DS340 Final Project Emotion Classification

Matthew Batacan & Daniel Skahill

Code Base: [Here](https://drive.google.com/drive/folders/1AR2lWJOIurJVRiCBHUIgCsiTX6hYcMQU?usp=sharing) (Notebook with all code can be found under the folder labeled “Final Code”)

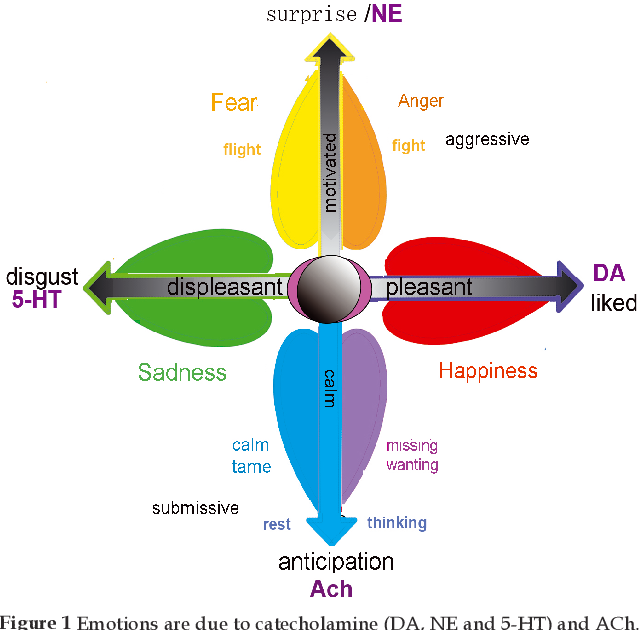
December 8th, 2022

1. **Introduction** - *What is the problem you were trying to solve and why did you want to solve it? Also, include any background information necessary to understand the problem (such as explaining sports statistics used later in the paper).*

Neural Networks have recently become the leading technology for computer vision tasks because they are good at classifying images, detecting objects, and abstracting features. Because they are really good at facial recognition and identifying facial features, the objective of our project was to see if neural networks could detect emotion from facial expressions. After doing research, we found that while similar projects had been completed before, they had yet to achieve stellar accuracy. We hoped that in this project we could achieve accuracy on par with or above what had been accomplished already.

Emotion recognition carries implications for various fields. One field of interest is psychology because “deficits in emotion recognition are associated with a range of psychiatric conditions ”(Banziger). Another potential use for emotion recognition technology could be sales or customer service to see how a customer responds to a product or the help they are receiving. It could also help companies determine how positive a user experience is in a physical or virtual environment.

In terms of the data used, we originally started our experimentation using a Kaggle FER (Facial Emotion Recognition) dataset which came with around 24,000 48x48 images belonging to 7 classes. Later we turned our focus to using a higher resolution dataset (224x224) from AffectNet (Mollahosseini) which came with over 200,000 images belonging to 8 classes. In both of these datasets, certain emotions such as contempt, disgust, and surprise had very few observations compared to the others. According to a paper from the International Journal of Neurology Research, specific neuromodulators in the brain trigger emotions, and when plotted on a 2D plane, there are four main quadrants of emotions: happy, sad, anger, and fear (Gu) (Figure 1). So, to narrow our focus groups and eliminate bias from a small sample size, these are the classes we chose (in addition to neutral).



1. **Methodology** - *What did you do? Include details like the parameters of machine learning here.*

As described above, our first step was to acquire data. We discovered a dataset from Kaggle which offered about 24,000 48x48 training images belonging to 7 different emotion labels. After we removed emotion labels, we were left with about 20,000 images with a relatively even distribution.

We began our testing on this dataset, trying out some common architectures that we had researched, such as VGG-16, but we didn’t receive very strong results. We hypothesized that there were perhaps too many layers that would cause the network to look at features that were insignificant for emotion detection. However, the simple architectures we created ourselves also didn’t work well, so we needed to search for the right balance. One drawback that we found with the initial Kaggle data was that there was not a lot of diversity. Most of the people in the images were of Caucasian descent, which we knew could lead to bias and inaccuracy. The dataset also contained some hand drawings of certain emotions which could also lead to bias. Since the purpose of our neural network was to be able to use it on real people, we wanted to find a better dataset that met our criteria.

The next dataset we found to train on, was the FER2013 dataset. This dataset consisted of around 31,000 images after reducing to our 5 emotion labels. We found that this data was more diverse than our initial dataset.

| Model 1: Configuration (Trained on Kaggle Dataset, FER2013, and AffectNet)      Input: 48x48 grayscale images belonging to 5 classes  Optimizer: Adam  Learning rate: .001  Learning rate and optimizer were chosen after running the network with RMSprop, Adam, SGD along with learning rates ranging from [.1, .0001] and finding that Adam with a learning rate of .001 were optimal for this architecture.  *\*all layers are CONV2D except for the final 2 which are dense layers* |
| --- |

We began searching for a higher resolution dataset with more diversity. Research led us to the AffectNet (Mollahosseini) database–which we had to apply for use–and we were fortunate enough to gain access to it. The dataset was not in an acceptable format for Keras to train on, so we first had to do some basic manipulation of the labels and image directories. While the total database contains about 200,000 images across 8 emotion labels, nearly two-thirds of those are from the “happy” and “neutral” categories. Because we didn’t want to bias our results, we selected a subset of the database for a total of about 48,000 images (much more evenly distributed).

We performed the majority of our testing using this dataset, and when we retrained the first model, it received a significantly higher accuracy than when run with the initial Kaggle dataset. Throughout our neural network testing, we tried many different architectures, but the Venturi model (Verma)–which was adapted from a research paper–and 32-512 model were determined to be the best (shown below). For each model, we experimented with multiple different learning rates and optimizers and concluded that for the Venturi and 32-512 models, RMSprop was the best optimizer with a learning rate of .001.

| Model 2 Configuration (Trained on AffectNet)    96x96 512 256 128 128 256 512 256 5    Input: 96x96 grayscale images belonging to 5 classes  Optimizer: RMSprop  Learning rate: .001  *\*all layers are CONV2D except for the final 2 which are dense layers* |
| --- |
| Model 3 Configuration (Trained on AffectNet)    224x224 32 64 126 256 512 1000 5    Input: 224x224 grayscale images belonging to 5 classes  Optimizer: RMSprop  Learning rate: .001  *\*all layers are CONV2D except for the final 2 which are dense layers* |

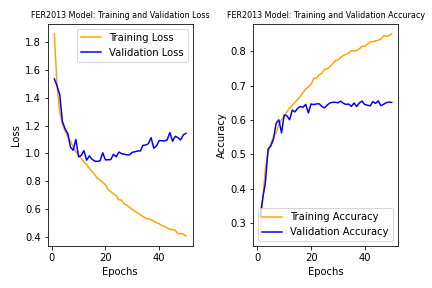
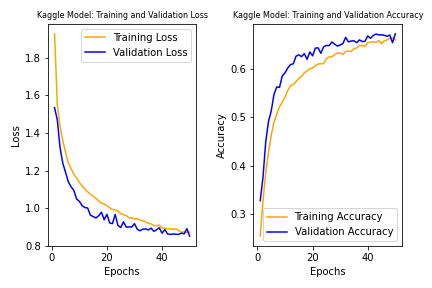
Once we had finalized our models, we began to work on implementing our neural networks with videos. The aim was to be able to use the trained neural network to show how many times emotions changed throughout a video. We utilized OpenCV’s CV2 library to open videos as a sequence of image frames, detect faces in each frame, and crop the frame to the specified size for the neural network. Once we had the cropped face sequence, we used our neural network to predict the emotion in each frame (because the accuracy of the neural network was about 70%, we chose to look at every 8th frame to predict and made the assumption that emotion would last at least 8 frames). After the predictions were made, we translated them into statistics and a Gantt chart to show each emotion's relative duration and time of occurance.

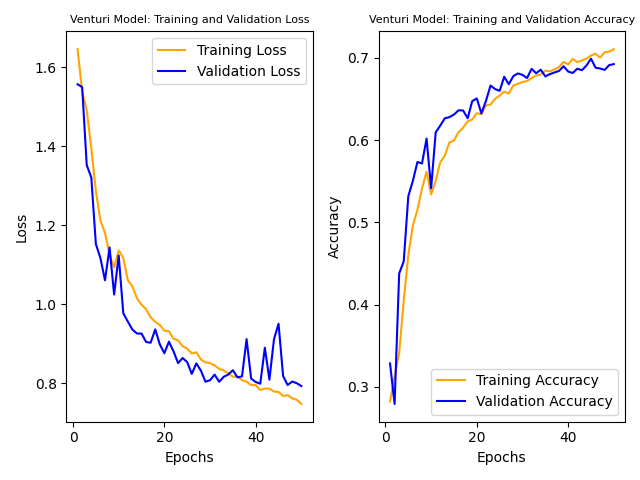
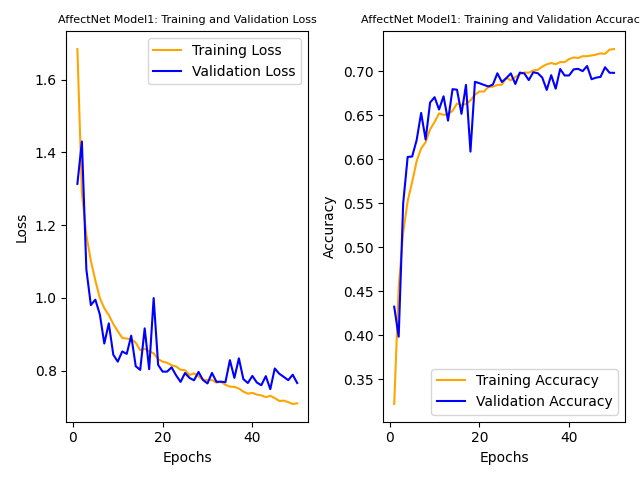
In the live stream extension of the project, we utilized many of the same components from our video analysis. Instead of the input being a video, we took input from the webcam and predicted emotions in real time. To do this, we first had the webcam running, and then used OpenCV’s Haar cascades to take in the webcam image and detect the face (Wortherspoon). To test if the face detection was working with input from the webcam, we implemented a test where we could take a picture using the webcam, have it detect the face, and then predict an emotion. To keep it flexible while testing which models worked the best, we took into consideration the need to be able to resize the images to fit the model input size. After achieving success with images from the webcam, we extended it to show the live prediction. One thing to note about this is that the detection is a couple frames behind real-time because it needs to detect the face before being able to predict the emotion and display it.

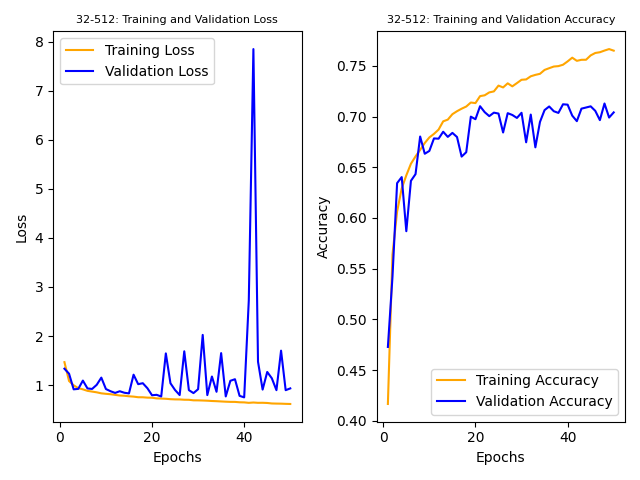
Lastly, we experimented with an ensemble method that would take the max probability of predicted emotion or the highest weighted average emotion between the models. However, we were not able to do lots of fine-tuning on the ensemble method and did not have the time permitted to further improve this method.

1. **Results** - *What were your results? However you evaluated your system, put the results here.*

Here are some graphs that depict the Training and Validation accuracy and loss of our neural networks.





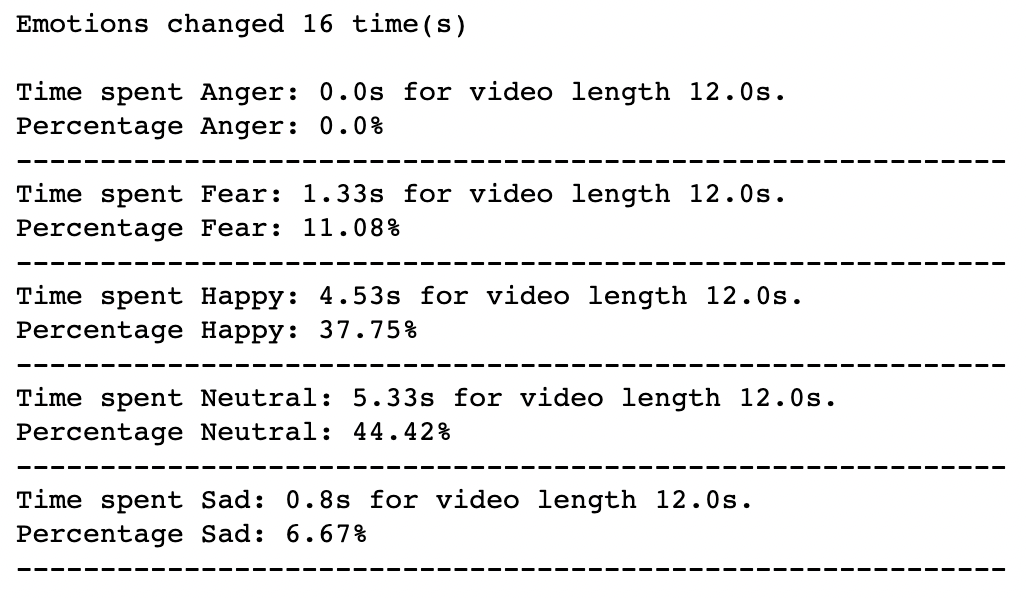


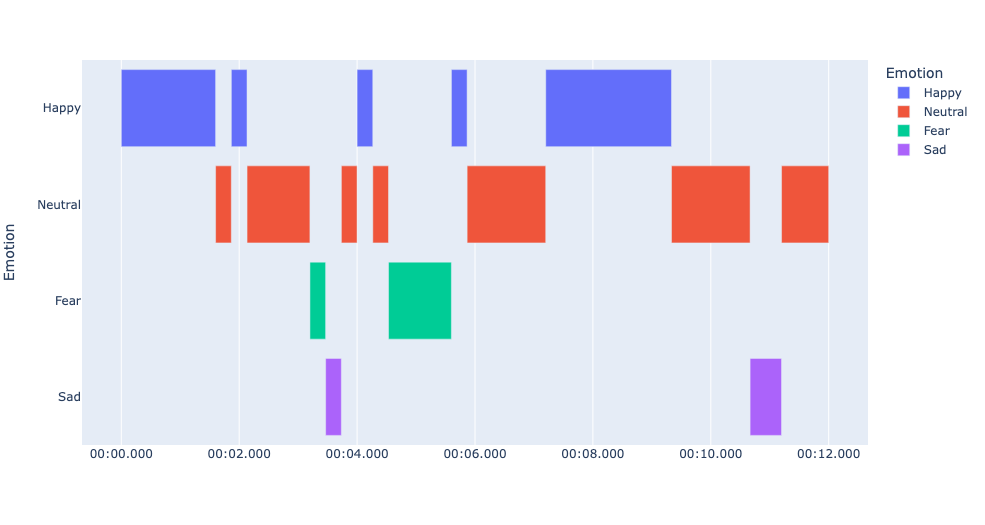
*The first 3 graphs show model 1 trained on the 3 datasets. 4&5 show models 2&3 respectively*

As evident by the graphs, we achieved an accuracy of about 70% for three of our models, however the loss varied from model to model. Ultimately, the first model presented in this paper–trained with the AffectNet data–produced the best results. The third model (32-512) achieved an accuracy slightly higher, but the loss was significantly greater and not as consistent. When trained and tested on the FER2013 dataset, the first model did worse than the others, which emphasizes the importance of quality data.

| **Results** | | | | |
| --- | --- | --- | --- | --- |
| **Model** | **Loss** | **Accuracy** | **Image Resolution** | **Dataset** |
| 1 | .8506 | .6731 | 48x48 | Kaggle |
| 1 | 1.1442 | .6516 | 48x48 | FER2013 |
| 1 | .7661 | .6981 | 96x96 | AffectNet |
| 2 (Venturi) | .7924 | .6922 | 96x96 | AffectNet |
| 3 (32-512) | .9326 | .7041 | 224x224 | AffectNet |

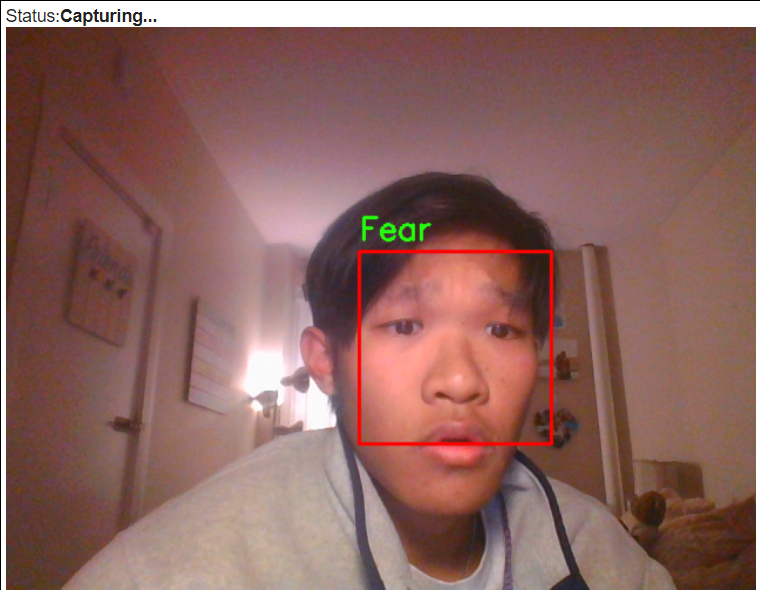
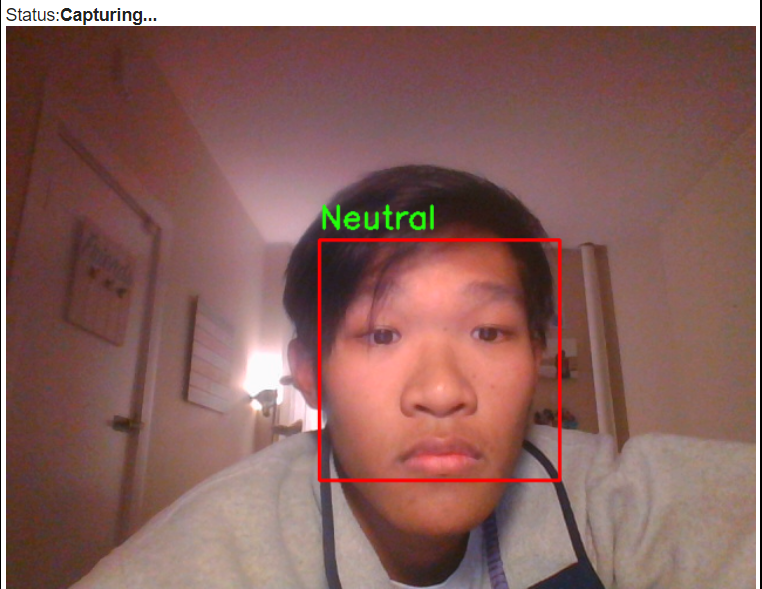
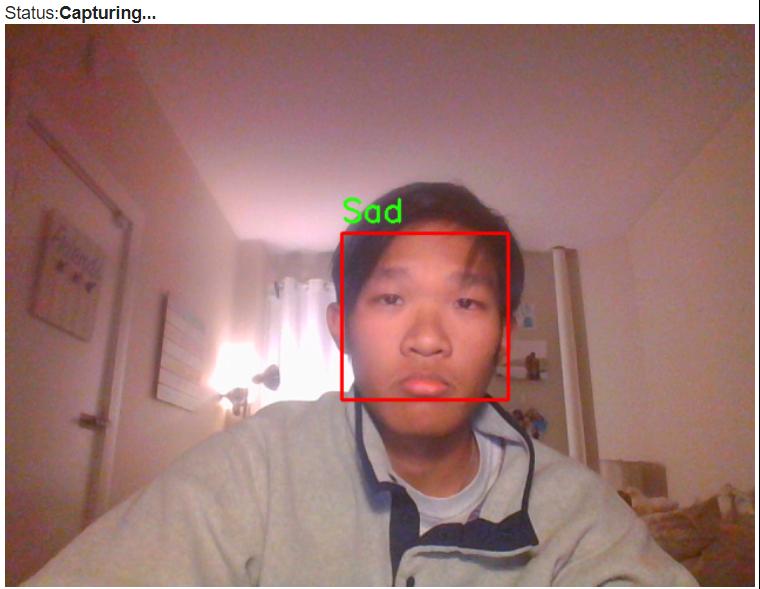
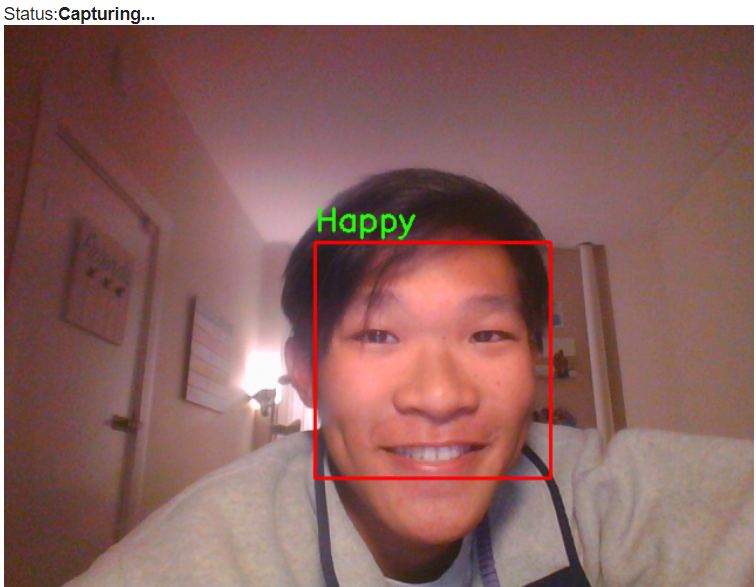
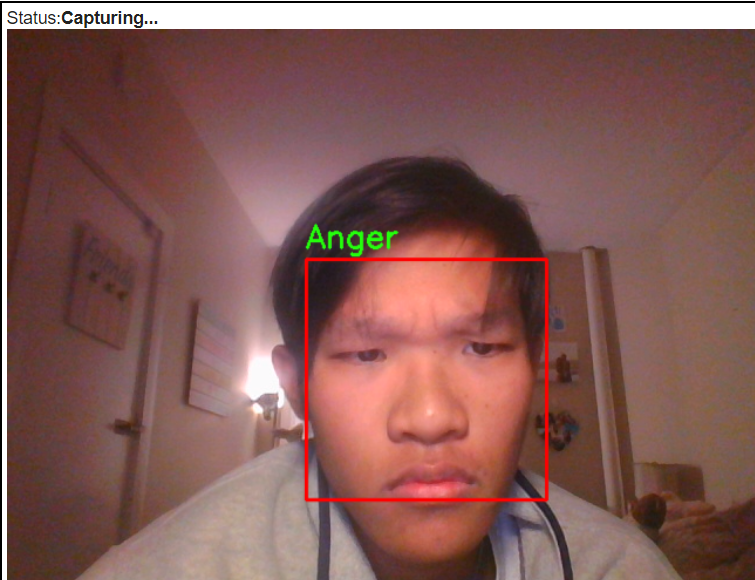
In connecting our model to predict emotions from a video, we input a video like the one found [here](https://drive.google.com/file/d/1wvlicLpSBvF0AscZQZe6w8sCapxyvyHs/view?usp=sharing), and output the following graph along with some statistics about each emotion.





When observing the video and gantt chart side by side, we can see that the analysis does a relatively good job of matching up the predicted emotions with actual emotions at roughly the right time; however, one emotion it missed in this example was “Anger.” The accuracy of the video analysis is dependent upon OpenCV’s facial detection module finding the correct face frame, so if any errors occur in that step, it will affect the statistics as well. In general, training the network on photos and extending it to video yielded similar results with ~70% accuracy (no formal accuracy tests were completed, this is purely based on estimation and trials).

Our final venture with the trained models was to connect them to a live video feed so that we could predict emotions in real-time (we expected there might be a slight lag). We were successful in hooking up the model to a webcam and displaying the emotion predicted for that moment on screen, but, the weaknesses of each model were especially evident in the live evaluation as we could see that certain models struggled to predict specific emotions. While some of the models were constantly changing the predicted emotion, others found it hard to predict outside of 2 or 3 emotions.



1. **Conclusions** - *What did you conclude from your results? Is there anything else your team learned from the experience?*

Overall, we are happy with the results of our project as we were able to achieve all three goals that we had set for ourselves at the start: scoring a respectable accuracy in detecting facial emotions, connecting the neural network to predict videos, and predicting emotions from a live feed. The success of our goals did not come without challenges as we were quick to learn the importance of quality data, parameters, and optimizers when working with neural networks. We also discovered that creating a neural network architecture from scratch is very difficult, and it is often more beneficial to research tested architectures and use them as a starting point.

Obtaining quality data is critical for AI projects as they are the backbone for building neural networks and machine learning infrastructure. Finding quality data requires effort as certain criteria cannot be sacrificed--especially when it comes to diversity, credibility, and accuracy. Without diverse, accurate, credible data, any abstractions made can easily be questioned and proved insignificant.

Parameterization and utilization of the correct optimizer are also important in building neural network architectures, and they can have a significant impact on the accuracy and loss of a model even if the structure itself is adequate. Running tests with different combinations of the two can greatly improve the performance of a model.

While we hoped to achieve an accuracy of around 80%, our best models were still 3.5 times stronger than random guessing. Much of the process involved trial and error, but along the way we learned many valuable lessons about developing neural networks and machine learning models. In connecting our models to videos and a live web cam, we were able to observe the power of utilizing models beyond collecting and reporting metrics. Our final takeaway is that emotion detection from facial expressions is a complicated task. People express emotions in unique ways, so there is a lot of variation across emotion categories. This variation makes it difficult for an AI to generalize as there are nuances to each individual’s expression of emotion.

***Contributions:*** *We both did an equal amount of work throughout this project and developed the final product together.*

Works Cited

Akhand, M. A. H., et al. “Facial Emotion Recognition Using Transfer Learning in the Deep

CNN.” Papers With Code,

https://paperswithcode.com/paper/facial-emotion-recognition-using-transfer.

Banziger, Tanja. “Measuring Emotion Recognition Ability.” *SpringerLink*, Springer Netherlands,

1 Jan. 1970, https://link.springer.com/referenceworkentry/10.1007/978-94-007-0753-5\_4188#:~:text=Emotion%20recognition%20has%20important%20implications,bipolar%20disorder%2C%20major%20depression).

Brownlee, Jason. “How to Perform Face Detection with Deep Learning.”

*MachineLearningMastery.com*, Machine Learning Mastery, 23 Aug. 2020, https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/.

Gu, Simeng, et al. “Differentiation of Primary Emotions through Neuromodulators: Review of

Literature.” *International Journal of Neurology Research*,

http://www.ghrnet.org/index.php/ijnr/article/view/986/1266.

Khaireddin, Yousif, and Zhuofa Chen. “Facial Emotion Recognition: State of the Art

Performance on FER2013.” ArXiv.org, 8 May 2021, https://arxiv.org/abs/2105.03588.

Liu, Junxiu, et al. “EEG-Based Emotion Classification Using a Deep Neural Network and Sparse

Autoencoder.” Frontiers, Frontiers, 12 June 2020,

https://www.frontiersin.org/articles/10.3389/fnsys.2020.00043/full\.

Verma, Abhishek, et al. “Modified Convolutional Neural Network Architecture Analysis for

Facial Emotion Recognition.” *IEEE Explore*,

https://ieeexplore.ieee.org/abstract/document/8787215.

Wotherspoon, Jack, director. *How to Use Webcam In Google Colab for Images and Video (FACE*

*DETECTION)*. *YouTube*, YouTube, 15 Dec. 2020, https://www.youtube.com/watch?v=YjWh7QvVH60. Accessed 7 Dec. 2022.

Datasets

Mollahosseini, A. et al, "AffectNet: A Database for Facial

Expression, Valence, and Arousal Computing in the Wild," in IEEE

Transactions on Affective Computing, 2017, <http://mohammadmahoor.com/affectnet/>

<https://www.kaggle.com/datasets/ananthu017/emotion-detection-fer>

<https://www.kaggle.com/datasets/deadskull7/fer2013/code>