Image-based Face Detection and Facial Expression Recognition An Application with Deep Learning

Hongji Yuan HJYUAN@SEAS.UPENN.EDU
Dat Nguyen DATNGUYEN@SEAS.UPENN.EDU
Tuan Nguyen TMNGUYEN@SEAS.UPENN.EDU

Abstract

Face detection and facial expression recognition have many potential benefits in healthcare, automotive, gaming, and many other industries. In this paper, we implemented a face detection model using *Detectron2* [1] and several different machine learning algorithms such as k-nearest neighbors (”KNN”), multi-layer perceptron (”MLP”), and convolutional neural net (”CNN”) for the facial expression recognition. Our models are trained on two separate image-based data sets. We were able to achieve a 70% AP50 with our face detection model and our best facial expression recognition model achieved an accuracy of 62%.

1. Motivation

Not long ago facial detection and facial expression recognition seemed like a sci-fi fantasy but today we can unlock our phone by just looking at it. The idea of being able to detect human faces has long been studied, with its roots formed in the 1960s, when Woodrow Wilson Bledsoe developed a system of measurement to classify photos of faces. Throughout the centuries, new technologies were invented to improve the ability to locate a face and then identify different features which paved the way for modern automated systems. Facial detection, while having many benefits such as enhanced security with facial bio-metrics and automation of identification, is also the first step to perform facial expression recognition.

Human emotion detection has a critical part in the interpersonal relationships. Emotions can generally be extracted from speech, hands and gestures of the body as well as through facial expressions. Being able to understand human emotions could improve the quality of the communications between human and machine. Furthermore, there is a wide range of industries that could benefit from emotion recognition such as healthcare, automotive, gaming, and many more.

The difference between object detection algorithms and classification algorithms is that in detection algorithms, we try to draw a bounding box around the object of interest (in this case, human face) to locate it within the image. Also, there could be many bounding boxes representing different faces in the image and the number of faces are unknown. To make the problem even more complicated, a human face is capable of expressing multiple emotions at the same time, making it challenging to detect even for a human being. Machine learning attempts to solve this by using different algorithms to extract certain features from the pixels of the image to come up with a prediction.

While there are about 27 different human emotions, in this project, we plan to work with labeled data sets with seven distinct human emotions: happiness, sadness, fear, anger, surprise, disgust, and neutral. We aim to implement a two-step model: first, localize human faces in an image and second, recognize emotion expressed in those faces.

2. Related Work

2.1. Face Detection

There have been many researches and studies around face detection with many proposal methods and techniques. Yan et al. [2] classified single image detection methods into four main categories, with the caveat that a face detection algorithm could fall into two or more of those groups.

- Knowledge-based methods
- Feature Invariant approaches
- Template Matching methods
- Appearance-based methods

Knowledge-based method depends on the set of rules, and it is based on human knowledge to detect the faces. For

University of Pennsylvania, CIS 520 Course Project.

example, a face must have a nose, eyes, and mouth within certain distances and positions with each other. Kouzani et al. [3], in their paper "Commonsense knowledge-based face detection", proposed a system that could detect 89% of the faces in the image database.

Figure 1: The detected human faces in an input image.

A downside of this methods is the difficulty in building an appropriate set of rules. There could be many false positive if the rules were too general or too detailed. This approach alone is insufficient and unable to find many faces in multiple images.

Feature-based method locates faces by extracting structural features of the face. It is first trained as a classifier and then used to differentiate between facial and non-facial regions. The idea is to overcome the limits of our instinctive knowledge of faces. This approach divided into several steps and even photos with many faces they report a success rate of 94%. According to Yan: "One problem with these feature-based algorithms is that the image features can be severely corrupted due to illumination, noise, and occlusion. Feature boundaries can be weakened for faces, while shadows can cause numerous strong edges which together render perceptual grouping algorithms useless."

Appearance-based methods rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face and non-face images. The learned characteristics are in the form of distribution models or discriminant functions that are used for face detection. The appearance-based model further divided into sub-methods for the use of face detection, some of which are as follows: Eigenface-based, Disitrbution-based, Neural Networks, Support Vector Machine, Hidden Markov Model, Inductive Learning, etc.

2.2. Facial Expression Recognition

KNN is an intuitive and easy-to-implement machine learning algorithm that can be used to solve many classification problems. KNN had also been used for facial expression recognition [4] which uses physiological signals as the predictors to recognize both basic and complex emotions. The physiological signals are extracted using the neural network proposed in Tayari-Meftah et al. research [5] which consists of three main modules: psychological module, formal computational module, and recognition module.

CNN While facial expression recognition is a classical classification problem where the data instance is classified into categories (emotions) based on the input features (image pixels), it could be a very challenging task for classical machine learning methods due to the high dimensionality of the image inputs. In deep learning, a convolutional neural network is commonly applied to analyzing images [6]. Many researchers have developed CNN-based methods [7; 8] for human facial expression recognition.

Transfer Learning The convolutional neural networks have successfully established many models for image classification, but it requires a lot of training data. Instead of training the entire CNN from scratch, it is common to use a pre-trained model on a very large dataset either as an initialization or a fixed feature extractor for the task of interest[9]. For our facial expression recognition task, we created one of our model by using the ResNet [10] pre-trained model as an initialization and fine-tuned the model using the FER-2013 dataset.

3. Dataset

3.1. Dataset 1

Our first data set ('Dataset 1'), which is provided and maintained by Dataturks, has about 400 images with a bit more than 1000 faces. The data is available as a JSON file with 2 main components, an URL address of the image, and the face labels and bounding boxes of that image. We extracted those information from the JSON file, made an annotate function, and showed the images with the correct bounding boxes for the faces.

Figure 2: Sample images from Dataset 1.

3.2. Dataset 2

For facial expression recognition task, we used the FER-2013 dataset [11]. The data consists of 48x48 pixel gray-scale images of human faces. The faces have been processed to be centered in the image and occupies about the same amount of space in each image. Each face is labeled as one of seven categories (Angry, Disgust, Fear, Happy, Sad, Surprise, and Neutral). Example images are shown in Figure 3. For the training process, we normalized the images to (-1, 1) range.

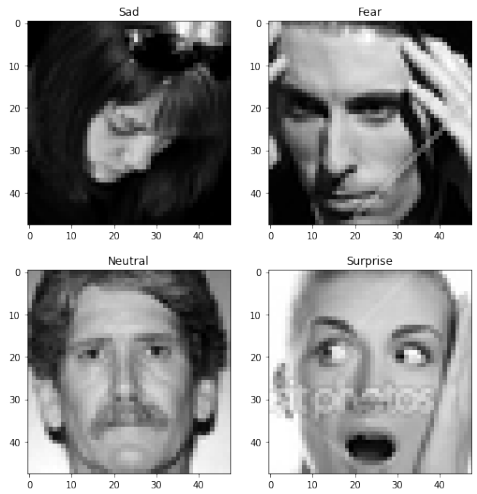


Figure 3: Example images with emotion labels.

4. Problem Formulation

Our goal is to identify facial expressions from images. We separated the problem into two parts.

Face Detection The first part is human face detection which involves several object detection algorithms. The difference between object detection algorithms and classification algorithms is that in detection algorithms, we try to draw a bounding box around the object of interest (in this case, human face) to locate it within the image. Also, there could be many bounding boxes representing different faces in the image and the number of faces are unknown. Because the number of occurrences of the faces is not fixed, we cannot use a standard CNN followed by a fully connected layer. A naive approach to solve this problem would be to take different regions of interest ("RoI") from the image, and use a CNN to classify the presence of the face within that region. The problem with this approach is that the faces might have different spatial locations within the image and different aspect ratios. Hence, we would have to select a huge number of regions and this could computationally blow up very fast.

To bypass the problem of selecting a huge number of regions, Ross Girshick et al. [12] proposed a method where we use selective search to extract just 2000 regions from the image which are called region proposals. Instead of trying to classify a huge number of regions, we can work with only 2000 regions. The same authors came up with an algorithm called Fast R-CNN that is significantly faster in training and testing over R-CNN. Both R-CNN and Fast R-CNN use selective search to find out the region proposals, which could be slow and time-consuming, affecting the performance of the network.

To address this issue, Shaoqing Ren et al. [13] proposed a new algorithm called Faster R-CNN. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

We implemented a pre-trained model from Detectron2 Model Zoo called Faster R-CNN - X101-FPN model to extract the co-ordinations of the bounding boxes. Due to the scope of this project, we won't go into too much details about the architecture of the X101-FPN model and just use it as a black box.

Facial Expression Recognition The second part is a supervised classification to perform emotion detection using extracted faces. We trained our own emotion detectors as well as using state-of-the-art models, so that we could build a connection between what we know vs. what is being used in practice. Since this is a classification problem, the most intuitive approach would be to use KNN which was chosen as our baseline model. To optimize KNN, not only different configurations of model hyperparameters were evaluated, but different data representations were also considered. This is because how we choose to represent the face images does have an impact on distances between data points. Moreover, because most image-processing applications use neural network models, we tried different types of feed-forward neural networks such as MLP and CNN as well as evaluate different loss functions like ReLU, Sigmoid and Tanh. Our goal is to determine a neural network's configuration that can extract the most emotion-related features from a face image.

5. Methods

As described in *Problem Formulation*, we tackled the problem by training two separate models. The first model detects human faces within a given image, and the second model classifies the emotions expressed in those faces.

For the face detection model, we selected a Faster R-CNN based model, the X101-FPN, available from Facebook's *Detectron2* package, Model Zoo. The X101-FPN model was pre-trained on a very famous COCO dataset. X101-FPN has the best box AP and mask AP but the downside is that it takes a lot of time to train. As the model is trained for object detection task, we used it as our baseline model and trained it with the 1000 faces in Dataset 1. For each of the detected faces, the model outputs the top-left and bottom-right (x,y) coordinates of the rectangular box which bounds the detected face. From the resulting coordinates, the detected faces can be cropped from the image, resized into the appropriate dimensions, and then fed into the second model.

For the second model, the goal is to classify the emotions of the detected human faces from the first model. The most intuitive approach for such problem is KNN. The hyperparameters that were tuned to optimize the baseline model include weightings, norms and number of neighbors. Different image representations of using raw pixels array vs. color-histogram were also considered when testing these three hyperparameters. This is because knowing which representation produces the most accurate model can help us rationalize our choices of the alternative models.

Since most image-processing applications use neural network models, MLP and CNN are thus chosen to be our main models to improve the baseline model's accuracy. Different architectures of MLP and CNN will be explored as we work more on the project.

We trained two separate CNN models, one based on a simple architecture and the other one based on the ResNet50. Both networks are trained and tested on the FER dataset. Details about the training process and results are reported in Section 6.

6. Experiments and Results

6.1. Face Detection

As describe in Section 3.1, we first pre-processed Dataset 1 to be in the right format for the X101-FPN model to train on. We mostly referenced the configuration provided by *Detectron2* for our setup and also utilized warm-up iteration which ensures that the system will arrive at steady state prior to taking measurements. See below for our full setup:

For object detection, we used the concept of Intersection over Union ('IoU') and Average Precision ('AP') to evaluate the performance of the model. IoU computes intersection over the union of the two bounding boxes; the bounding box for the ground truth and the predicted bounding box.

X101-FPN Training Setup	
Number of Workers	4
Image per Batch	4
Base Learning Rate	0.00025
Warm-up Iteration	1200
Max Iteration	2200
Gamma	0.05
Batch Size per Image	32
Number of Classes	1 (face)

Table 1: Detectron2 Configuration

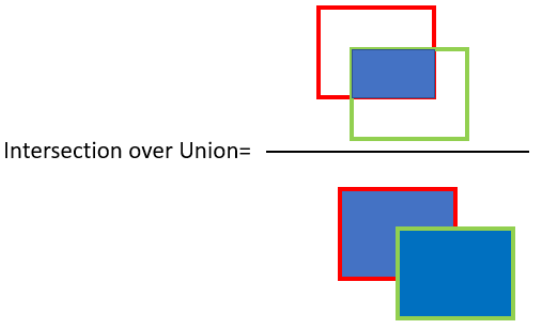


Figure 4: A representation of IoU, courtesy of University of Pittsburgh.

We then set a threshold value for the IoU to determine if the object detection is valid or not. For example, if we set IoU to 0.5, if IoU is ≥ 0.5, we classify the object detection as True Positive. if Iou ≤ 0.5, then it is a wrong detection and we classify it as False Positive. When a ground truth is present in the image and model failed to detect the object, classify it as False Negative. We then used Precision and Recall as the metrics to evaluate the performance.

$$\text{Precision} = \frac{TruePositive}{TruePositive+FalsePositive}$$
$$\text{Recall} = \frac{TruePositive}{TruePositive+FalseNegative}$$

The general definition for the Average Precision (AP) is finding the area under the precision-recall curve.

$$\text{AP} = \int_0^1 p(x)\,dx$$

Our model was able to achieve a 70% AP50 (AP with IoU set to 0.5), which is considered decent for object detection. See below a few examples generated by our model.

Figure 5: Sample images result from face detection model.

And another example that is not originally from our dataset...

Figure 6: Human face detection, courtesy of Prof. Ungar.

6.2. Facial Expression Recognition

6.2.1. K-Nearest Neighbors

Because of its intuitive approach and simplicity, KNN is chosen to be the baseline of our emotion classification problem. We used the KNeighborsClassifier from Scikit-learn to develop and evaluate the baseline model. The classifier's performance heavily depends not only on the selected features of the input image, but also on the hyperparameters of the model. We tried two methods of extracting features of the input image [14]: The first method is to represent the image using its flatten raw pixel array, and the second method is to convert the image into its color histogram. For each method, different settings of hyperparameters are considered to evaluate the model accuracy.

The hyperparameters of our KNN classifier were tuned by considering different configurations of the weightings, norm types, and number of neighbors. Since the dimensionality of the input is quite large (48*48 features for flatten raw pixel representation, and 255 features for color histogram representation), the curse of dimensionality will make all the data points further apart and thus choosing the hyperparamaters that can detect slight differences in the input vector is crucial for the model's accuracy. To satisfy this requirement, the weightings were made to be the inverse of the distance or "weighting by distance" in which closer neighbors of a query point will have a greater influence than neighbors that are further away. The norm type is selected by evaluating the model's accuracy with respect to different norms, and the results are shown in Figure 7:

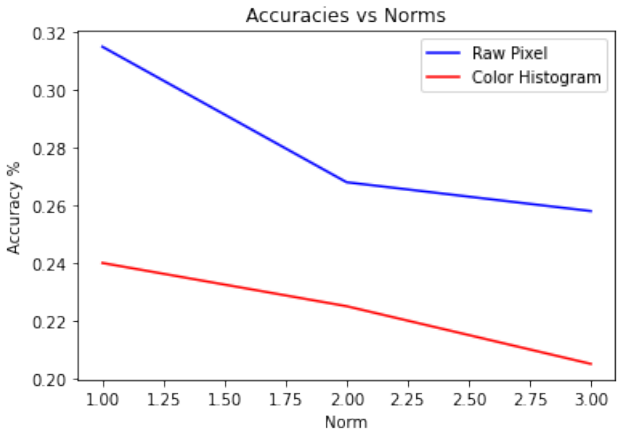


Figure 7: KNN accuracies with respect to L1, L2, L3 norm.

To recognize the emotions from a facial expression, the layouts of mouth and eyes are what really matter, and other features such as skin colors, hair styles, glasses, backgrounds, etc can be considered as noise. Unfortunately, these noisy features occupy most of the space of a given image. For the model to have a good performance, it needs to somehow discard the noise or at least not to amplify it. Therefore, the trend-lines of Figure 7 make sense because L2,L3 and even higher norms amplify those noisy features and thus result in lower accuracy compared that of L1 norm which does not amplify the noise. As a result, L1 norm was chosen as the metric to measure distances.

Similarly, the number of neighbors is selected based on the

model accuracy with respect to different number of neighbors. The experiment results are illustrated in Figure 8:

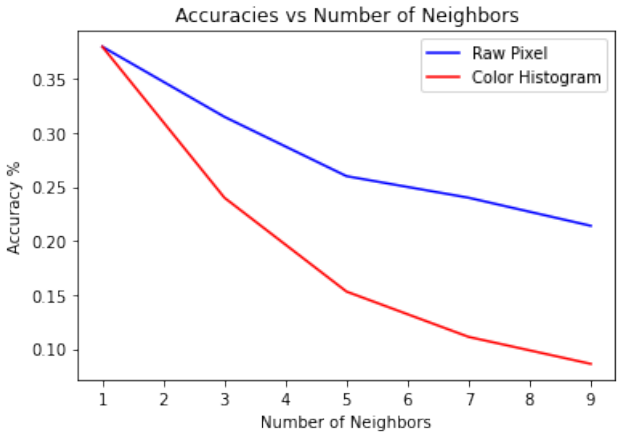


Figure 8: KNN accuracies with respect to K = 1,3,5,7,9.

From Figure 8, increasing the number of nearest neighbors results in the decrease in the model accuracy. Because the curse of dimensionality causes the data points to be far away from each other, this undermines the statistical significant of the data. Therefore, considering more neighbors only add noise to our model which results in a lower accuracy . When $K = 1$, the model has the highest accuracy of 38%. However, this performance is resulted from the model over-fitting the data, and thus we did not consider this as the true accuracy of the baseline model. When $K \geq 3$, we can see that different representations of images can result in different model accuracies. Specifically, the KNN model that uses raw pixels as its input outperforms the model that uses color histogram. From this result, we learn that to "decode" the emotion of a face image, it is more effective to consider the contours (raw pixel array) of the image rather then the colors (color histogram) presented in the image. This result gives us more confidence in our choice of using Neural Network models as the alternatives which have been proven to be very effective in analysing images in their raw pixel forms. Based on Figure 8, when $K = 3$ and the input image is represented using its raw pixels, the baseline model has the next best accuracy of 31.5%. Please note that, given seven emotions, a random classifier would have a 14.3% accuracy.

In summary, given the hyperparameter setting of raw pixel representation, weighting by distance, L1 norm and $K = 3$, the KNN baseline model has the best testing accuracy of 31.5%

6.2.2. Multi-layer Perceptron (MLP)

The MLP is our second baseline model which is implemented using Pytorch and has the architecture described in Table 2:

MLP Layers	
Fully-connected 1	Output Channels = 1024
Fully-connected 2	Output Channels = 1024
Fully-connected 3	Output Channels = 64
Fully-connected 4	Output Channels = 32
Fully-connected 5	Output Channels = 7

Table 2: MLP Architecture

The architecture is inspired by the network in [15] which only uses fully-connected layers to classify the MNIST dataset and achieves the testing accuracy of 96.07%. Since the inputs of [15] are 28×28 images, and the size of our training images is 48×48, we thus scaled up the number of neurons/filters in [15] from {256,256,128,32,10} to {1024,1024,64,32,7} and adjusted the final layer to better suit our application. However, because detecting emotions is a lot more indirect than detecting digits from raw pixels, different activation functions will be evaluated to determine which function would help to extract the most emotion-related features from input images.

To select the best activation function for our problem, we trained the model using either Sigmoid, ReLU or Tanh function and ran the model through 50 epochs. The activation function, which not only results in the highest testing accuracy but also has the best loss convergence, will be selected. The results of our experiment are recorded in Figure 9, 10, 11 and Table 3:

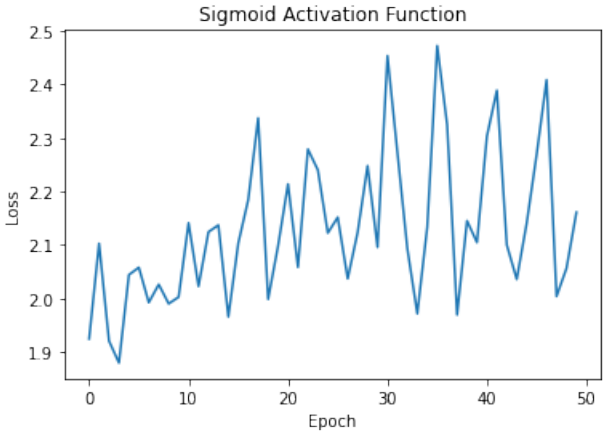


Figure 9: Training Loss vs. Number of Epochs Using Sigmoid Activation Function

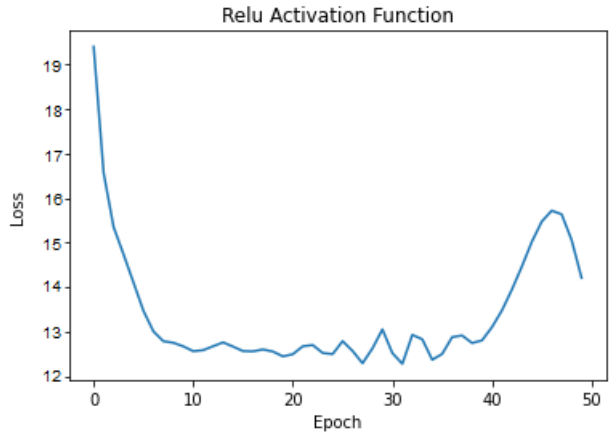


Figure 10: Training Loss vs. Number of Epochs Using ReLU Activation Function

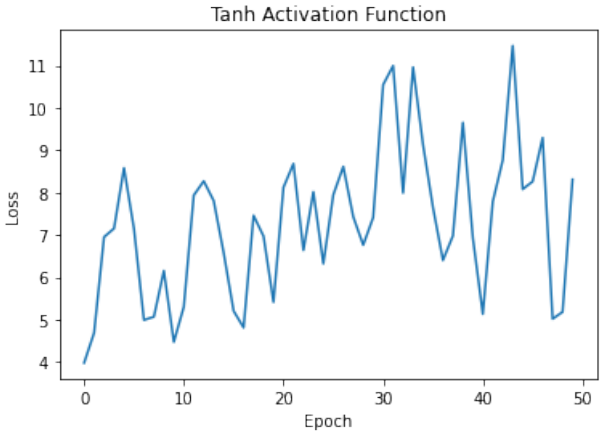


Figure 11: Training Loss vs. Number of Epochs Using Tanh Activation Function

Activation Function	Testing Accuracy %
Sigmoid	17.49
Relu	22.81
Tanh	20.19

Table 3: Testing Accuracies

Notice that the losses of figures 3,4,5 are relative with respect to what activation function is being used. For example, sigmoid function constrains the values of neurons to be between 0 and 1 while tanh function allows neurons to output numbers from $-\infty$ to $+\infty$. These differences in the range of the model's output result in different ranges of cross-entropy loss. Nevertheless, we can see that models using sigmoid or tanh function did not converge while the model using relu function did. Despite also having the highest testing accuracy of 22.8%, the relu-model seems to be stuck at a local minimum which causes the model to mostly output emotion as "happy". The "happy" emotion is also the majority of the dataset. From this result, we learn that it is not effective to use only fully-connected layers to recognize emotions from raw pixels. Because there are many irrelevant features such as background colors and skin colors, weighting all pixels would only cause the model to stuck at some local minimum or even unable to converge at all. As a result, using convolutional neural network (CNN) would be our next best approach, since CNN can detect more relevant features such as eyes and mouths by considering local receptive fields instead of the entire raw pixel array.

In summary, the MLP model, which has the architecture described in table 1 and uses ReLU as its activation functions, results in the best testing accuracy of 22.8%.

6.2.3. Convolutional Neural Networks

Model The architecture for the two networks we used are presented in Table 4. For the ResNet-based CNN model, we replaced the fully-connected layers in the initial ResNet50 model. Both models were trained on the FER data set using Adam optimizer with learning rate of 0.001 and the loss is calculated using cross-entropy loss.

Performance as shown in Table 5, while the architecture is simple, our CNN model outperformed the other two machine learning methods we tested by a large margin. Using ResNet18 architecture and initialization, we were able to get 62.8% accuracy on the test dataset and the model get 55.2% accuracy only after the first training epoch, which outperformed the best accuracy of our simple CNN model. Since the ResNet-based CNN gave the best accuracy across our tested models, the result discussed in later sections of this report will focus on the performance of this model.

Model	Accuracy(%)
KNN	31.50
MLP	22.81
simple CNN	54.79
ResNet CNN	62.77

Table 5: Model performance

Analysis As shown in Figure 15 the 'Happy' expression has the highest accuracy of 83.84% while the model's performance on 'Neutral', 'Sad', 'Fear', 'Angry' expressions are relatively poor and the accuracies are around 55%. As we can see in the confusion matrix presented in Figure 16, our model could be confused with those four expressions and often predict one of them as the other. Figure 14 presents some prediction results made by the ResNet-based CNN model. We can see from the result that the happy faces like Figure 14a and Figure 14b are obvious, but 'Sad',

Simple CNN	
Conv1	Kernel size = 5x5x64, stride = 1, pad = 0. Followed by ReLU
MaxPool1	Kernel size = 5x5, stride = 2
Conv2	Kernel size = 3x3x64, stride = 1, pad = 0. Followed by ReLU
MaxPool2	Kernel size = 3x3, stride = 2
Conv3	Kernel size = 3x3x128, stride = 1, pad = 0. Followed by ReLU
MaxPool3	Kernel size = 3x3, stride = 2
Fully-connected 1	Output Channels = 1024. Followed by ReLU
Fully-connected 2	Output Channels = 1024. Followed by ReLU
Fully-connected 3	Output Channels = 7. Followed by ReLU
ResNet-based CNN	
Convolutional layers	Convolutional layers from ResNet50, pre-trained weights are loaded.
Fully-connected 1	Output Channels = 1024. Followed by ReLU
Fully-connected 2	Output Channels = 1024. Followed by ReLU
Fully-connected 3	Output Channels = 7. Followed by ReLU

Table 4: CNN Architecture

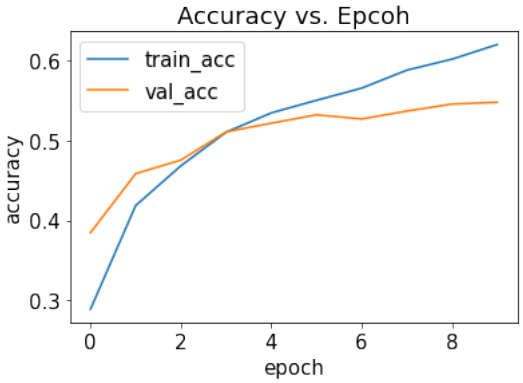


Figure 12: Simple CNN loss curve

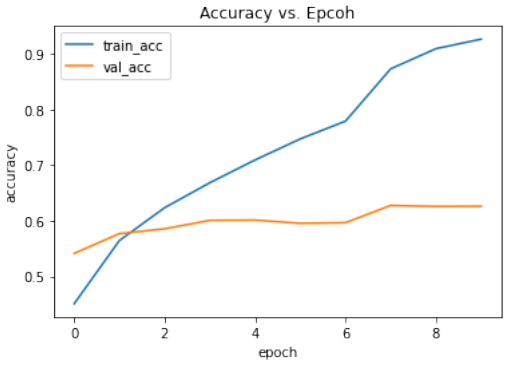


Figure 13: ResNet-based CNN loss curve

'Fear', 'Angry' faces are harder to differentiate from each other. Images like Figure 14d, 14e and 14f could even confuse a human agent. This could explain the difference in the per class accuracy.

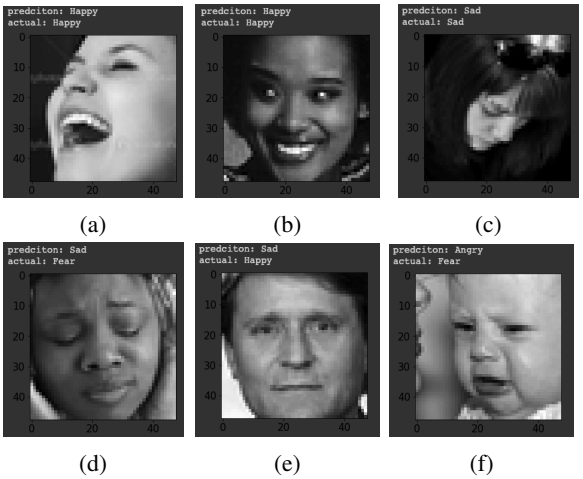


Figure 14: Example of predictions made by our model, (a), (b) and (c) are predicted correctly, (d), (e) and (f) are wrong.

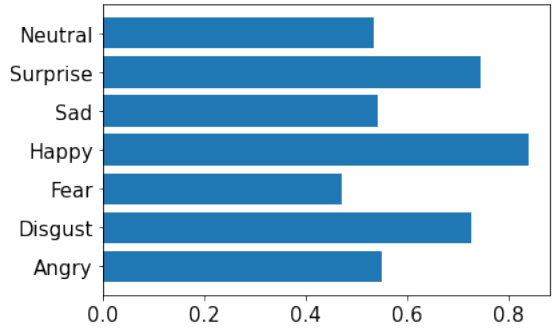
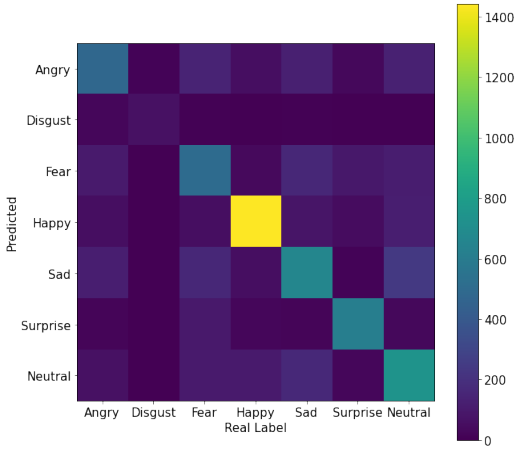


Figure 15: Per class accuracy



	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	479	15	144	51	127	29	130
Disgust	26	64	9	2	8	1	4
Fear	103	3	502	38	159	91	114
Happy	51	0	52	1442	83	50	123
Sad	123	5	158	55	664	15	238
Surprise	22	1	97	28	18	615	33
Neutral	65	0	105	104	166	26	740

Figure 16: Confusion matrix

7. Conclusion and Discussions

Face detection is the first step in many face-related tasks, thus, we see some potential opportunities to leverage our first model for a future research on celebrity face recognition, for example. Even though the model did quite well for the most part, there are some instances where the model either failed to identify faces or it "over-detected" and identified multiple bounding boxes for the same face. This could be overcome by having more training data (or augmenting existing data) or applying a technique called non-maximum suppression.

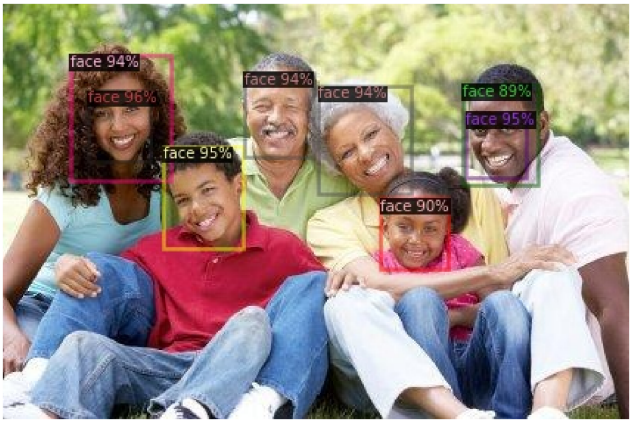


Figure 17: An example of "over-detecting".

KNN Implementing KNN as baseline model teaches us a lot about the inherent limitations of the model as well as the challenges of facial expression recognition itself. KNN offers an intuitive solution to our problem, but its performance is negatively affected by the curse of dimensionality. We overcome this problem by choosing the hyperparameters such as weighting by distance, L1 norm and $K = 3$ which do not amplify noise and can capture subtle change in distances between data points. From Section 6.2.1, we learn that raw-pixel representation results in a higher testing accuracy of 31.5% compared to 24.7% of color-histogram representation.

MLP is chosen as our next baseline model which is made of five fully-connected layers $\{1024,1024,64,32,7\}$. Different activation functions were evaluated, and we found that Relu function not only makes the model to converge but also has the best testing accuracy of 22.8%. The fact that the MLP model performs not as good as the KNN model goes against our expectation. This could be explained by the limitation of fully-connected layers in filtering out noise in images, while using L1 norm in KNN can at least help not to amplify the noise.

Using ResNet The best accuracy we obtained is 62.8% from the ResNet-based CNN model. The pre-trained ResNet model we are using as our initialization is trained on more than a million images from the ImageNet database. Therefore, it has already learned very good kernels in its convolutional layers and is able to extract useful features from the facial expression images without further training. That might be the reason why it outperforms our simple CNN model after the first training epoch. Since training a deep neural network could take a huge amount of computational power, transfer learning technique is very important for machine perception tasks. Lots of recent research are using pre-trained models as backbone and developing new networks from that[16; 17].

FER Dataset As discussed in the previous section, the performance drop for classifying "Neutral", "Sad", "Fear" and "Angry" could be resulted from the confusing labels in the FER-2013 dataset. This is also reported by Microsoft Research as they released the FER+ dataset with new labels collected by crowd sourcing[18]. Some re-labeled images are shown in Figure 18. For future works, we can train our model on the FER+ dataset and it should improve our prediction accuracy.

Our final thought Facial expression recognition is a very challenging task, even for a human being. A trained human can be excellent at hiding true emotions and there could be no way to find out through facial expressions. Furthermore, a human face could show multiple expressions at the same time. An extension to this project could be to build a classifier that shows the probability of each emotion. With

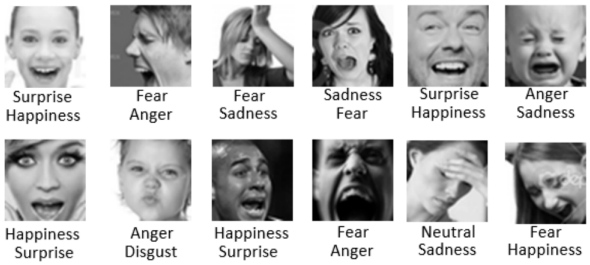


Figure 18: FER vs FER+ examples. Top labels are FER and bottom labels are FER+.

that said, machine learning is getting closer at matching human's level, which could be exciting and terrifying at the same time.

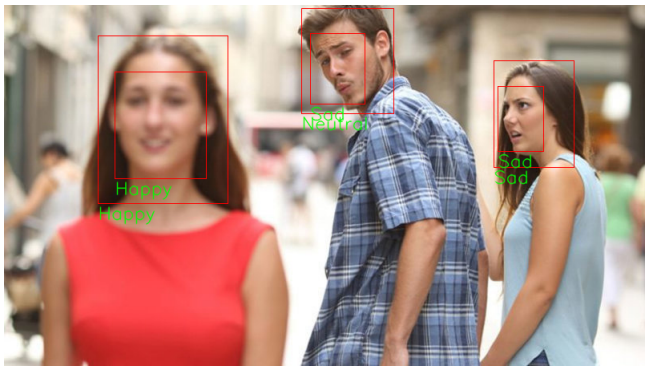


Figure 19: An example of potential multiple expressions in one face.

Acknowledgments

We would like to thank Dr. Ungar and the CIS 520 TAs for their advice and encouragement through out this project.

References

[1] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." https://github.com/facebookresearch/detectron2, 2019.

[2] Ming-Hsuan Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34–58, 2002.

[3] A. Z. Kouzani, F. He, and K. Sammut, "Commonsense knowledge-based face detection," in *Proceedings of IEEE International Conference on Intelligent Engineering Systems*, pp. 215–220, 1997.

[4] I. T. Meftah, N. L. Thanh, and C. B. Amar, "Emotion recognition using knn classification for user modeling and sharing of affect states," *Neural Information Processing Lecture Notes in Computer Science*, p. 234–242, 2012.

[5] I. Tayari-Meftah, N. Le-Thanh, and C. Ben-Amar, "Sharing emotional information using a three layer model," 2011.

[6] M. Valueva, N. Nagornov, P. Lyakhov, G. Valuev, and N. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Mathematics and Computers in Simulation*, vol. 177, pp. 232 – 243, 2020.

[7] B. Zhang, C. Quan, and F. Ren, "Study on cnn in the recognition of emotion in audio and images," in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pp. 1–5, 2016.

[8] B.-K. Kim, H. Lee, J. Roh, and S.-Y. Lee, "Hierarchical committee of deep cnns with exponentially-weighted decision fusion for static facial expression recognition," in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, (New York, NY, USA), p. 427–434, Association for Computing Machinery, 2015.

[9] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[11] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio, "Challenges in representation learning: A report on three machine learning contests," 2013.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2014.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016.

[14] A. Rosebrock, "k-nn classifier for image classification," Apr 2020.

[15] P. Mahajan, "Fully connected vs convolutional neural networks," Oct 2020.

[16] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," 2018.

[17] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2018.

[18] E. Barsoum, C. Zhang, C. Canton Ferrer, and Z. Zhang, "Training deep networks for facial expression recognition with crowd-sourced label distribution," in *ACM International Conference on Multimodal Interaction (ICMI)*, 2016.