Facial Emotion Recognition as an Aid for Autism

Agni Kumar (6.869)
Massachusetts Institute of Technology
Cambridge, MA 02139
agnik@mit.edu

Shannon Duffy (6.819)
Massachusetts Institute of Technology
Cambridge, MA 02139
sduffy@mit.edu

Abstract

We present the designs of various artificially intelligent systems capable of recognizing emotions through facial expressions. Various neural network architectures were customized, trained, and subjected to classification tasks, after which the best performing networks were further optimized. A variety of datasets (fer2013, Jaffe, CK+), as well as one of our own creation, was used to train the models. We applied transfer learning on the fully-connected layers of an existing convolutional neural network that was pretrained for human emotion classification, as well as evaluated custom neural networks. We achieved an overall top test accuracy of 65.7%. The applicability of the final model is portrayed in our building of a live video application capable of instantaneously classifying users' emotions and superimposing faces with appropriate emojis. Through this project, we worked to develop the foundations of a product capable of aiding children with autism, who often struggle to read the subtle nonverbal cues contained in people's expressions.

1. Introduction

One of the current top applications of artificial intelligence using neural networks is the recognition of faces in photos and videos. Such techniques involve processing visual data and searching for general patterns present in human faces. Facial recognition has several applications, from surveillance purposes to crowd management to automatic blurring of faces on Google Street View footage to automatic recognition of Facebook friends in photos. A more advanced development in this field is emotion recognition. Emotions facilitate interactions between human beings, and understanding them can bring valuable context to social communications.

While emotions can be identified through a variety of techniques (involving the examination of body language, voice intonation, electroencephalography waveforms, etc.), a practical method involves studying facial expressions [2]. In this paper, we review the design choices of existing emotion recognition models and enhance prior techniques with some new ideas. Given that there are seven types of human emotions cited to be universally recognizable across most cultures (happiness, sadness, surprise, anger, disgust, fear, contempt), a tool to detect changing emotions from facial expressions would be widely applicable [5]. Some of the most promising applications of emotion recognition involve helping children with autism read faces; provided with a simple cue, such as emojis overlaid onto or near faces within their fields of view, autistic individuals could increasingly learn to notice and interpret the social world around them.

We note that the task of emotion recognition is particularly difficult for two main reasons: (1) that there does not exist a large database of training images, and (2) that classifying emotion can be difficult depending on whether the input image is static or a transition frame into a facial expression, a challenging issue for real-time detection involving dynamic varying of expressions.

As underlined within the literature, a highly promising concept for facial expression analysis is the use of deep convolutional neural networks (CNNs). Given a limited amount of available processing resources, we subjected multiple architectures to an emotion classification problem. Most classification problems involve examining static images of facial expressions, but we investigated the application of CNNs to emotion recognition in real time with a video input stream, optimizing the best-performing network for efficient computation for frame-by-frame classification. In optimizing our system, we accounted for variations in lighting and subject position as much as possible, and were able to successfully implement an application wherein emojis indicating expressions are superimposed over faces in real time.

2. Related Work

Over the last two decades, researchers have significantly advanced algorithms used for interpreting pictures. Approaches may be grouped into two main categories: those that involve pre-programmed feature extractors used to an-

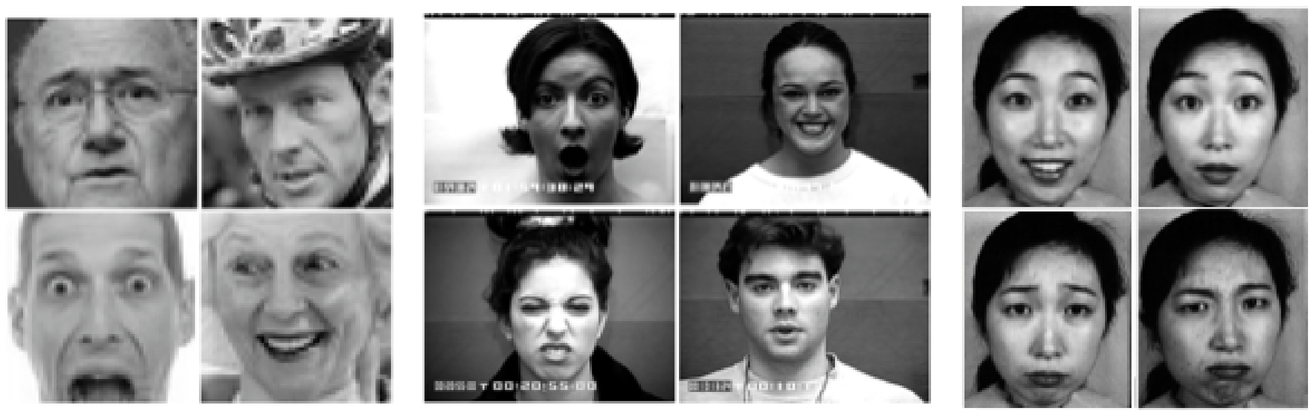

Figure 1. From left to right, examples of images from the fer2013, CK+, and JAFFE datasets

alytically break down several elements in the picture in order to categorize the object shown, and those that involve self-learning neural networks providing a black-box identification technique in which the system itself develops rules for object classification by training on labeled sampled data.

2.1. Common CNN Designs

Commonly used CNNs for emotion recognition include a set of fully connected layers at the end, which tend to contain most of the parameters. Recent architectures such as Inception-v3, reduced the amount of parameters in their last layers by including a Global Average Pooling operation, which forces the network to extract global features from the input image [16]. Additionally, modern CNN architectures like Xception employ experimental characteristics such as the use of residual models and depth-wise separable convolutions, which reduce the amount of parameters by separating the process of feature extraction and combinations within convolutional layers [9]. The state-of-the-art model for the fer2013 dataset is comprised of a CNN in which 98% of all parameters were located in the last fully connected layers; trained with square hinged loss, the model achieved an accuracy of 71% using approximately 5 million parameters [6].

2.2. Recent Developments

Recent top submissions to the Emotions in the Wild (EmotiW 2018) contest for static images used CNNs to generate up to 61% test accuracy [4]. Another recent development addressed two important problems relating to facial image recognition: the scarcity of data for training deep CNNs, and the variation in appearance caused by differing training image illuminations. Levi et. al utilized a Local Binary Patterns (LBP) operator, which possesses a robustness to monotonic gray-scale changes caused by illumination variations, to transform the image before inputting them into a CNN. This special data preprocessing was applied to to various public models like VGG_S (achieving 40% accuracy), after which the model was re-trained on the CASIA WebFace dataset and transfer-learned on the Static Facial Expressions in the Wild (SFEW) dataset (achieveing 55% accuracy) [13, 8]. We chose to the use this modified, pre-trained, freely available neural network as a starting point, later following up with proposals for performance improvements.

3. Approach

To develop a working model, we used three different datasets. The first was the JAFFE (Japanese Female Facial Expression) database, which contains 213 images of 60 Japanese females displaying the emotions angry, disgust, fear, happy, neutral, sadness, and surprise [10]. This dataset served well for the initial training because the images were cropped to show only the face, and subjects were looking directly at the camera. The second database that used was the extended Cohn-Kanade (CK+) database, which contains 5,876 images of 210 individuals displaying the emotions, angry, disgust, fear, happy, sadness, surprise, like JAFFE and contempt instead of neutral [12]. It includes both posed and unposed expressions, adding great diversity. The final database that we used was fer2013, which is a Kaggle database from Kaggle's "Representation Learning: Facial Expression Recognition Challenge" [1]. The database contains 35,000 images with subjects displaying the same emotions as JAFFE: angry, disgust, fear, happy, sad, surprise, and neutral. This database was the most extensive, so we used it for most of our final rounds of testing. Examples of photos from each database can be seen in Figure 1, and a breakdown of the fer2013 dataset is given in Figure 2.

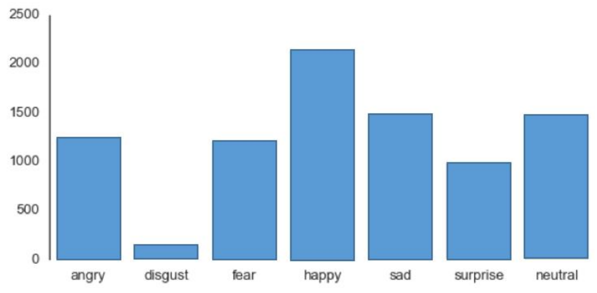

Figure 2. Image distribution of the fer2013 dataset

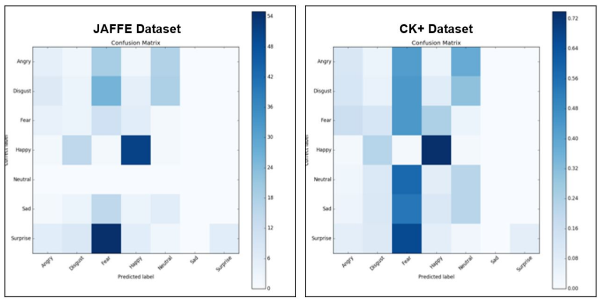

Figure 3. Confusion matrices for the unaltered VGG_S network on the JAFFE and CK+ datasets

3.1. Applying Transfer Learning

We initially ran the aforementioned VGG_S network on the JAFFE and CK+ datasets, achieving accuracies of 14.08% and 24.27% respectively. Confusion matrices are depicted in Figure 3. We see that many facial expressions were incorrectly classified as 'fear' for both datasets. We hypothesized that the low accuaracies may be due to the facial expressions in the JAFFE dataset being quite subtle and thus difficult to detect emotion from, and possibly because there were too few images with 'fear' and 'disgust' labels in both the CK+ and JAFFE databases.

To improve facial emotion classification accuracy, we applied transfer learning, using the pre-trained Transfer-LearningNN with an Inception-v3 base model [14]. Our model added a couple top layers to the original model to match the number of target emotions to be classified and reran the training algorithm on the fer2013 dataset.

3.2. Creating Custom CNNs

We began by experimenting with a simple CNN (CNN-1), composed of an input, three convolution layers, one dense layer, and an output layer. As we hypothesized that the simple network architecture would would fail to pick up on the subtler details in facial expressions, we followed up by building a deeper architecture. After fine-tuning the number and configuration of convolutional and dense layers, as well as the droupout percentage in dense layers, we settled on a network nine layers deep, with maxpooling after every three convolutional layers, followed by two sequential dense layers (with 15% dropout) and a softmax output (CNN-2). We hypothesized that this new architecture would capture fine details with more meticulousness than before.

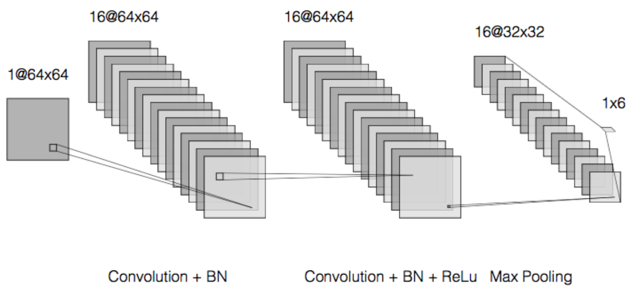

Figure 4. Depicts one module of the custom CNN

We designed another custom model as well, shown in Figure 4 (CNN-3). This module consisted of four modules where each module had two convolutional layers, ReLu activation, and a max pooling layer. We then tried to improve the CNN by adding two dense layers at the end with 50% dropout between the two dense layers, then by adding only one dense layer. For our next step, we utilized several popular models: ResNet-18, VGG_S, Inception-v3, Xception. We trained our dataset using the ResNet-18 network architecture with Adagrad and Adam optimizers. For the Adam optimizer, we tried learning rates of 0.001 and 0.0014, as gathered from a prior research finding [7, 11].

The Inception-v3 module computes 1x1, 3x3, and 5x5 convolutions within the same module of the network and the output of these filters is fed into the next layer [15]. We thought that the Inception-v3 model would therefore be good to try in emotion recognition because multiple features are extracted in one module. There are multiple features that go into the portrayal of an emotion (mouth, eyes), therefore we thought that this model would perform well at detecting those multiple features.

The Xception model extends the Inception-v3 model by replacing the modules with depthwise separable convolutions [3]. The Xception model performs well because the spatial and depthwise information in a layer can be decoupled. Therefore, if it was found that Inception-v3 performed well, we hypothesized that Xception would perform better or comparably.

3.3. Developing Video Application

We connected a video stream to the best-performing custom CNN using a standard webcam. The model was trained on a dataset we created, comprised of 300 images of ourselves for each emotion captured in 3 different light condi-

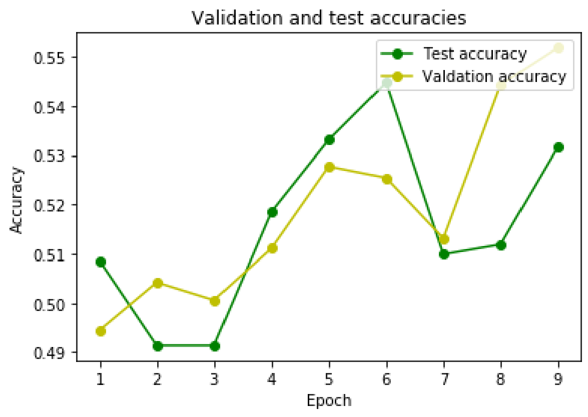

Figure 5. Validation and test accuracies when employing transfer learning

tions (100 of each). We also applied a Haar-Cascade filter provided by OpenCV to crop the input image faces, which significantly improved test and training accuracy.

4. Experimental Results

With transfer learning, a final test accuracy of 53.17% was achieved. A plot of the validation and test accuracies over training epochs is given in Figure 5. As the accuracies follow the same general trend and are close together in magnitude, overfitting is minimal.

For our custom CNNs, we found that CNN-1 performed poorly; a low test accuracy of 15.14% indicated that it was effectively randomly guessing emotions. Modifications to this simple model heightened performance substantially, and CNN-2 achieved a 59.18% accuracy. As for CNN-3, the final results were that the neural nets tested performed very similarly, but Xception and Resnet-18 had the best accuracy. As shown in Figure 6, after 100 epochs, the custom CNN had an accuracy of 61.7%. The training accuracy is consistently below the testing accuracy, so it seems that the model was simple enough that the data was not overfit. We thought that adding dense layers would increase the accuracy, however we found that with one dense layer, the accuracy dropped to 57.1% and with two dense layers, the accuracy dropped to 52.7%. The accuracy per epoch is shown in Figure 7 and 8, and it seems that with the addition of the dense layers, the model is fitting the data worse because the accuracy of the training data gets much lower.

The Xception model gave one of the highest accuracies with a test accuracy of 65.6%. Due to the high accuracy of the modified Inception-v3 model, we expected the Xception model to also perform well because it uses the same idea of extracting multiple features in a module and extends it with depthwise separation. Additionally, looking at Figure 9, the model didn't seem to overfit that much because the test accuracy is within a few percent of the training accuracy.

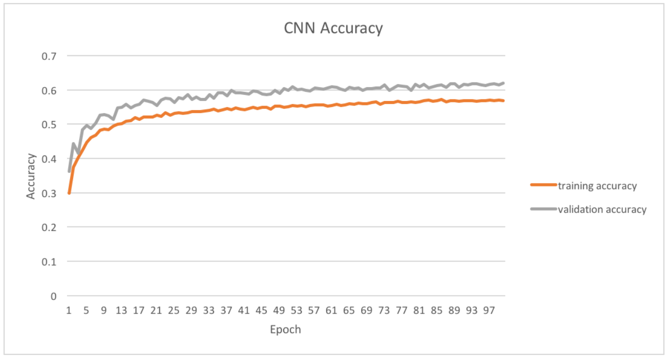

Figure 6. The accuracy of the custom CNN model on the train and test datasets after 100 epochs.

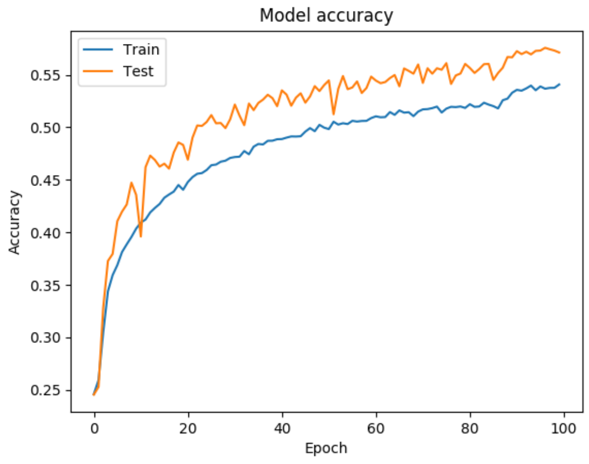

Figure 7. The accuracy of the custom CNN with one dense layer on the train and test datasets after 100 epochs.

We tested ResNet-18 by varying the hyperparameters and optimizer and found that ResNet-18 performed best with the Adam optimizer and 0.001 learning rate. We started by testing the adagrad optimizer, which had a test accuracy of 64.6%. However, it seems like the model overfit the data. As shown in Figure 10, at around epoch 20, the accuracy of the training dataset starts to improve much more than the accuracy of the test dataset, which means that the model was fitting to the training set. The second optimizer that we tried was the Adam optimizer and we tested using a learning rate of 0.001 and 0.0014. With a learning rate of 0.001, the test accuracy was 65.7% and with a learning rate of 0.0014, the test accuracy was 64.7%. For a learning rate of 0.001, it seems best to train the model for around 70 epochs because it is enough epochs such that the test accuracy is around the highest, but it is not so many epochs that the model starts to overfit to the training set. As seen in

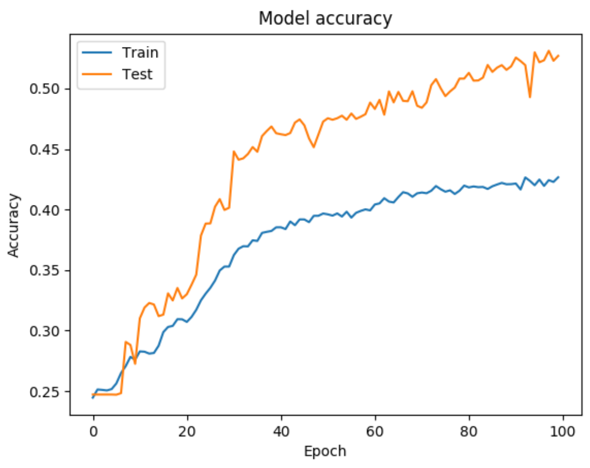

Figure 8. The accuracy of the custom CNN model with two dense layers and a 50% dropout in between on the train and test datasets after 100 epochs.

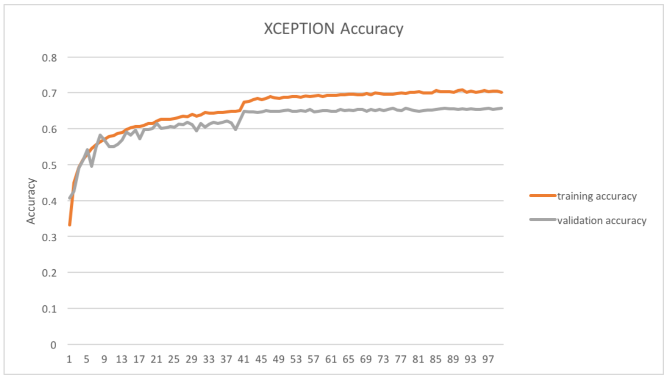

Figure 9. The training and validation accuracy training using an Xception model.

Figure 11, after around epoch 70, the training accuracy continues to increase, but the test accuracy stays the around the same. Similarly, from Figure 12, when using the Adam optimizer and 0.0014 as a learning rate, the model performed best after around 40 epochs, so the same reasons as above: the test accuracy is around the highest and the training accuracy is still relatively similar to the test accuracy.

We have summarized the test accuracies of the models discussed above in Table 1.

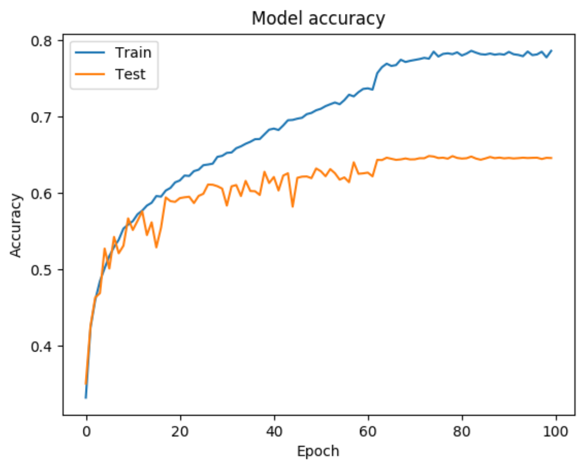

Figure 10. The accuracy of the ResNet-18 model with the Adagrad optimizer and learning rate of 0.001 on the test and training datasets.

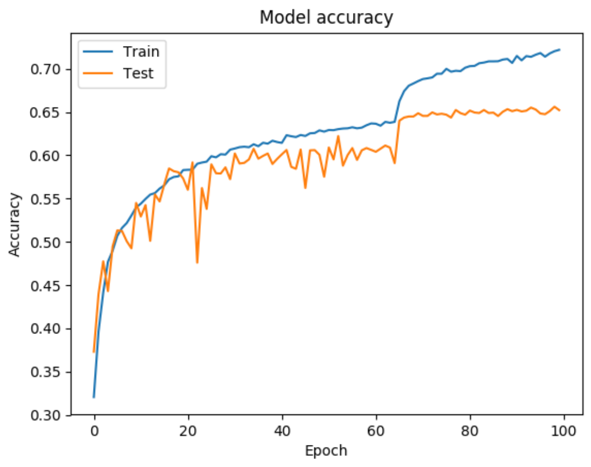

Figure 11. The accuracy of the ResNet-18 model with the Adam optimizer and learning rate of 0.001 on the test and training datasets.

5. Conclusion

If we were to continue the project, there are a few steps that we would take to try to further improve the accuracies of the emotion recognition. First, we would augment the dataset to add images that have been filtered with a Gaussian blur. This would mimic noise that a user may produce when user their webcam and help to prevent overfitting to the training set, therefore we think that it would make the model perform more accurately on the test set. Additionally, we would incorporate time-delayed 3D-convolutional neural networks, which use temporal information as part of the training samples. Instead of inputting one image to the model, we would include images that led up to that pose. One training sample would contain n images from a series and the emotion label would be the emotion shown on the last image. By capturing the expressions leading up to an emotion, we hope to help the model better differentiate between the different emotions. A demonstration of

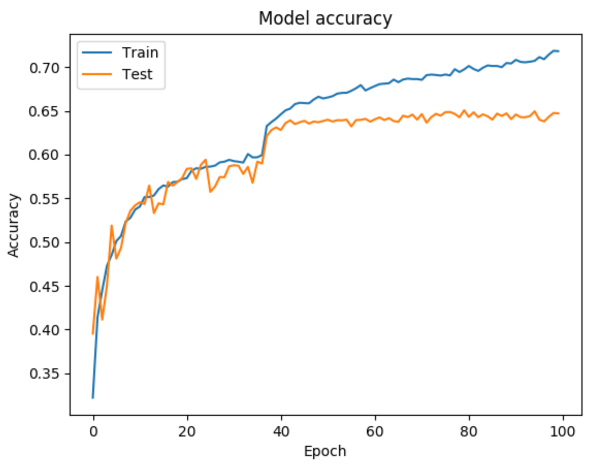

Figure 12. The accuracy of the ResNet-18 model with the Adam optimizer and learning rate of 0.0014 on the test and training datasets.

Model	Accuracy
Transfer learning	53.2%
CNN-2	59.2%
CNN-3	61.7%
CNN-3 with dense layer	57.1%
CNN-3 with two dense layers	52.7%
Xception	65.6%
ResNet-18 with Adagrad	64.6%
ResNet-18 with Adam (0.001 learning rate)	65.7%
ResNet-18 with Adam (0.0014 learning rate)	64.7%

Table 1. Accuracy results of various models after optimizations

the video application is given below, and can be viewed within Adobe Reader by clicking on the frame displayed.

6. Division of Labor

For the research portion, I (Agni Kumar) worked on applying transfer learning, implementing and training CNN-1 and CNN-2, and developing the video application. I also created figures for these experiments and wrote various parts of the paper.

References

[1] Challenges in representation learning: Facial expression recognition challenge, 2012.
[2] P. Abhang, S. Rao, G. W. Gawali, and P. Rokade. Emotion recognition using speech and eeg signal a review. 2011.
[3] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 2016.
[4] A. Dhall, A. Kaur, R. Goecke, and T. Gedeon. Emotiw 2018: Audio-video, student engagement and group-level affect prediction. 2018.
[5] P. Ekman. Universals and cultural differences in facial expressions of emotion. 1971.
[6] I. Goodfellow. Challenges in representation learning: A report on three machine learning contests. 2013.
[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.
[8] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.
[9] A. Howard. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 2017.
[10] J. Kumari, R. Rajesh, and K. Pooja. Facial expression recognition: A survey. volume 58, 08 2015.
[11] D. Mack. How to pick the best learning rate for your machine learning project, 2018.
[12] M.-A. Quinn, G. Sivesind, and G. Reis. Real-time emotion recognition from facial expressions. 2017.
[13] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.
[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.
[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. in proceedings of the ieee conference on computer vision and pattern recognition. 2016.