# Project Proposal: Emotion Recognition Using Deep Convolutional Neural Networks

Rudraksh Tuwani

**Domain Background**

The project that I'm proposing is in the domain of Computer Vision, specifically facial recognition. Traditionally, researchers in this domain used to explicitly define features for both localisation and recognition of faces. However, in recent times, an increasing number of experts in this field have begun to use Convolutional Neural Networks (CNNs) for the above tasks, in order to deal with variable lighting, different postures or situations where hardcoding features may not be viable. CNNs are designed to mimic the visual cortex and have achieved state of the art in recent times.

This project is inspired from: http://cs231n.stanford.edu/reports/2016/pdfs/005_Report.pdf

**Problem Statement**

The aim of this project is to develop a Convolutional Neural Network model to detect and classify the emotion expressed in static images of human faces into seven broad categories, namely:

1. Angry (0)
2. Disgust (1)
3. Fear (2)
4. Happy (3)
5. Sad (4)
6. Surprise (5)
7. Neutral (6)

Mathematically, the aim is to develop a model which can detect the emotion expressed in a static image by taking linear/ nonlinear combinations of pixel values. The dataset (described below) hasn't been preprocessed much and as a result contains variable lighting, different poses, and even watermarks on some images.

**Dataset**

The dataset for this task comes from an archived Kaggle Competition:
https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge

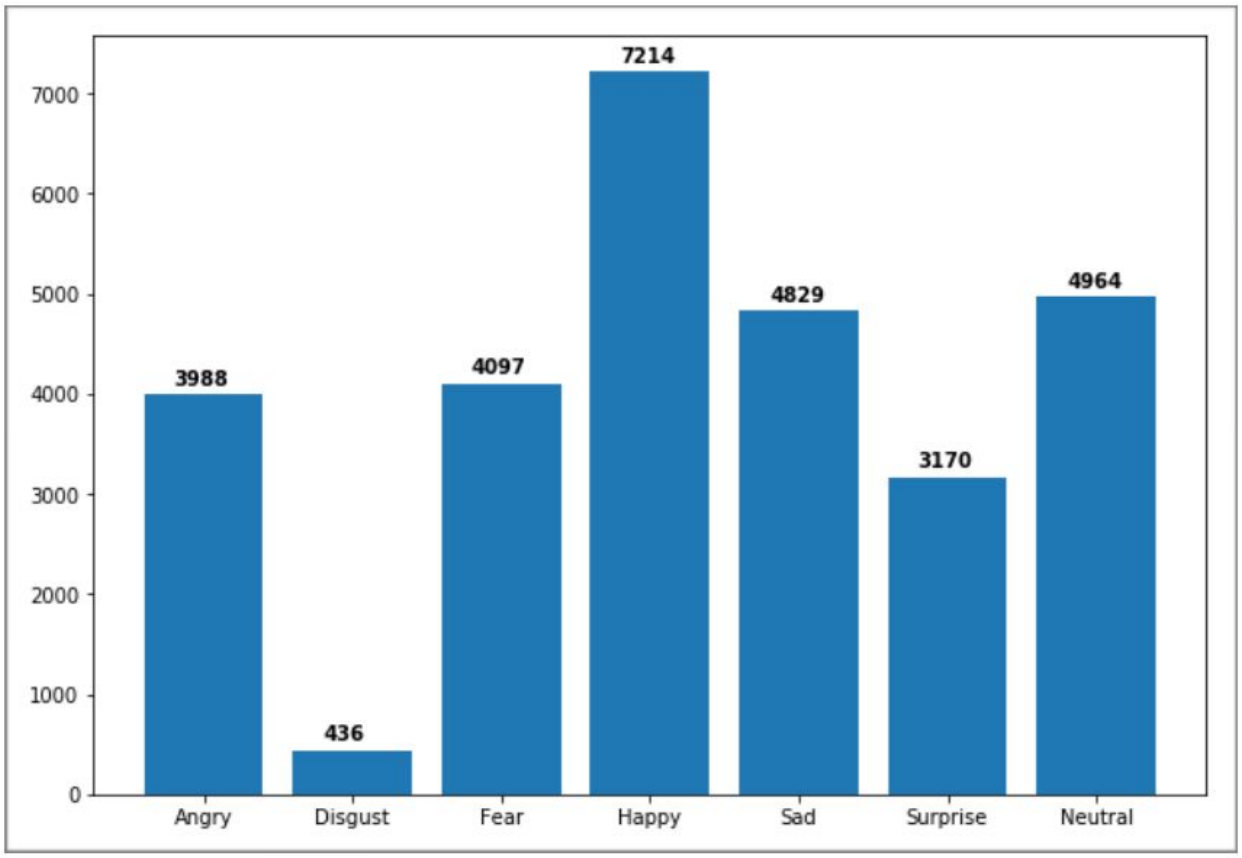
It contains 48 x 48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The data is available in the form of a .csv file with two columns: "emotion" and "pixels". The "emotion" column contains a numeric code ranging from 0 to 6, inclusive, for the emotion that is present in the image. The

"pixels" column contains a string surrounded in quotes for each image.

After removing the corrupted images, there were a total of 35,875 images left in the dataset with the following split recommended:

- Training: 28,698 images.
- Validation: 3,588 images
- Testing: 3.589 images

The following is the number of images in each category in the Training set:




As described above, the images are available in low resolution with almost no standardisation as to the lighting conditions, poses etc.

**Solution Statement**

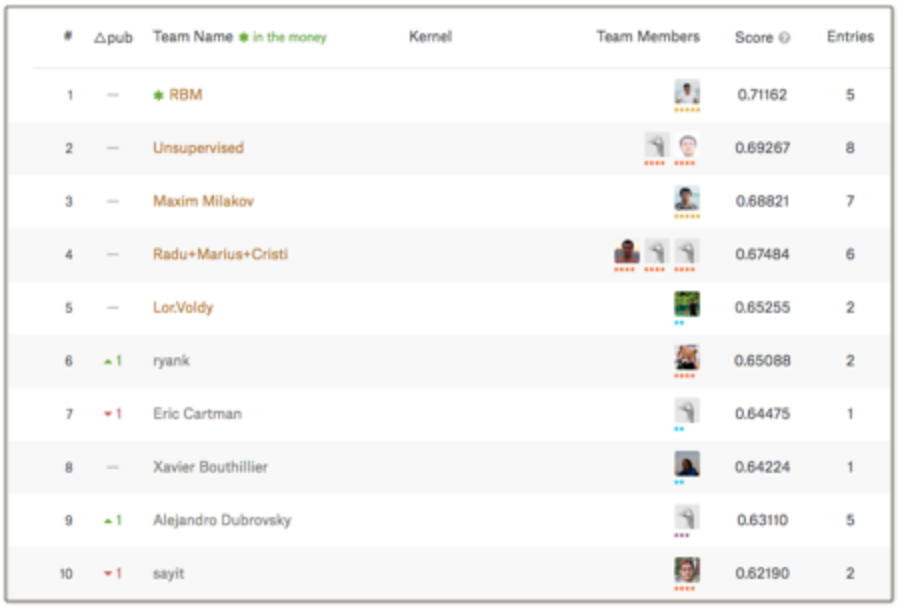
The solution will have two stages:

1. Preprocessing the images.
2. Building a CNN on the preprocessed images with specifics:
    a. Architecture: I intend to use a variant of the popular VGG-16 style ConvNet for this task.
    b. Loss: The intended loss function for training the model is multi class cross entropy.

Additionally, I will also be experimenting with Image Augmentation to make the model more robust.

**Benchmark Model**

The best accuracy on the test set was 0.71162 on Kaggle. Since I am limited by computational resources, my goal is to achieve at least 0.6 accuracy on the test set.



| # | △pub | Team Name ✱ in the money | Kernel | Team Members | Score | Entries |
|---|---|---|---|---|---|---|
| 1 | — | ✱ RBM | | | 0.71162 | 5 |
| 2 | — | Unsupervised | | | 0.69267 | 8 |
| 3 | — | Maxim Milakov | | | 0.68821 | 7 |
| 4 | — | Radu+Marius+Cristi | | | 0.67484 | 6 |
| 5 | — | Lor.Voldy | | | 0.65255 | 2 |
| 6 | ▲1 | ryank | | | 0.65088 | 2 |
| 7 | ▼1 | Eric Cartman | | | 0.64475 | 1 |
| 8 | — | Xavier Bouthillier | | | 0.64224 | 1 |
| 9 | ▲1 | Alejandro Dubrovsky | | | 0.63110 | 5 |
| 10 | ▼1 | sayit | | | 0.62190 | 2 |

**Evaluation Metrics**

I use multi class cross entropy as the evaluation metric. It is defined as follows:

$$H_{y'}(y) = -\sum_i y_i' \log(y_i)$$

where y is our predicted probability distribution, and y′ is the true distribution (the one-hot vector with the digit labels). In some rough sense, the cross-entropy is measuring how inefficient our predictions are for describing the truth.

This is typically not available for the benchmark model. However, for comparison between benchmark model and my own, I'll be using accuracy metric. Accuracy is simply the ratio of observations we correctly classify to the total number of observation.

**Project Design**

Following will be the steps required:

1. We need to first extract images from the .CSV file available on Kaggle.
2. It is always recommended to get familiar with the dataset at hand and know in advance what may go wrong. So, the second step is to perform exploratory data analysis by manually shifting through the image files and their corresponding labels.
3. The above step should help us understand the kind of pre-processing required before the data can be fed to the model. In this step, we will perform the pre processing. Typical examples of preprocessing image data of faces is Data Normalisation, Face centering, ZCA etc.
4. Get the baseline/benchmark model's evaluation metric value. Our target is to better it.
5. Choose the model with the best validation loss as the final model.