An Ensemble of Face Recognition Algorithms for Unsupervised Expansion of Training Data

Jeffrey Dale*
Department of Computer Science
Missouri State University
Springfield, MO, USA
jdale@ieee.org

Anthony Clark
Department of Computer Science
Missouri State University
Springfield, MO, USA
anthonyclark@missouristate.edu

***Abstract*—Facial recognition is a classical problem in computer vision. The accuracy of face recognition algorithms is crucial in practice, as systems are increasingly secured with biometric locks. However, the performance of these algorithms is heavily dependent upon the size of the training data. This paper proposes an unsupervised ensemble method for expanding the set of training faces when only a single labeled face per subject is known. We show that the ensemble's confidence measure is sufficient to expand the training set to the point where more sophisticated algorithms can take over in classification.**

***Index Terms*—confidence, unsupervised, face, ensemble**

I. Introduction

Modern face recognition methods are primarily built on deep learning frameworks such as convolutional neural networks [1] or generative adversarial networks [2]. While such methods boast outstanding accuracy, they require a large number of training samples to achieve this performance. Even Facenet [3], using the Inception-ResNet architecture [4] and trained for feature extraction to 99.65% accuracy on the large VGGFace2 dataset [5], does not perform well on new datasets when the number of training faces per subject is very small. However, with just a few more training faces, Facenet consistently achieves perfect or near-perfect accuracy.

In this paper, we propose a confidence measure to estimate the likelihood that a predicted label is correct. We use this confidence measure to design an ensemble that gathers high confidence labels and faces by assuming that, under certain circumstances, the ensemble's predicted label is correct. Newly labeled images are then used to train more sophisticated algorithms, or optionally to retrain the component algorithms of the ensemble.

An example use case for the proposed ensemble is, in conjunction with face location/detection methods, to tag subsets of people from a known group in a set of photographs or video feed. One scenario where this would prove useful is in smart surveillance devices. A user could place a camera in or around their house, and after being provided with one picture of each member of the household, the system would progressively get better at identifying these people. Setting a threshold on the association of faces to known labels, the system could then be used to send alerts when unknown people appear on camera.

II. Background

In this section, we provide a brief overview of the individual facial recognition algorithms used in the proposed ensemble. Our choice of algorithms stems largely from popularity and availability of implementation. For these reasons, we first chose to use the three facial recognition algorithms bundled with OpenCV [6]: Eigenfaces [7], Fisherfaces [8], and Local Pattern Binary Histograms [9]. OpenCV provides a unified interface to each of these algorithms, including methods for training and prediction. The prediction methods not only return an algorithm's prediction for a given face, but also the computed distance between the given face and the face associated with the predicted label. In addition to the three algorithms from OpenCV, we implemented the Randomized PCA [10] algorithm to add to the diversity of the ensemble.

We want to emphasize that the choice of algorithms in this paper is not necessarily a recommendation for what should be used in practice. Instead, the emphasis in this paper is on our proposed metric for evaluating the **confidence** of a face recognition algorithm and on our confidence-based ensemble methodology. Thus, flexibility was of the utmost concern when developing this approach. One should consider many factors relating to the application domain when building the ensemble, such as available time to train, desired prediction times, availability of training data, and number of faces to learn. While we use four component algorithms with one training face per subject, both the number of component algorithms and number of training faces can also change as needed. A thorough suvey on single-face-per-subject recognition is provided by Tan et al. [11].

978-1-7281-1360-9/18/$31.00 ©2018 IEEE
DOI 10.1109/CSCI.2018.00072

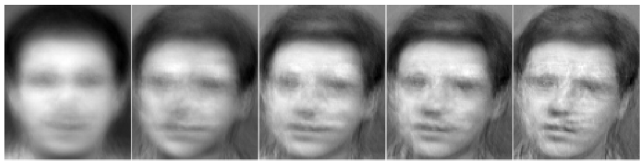


Fig. 1: Eigenfaces reconstructions using increasing numbers of components. From left to right the images were reconstructed using: 1, 25, 50, 75, and 100 principle components.

A. Eigenfaces Algorithm

Eigenfaces is one of the oldest and most well known face recognition algorithms to date. The idea of Eigenfaces is to find characteristic faces for each subject in the database. To combat the high dimensionality of the "face space," principal component analysis (PCA) [12] is used to only consider the components which contribute the most. To associate a new face with a face from the already labeled training set, a nearest neighbor is determined after projecting the new face onto the PCA subspace and calculating the shortest distance between this projection and all images from the labeled training set.

The Eigenfaces algorithm takes a single parameter: the number of components to use in principal component analysis. Using fewer components improves computation time, but makes it more difficult for the algorithm to distinguish between two similar faces. For larger datasets, one should choose a larger number of components. Figure 1 illustrates the effect of varying the number of components.

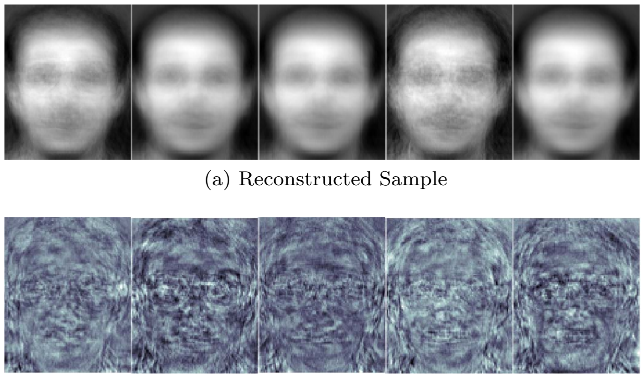
B. Fisherfaces Algorithm

In a similar fashion to Eigenfaces, Fisherfaces [8] uses linear discriminant analysis (LDA) [13] to classify faces. LDA is similar to PCA, however LDA maximizes between-class distances and minimizes within-class distances, whereas PCA does only the latter. As a consequence of using LDA instead of PCA, Fisherfaces is more invariant to illumination changes than Eigenfaces. Just as in Eigenfaces, classifying a new face with Fisherfaces is done using nearest neighbor classification.

Fisherfaces is also parameterized by the number of components to use during linear discriminant analysis. As demonstrated in Figure 2, the effect of this parameter is similar to that of PCA in the Eigenfaces algorithm.

C. Local Binary Pattern Histograms

The Local Binary Pattern Histograms (LBPH) algorithm [9] works by analyzing the intensities of pixels surrounding a given pixel. Each surrounding pixel with an intensity greater than or equal to the intensity of the given pixel is given a value of 1; all other surrounding pixels are set to 0. These 0 and 1 bits can be converted to a single integer value that is then used to represent the value for the given pixel; this process is depicted in Figure 3a. LBPH can be computed using a circle around the pixel in question, defined by a radius parameter with


(a) Reconstructed Sample

(b) Fisher Faces

Fig. 2: (a) Fisherfaces reconstructions using increasing numbers of components. Since the variation between reconstructed faces is less noticeable, we have also provided (b) an example of the generated Fisherfaces.

a specified number of points (neighbors) to sample in this circle.

Since LBPH only compares relative pixel intensities, it is almost entirely invariant to changes in illumination (illustrated in Figure 3b).

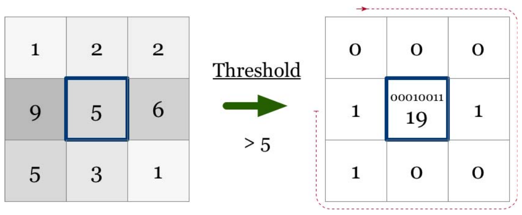
D. Randomized PCA Algorithm

Randomized Singular Value Decomposition (SVD) [10] was proposed as a way to calculate an approximate singular value decomposition of a matrix when relatively few singular vectors are desired. This approach utilizes a random Gaussian test matrix to make the computation of the approximate SVD of a large matrix more time and resource efficient.

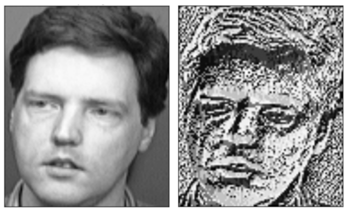
Randomized SVD can be applied to the face recognition problem as a drop-in replacement for the calculation of the principle components in Eigenfaces [14], yielding a different method of recognizing faces: Randomized PCA. The method of reconstructing these faces from principal components, shown in Figure 4, is the same as in Eigenfaces. While Randomized PCA for face recognition is very similar to Eigenfaces, with both attaining similar accuracy, the different method of computing principal components changes which faces the algorithm is capable of recognizing. This makes Randomized PCA a valuable addition to our proposed ensemble.

E. Ensemble Methods

The goal of ensemble methods is to improve performance (i.e., increase accuracy) by combining existing techniques for solving a problem. The general notion is that through a set of techniques, some techniques will have an advantage in certain situations, and through some form of consensus the advantages can be combined. In the context of face recognition, we combine the predictions of Eigenfaces, Fisherfaces, LBPH, and Randomized SVD to yield results that are more accurate than any individual algorithm on its own. Some common techniques in designing ensemble methods



(a) LBPH thresholding



(b) LBPH representation

Fig. 3: (a) An example of the thresholding process used during LBPH. The given pixel has a thicker border. The left grid shows the original pixel intensities, and the right grid shows the thresholded values. The dashed line indicates how the final given pixel value is calculated. (b) an LBPH representation of a face from the AT&T dataset.

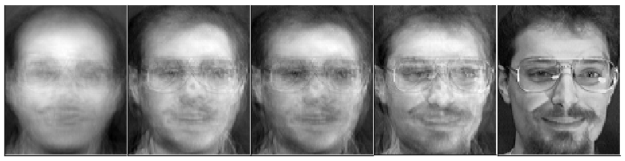


Fig. 4: Reconstruction of a face from the AT&T faces dataset using Randomized PCA with 1, 5, 10, 20, and 50 components, respectively. Compare to Figure 1 which appeared more blurry even with higher numbers of components.

include the Bayes Optimal Classifier [15] (typically not feasible in large problems), Bayesian Model Averaging [16], Boosting [17], and Stacking [18]. Our proposed ensemble does not quite fit into any of these categories, but borrows techniques from both Stacking and Boosting.

III. Confidence Measure

The primary purpose of this paper is to develop a measure that, given the predicted labels and associated distances from several existing algorithms, returns one predicted label and a confidence value that describes the overall confidence of the component algorithms. For classification tasks, it is common to use a simple voting strategy called majority voting. If two methods assign a sample to class "A" and one method assigns the sample to class "B", a majority voting strategy would simply assign the sample to class "A" as it had the most votes. We can improve this voting strategy by assigning weights to each vote. Choosing these weights is not trivial, however.

In the proposed ensemble, we utilize the distance measure returned by each algorithm to improve voting. Consider a face with an unknown label that is presented to the algorithm. We will denote this face as a random vector $\boldsymbol{X}_f$. Similarly, we will denote the single known face for each subject as $\boldsymbol{Y}_i$, $i = 1, 2, ..., k$ where k is the number of unique subjects present in the dataset.

Note that $\boldsymbol{X}_f$ and $\boldsymbol{Y}_i$ are usually projections of a raw face vector onto another subspace, however the theory presented hereafter is unchanged in these cases. Also, when there is more than one training face for each subject, $\boldsymbol{Y}_i$ is intuitively thought of as an "eigenface", which is a generalization of features found in each training face.

We define Z_f to be the distance between $\boldsymbol{X}_f$ and its nearest neighbor face $\boldsymbol{Y}_i$ is given by

$$Z_f = \min_{i=1,2,\ldots,k} \|\boldsymbol{X}_f - \boldsymbol{Y}_i\|_2^2 \ . \tag{1}$$

We can see that Z_f is in fact a random variable, but its distribution function $f_Z(z)$ is unknown.

To estimate the distribution of Z, we use kernel density estimation to approximate $f_Z(z)$ using a Gaussian kernel [19]. Suppose an algorithm returns a predicted label for a face with distance z; we define the confidence of this prediction to be the probability that a randomly classified face has a higher distance value, that is,

$$C(z) = \Pr(Z \geq z) = \int_z^\infty f_Z(t)\ \mathrm{d}t \ . \tag{2}$$

To approximate the improper integral in (2), we transform the limits of integration and approximate using Gaussian quadratures [20]:

$$C(z) = \int_0^1 f_Z\left(z + \frac{t}{1-t}\right) \frac{\mathrm{d}t}{(1-t)^2} \ . \tag{3}$$

The confidence value $C(z)$ ranges from zero to one, with a value of one indicating a high likelihood that the predicted label associated with the distance z is correct.

IV. Proposed Ensemble

The key observation motivating the ensemble is that the model for each algorithm can be retrained on testing data when all algorithms agree on the same label. Formally, the steps of the ensemble are as follows:

1) Divide the dataset into testing and training. The training dataset consists of one randomly selected image (with a known label) for each subject, and the testing dataset contains all remaining data. The true labels of the testing dataset are only used in evaluating the performance of the ensemble.
2) Train each algorithm in the ensemble using the training dataset with the known labels.

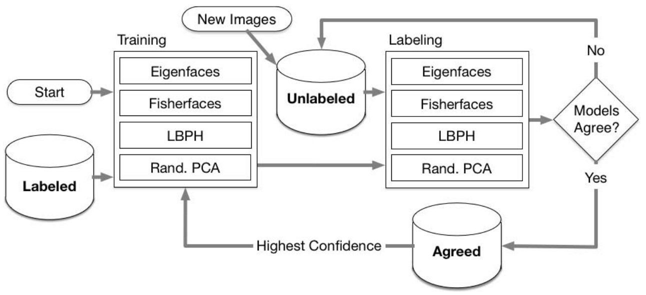


Fig. 5: Flow chart of proposed ensemble method.

3) Obtain the distance to the nearest neighbor match of each algorithm's prediction, and use these distances as random samples to estimate the density function of Z from equation (1) using kernel density estimation.
4) Predict the label of each image in the testing dataset using component algorithms. For all cases where each algorithm agrees on labels, store the predicted label with the sum of the confidence values of each algorithm's prediction.
5) Order each agreement-confidence tuple by sum of confidence in descending order.
6) Discard all agreements whose confidence is lower than a given threshold.
7) Update the models of each algorithm with the remaining faces and agreed-upon labels after step 6.
8) Repeat steps 3-7 as many times as desired. The number of repetitions will be referred to as the number of *passes*. The density function is updated at each pass, since the distribution of distances will change as each algorithm's model is updated.
9) Each algorithm votes on the labels of the remaining testing images. At this stage, the algorithms have some disagreement regarding the label of all remaining images. The weight of each vote is the confidence measure of that algorithm's distance, as described in equations (2) and (3).

This process is visualized in Figure 5. The primary use case of this ensemble is to augment a small existing training set with high confidence matches. The ensemble can also be used directly as a face recognition algorithm. Using kernel density estimation makes no assumptions about the distribution of distances, so in theory, any algorithm that returns a distance measure that is smaller for closer matches will fit into the ensemble as-is. Adding more algorithms to the ensemble scales the complexity linearly with the complexity of the added algorithm, and doing so is a simple process. When using a larger number of algorithms, one may wish to assume a prediction is correct when a strong majority agrees on a label, rather than requiring 100% agreement. The Python implementation of this ensemble is available via GitHub at https://github.com/jeff-dale/face-rec-ensemble/.

TABLE I: Accuracy on each dataset after 5 replicate experiments. MV Ensemble indicates the ensemble where labels are assigned solely by majority voting.

	AT&T Faces		**Yale Faces**	
Method	Best	Average	Best	Average
Eigenfaces	74.17%	69.72%	13.03%	12.05%
Fisherfaces	73.33%	69.33%	11.78%	11.01%
LBPH	83.61%	81.95%	29.63%	27.31%
Rand. PCA	74.17%	69.95%	13.12%	11.98%
MV Ensemble	74.44%	70.33%	13.16%	12.05%
Best Guess	86.39%	84.67%	29.97%	27.73%
Ensemble P0	76.94%	73.33%	16.60%	15.40%
Ensemble P1	96.39%	94.33%	80.47%	79.10%
Ensemble P2	98.33%	95.73%	83.32%	82.57%
Ensemble P3	98.61%	95.73%	85.16%	84.06%

V. Experimental Results and Discussion

The proposed ensemble was tested on two benchmark datasets: the AT&T faces dataset [21] and the Extended Yale Database B [22]. The AT&T faces dataset has 400 faces from 40 subjects (10 images per subject) and the Extended Yale Database B has 2424 faces from 38 subjects with a varying number of faces per subject.

The parameters of Eigenfaces, Fisherfaces, LBPH, and Randomized PCA were tuned using the differential evolution (DE) algorithm [23]. The exact parameter values can be seen in our code repository.

The remainder of this section is broken into two parts: (1) using the ensemble to augment the training set and (2) using the ensemble as a face recognition algorithm.

A. Ensemble for Training Set Augmentation

As previously mentioned, an important trait of the proposed ensemble is its ability to quantify its confidence in a prediction as a function distance measures of its component algorithms. To show that this confidence measure is performing as expected, Figure 6 shows a receiver operating characteristic (ROC) curve obtained by varying the confidence cutoff threshold. It is clear from Figure 6 that false positives in classification are rare when the confidence measure is high, as is essential in a good confidence measure.

Using this approach to augment existing training data allows more powerful classifiers, such as Facenet [3], to take over in classification beyond a certain point. When running Facenet with a model trained on VGGFace2 [5], it only took three training faces per subject for classification accuracy to exceed 97%, however with fewer than three training faces per subject, classification accuracy was less than 60%. Based on the ROC curve in Figure 6, we see that the ensemble is capable of providing the additional labels needed by Facenet to perform well, with a high

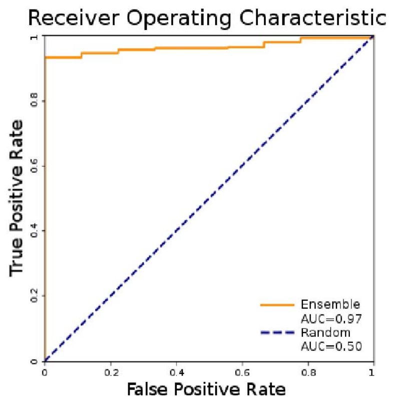


Fig. 6: ROC curve showing how varying the confidence threshold in step 6 of Section IV affects the true and false positive rates of the ensemble's predicted labels on the AT&T dataset on the first pass. True positives are when agreements have the correct label and false positives are when agreements have the incorrect label. The blue dotted line indicates the expected performance of a random classifier.

likelihood of having no added false positives (incorrectly labeled faces).

B. Ensemble for Face Recognition

Table I shows the accuracy achieved by the proposed ensemble on the AT&T and Yale faces datasets. In the table, we provide accuracy results for each component algorithm in the ensemble, as well as results for two baseline ensemble methods and our proposed method. Since our proposed method includes multiple passes, we have included the accuracy achieved after each pass. From the table, we observe that each individual algorithm had average accuracy of around 70 to 80% on the AT&T faces dataset. The naïve majority voting (MV) ensemble, which uses equally weighted votes for each algorithm, performed better than Eigenfaces, Fisherfaces, and Randomized PCA on average, but not better than LBPH. For the sake of comparison with our proposed method, we have also introduced a new ensemble method called *Best Guess*, which simulates an ensemble with perfect voting, i.e. always correctly labels a face if any single algorithm correctly labeled that face. The Best Guess ensemble is clearly only for benchmarking, as it requires ground truth information on the accuracy of each algorithm. It is used in this paper to demonstrate that the proposed ensemble can achieve higher accuracy than *any* voting-only ensemble (with no re-training). This unimplementable method gives an accuracy of around 85% on the AT&T dataset.

On the AT&T faces dataset, the *P0* ensemble (ensemble pass 0, confidence-based voting and no re-training) achieved an average accuracy higher than the majority voting ensemble. These results indicates that our confidence-based voting has merit. After 1 pass of re-training, the proposed ensemble outperforms all methods shown in the table, including the Best Guess ensemble. After 2 passes, we begin to notice diminishing returns on the accuracy of the ensemble, and after 3 passes, we achieve a peak average accuracy of just over 95%. The point of diminishing returns can be more clearly seen in Figure 7.

TABLE II: Number of gathered labels and classification time per face (in seconds) for each pass of the ensemble. Time was measured on an standard Intel i7 processor running all 5 replicates of the ensemble in parallel.

	AT&T Faces		**Yale Faces**	
Number of Passes	Labels	Time	Labels	Time
Ensemble P0	40	0.12	38	0.34
Ensemble P1	297	0.17	976	1.59
Ensemble P2	358	0.17	1460	2.21
Ensemble P3	370	0.17	1637	2.44

We observe similar results for the Extended Yale Database B, however, with lower accuracies overall (Table I). The lower accuracies can be attributed to the fact that the Extended Yale Database B is much larger than the AT&T database, and contains subjects in multiple poses with varying illumination conditions. We see that for larger datasets, only one retraining pass is needed to get a large jump in accuracy. Using more than one or two passes is only recommended when training time is not a factor and accuracy is absolutely crucial.

Since performance peaks after three passes for our test datasets, we chose to limit our ensemble to three passes. Forcing the ensemble to continually retrain will lead the ensemble to eventually choose poor faces to add to the training set, particularly when using stochastic algorithms like Randomized PCA, which can potentially assign different labels to the same face when run multiple times on the same data. Figure 7 clearly shows that the confidence measure and retraining is useful, however, it also shows that at some point further passes quickly becomes unnecessary.

Table II shows how the number of gathered labels grows after each pass, as well as the average time taken to classify each face in the testing set. Even with the heavy computations in our algorithm being done in Python (with Numpy/Scipy), such as the repeated kernel density estimation and quadratures, we see that faces can very easily be classified in real time on the AT&T dataset. After several passes on the Yale dataset however, classification slows down. Some ways to improve the classification time include (i) using a faster language than Python, (ii) parallelizing the ensemble predictions, and (iii) utilizing the GPU for batch testing.

VI. Conclusion

In this paper, we presented: (i) a universal method for measuring an algorithm's confidence based on distance values, (ii) a voting strategy based on this confidence

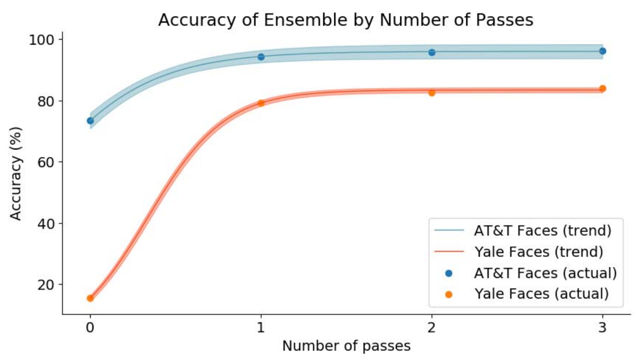


Fig. 7: Average accuracy of the proposed ensemble on the AT&T and Yale faces datasets by number of passes. Each point on the graph is a data point, while the lines are generated by fitting a logistic curve to the points. The shaded area is the standard deviation of each replicate experiment, also fitted with a logistic curve for smoothness.

measure, (iii) an ensemble method that uses the proposed confidence-based voting to achieve high accuracy on well-known face recognition datasets, and (iv) the ability of this ensemble to effectively expand small training data for use in larger scale classifiers. This ensemble is robust and flexible, and is useful in a variety of different real-world scenarios. Our experiments show that confidence-based voting achieves higher accuracy than majority voting in the average case, and returns a useful continuous value between zero and one to quantify its prediction, providing more than a "yes or no" in its decisions. The ensemble method outperforms each of its components by a large margin, and even achieves higher accuracy than an infeasible Best Guess ensemble, allowing the ensemble to correctly identify faces where all components of the ensemble failed.

Most beneficial of all, *any* face recognition algorithm that provides a distance or confidence value to its predictions can be dropped into the ensemble. One could even create a so-called "meta-ensemble" containing other ensembles. Even when component algorithms have individually low accuracy, as in our experiments with the Yale dataset, the ensemble is still able to gather more high-confidence labels and achieve high accuracy.

Finally, our ensemble method can easily be updated with new algorithms–a useful characteristic in the rapidly changing field of computer vision. When a new state of the art technique is discovered, simply drop it into the ensemble and observe the increased performance.

References

[1] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 1

[2] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning gan for pose-invariant face recognition," in *CVPR*, vol. 3, no. 6, 2017, p. 7. 1

[3] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823. 1, 4

[4] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, vol. 4, 2017, p. 12. 1

[5] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference on*. IEEE, 2018, pp. 67–74. 1, 4

[6] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000. 1

[7] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991. 1

[8] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 711–720, 1997. 1, 2

[9] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002. 1, 2

[10] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011. 1, 2

[11] X. Tan, S. Chen, Z.-H. Zhou, and F. Zhang, "Face recognition from a single image per person: A survey," *Pattern recognition*, vol. 39, no. 9, pp. 1725–1745, 2006. 1

[12] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal component analysis*. Springer, 1986, pp. 115–128. 2

[13] A. J. Izenman, "Linear discriminant analysis," in *Modern multivariate statistical techniques*. Springer, 2013, pp. 237–280. 2

[14] P.-G. Martinsson, V. Rokhlin, and M. Tygert, "A randomized algorithm for the decomposition of matrices," *Applied and Computational Harmonic Analysis*, vol. 30, no. 1, pp. 47–68, 2011. 2

[15] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, "The multilayer perceptron as an approximation to a bayes optimal discriminant function," *IEEE Transactions on Neural Networks*, vol. 1, no. 4, pp. 296–298, 1990. 3

[16] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky, "Bayesian model averaging: a tutorial," *Statistical science*, pp. 382–401, 1999. 3

[17] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*. Springer, 1995, pp. 23–37. 3

[18] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992. 3

[19] B. W. Silverman, *Density estimation for statistics and data analysis*. CRC press, 1986, vol. 26. 3

[20] A. H. Stroud and D. Secrest, "Gaussian quadrature formulas," 1966. 3

[21] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*. IEEE, 1994, pp. 138–142. 4

[22] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 643–660, 2001. 4

[23] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997. 4