# Emotion AI, Real-Time Emotion Detection using CNN

**Tanner Gilligan**
M.S. Computer Science
Stanford University
tanner12@stanford.edu

**Baris Akis**
B.S. Computer Science
Stanford University
bakis@stanford.edu

**Abstract**

In this paper, we describe a Convolutional Neural Network (CNN) approach to real-time emotion detection. We utilize data from the Extended Cohn-Kanade dataset, Japanese Female Facial Expression data set, and our own custom images in order to train this model, and apply pre-processing steps to improve performance. We re-train a LeNet and AlexNet implementation, both of which perform with above 97% accuracy. Qualitative analysis of real-time images shows that the above models perform reasonably well at classifying facial expressions, but not as well as the quantitative results would indicate.

## 1 Introduction

The ability to confidently detect human emotions can have a wide array of impactful applications, and therefore emotion recognition has been a core area of research in computer vision. We wanted to focus on the issue of emotion recognition, and build a real-time emotion detection system.

When we began to work on the area of emotion detection, we quickly realized that there is an innate problem which is that all data sets are based on "acted" emotions instead of "real" emotions. Many of these data sets such as CK+ ([Lucey et al., 2010]) and JAFFE ([Lyons et al., 1998]) are collections of actors who demonstrated core emotions in front of a camera. Therefore the field isn't detecting real emotions, but rather detecting the emotion that the subject is acting or the observer is perceiving. This problem was also very obvious while testing our model, as we saw confidence scores increase as the subject portray very exaggerated facial expressions that would be defined as "fake" by a human.

When we discussed possible applications of a successful emotion recognition tool, one application we though of is to use emotion labels and prediction scores combined with social science on emotion research led by Paul Ekman [Ekman, 1992] to predict emotion intensities. As indicated in Frijda et al. [1992] emotion intensity prediction is a really hard problem and a very valuable insight for the field of psychology. The main reason we didn't pursue emotion intensity prediction is that there were no existing data sets or research that can serve as the ground truth.

Therefore we concentrated on building a successful emotion recognition model that can work in real-time. In this project we built a model that uses Convolution Neural Networks to successfully classify faces as one of the core seven emotions: anger, contempt, disgust, fear, sadness, happiness, surprise, and neutral [Darwin et al., 1998].

## 2 Background

### 2.1 Previous Works

Emotion Recognition in the Wild (EmotiW) Challenge is the leading academic challenge on emotion recognition and labeling. We concentrated on the winning papers of the 2014 and 2015 challenge. Its important to highlight that the papers demonstrating the best results for the 2015 Image based Static Facial Expression Recognition Sub-challenge used CNNs. Yu and Zhang [2015] proposes a CNN architecture specialized on emotion recognition performance. They propose two novel constrained optimization frameworks to automatically learn the network ensemble weights by minimizing the loss of ensembled network output responses. Kim et al. [2016] took a different approach by creating a committee of multiple deep CNNS. They also created a hierarchical architecture of the committee with exponentially-weighted decision making process.

There were also a wide variety of other papers that suggested alternative methods to CNN, but didnt perform as well. Most of these papers use support vector machines (SVM) or largest margin nearest neighbor (LMNN) for classification. The main difference between these were the feature descriptors. [Dhall et al., 2011] used a system that extracts pyramid of histogram of gradients (PHOG) and local phase quantization (LPQ) features for encoding the shape and appearance information. [Yao et al., 2015] used AU (Action Unit) aware features that were generated after finding pairwise patches that are significant to discriminate emotion categories. Yaos main insight was that previous research groups neglected to explore the significance of the latent relations among changing features resulted from facial muscle motions. This approach delivers results better than the winning team of 2014 but falls short compared to 2015 winners results. [Shan et al., 2009] concentrated on person-independent facial expression recognition and used Local Binary Patterns (LBP) descriptors. Many of these algorithms that used feature based models aimed to mimic Ekman's suggestions for human emotion at [Ekman et al., 2013].

### 2.2 Improvements on Previous Works

Since many of the most successful emotion recognition applications used CNN's, we also decided to use CNNs as our model to target the emotion recognition problem. We already started seeing over 90% train and test accuracy by only using CK+ data set. Similar to many other papers shared below, we added JAFFE data set and this increased our accuracies. Different than most other papers in this area we also created data samples on our own, and added these to our data sample as well. Surprisingly this increased accuracies even further. We were able to use these additional samples since we were using CNNs, and the accuracy aren't dependent on very specific features like AU-intensities.

It was hard to find accurate benchmarks since a lot of the papers were data set dependent. Therefore we concentrated on research that also trained their models on CK+ data set, and ideally added JAFFE as well. In these papers, we saw results ranging from 45% and 97%, which is in line with our final results of around 97% test-set accuracy.

Chew et al. [2011] used face tracking and constrained local models (CLM) with CK+ data set and had testing accuracy ranging 45.9% (sad) and 93.6% (surprise). Jeni et al. [2013] also used CK+ data set and they were able to reach 86% average accuracy on continuous AU (Action Unit) intensity estimation. Velusamy et al. [2011] used most discriminative AUs for each emotion to predict emotions and reached to 97% with CK+ data set but only 87.5% with JAFFE. Since this approach was really dependent on discriminative physical expression of emotions it didn't do as strongly in JAFFE database. Islam and Loo [2014] Utilized displacement of points on the face between neutral and expressive emotions and was able to correctly classify 83% (fear) to 97% (happiness) of the emotions. Again this approach was able to successfully classify emotions like happiness and surprise since large displacements in the mouth region created discriminative results.

Our final results are very promising since they result in accuracies that are less dependent on the actors or emotions being portrayed.

## 3 Approach

### 3.1 Overview

In order to accomplish our task of developing a real-time emotion detection system, we had to get several components working independently. We also had to conduct research to better understand the basis for our problem, and how we could improve our results once we got things working. Below summarizes our general approach:

- Build Dataset: We collected labeled facial-expressions data sets from multiple sources, and processed their labels and images into a common format. We then introduced custom images of ourselves and a friend to further enrich the data set.
- Pre-process Images: We ran facial-detection software to extract out the face in each image. We then re-scaled the croppings, and manually eliminated poor images. As a pre-processing step for the CNN, we also applied a Gaussian filter to the images, and subtracted the mean-image of the training set from each image. In order to get more out of our limited training data, we also augmented the images to include reflections and rotations of each image, with the hope that this would improve robustness
- Construct CNN: We utilized pre-trained versions of AlexNet and LeNet in Caffe on

AWS, where we re-trained the first and last layer. We also had to experiment with various learning rate methods and parameters in order to generate a non-divergent model.

- Develop real-time Interface: OpenCV allowed us to get images from our laptop's webcam. We then extracted the face as before, pre-processed the image for the CNN, and sent it to AWS. On the server, a script would run the image through the CNN, get a prediction, and the results would be pulled back to local.

### 3.2 Implementation

#### 3.2.1 Dataset Development

**CK+ Dataset** The first step in developing our emotion-detection system was to acquire data with which to train our classifier. We sought to find the largest data set we could, and we selected the Extended Cohn-Kanade (CK+) data set. This data set is composed of over 100 individuals portraying 7 different labeled emotions: anger (1), contempt (2), disgust (3), fear (4), happy (5), sad (6), and surprise (7). In addition, we also introduced an addition class-0 to represent a neutral expression. One feature we really liked about this data set is that for each person displaying an emotion, the directory contains 10 to 30 images demonstrating that individual's progression from a neutral expression to the target emotion. This is good because it allows us to have multiple degrees of intensity for each emotion represented in our data set, as opposed to only the most extreme examples. We originally elected to take the first two images of each sequence and label them as neutral, and the last three and label them as the target emotion. We found that this greatly limited our training set size, however, as we were left with fewer than 1000 training images. To combat this, we looked more closely at the images and decided to take the last third of each sequence as the target emotion, as opposed to just the last three.

**Excluding Contempt** Upon testing this 8-class classifier, we found that it tended to over-predict "contempt". This manifested in quantitatively lower recall and precision scores for the contempt class, as well as qualitative worse predictions when we fed it live images. We conducted further research on this and found that many papers on emotion detection ignore the contempt class, as they say it is merely a combination of fear and disgust. Taking this into account, we dropped all instances of contempt from our data set, and re-split it. Thankfully contempt was the smallest class in terms of image count, so we didn't lose a substantial part of our data set.

**JAFFE Data Set** After eliminating contempt, we again tested our model qualitatively and quantitatively. We found that we were doing very well quantitatively, our precision, recall, and accuracy were all well over 90% on both test and train, but our qualitative results were still rather poor. Since the network was doing very well on data it was given, but was not generalizing well, we decided to find additional data sources. One of the research papers we investigated combined the CK+ with the Japanese Female Facial Expression (JAFFE) data set, and was able to achieve improved results. Unfortunately, the data set only contained around 250 images, but it was still able to boost the model's performance by a few percent.

**Custom Images** Since the real-time interface was being tested solely by us, we also decided to add ourselves to the data set to see if it would improve qualitative results. We found a friend to help us, and the 3 of us proceeded to take an additional 20 images for each class, further increases our data set size. After this final inclusion, our final data set sizes were:

| Set | Size |
|---|---|
| Train | 2104 |
| Val | 300 |
| Test | 601 |

Including the images from the JAFFE data set and the ones we custom-made, we were able to again boost our quantitative results by a small margin, and our qualitative results also noticeably improved.

#### 3.2.2 Data Pre-processing

Given the non-homogeneity of the data set, we had to pre-process the data into a common format. We first converted all images to grayscale. We then utilized OpenCV to detect faces within each image, which returned to us a set of bounding boxes to examine. In cases where no bounding box was found, we set that image aside and ran it again using different detection parameters until we were able to successfully detect the face. In cases where multiple bounding boxes were returned, we analyzed the sizes and locations of the boxes, and selected the one that was largest and/or most central in the image. Using this approach, we were able to extract the facial component of every image in our data set. Once extracted, we re-scaled the images to a common 250-by-250 size.

In order to help the CNN perform better, we also applied statistical pre-processing to the images. The first step we took was applying a small 5-by-5 Gaussian filter over the images, which is meant to help smooth-out noise while still preserving image's edges and distinctive features. The second step we took was subtracting the training-set's mean image from every image. This is beneficial because the distribution of pixel values becomes centered at 0, and is common practice for training data fed to any machine-learning model.

Since we only have 2104 distinct training image's, and Convolutional Neural Networks tend to perform better with more data, we sought to find ways to enrich this data set. To do this, we augmented each image in two ways. First, we mirrored the image across the Y-axis, which produced a similar but not identical training point. In addition, we also introduced slight rotations of 10 degrees in either direction for each image, which helped to boost our training-set size, and improve robustness.

#### 3.2.3 CNN Construction

In order to develop our Convolutional Neural Network, we decided to utilize pre-trained models. We believed that this would lead to better results for our project, since these pre-trained networks are much deeper that we could develop, and would thus have much better feature-detection power. In researching existing networks, we couldn't find any that dealt directly with facial detection or recognition, so we chose to use networks with varying initial applications.

The first network we used was LeNet, which was trained on the MNIST data set. The MNIST data set is composed of hand-written numbers, and the objective of the model is to classify each image as a digit. The second network we looked at was AlexNet, which was developed and trained for the ImageNet Challenge. This challenge seeks to classify images into one of 1000 categories, ranging from animals to beverages. Even though neither of these networks deals directly with faces, our hope is that the lower level features learned by these networks, such as edges and curves, can be transferred from these data-rich environments to our data-poor environment.

Since LeNet and AlexNet were trained with different intentions than our own, we needed to tweak the networks slightly. First, since our input images were neither color nor the 227-by-227 dimension utilized by these networks, we had to change the input data-layer and retrain the first convolutional layer to account for this. Second, since we are only predicting 7 classes rather than the 1000 originally used, we needed to retrain the final softmax layer. As a result of us having far fewer training images than these networks originally had, we also had experiment with different learning rate hyper-parameters in order to induce convergence, as the original hyper-parameters often diverged. We ultimately settled on a ”fixed” learning rate policy, with base learning rate of 0.001 with a momentum of 0.9 and weight_decay of 0.0005. Note that even though a ”fixed” learning rate policy is used, the SGD solver of Caffe still uses the momentum and weight_decay to steadily reduce weight updates over time.

#### 3.2.4 Real-Time Interface

In order to create the real-time interface, we needed to gather local images and run them through the CNN on AWS. To accomplish this, we utilized OpenCV to extract the images from out laptop's webcam. The images were them pre-processed in the same manner as our data set: convert it to grayscale, extract the facial component, and re-scale to 250-by-250. We chose to use 250-by-250 images because AWS would only allow us to send at most 65KB in a single file, so the 250-by-250 images fit within this constraint.

On AWS, we had a server script that has our trained model loaded into memory, and waits for an incoming file. When received, the image has a Gaussian filter applied, and the mean image is subtracted, just as with the rest of the data set. The image is then augmented via a mirroring and rotation, and the set of images is fed into the neural network. A prediction is produced for each image, and we select our prediction to be the most common class label among the images. If there is a tie, we select the class with the highest sum of class scores among the maximal classes.

One limitation of AWS is that it does not allow you to send data directly to a local computer, so our script could not simply send the results back when the computation was finished, or even signal to us that it was done. We had our first script write the results to a file and have a second socket that listens the file. We had another local script that called the second AWS server after sending images were done to retrieve the results and this combinations of four scripts and two sockets created a close to real-time interface.

## 4 Experiments and Results

### 4.1 Result Metrics

For our project, we have two metrics we use to evaluate the performance of our model. The first is the obvious quantitative results, such as precision, recall, and accuracy, in which we examine the statistic success of our model in predicted our labeled data set. The second success measure we use is how well it is able to classify our live-streamed images. Since there are no labels for these images, only us knowing which expression we are trying to portray, it is difficult to quantitatively examine these results without have to hand-label a second data set (which we do in some instances). Furthermore, since neither of us are actors, our expressions could also be poor portrayals of the target emotions, but this is also a more realistic application of the system. In the below analysis, we describe our results with respect to both of these metrics.

### 4.2 8-class Prediction

In our first round of experimentation, we used the full 8-class CK+ data set including contempt. Here, the classes are:

0: Neutral
1: Anger
2: Contempt
3: Disgust
4: Fear
5: Happiness
6: Sadness
7: Surprise

We used only LeNet for our initial exploration, and after tuning hyperparameters, achieved the following results on our test set:

Confusion matrix

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 149 | 0 | 2 | 1 | 0 | 0 | 1 | 0 |
| 1 | 61 | 0 | 0 | 0 | 0 | 1 | 0 |
| 4 | 0 | 5 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 51 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 27 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 82 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 33 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 80 |

Precision

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| .94 | 1.0 | .71 | .98 | 1.0 | 1.0 | .97 | 1.0 |

Recall

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| .97 | .98 | .56 | .94 | 1.0 | .99 | 1.0 | .99 |

| Dataset | Acc. |
|---|---|
| Train | 0.988 |
| Val | 0.972 |
| Test | 0.972 |

From the above results, specifically the precision and recall of class 2 (contempt), we can see that this class is clearly performing the worst. Upon inspecting the same statistics on both our training and validation set, we find similar results. In addition, our qualitative analysis also indicates that the contempt class was causing issues, as most images we sent it to predict were classified as contempt, even though it is the minority class. On a sample of 20 images we sent to be predicted, 4 were classified correctly, 3 were classified incorrectly, but not as contempt, and the remaining 13 were all labeled as contempt. This revelation is what drove us to investigate literature further, and found that most emotion detection researchers tend to discard the contempt class. We followed this example, and retrained our model excluding this class

### 4.3 7-class Prediction

After retraining out model, we achieved improved results. Due to the fact that contempt made up such a small portion of the training data, the changes in accuracy and precision aren't very much, but they are substantial when considering that only about 1% of the data set was altered. Upon extracting our summary statistics, from the model, and using the same class labels as above (dropping contempt), we obtained the following results:

Confusion matrix

| | | | | | | |
|---|---|---|---|---|---|---|
| 128 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 51 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 57 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 34 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 78 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 32 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 100 |

Precision

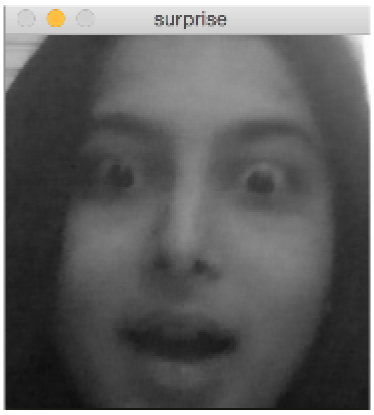
| | | | | | | |
|---|---|---|---|---|---|---|
| 0.96 | 1.0 | 1.0 | 0.97 | 1.0 | 1.0 | 0.99 |

Recall

| | | | | | | |
|---|---|---|---|---|---|---|
| 0.98 | 0.98 | 0.95 | 1.0 | 1.0 | 1.0 | 0.99 |

| Dataset | Acc. |
|---|---|
| Train | 1.0 |
| Val | 0.975 |
| Test | 0.986 |

It is difficult to directly compare the results from the 7-class and 8-class case since the data was re-segmented when removing contempt. Despite this, one can clearly see that the precision, recall, and accuracy for every class increased between the two runs. This indicates that excluding contempt not only improve performance by not misclassifying contempt images, but also preventing other images from being confused with contempt, and were thus classified correctly. Our qualitative results also improved as a result of this change, and we were able to get more correct predictions. In a sample of 20 webcam images we sent to the network, 8 were classified correctly, and we had 100% accuracy on surprise images. An example of a correctly classified image is below:



## 4.4 JAFFE Predictions

Once we removed the contempt class from our data set, we were able to add the Japanese Female Facial Expression (JAFFE) data set as well, since it includes images for the remaining 6 emotions. The inclusion of this data set resulted in the following:

Precision

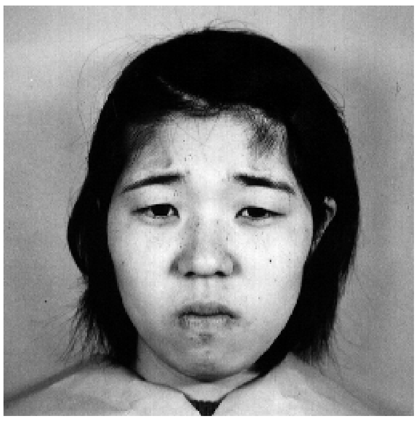
| | | | | | | |
|---|---|---|---|---|---|---|
| 0.98 | 0.95 | 0.88 | 0.97 | 0.99 | 0.75 | 1.0 |

Recall

| | | | | | | |
|---|---|---|---|---|---|---|
| 0.97 | 0.98 | 0.92 | 0.91 | 0.93 | 1.0 | 0.92 |

| Dataset | Acc. |
|---|---|
| Train | .979 |
| Val | 0.969 |
| Test | 0.945 |

We can see that the JAFFE data set noticeably reduced our accuracies across all three splits. In addition, the precision of class 5 (sadness) took a big hit, dropping from 1.0 to 0.75. In an attempt to better understand why this occurred, we looked at some of the JAFFE images that were labeled as sad. We found that some of these images were rather poor or subtle examples of a sad expression, and could easily be confused with neutral by just looking at them. Below is an example of an image that is labeled as sad, but was incorrectly classified as neutral by our model:



We decided to leave these bad images in the data set because, even though they are poor examples of the target emotion, they are still valid expressions of them. By excluding them, we would be hand-picking our data set to only train on exaggerative expressions that would likely never be present in the real world. Instead, we elected to simply add more data to our data set in the hopes that it would both disambiguate some of the JAFFE images, as well as improve our overall performance.

### 4.5 Custom Images

In the final iteration of constructing our data set, we added a total of 420 images equally split among the 7 classes. In addition to providing additional, unique images to the model, it also helped to balance the class distribution, which is an issue we had previously been unable to address. By adding these images into our data set, we were able to achieve results similar to the 7-class model in nearly every category, which is significant given we retain the JAFFE data set which previously decreased our performance. In addition, this was the first instance where AlexNet outperformed LeNet, so below we show AlexNet's results:

Confusion matrix

| | | | | | | |
|---|---|---|---|---|---|---|
| 136 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 74 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 59 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 40 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 117 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 54 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 112 |

Precision

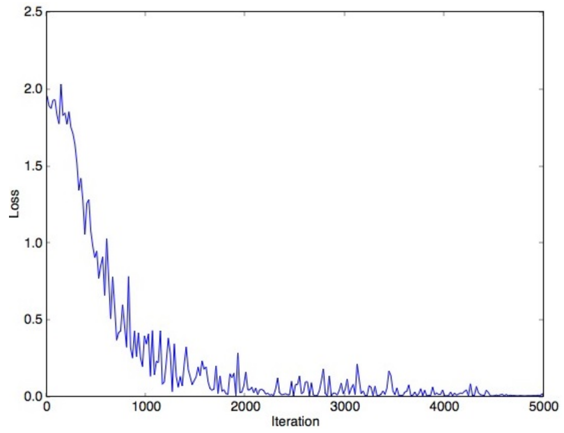
| | | | | | | |
|---|---|---|---|---|---|---|
| 0.99 | 0.97 | 0.98 | 0.95 | 0.99 | 1.0 | 0.99 |

Recall

| | | | | | | |
|---|---|---|---|---|---|---|
| 1.0 | 1.0 | 0.98 | 0.91 | 1.0 | 0.96 | 0.98 |

| Dataset | Acc. |
|---|---|
| Train | 1.0 |
| Val | 0.99 |
| Test | 0.985 |

AlexNet Loss



In addition to quantitative improvements in our model, we also experienced qualitative improvements results as well. When observing the live-stream of predictions being returned to us by our network, the results were much better than with our previous models. When portraying extreme expressions of surprise or sadness, the model correctly classified them with near perfect accuracy. When making expressions that were less exaggerative, the model was not able to classify the images very well, as one might expect. This is because the key features that differentiate the classes are not readily apparent, so the model can not predict as well. One class that the model qualitatively performs very poorly at is happiness. In our numerous attempts to elicit a prediction of happiness from our model, we nearly always failed. Interestingly, when a friend of ours attempted to do the same, she was able to consistently get predictions of happiness when we couldn't. We are unsure why this would occur, but it could be caused by an underlying artifact of our data set, such as a woman's facial features being more highly associated with a happy expression.

In an attempt to quantify our relatively qualitative results, we tried to classify 50 live-stream images. Of those we sent, the network correctly classified 28, typically being those with the most exaggerative expressions. As previously discussed, surprise and sadness performed the best, while happiness performed the worst. In addition to looking at the predicted class, we also analyzed the class scores output for incorrectly classified image. In comparing the class scores produced by our earlier models to those produced our final model, we noted a respectable increase in the score for the correct class. In every case we examine, the final model produced class scores such that the correct class was either the second or third maximal score, while our previous models had no such guarantee. This shows that even though we couldn't correctly classify the images, our predictions were at least closer to being correct. In summary, we were able to achieve improved results on our life-streamed images, but not nearly as well as our quantitative results would indicate.

## 5 Conclusion

In this paper, a CNN-based emotion detection model is proposed that utilizes facial-detection software and cloud computing to accomplish its task. The final model resulted in accuracies comparable to the state-of-the-art papers in the field, reaching as high as 98.5% accuracy on our custom data set, and 97.2% on the original CK+ data set. Our code base can be fount at `https://github.com/barisakis/cs231a_eai`. In addition, our model also exhibits more balanced accuracy results across the emotion spectrum. Lastly the proposed model still worked significantly well with non-actor subjects, especially for physically expressive emotions like sadness, happiness and surprise.

One future area of work is to create a user interface where users can iteratively train the model through correcting false labels. This way the model can also learn more from real world users who express various emotions in different ways. In addition, including a layer in the network that accounts for class imbalance could provide addition improvements over our results. We attempted implement latter of these, but were unable to get it working.

Another area of interest to explore is predicting on a continuous scale the intensity of emotions being portrayed. We believe that we already have a reliable recognition algorithm, so by incorporating knowledge from the social sciences on emotion, a more powerful predictor could be built. In order develop and train such a predictor, however, one would need an annotated data set with which to work. One possible means for creating such a data set would be to aggregate people's opinions of an image using a service like Amazon Mechanical Turk, and then average the responses together to produce an intensity measure for each emotion.

Even though many state of art algorithms are very good at detecting facial expressions, these images are often exaggerative and un-realistic. As such, we believe that there is need for better data sets aimed at understand 'real' emotions.

## References

S. W. Chew, P. Lucey, S. Lucey, J. Saragih, J. F. Cohn, and S. Sridharan. Person-independent facial expression detection using constrained local models. In *Automatic Face Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 915–920, March 2011. doi: 10.1109/FG.2011.5771373.

Charles Darwin, Paul Ekman, and Phillip Prodger. *The expression of the emotions in man and animals*. Oxford University Press, USA, 1998.

Abhinav Dhall, Akshay Asthana, Roland Goecke, and Tom Gedeon. Emotion recognition using phog and lpq features. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 878–883. IEEE, 2011.

Paul Ekman. An argument for basic emotions. *Cognition & emotion*, 6(3-4):169–200, 1992.

Paul Ekman, Wallace V Friesen, and Phoebe Ellsworth. *Emotion in the human face: Guidelines for research and an integration of findings*. Elsevier, 2013.

Nico H Frijda, Andrew Ortony, Joep Sonnemans, and Gerald L Clore. The complexity of intensity: Issues concerning the structure of emotion intensity. 1992.

Md Nazrul Islam and Chu Kiong Loo. Geometric feature-based facial emotion recognition using two-stage fuzzy reasoning model. In *Neural Information Processing*, pages 344–351. Springer, 2014.

L. A. Jeni, J. M. Girard, J. F. Cohn, and F. De La Torre. Continuous au intensity estimation using localized, sparse facial feature space. In *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, pages 1–7, April 2013. doi: 10.1109/FG.2013.6553808.

Bo-Kyeong Kim, Jihyeon Roh, Suh-Yeon Dong, and Soo-Young Lee. Hierarchical committee of deep convolutional neural networks for robust facial expression recognition. *Journal on Multimodal User Interfaces*, pages 1–17, 2016.

Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 94–101. IEEE, 2010.

Michael Lyons, Shota Akamatsu, Miyuki Kamachi, and Jiro Gyoba. Coding facial expressions with gabor wavelets. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 200–205. IEEE, 1998.

Caifeng Shan, Shaogang Gong, and Peter W McOwan. Facial expression recognition based on local binary patterns: A comprehensive study. *Image and Vision Computing*, 27(6):803–816, 2009.

S. Velusamy, H. Kannan, B. Anand, A. Sharma, and B. Navathe. A method to infer emotions from facial action units. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2028–2031, May 2011. doi: 10.1109/ICASSP.2011.5946910.

Anbang Yao, Junchao Shao, Ningning Ma, and Yurong Chen. Capturing au-aware facial features and their latent relations for emotion recognition in the wild. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 451–458. ACM, 2015.

Zhiding Yu and Cha Zhang. Image based static facial expression recognition with multiple deep network learning. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 435–442. ACM, 2015.