1 Notation

In this brief section we will standardize the notation used in this document.

α, β, γ: the first three lowercased greek letters will denote constants

θ: this privileged letter denotes parameters to be learned

ϕ: this privileged letter denotes feature mapping

f: lower cased alphabetical letters will denote functions

$\boldsymbol{v}$: lower cased bolded letters will denote vectors

$\boldsymbol{A}$: upper cased bolded letters will denote matrices

$\mathbb{B}$: upper cased blackboard letters will denote sets

$\mathfrak{C}$: upper cased gothic letters will denote

2 Measuring Attractiveness

Dates: May 28th, 2017 - June 4th

2.1 Problem Statement

We wish to construct a measure of attractiveness individualized for each person, so that for every person k, and a given set of faces $\mathbb{I}$, we have a function f_k so that:

$$f_k : \mathbb{I} \to \mathbb{S},$$

where $\mathbb{S} = \{1, \ldots, N\}$ is some rating system for appropriately large N. Towards this end, we need to:

1. Detect face
2. Place bounding box on face and extract this bounding box
3. Normalize face
4. Extract appropriate features from face
5. Relate feature represenation of face to $\mathbb{S}$.

In this next few sections we will describe each step and detail, and evaluate each open source packages for (1). how the problem is posed, (2). performance on current data set, and (3) complexity in terms of run-time, disk, or network calls.

2.2 Face Detection

We will evalute the following libraries:

1. FaceNet
2. YOLO900
3. OpenCV

The evaluation set will have both positive and negative examples. The positive example will be $1,000$ images from the Tinder-Female dataset, the images are manually selected so that a face is gauranteed to appear somewhere in the image (not a given). The negative data set will be composed of $1,000$ stock images where no faces appear. We will select the library that gives the lowest *false negative* performance.

2.2.1 FaceNet in Torch (OpenFace)

.

This library is found here: https://cmusatyalab.github.io/openface/. It is a Torch implementation of deep neural net described in the paper ”FaceNet: A Unified Embedding for Face Recognition and Clustering” by Schroff, et. al. The documentation appears resonable.

Now we will review the method in FaceNet.

2.2.2 FaceNet in Tensorflow

.

There is one implementation found on github at: https://github.com/davidsandberg/facenet. There is no documenation, although the code appears reasonably organized. This gives me the possiblity of training my own classifer on the proprietary Tinder data set.

2.2.3 YOLO9000

.

2.2.4 OpenCV

.

This is the most mature of all libraries, and thus the documentation is complete. although it does not use convolutional neural nets, installing OpenCV is very convoluted indeed.

2.3 Normalization via Appearence Based Models

2.4 Existing Solutions

2.5 Implementation

2.6 Results