Machine Learning Advanced Nanodegree

Capstone Proposal

Aditya A Kulkarni **December 21st, 2018.**

Domain Background

Tracking, detecting and recognizing faces is one of the challenging and sought after tasks in computer vision. Until a decade ago using computers to work on images and video was a difficult task, then came gpus which boosted the use of neural networks and deep learning. This lead to a major push in the ability of machines to do computer vision tasks. Facial keypoints detection belongs to one of these tasks and can be used as a building block in several applications, such as:

- tracking faces in images and video
- analysing facial expressions
- detecting dysmorphic facial signs for medical diagnosis
- biometrics / face recognition

There are several industries benefiting from this technology. Law enforcement agencies are using face recognition to keep communities safer. Retailers are preventing crime and violence. Airports are improving travelers' convenience and security. And mobile phone companies are using face recognition to provide consumers with new layers of biometric security.

Related research paper: http://cs231n.stanford.edu/reports/2016/pdfs/007_Report.pdf

Link to the data source : https://www.kaggle.com/c/facial-keypoints-detection/data

Problem Statement

The objective of this task is to predict keypoint positions on face images and I'll be treating this problem as a regression problem. Detecting facial keypoints is a very challenging problem. Facial features vary greatly from one individual to another, and even for a single individual, there is a large amount of variation due to 3D pose, size, position, viewing angle, and illumination conditions. Computer vision research has come a long way in addressing these difficulties, but there remain many opportunities for improvement.

Datasets and Inputs

The data set for this competition was graciously provided by Dr. Yoshua Bengio of the University of Montreal. I obtained the dataset from kaggle competition:-

https://www.kaggle.com/c/facial-keypoints-detection/data

The data is in the form of csv files. The each row in the file contains values for each predicted keypoint specified by an (x,y) real-valued pair in the space of pixel indices. There are 15 key points, which represent the following elements of the face:

1. Left_eye_center
2. Right_eye_center
3. left_eye_inner_corner
4. left_eye_outer_corner
5. right_eye_inner_corner
6. right_eye_outer_corner
7. left_eyebrow_inner_end
8. left_eyebrow_outer_end
9. right_eyebrow_inner_end
10. right_eyebrow_outer_end
11. nose_tip
12. mouth_left_corner
13. mouth_right_corner
14. mouth_center_top_lip
15. mouth_center_bottom_lip

Left and right here refers to the point of view of the subject.

In some examples, some of the target keypoint positions are missing (encoded as missing entries in the csv, i.e., with nothing between two commas).

The input image is given in the last field of the data files, and consists of a list of pixels (ordered by row), as integers in (0,255). The images are 96x96 pixels.(i.e 9216 values in the image column)

Data files

- **training.csv:** list of training 7049 images. Each row contains the (x,y) coordinates for 15 keypoints, and image data as row-ordered list of pixels.
- **test.csv:** list of 1783 test images. Each row contains ImageId and image data as row-ordered list of pixels
- **submissionFileFormat.csv:** list of 27124 key points to predict. Each row contains a RowId, ImageId, FeatureName, Location. FeatureName are "left_eye_center_x," "right_eyebrow_outer_end_y," etc. Location is what you need to predict

After a brief study I think I'll be splitting the training the data to create train and validate sets. Also, at the moment I think I'll be using the keypoints which are common in most of the images. I'll be using all the examples in the dataset.

Solution Statement

A model will be built with Keras using tensorflow in the backend and will be trained on training data provided. Specifically a CNN will be implemented as they have proved to work well on problems related to images. The model will be optimised to minimize Root mean square error as defined in the Evaluation Metrics section. Predictions will be made on the test data set which is also provided.

Benchmark Model

The competition's overview page has a getting started with R section which provides a benchmark model using the evaluation metric mentioned below.

The score it 3.758999 for the benchmark model and I'll attempt build a model and improvise on this metric score.

The link for the benchmark model page is:-

https://www.kaggle.com/c/facial-keypoints-detection#getting-started-with-r

Evaluation Metrics

The evaluation metric used by the competition to score the submissions is Root Mean Square Error. RMSE is very common and is a suitable general-purpose error metric. Compared to the Mean Absolute Error, RMSE punishes large errors:

$$RMSE = \sqrt{1/n \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2}$$

where y hat is the predicted value and y is the original value.

Project Design

From the description and problem statement it can be inferred that it is a computer vision task and as mentioned above a CNN will be used to arrive at a solution. Initially data exploration will be carried out to understand possible labels, range of values for the image data and order of labels. This will help preprocess the data and can end up with better predictions. The preprocessing consists of converting the ‘Image’ column into a matrix and then get it into proper dimensions so that it is appropriate input for the neural network model. After this, CNN will be implemented in Keras using tensorflow as backend.

The first CNN architecture will be a sequence of convolution and maxpooling layers with a couple of dense layers in the end. Finally necessary predictions on the test data will be carried out and will be evaluated using the evaluation metric.

A general skeleton would be:

1. Importing the data from csv file and preprocessing the Image column to get the image in a matrix form.
2. Choose the images from which most information can be taken and the model can learn well.
3. Building a sequential model with convolution layers, with max pooling, dropout layers if required.
4. Test it on the test data and calculate the RMSE, reiterate and try to build a better model.
5. Plot the images with facial points on them.

Though creating the architecture is an option it can be really time consuming and resource hungry, thus transfer Learning is considered as an alternative as it saves a lot of training time and can get same or better result within less time.