CSE 881 FINAL PROJECT (Cover Page)

PROJECT TITLE: EXPRESSION RECOGNITION

PROJECT TYPE: PROTOTYPE DEVELOPMENT

DIFFICULTY LEVEL: MODERATE

Ground truth labels + 0.5
Data preprocessing + 1
Evaluation + 0.5
Prototype development + 1

SUMMARY OF TEAM MEMBER PARTICIPATION:

Name	Responsive to emails	Attended project meetings	Participate in data collection and preprocessing	Participate in coding	Participate in analysis experiments	Writing final report	Class presentation	Completed Assigned Tasks
Shaojun Wang	3	3	3	3	3	3	3	3
Xiaoyan Li	3	3	3	3	3	3	3	3
Jicheng Li	3	1	0	0	0	0	0	0

SUMMARY OF TEAM MEMBER PARTICIPATION

Name	Roles and Contributions
Shaojun Wang	data collection, preprocessing, intermediate report, prototype development, traditional classifiers, final report
Xiaoyan Li	CNN, presentation slides, final report
Jicheng Li	project topic formulation

I approve the content of the final report (please add your signature below):

Shaojun Wang:_____

Xiaoyan Li :_____

Jicheng Li :_____

Expression Recognition

CSE 881 Final Project

Shaojun Wang
Michigan State University
wangsha8@msu.edu

Xiaoyan Li
Michigan State University
lixiaoy5@msu.edu

Jicheng Li
Michigan State University
lijichen@msu.edu

ABSTRACT

Face recognition is a hot topic in machine learning. It is useful in many situations. Expression recognition is also a useful sub-field and have plenty of practical use. In this project, the team trained traditional classifiers and convolutional neural network on CK+ dataset and tested on AffectNet. Results have shown even a simple convolutional neural network could result in high training accuracy, but the accuracy on test set is poor. The traditional classifiers did not perform well on training nor test data.

KEYWORDS

Expression Recognition, Facial Expression, Expression Detection

ACM Reference Format:
Shaojun Wang, Xiaoyan Li, and Jicheng Li. 2021. Expression Recognition: CSE 881 Final Project. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Recognizing facial expression comes in natural for humans, but it is not necessarily an easy task for machine. It is challenging for machine because such task often involves high dimensional space. It the images are 50×50, then the dimension is already at a whooping 2500. There could be many potential application for such automatic expression detector. It could be used in lie detector to monitor change in micro expression, be implemented on online psychological consulting service to detect the status of the subject, and be applied in public safety to identify potential perpetrator a head of time. This project is an endeavor to develop classifiers to recognize human facial expression. The models developed did not perform well on the test set; nevertheless, over the course of this project, the team made great contribution and learned many frameworks, classifiers, and preprocessing techniques.

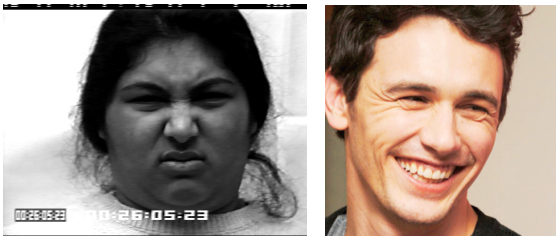
2 RELATED WORK

There are multiple existing works on expression recognition. One of the recent work [4] involved multiple facial expression datasets. The researchers built models on these datasets separately, and the highest accuracy reported was 92.45%. The researchers also built models on the combined dataset, and the highest accuracy reported was 73.58%. One of the classical work [5] was able to achieve accuracy of 92.30% using multilayer perceptron on a dataset with posed expression. Little research was found on testing models built on posed expression with real life expression.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3 DATA COLLECTION

The datasets used in this project are CK+ [1] and AffactNet [2]. CK+ contains 593 image sequences collected under experiment setting, and 327 images are labelled with emotion. The images that did not come with a label were manually labeled based on visual inspection. Figure 1a shows one of the images in this dataset. AffectNet contains 460,000 manually annotated images, and these images were collected in the wild. These images were scrapped off from the Internet. Figure 1b shows one of the images in AffectNet. The expression labels of AffectNet is a super set of CK+, and the disjoint images were removed. In the end, the labels are the followings: Angry, Contempt, Disgust, Fear, Happy, Sadness, and Surprise. CK+ is 1.7 G in size and AffectNet is 55 G in size. CK+ was used as the training set, and AffectNet was used as the test set. The purpose for that is to see, when models were train on small data set with subjects posing expressions, how they would perform in real life scenario. The aspiration was the exaggerated expressions could expose important attributes of these expressions and help the classify to perform well in real life scenarios.



(a) CK+ Image **(b) AffectNet Image**

Figure 1: Raw Image

4 DATA PREPROCESSING

A number of preprocessing methods were applied to the raw images. Classifiers were trained on images went through gray scale, Gaussian blur, and Canny edge detection. First, all images in the training set were examined visually to verify the correctness of labels and images with no label were labeled manually according to the expression code. Secondly, all images were processed and converted to gray scale. The faces were extracted and cropped to the same size and normalized. Finally images were converted to csv file. The performance of the classifiers will be compared to

investigate which preprocessing method is more effective. These preprocessing methods are explained in the following subsections.

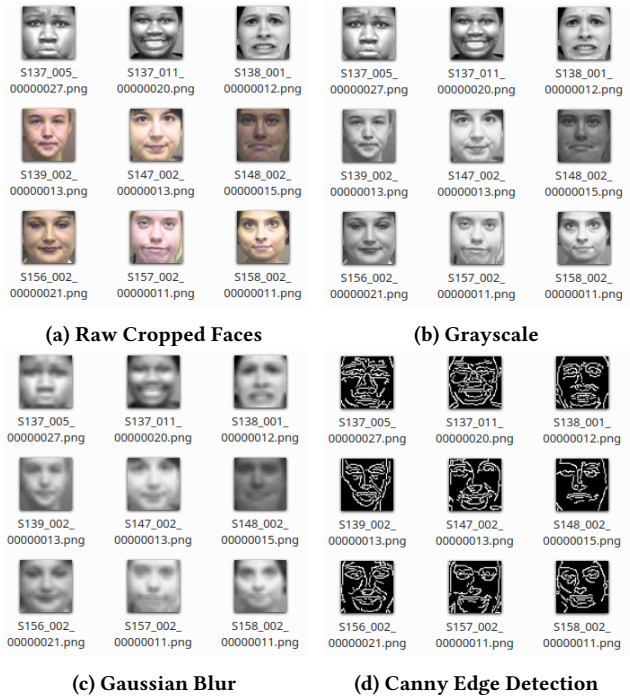


Figure 2: Preprocessing

4.1 Cropping

The raw images are not suitable for training directly. As shown in figure 1a, the face region does not occupy the entire frame, and it includes non-relevant information. Python package dlib was used to crop square regions of the faces out of the images. The cropped faces were resized to 50×50. The result can be seen in Figure 2a.

4.2 Grayscale

Many of the original images are in RGB. Color does not play an important role in identifying facial expression. A filter was applied to the cropped faces to convert them into grayscale. The result can be seen in Figure 2b.

4.3 Gaussian Blur

A clear image is definitely preferred for human. However, it is inevitable to have noise in the images, and this is undesirable for training. To alleviate the impact of noise, a 5×5 Gaussian kernel was applied to the grayscale images. This may help the machine to learn the overall shape of the expression instead of focusing on individual pixels. The result can be seen in Figure 2c. In this figure, the expressions are still clear to human eyes.

4.4 Canny Edge Detection

Cropping the faces out of the images does help to reduce the impact of the background, but there is still some leftover. The performance of the classifier could vary drastically in different lighting conditions as well. To recognize expression, all machine needs might be just the contour of the face. Canny edge detection was applied to the grayscale images. The result can be seen in Figure 2d. Although the expressions in the images are hardly recognizable by human, it could potentially work for machine. If this method is proven to be successfully, the classifier will be robust to noise and change in environmental settings.

5 METHODS

In this project, two approaches were adopted, transitional classifiers and convolutional neural network (CNN). The traditional classifiers used are Gaussian process classifier (GP), logistic regression classifier (LOGREG), support vector machine classifier (SVM), multilayer perception classifier (MLP), K nearest neighbors classifier (KNN). CNN was chosen because it is regarded as the state of art of image recognition, and it is trained based on grayscale transformation of the CK+ dataset. The number of each label for the training set are the followings: 45 Angry, 18 Contempt, 59 Disgust, 25 Fear, 69 Happy, 28 Sadness, and 83 Surprise.

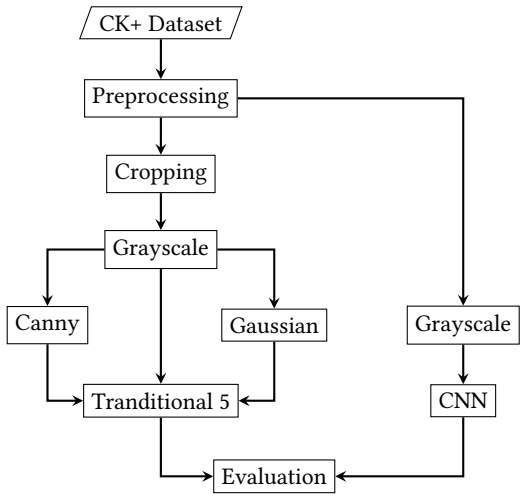


Figure 3: Workflow

5.1 Classical Classifiers

These five classifiers were trained and the hyper-parameters were tuned using Python package Auto Tune Models (ATM) [3]. The five classifiers were trained separately on datasets that went through grayscale, Gaussian blur, and canny edge detection preprocessing. The ATM package models the performance score as Gaussian distribution based on hyper-parameter values. The value of hyper-parameter to evaluate is determined based on Bayesian optimization. In total, 300 candidate models were evaluated.

5.2 Convolutional Neural Network

Convolutional neural net work is a popular model which has been widely used for image classification, such as face recognition and object recognition. The kernel in CNN model is effective to capture local edge features which is difficult to capture by traditional models.

Another advantage of CNN model is pooling layer. The pooling layer could increase resistance to noise variations in images. Also, large scale CNN model is easy to achieve high accuracy especially when the dataset is large.

In this project, the CNN model is composed of six layers. For the first convolutional layer, input channel is 1 and output channel is 16, kernel size is 5, stride is 1, and padding is 0. The second convolutional layer is RelU layer. The third layer is a max pooling layer with kernel size 4. The fourth layer is to convert the 3-d tensor to 1-d. The fifth layer is a fully connected layer, with 87616 inputs and 8n outputs, the sixth layer is another fully connected layer with 8 inputs and 1000 inputs .The last layer is another fully connected layer with 1000 inputs and 8 outputs. The 8 output numbers is computed by softmax function as probability. Comparing with true labels, cross entropy loss is used to calculate loss. Stochastic gradient descent is used to update parameters. For each training process, batch size 10 is used and learning rate is set to be 0.01. In testing process, accuracy is calculated, confusion matrix heat map is generated.

6 RESULT

The classifiers were evaluated on 10% of AffectNet based on random draw. The number of labels are the followings: 2324 Angry, 382 Contempt, 401 Disgust, 625 Fear, 12970 Happy, 2455 Sadness, and 1383 Surprise. The confusion matrices reported in the following sections are visualized; lighter regions indicate large number, and darker regions indicate small number.

6.1 Gaussian Process Classifier

GP did not perform significantly better than random guess on the training set, but it performed drastically better on the test set. The improvement is questionable. Judging from the confusion matrices, GP might have gained this improvement purely for classifying a large amount of images as happy.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Training	0.2626	0.2155	0.1779	0.1429
Testing	0.3982	0.4847	0.2975	0.1429

Table 1: Training Accuracy vs Test Accuracy

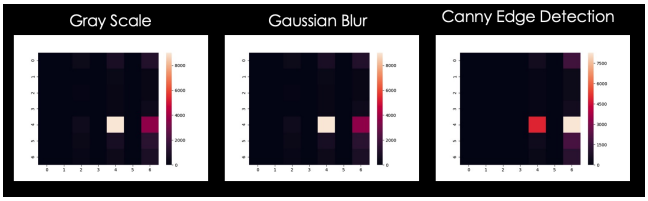


Figure 4: GP Confusion Matrix

6.2 K Nearest Neighbors Classifier

KNN performed poorly on the training set and the test set. Canny edge detection had helped KNN to retain some performance on the test set.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Training	0.3314	0.3260	0.2007	0.1429
Testing	0.0599	0.09444	0.2799	0.1429

Table 2: Training vs Test Accuracy

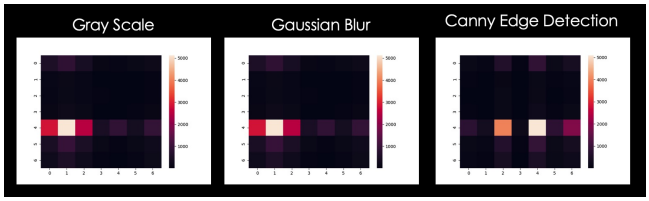


Figure 5: KNN Confusion Matrix

6.3 Logistic Regression Classifier

LOGREG had great performance on the training set, but it did not outperform random guess significantly on the test set. Preprocessing did not help LOGREG to do any better.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Training	0.5378	0.6099	0.3876	0.1429
Testing	0.2635	0.2112	0.2516	0.1429

Table 3: Training vs Test Accuracy

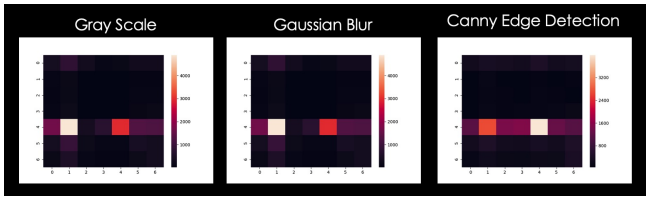


Figure 6: LOGREG Confusion Matrix

6.4 Multi-layer Perception Classifier

MLP had great performance on the training set, but it did not outperform random guess significantly on the test set. Canny edge detection helped MLP a great deal to retain some performance on the test set.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Training	0.5801	0.6038	0.4083	0.1429
Testing	0.1932	0.1899	0.3125	0.1429

Table 4: Training vs Test Accuracy

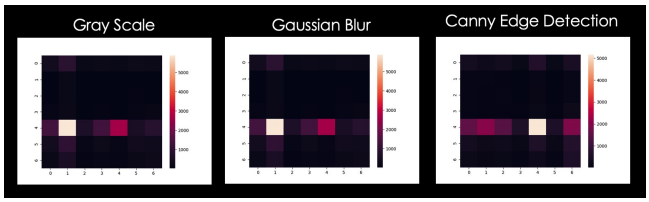


Figure 7: MLP Confusion Matrix

6.5 Support Vector Machine Classifier

The observation is similar to MLP. SVM had great performance on the training set, but it did not outperform random guess significantly on the test set. Canny edge detection helped SVM a great deal to retain some performance on the test set.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Training	0.5419	0.6025	0.3796	0.1429
Testing	0.1724	0.1457	0.2791	0.1429

Table 5: Training vs Test Accuracy

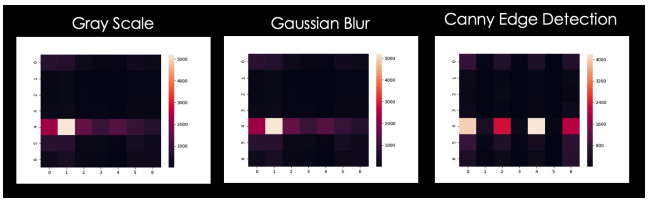


Figure 8: SVM Confusion Matrix

6.6 Traditional Classifiers Overall

For the 5 traditional classifiers, the overall observations are performance decreasing greatly on test set, Canny edge detection did not help training accuracy but helped test accuracy, and test accuracy results were heavily influenced by happy label.

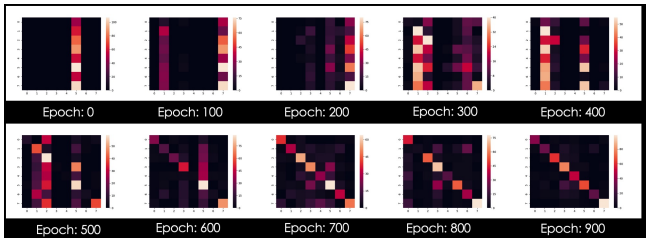
For each preprocessing method, an ensemble classifier was made based on the majority vote of the five classifiers. The result once again highlighted the importance of Canny edge detection. Although Canny edge detection did not help the training accuracy, it does play an important role in helping the models to operate in noisy environment.

	Grayscale	Gaussian Kernel	Canny Edge	Random Guess
Testing	0.2081	0.1977	0.3207	0.1429

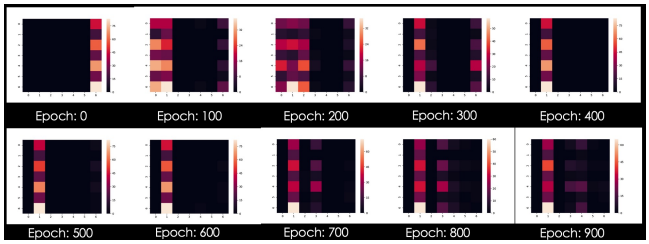
Table 6: Training vs Test Accuracy

6.7 CNN

Figure 9 illustrates how the confusion matrices change over the course of training. In the two subfigures, it is obvious that, as the training progresses, training accuracy increased gradually, but the test accuracy remained subpar. Figure 10 and Figure 11 shows the training accuracy and test accuracy. The training accuracy could achieve as high as over 90%. The trend is training accuracy kept improving, but it did not seem to converge for the test accuracy. The high accuracy of training data is likely to be the result of over fitting. Although at the end, the test accuracy seemed to improve, the performance was still marginal. To achieve a high test accuracy, it might require a large number of iteration, if it ever could.



(a) Training Set Confusion Matrix



(b) Test Set Confusion Matrix

Figure 9: CNN Confusion Matrix

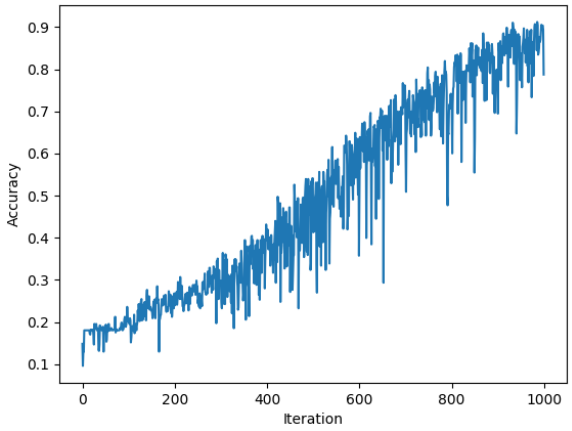


Figure 10: Training Set Accuracy

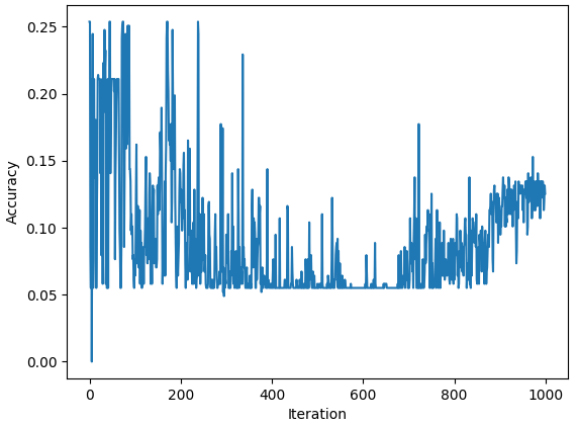


Figure 11: CNN Test Accuracy

7 WEB PROTOTYPE

A web prototype has been developed. This site allows users to upload image to process expression classification. This is a drag and drop process. The server handles preprocessing including cropping face, resizing, performing Canny edge detection. The server classifies the image based on ensemble of KNN, LOGREG, MLP, and SVM, and the web interface displays the label on a popup. GP was not used in the ensemble because it is unclear where its performance on the test set came from. The site only runs on localhost currently due to difficulty with deployment.

8 CONCLUSION

Despite the effort, this project was not able to produce a good model for real life use based on posed expression. The results have a number of implications. First of all, to recognize natural expression, the models might have to be trained on natural faces. Secondly, the performance of the models severely deteriorates on the test set, suggesting the models might be trained in a way to recognize the statistical attributes of the dataset instead of the expressions. Last but not least, the size of the training set might not be adequate to achieve the task. Even though the project itself was not successful, there are important lessons learned. KNN has proven not a good classifier for expression recognition, and perhaps this conclusion could be extended to image classification in general. It is possible that certain preprocessing technique does not help training performance but improves performance on the test set; in this case Canny edge detection helped the models overall in noisier environment than training set. CNN remains a strong contender in image classification; however, when the settings of the test set vary significantly, the performance will suffer.

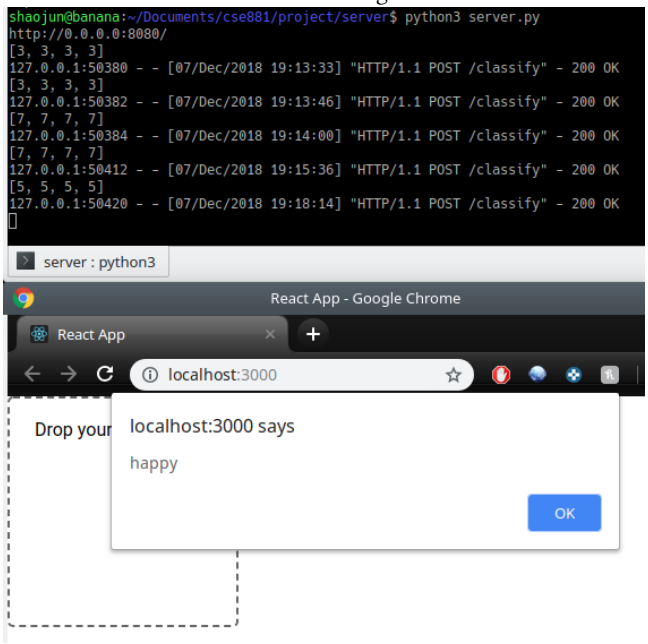
A SOURCE CODE

The source code is hosted at the following link:
https://github.com/wangsha8/cse881project

"cnn" directory contains work associated with CNN classifier. "dataset" directory contains the training data. The "test" subdirectory contains a link to the test data (the size exceeds the limit of github). "traditional5" directory contains work associated with the traditional classifiers. "website" directory contains work associated with prototype.

B WEBSITE INTERFACE

The web interface looks like the followings:



All the user needs to do is drag an image in. The server will process the image. The image does not have to contain a face, and in this case server will send back a warning. The image does not have to be preprocessed either. The server will crop the face out, apply canny edge detection, and give back the expression label based on ensemble of KNN, LOGREG, MLP, and SVM.

Please contact the project team for live demo.

REFERENCES

[1] Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. 2010. 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*. IEEE, 94–101.

[2] Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. 2017. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *arXiv preprint arXiv:1708.03985* (2017).

[3] Thomas Swearingen, Will Drevo, Bennett Cyphers, Alfredo Cuesta-Infante, Arun Ross, and Kalyan Veeramachaneni. 2017. ATM: A distributed, collaborative, scalable system for automated machine learning. In *IEEE International Conference on Big Data*.

[4] Jiabei Zeng, Shiguang Shan, and Xilin Chen. 2018. Facial expression recognition with inconsistently annotated datasets. In *Proceedings of the European conference on computer vision (ECCV)*. 222–37.

[5] Zhengyou Zhang, Michael Lyons, Michael Schuster, and Shigeru Akamatsu. 1998. Comparison between geometry-based and gabor-wavelets-based facial expression recognition using multi-layer perceptron. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*. IEEE, 454–459.