UCI University of California, Irvine

Emotion Classification through Deep Learning

Matthew Stanfield
George Gabricht
Caleb Pitts
Daniel Tseng

UCIrvine

Project Roadmap

- Abstract & Overview
- Datasets & Augmentation
- Model Structure
- Results & Improvements
- Conclusion

Detecting Human Emotions

The relatively easy task of emotion detection is non-trivial for a computer.

https://media.gettyimages.com/photos/sad-man-in-a-light-blue-tshirt-on-a-white-background-picture-id184918973

https://www.news.ucsb.edu/sites/default/files/styles/article_horizontal/public/images/2014/angry%20face.jpg?itok=w8rAIJs-

https://image.shutterstock.com/image-photo/woman-smiling-perfect-smile-white-260nw-268932410.jpg

Abstract

- For our project, we wanted to build a system that could take in webcam feed of a person's face and classify the emotions the person is feeling.
- We knew that to accomplish this goal, we needed to learn from the similar approaches to the problem in the past.
- Our system is capable of extracting facial images from webcam feed and classifying an emotion based on the image's facial features.
- We accomplished this goal with the help of Convolutional Pooling, Data Augmentation, and extensive trials of various models and parameters.
- In the end, our model consistently earns a 95% accuracy on Test Data and performs fairly accurately on live data.

Overview

Can we train a neural network to predict an emotion from a picture?

Neural Network

Happy

https://content.thriveglobal.com/wp-content/uploads/2018/01/Happy_guy.jpg

Project Roadmap

- Abstract & Overview
- **Datasets & Augmentation**
- Model Structure
- Results & Improvements
- Conclusion

Our Datasets for the Problem

FER2013

- ~ 36,000 images
- These images span 6 Basic Expressions + Neutral
- The dataset images are size (48, 48)
- This dataset proved most useful during training.

RAF DB

- ~ 30,000 images
- These images span 6 Basic Expressions as well as 12 Compound Expressions
- The dataset images are size (100, 100).
- This dataset was too small for its large # of expressions.

EXP-W

- ~ 90,000 images
- These images span 6 Basic Expressions + Neutral
- The dataset consisted of many pop culture pictures.
- This dataset proved to be less useful during training.

FER2013 Contents

FER2013 contains 35887 grayscale pre-cropped 48x48 images classified in 7 classes

Images per Class

Happy	Neutral	Sad	Fear	Angry	Surprised	Disgust
9073	6178	6051	5262	4960	3846	517

Happy or Angry?

Preprocessing and Data Augmentation

- In order to accurately process images through convolutional neural networks, a little bit of preprocessing is necessary.
- Given an image:
  - first the face must be isolated and cropped.
  - next the image must be resized to fit the shape of the training set (48, 48).
  - lastly, each image must be normalized between 0 and 1 (instead of between 0 and 255).
- However, a model is only as strong as its data, and preprocessing only gets you so far.
- Many datasets, including the ones we used, are very limited in size.
- To combat this issue, data can be augmented to produce similar data to help train the model.
- This works especially well w/ images, where simple operations yield valid new data.

Image Augmentation for Limited Size Datasets

Original Image

- Initially, to augment our dataset, we simply mirrored our training set, instantly doubling the images.
- Keras ImageDataGenerator is a valuable tool we discovered that allows you to specify margins for rotation, zoom, etc.
- This tool generates a potentially infinite number of images augmented from the original.

Generated Image

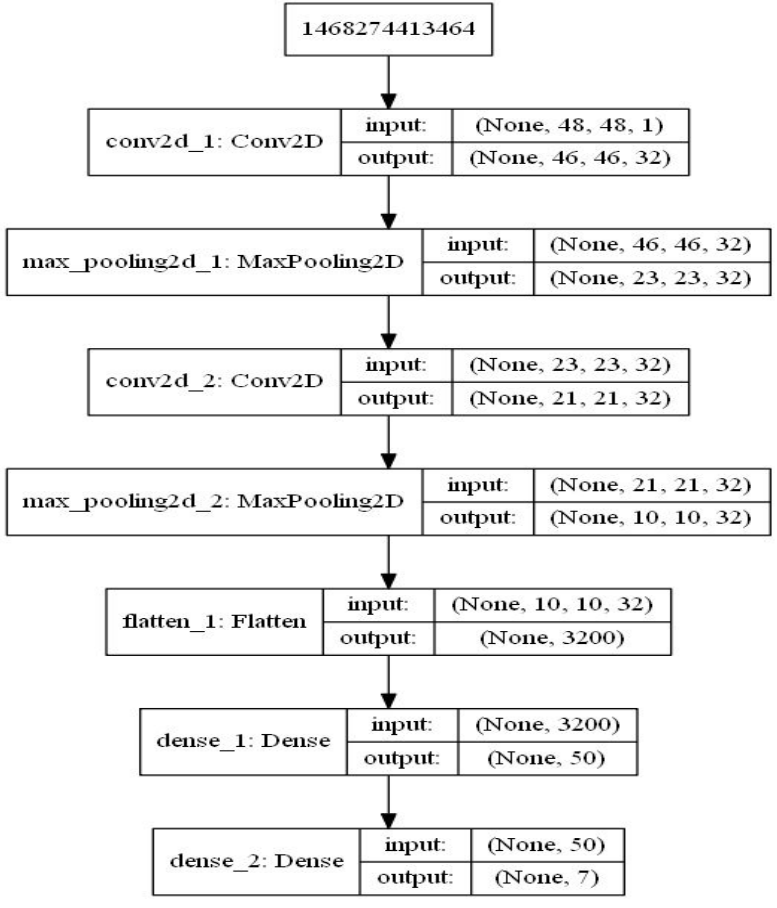
- The image to the right is an example image generated from the image above using ImageDataGenerator.

Project Roadmap

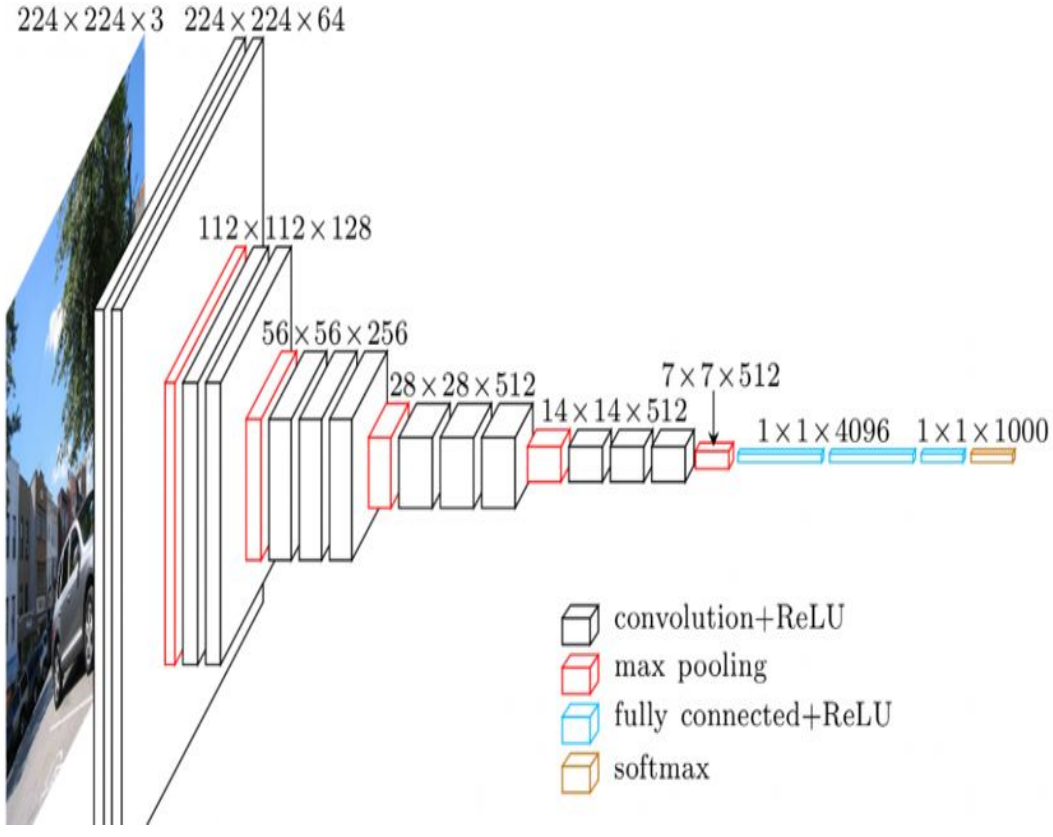
- Abstract & Overview
- Datasets & Augmentation
- **Model Structure**
- Results & Improvements
- Conclusion

Inspiration

- To get a basic idea of where to begin we implemented:
  - A model very similar to the Sequential MNIST model presented earlier this quarter.
  - A baseline model very similar to those used in FER2013 Kaggle Competitions.
- The diagram to the right is a commonly used model for the MNIST dataset.
  - From such models, we were able to learn techniques to improve our model.
  - One such technique, which proves to be very powerful, is Convolutional Pooling



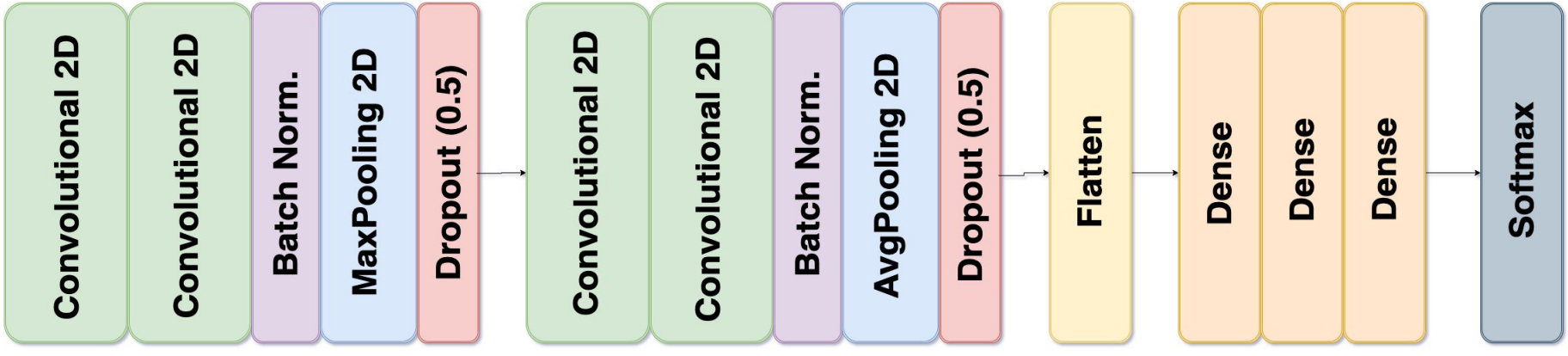
Convolutional Pooling



- Convolutional layers in a neural network summarize the presence of features in an input image.
- One issue with output feature maps is that they are biased towards the location of features.
- A common approach to address this issue is to down-sample the feature maps.
- Pooling layers provide an approach to down-sampling by summarizing the presence of features.

Our Model

- Our final model consists of the 15 layers illustrated below.
- Convolutional Pooling was used for feature extraction and down-sampling.
- Dropout was used with a rate of 0.5 to combat overfitting.
- After we flatten the images, 3 dense layers are used to facilitate feature learning, and an emotion is predicted through our softmax output layer.

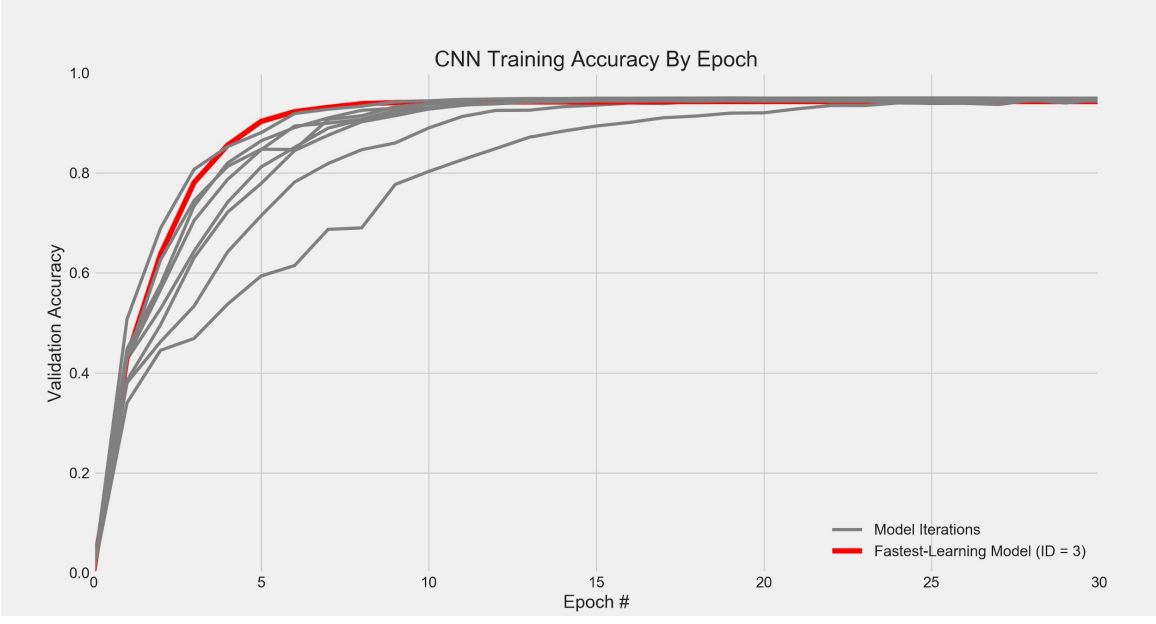


Project Roadmap

- Abstract & Overview
- Datasets & Augmentation
- Model Structure
- **Results & Improvements**
- Conclusion

Model Iterations Analysis

- We cross-validated 9 different models
  - across 5 folds
  - 30 epochs each fold
- All converged ~ 92% accuracy
- Model 3 learned the fastest, achieving ~ 90% accuracy after only 5 epochs



Model Results

❖ Confusion matrix (in counts)

	Predicted Angry	Predicted Disgust	Predicted Fear	Predicted Happy	Predicted Sad	Predicted Surprised	Predicted Neutral
Actual Angry	877.2	0.8	15.2	21.8	32.6	8.6	24.2
Actual Disgust	3.2	89.6	2.8	3.6	3.0	1.4	2.6
Actual Fear	16.0	1.2	911.0	19.8	29.8	21.2	26.6
Actual Happy	19.4	0.6	18.6	1680.0	19.8	10.8	25.6
Actual Sad	26.0	1.2	22.6	26.8	1100.4	7.8	33.6
Actual Surprised	6.0	0.4	16.6	14.2	6.8	771.6	12.2
Actual Neutral	13.6	0.2	18.0	32.4	31.0	11.6	1137.4

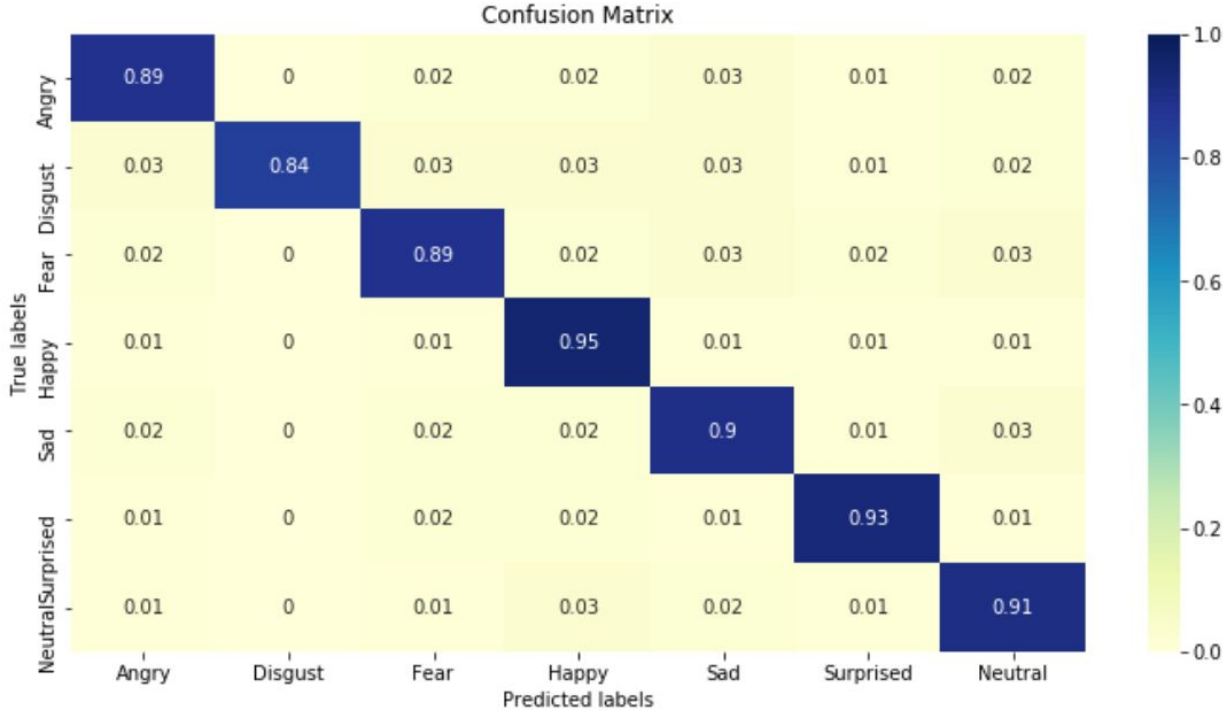
❖ Error by emotion label

	Error
Predicted Angry	0.09
Predicted Disgust	0.05
Predicted Fear	0.09
Predicted Happy	0.07
Predicted Sad	0.10
Predicted Surprised	0.07
Predicted Neutral	0.10

❖ **Average testing accuracy is 91.5%**

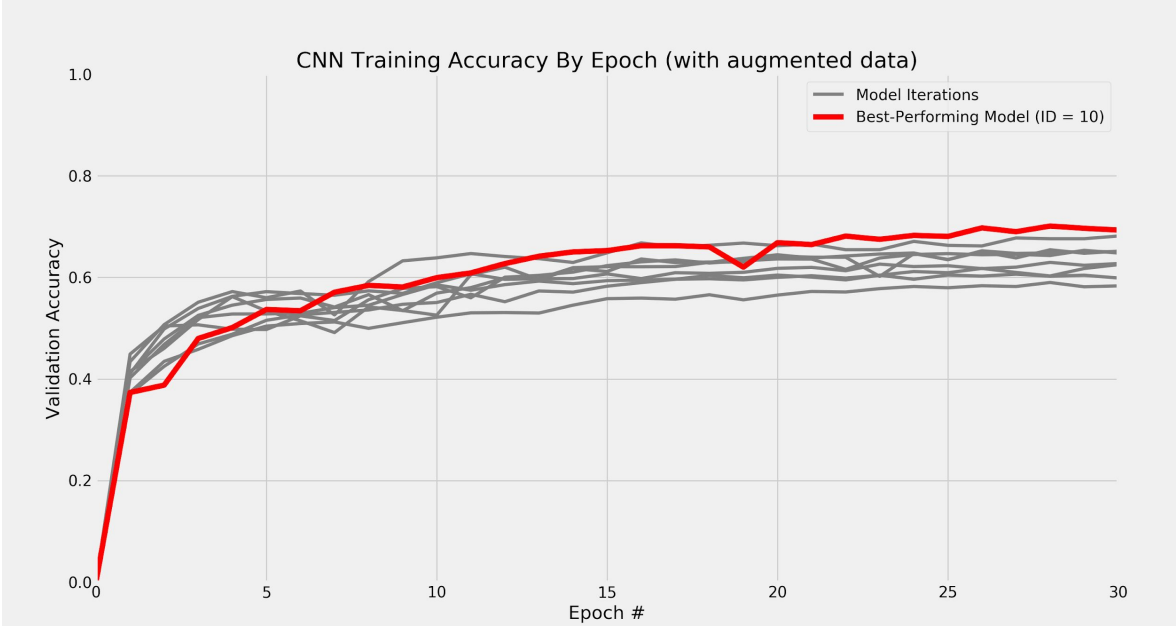
Model Results

- Confusion matrix (in proportions)



Testing our Models with Other Datasets

- Our model performs very well when predicting images similar to the FER2013 dataset.
- However, we did not perform so well when tested against other datasets.
- Furthermore, our models all seemed to perform poorly on combined datasets with generated training sets, shown to the right.



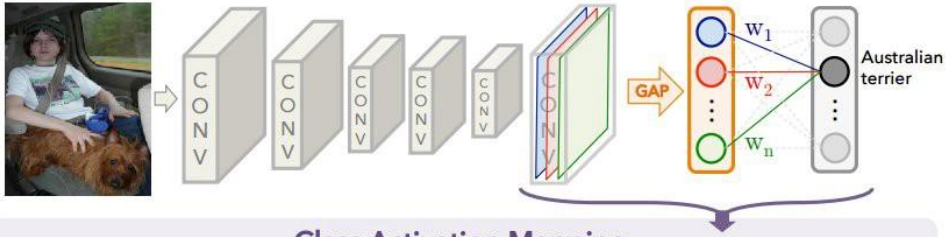
Project Roadmap

- Abstract & Overview
- Datasets & Augmentation
- Model Structure
- Results & Improvements
- **Conclusion**

Next Steps: Class Activation Maps

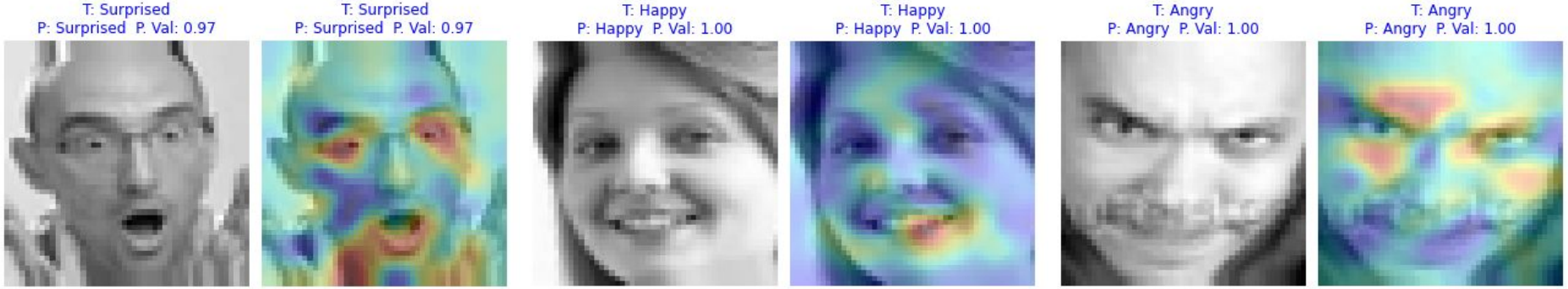
- Class Activation Maps made by finding which filters are important
- Trends occur in data for specific emotions.
- These trends can be used to improve on our model in the long run.

Learning Deep Features for Discriminative Localization



https://www.di.ens.fr/~josef/publications/Oquab15.pdf

https://arxiv.org/abs/1512.04150



Focus on eyes and Mouth

Focus on mouth and teeth

Focus on area around eyes

More Class Activation Maps

Demo: Camera Implementation

Summary

- Our goal was to classify people's emotions through images with a decent accuracy in real time.
- Our model achieves this goal with a 91.5% accuracy and performs fairly well on real time video.
- We utilize Convolutional Pooling as well as Keras ImageDataGenerator to help train our model for a wider range of users.
- For future projects, we would exploit Class Activation Maps, tackle the issues with combining our datasets.
- One possible feature to facilitate video support would be to add LSTM to our model and factor in some features from the last several frames.

Questions?

Thank You

Class Activation Maps NN Structure

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_12 (Conv2D)           (None, 48, 48, 64)        640
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 48, 48, 64)        36928
_________________________________________________________________
spatial_dropout2d_4 (Spatial (None, 48, 48, 64)        0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 24, 24, 64)        0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 24, 24, 128)       73856
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 24, 24, 128)       147584
_________________________________________________________________
spatial_dropout2d_5 (Spatial (None, 24, 24, 128)       0
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 12, 12, 128)       0
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 12, 12, 128)       147584
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 12, 12, 128)       147584
_________________________________________________________________
global_average_pooling2d_2 ( (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 903
=================================================================
Total params: 555,079
Trainable params: 555,079
Non-trainable params: 0
_________________________________________________________________
```