CS230

Human Emotion Classification & Prediction of the OASIS Image Dataset

Protip Roy
protip@stanford.edu
protip_roy@yahoo.com
December 08, 2019

Abstract

Images evoke human emotions, such as fear, anger, joy, sympathy, disgust etc. It might be assumed that such emotions are highly subjective, however studies such as IAPS[3] and OASIS[1] have shown that such responses are largely uniform across a large group of subjects. Project OASIS has published an open-access image dataset labeled by standardized normative emotional ratings. This projects goal is to build a CNN model and train it over the OASIS dataset in order to be able to predict such emotional classifications.

1 Introduction

Images of various kinds such as people, objects and scenes elicit a range of affective response from human viewers. Not only are these responses are of interest to behavioral science, neuroscience and psychology researchers, there is a significant commercial and daily-life ramification as well. A viewer's emotions involved in decision making in an e-commerce purchase or transaction, a voter's impression of a politician looking at a flyer, or forming social biases from viewing images within news articles - are just a few examples. It cannot be overstated that this aspect of emotion plays a very important part in our social and socio-economic life. Any form of automation or tooling which can predict or proxy such human responses will be of great value to the behavioral science community and to say the least - to e-commerce, news media and social media communities.
A basic requirement for this type of automation is a set of images or stimulus and associated standardized emotion ratings. Such a dataset must have been carefully chosen/categorized by behavioral scientists and a uniform rating standard have been invented. In order to be effective, the ratings must have been derived from responses gathered from a large pool of human subjects in a statistically proven manner.
OASIS [1] provides an open-access standardized stimulus set, containing medium resolution affective images in color, that encompasses a broad spectrum of themes. Each image is labeled by a normative rating system on two affective dimensions: *Valence* - the degree of positive or negative affective response and *Arousal* - the intensity of the affective response. Each rating is expressed in a 7-point Likert scale: 1 through 7.
The endeavor of this project is to build a NN whose input X shall be the OASIS[1] provided image set and output shall be a corresponding 7-level classification set which are then compared to the ground truth Y - the respective Valence (or Arousal) ratings provided by

OASIS[1]. The goal is obviously to find a suitable NN architecture and be able to train the model in order to be able to predict with a high degree of accuracy (80%).
An obvious challenge of this task is insufficiency of dataset - we will have to depend on a rather meager number of examples provided by OASIS. Although, a copious number of images may be scraped from internet, those images are not scientifically labeled - or even if so claimed, there is no guarantee of standardization. Thus we shall rely solely on OASIS dataset.

2 Related Work

There are a couple projects under CS230 which has undertaken similar endeavors:

- Is a Picture Worth 1,000 words? Visual Emotional Analysis using Transfer Learning and CNNs by Hasna Rtabi, Katherine Irena Kowalski, Noah Abraham Jacobson: report
- An Iterative Unfreezing Strategy for Predicting Human Emotional Response to Images by Ashwin Sreenivas, Jessica Zhao: report
-

My project differentiates from above by either the choice of network or the input data-set.

3 Dataset and Features

The original dataset comprises of 900 color images of resolution 400Hx500Wx3C and a spreadsheet(CSV) containing the associated ratings of the images. The images are grouped into 4 mutually exclusive categories - Animals, Objects People and Scenes. As described above, there are two primary ratings per image, *Valence* (the degree of positive or negative affective response that the image evokes) and *Arousal* (the intensity of the affective response that the image evokes). A few examples are depicted below:

1. Positive Valence (6.224)

2. Neutral Valence (3.396)

3. Negative Valence (1.327)

Each rating is expressed in a scale of 1.0 - 7.0 as floating point numbers. The data was collected on Amazon Mechanical Turk (MTurk) from 822 participants comprising of men and women belonging to the age group of 18 through 74. The 900 images were divided into 4 lists of 225 each. Further, a participant was presented with only a single list to rate. It was ensured that an image was rated by at least 45 to 50 individuals.
The study team then normalized the Mturk rating data and compiled a mean and a standard deviation for each image. It further separated the data into ratings obtained from men and women. The standard deviation ranged, for example, from 0.91 to 1.28 for Men-Valence. Which in turn, per authors of the OASIS paper, is an indicator of reliability.
For purpose of this project, I have rounded up or down the target rating(mean) extracted from the CSV in order to convert to an integral scale of 1 - 7. In effect, classifying the ground truth to

7 classes. Since we are given essentially 4 ratings per image: Valence-men, Valence-women, Arousal-men and Arousal-women I have chosen only one of the rating for training and development purposes - the Valence-Men.
I would have divided the set of 900 original images into a 90/10 Train/Dev split. However since 810 would have been quite a small number for a training set, data augmentation was deployed first. Following was the data-set preparation & augmentation procedure:

- Read in images in the order listed in the CVS file. Extract Valence-Men column from CSV.
- Down Scale images when applicable - CV2.Resize (Interploation=INTER_AREA) was used
- My code supports taking an N% of original image, flip them horizontally, then add the resulting N% augmented image to the main data set - Numpy.flipLR was used.
- Further take an M% from the augmented set, crop them randomly, then add them to the already flip-augmented set - basic Numpy array manipulation was used here.
- The full data set comprised of ~2000 images in total.
- Finally, the full set was shuffled randomly and was then split into a 90/10 Train and Dev set.

The initial goal was to feed the images to my model(s) in its native resolution without any pre scaling in order to preserve all feature attributes, however, that turned out to be quite ambitious as explained later in this report.
The OASIS dataset is available for download here: http://benedekkurdi.com/oasis.php

4 Methods

I have built two NN Models:

- Model V1: Started with a simple Tensorflow.v1 model comprising of 3 standard layers with a 7-class Softmax activation at the output. As a matter of fact I have leveraged most of the python code from our “Tensorflow Tutorial Assignment”. The model construct is as follows:
  - INPUT : (600000, m) # Flattened —> Transposed (m x 400 x 500 x 3)
  - L1 : Linear(25) —> Relu
  - L2 : Linear(12) —> Relu
  - L3 : Linear(7) —> Softmax —> OUTPUT

- Model V2: A CNN identical to the structure of VGG16 [2]. This model implements the VGG16 structure, i.e. stacked Convolution2D and MaxPool layers followed by 3 FC/Dense layers as shown in FIG 2 below. The last FC layers comprises of only 7 units in order to match the degree of classification of our ground truth. This is Implemented in Tensorflow.v1 as well. The model construct is as follows:
  - INPUT : (m, 100, 125, 3) # Down scaled to 25% of original
  - L1-1 : Conv2d(3, 3, 64) —> Relu
  - L1-2 : Conv2d(3, 3, 64) —> Relu —> MaxPool(2,2)
  - L2-1 : Conv2d(3, 3, 128) —> Relu
  - L2-2 : Conv2d(3, 3, 128) —> Relu —> MaxPool(2,2)
  - L3-1 : Conv2d(3, 3, 256) —> Relu
  - L3-2 : Conv2d(3, 3, 256) —> Relu
  - L3-3 : Conv2d(1, 1, 256) —> Relu —> MaxPool(2,2)
  - L4-1 : Conv2d(3, 3, 512) —> Relu
  - L4-2 : Conv2d(3, 3, 512) —> Relu
  - L4-3 : Conv2d(1, 1, 512) —> Relu —> MaxPool(2,2)
  - L5-1 : Conv2d(3, 3, 512) —> Relu
  - L5-2 : Conv2d(3, 3, 512) —> Relu
  - L5-3 : Conv2d(1, 1, 512) —> Relu —> MaxPool(2,2)
  - L6 : Dense(4096) —> Relu
  - L7 : Dense(4096) —> Relu
  - L8 : Dense(7) —> Softmax —> OUTPUT

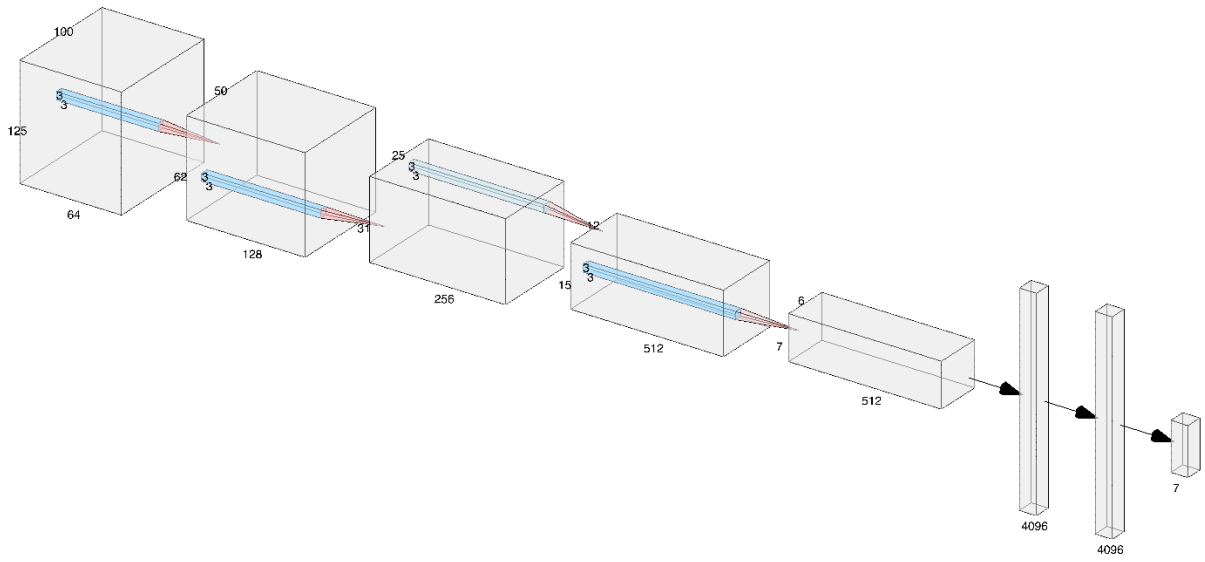

FIG 2: MODEL V2 - VGG16 CONVOLUTION MODEL

For both models, weights were initialized by Xavier Initialization while biases were initialized to zeros. Gradient descent was accelerated by Adam Optimizer with a learning rate of 0.0001.
Regularization in form of L2/weight-decay or drop-out was highly desirable but was not accomplished due to practical constraints.
The models and original data-set are available at GitHub link:
https://github.com/protip-roy/cs230_oasis.git
The top-level files are "cs230_oasis/v1/oasis_v1.py" and "cs230_oasis/v2/oasis_v2.py" respectively. "cs230_oasis/v1/image_proc.py" & "cs230_oasis/v2/image_proc.py" are the data pre-processors.

5 Experiments/Results/Discussions

Both models were trained on a p2.xlarge AWS VM featuring a single Tesla K80 GPU. The chosen AMI configuration was tensorflow_p36.
In case of Model V1 (Standard NN), I was able to load a full train set (1891) without any need for down scaling of image resolution (400x500). As shown in FIG 3, the cost flatlined at 1.70 with this model. However, it took about 13 hours to run 4000 epochs with a mini-batch size of 256. Train & Test accuracies are listed in the table below.
Although runtime of Model V2 (CNN- VGG16) fared much better it encountered compute resource limitations. As can be seen in FIG 4, given a very small training set (18), cost optimized to near 0 very quickly within 100 epochs. However, training could not be completed with a full training set (1891) of the original resolution images (400x500) even with a mini-batch size of 4. While training, the GPU crashed out with Out-Of-Memory(OOM) error. Thus, I resorted to down scaling the original images to 4:1 ratio - .i.e to 100x125x3.

With a full training set(1502 or 1434) of the down-scaled images the model performed very well

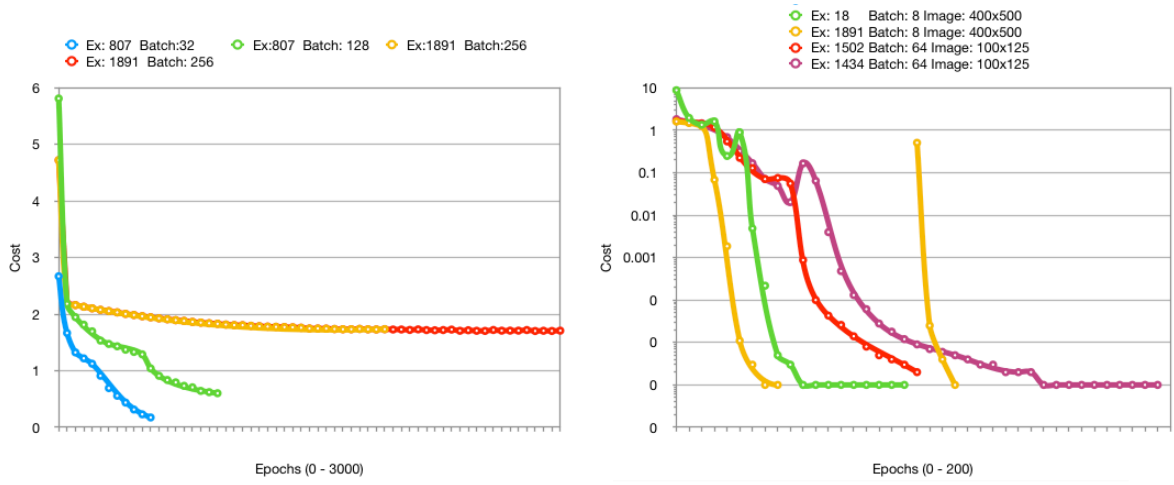

FIG 3: MODEL V1 (STANDARD) RESULTS **MODEL V2 (CNN) RESULTS**

in terms of cost optimization and run-time. The model reached to a zero cost within 200 epochs with a batch size of 64. Each run was completed with an hour. The table below captures a few performance data points for both models:

	Image Resolution	Train Set Size	Batch Size	Epochs	Training Accuracy	Dev/Test Accuracy
Model V1	400x500	807	32	600	0.94423795	0.22222222
Model V1	400x500	807	128	1000	0.87608427	0.3
Model V1	400x500	1891	256	2000	0.3395029	0.3270142
Model V1	400x500	1891	256	3000	0.3395029	0.3270142
Model V2	400x500	18	8	100	1.0	0.5
Model V2	400x500	1891	8	110	crash	crash
Model V2	100x125	1502	64	100	1.0	0.4011976
Model V2	100x125	1434	64	200	1.0	0.3625

Table above shows that even though Training Accuracy reaches 1.0 with the Convolution model, the test accuracy gets saturated at ~0.5. Clearly this model is over-fitting to the training set. Also, dev/test accuracy was compromised when original image resolution was down scaled.

6 Conclusion/Future Work

In order to address variance or overfitting noted above, next step would be to introduce Regularization (L2). TF.v1 API does not appear to offer an efficient way of regularization. Thus I plan on continuing the development on either TF.v2 (Keras) or PyTorch framework. Down scaling the original images clearly resulted in loss of features and in effect poorer model accuracy - I need to find a way of overcoming this hurdle.

References

[1] Introducing the Open Affective Standardized Image Set (OASIS)
Benedek Kurdi1 & Shayn Lozano1 & Mahzarin R. Banaji1
Published online: 23 February 2016 © Psychonomic Society, Inc. 2016

[2] VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION
Karen Simonyan* & Andrew Zisserman+
Visual Geometry Group, Department of Engineering Science, University of Oxford {karen,az}@robots.ox.ac.uk

[3] International Affective Pictures System (IAPS)

https://csea.phhp.ufl.edu/media.html#topmedia

Acknowledgements

The project idea came from my external mentor Mana lewis - mana@chezmana.com