and use the new trained model to predict the gender based on defined threshold

Refinement

Here are some of the steps I went through:

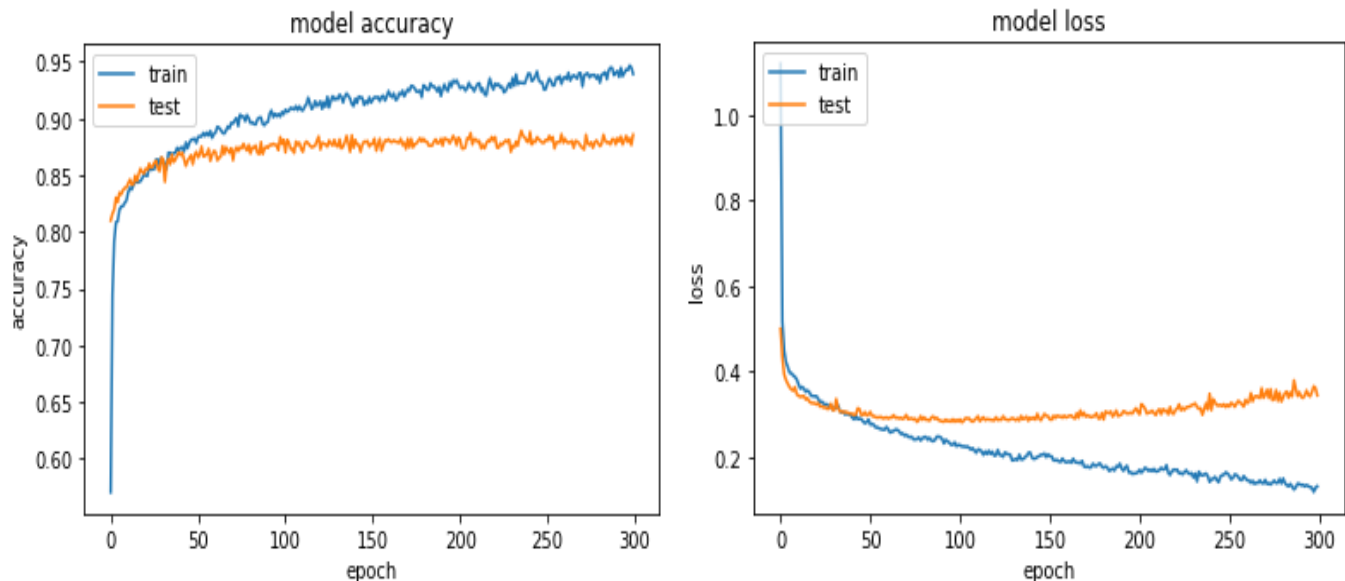
- First using VGG16 to extract features:
 - Then I used a dense layer with 512 neurons but the model over-fitted the training data very quickly
 - Adding a dropout layer with different rates, the best rate I got was 0.9 and 128 epoch with 1024 neurons dense layer to get best validation accuracy of 0.862
- Then tried using InceptionResNetV2 to extract features:
 - I also used a different dropout rates and different number of neurons in dense layer, best result I got was with 1024 neurons in the dense layer and dropout 0.8 after 300 epoch I got best validation accuracy of 0.889

IV. Results

Model Evaluation and Validation

During development a validation set was used to choose the best model.

The best model validation accuracy was 0.889 and on the test set achieved 0.8855 accuracy.



Here's a complete description of the final approach chosen:

- Using InceptionResNetV2 as a pretrained model to extract features and using preprocess_input provided by keras for this model to handle image before predicting.
- After extracting the features used a network with one dense layer of 1024 neurons and dropout layer of 0.8 and number of epochs was 300

To verify the robustness of the model I tested on the test set and also on couple of image from outside the dataset and it achieved pretty good results.

Justification

Given that our Benchmark model only achieved validation accuracy of 0.57 as described in the Benchmark section, and here we could achieve 0.889 validation accuracy so we make a big improvement

Also given that our dataset is only of faces images aligned but still I test it on general images of persons and it predict the right gender.

There's also still a lot of ways we can to increase this accuracy as we will discuss in the improvement section.

V. Conclusion

Free-Form Visualization

Here are 2 complete pictures of males that model can identify they are males even when that the trained model was trained on faces images only



Also two pictures of females the models could identify as females from the whole picture not just their faces



Reflection

The process for this project can be summarized using the following steps:

1. Think about interesting problem to solve, search about relevant work on this problem, find a good dataset to work on
2. Downloading the dataset and exploring it
3. Preprocessing the dataset and dividing it into training, validation and test set
4. Choosing a pretrained model to extract features and preprocessing the images for the model
5. Building a network that given the extracted features can classify whether it's male or female in the image and testing the model

I found step 1 to be somewhat difficult as it's hard to find good dataset to a problem you like.

Also step 2,3 depends on step 1 as there are some dataset that are not well organized and can be frustrating to work with and can cause the model training process to be hard or even impossible if there's a problem with the data that make the model can't learn

I liked step 5 as using features extracted from pretrained model increased the accuracy of the model by a big factor even on the first few epochs and even without training any layers on that model.

Improvement

To achieve a better accuracy we can try the following:

- We can fine-tune the last layers of the pretrained model we are using, but I couldn't do that as I'm working on CPU
- Increase the number of training images we are using, since I only used subset of the data
- Try different dataset
- For new images detect faces in them and align them before predicting