Ensemble Learning for Multiple CNN Architectures on Face Classification

Yidong He
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA
yidongh@andrew.cmu.edu

Zhefan Xu
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA
zhefanx@andrew.cmu.edu

Hongyi Zhou
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA
hzhou3@andrew.cmu.edu

Abstract

Face recognition performs the task of determining the identity of the image of a person's face. This task is essentially a N-way classification problem. A series of Convolutional Neural Network classifier has been developed to increase the accuracy of classification. In this project, we aim to further improve the accuracy by using an ensemble of three state-of-the-art CNN architectures ResNet, ShuffleNet v1, and MobileNet v2, each of which has been trained on the same dataset. For every face image, each ensemble member network produce 512 facial embeddings which are then concatenated together and fed into another consensus scheme network to decide the collective classification. Final results show that the ensemble network is able to achieve 15% to 20% higher accuracy than individual CNN network. This method is implemented in Python (using Pytorch) and is available at https://github.com/MeloHo/Ensemble-Learning-for-Face-classification.

1. Introduction

Generally, face classification performs the task of determining the ID of the face given an image of a person's face. Whereas, face verification is the task to compare two face images and determine whether they come from the same person. Intuitively, facial features vary extensively across people. We will train a Convolutional Neural Network(CNN) to extract the features from the images. After feature embedding, an ID can be assigned to the face with confidence scores for N-classes. Face recognition has many real-world applications, e.g., smart phones use it to check the identity of the user and validate payment, and law enforcement apply it to criminal detection.

In this project, deep neural networks are chosen to be the feature extractor, due to its capablility of making accurate predictions and classifications given a large set of training data. However, this capability comes with a cost. Firstly, neural networks are data sensitive and have very high variance. For a given Neural Network architecture, the selection of the weights w is an optimization problem with many local minima. While in search of the global minima, the algorithm might end up in local minimas that differ from one trial to another[5]. This stimulates immense randomness that stems from weight initialization and sequencing of training data. Secondly, deep neural networks tend to overfit the training data and have large generalization error, i.e, achieving high accuracy on the training set while predicting significantly less accurate on the test set.

One approach to tackle these issues of deep neural network models is to train multiple NN models and combine the prediction results. This is known as ensemble learning. Winners in machine learning competitions, such as Kaggle, usually use a large ensemble of models and beat any single model classifier.

The main idea of our ensemble method is to classify a given face image by obtaining a vector of facial embeddings result from each ensemble member and then using another high-level network to decide the final classification. Since each member network has different bias and makes different generalization errors, it is a good strategy to use the collective decision of the ensemble. As our result shows, the prediction of the ensemble is 15% to 20% less fallible than that of each individual network.

2. Background & Related Work

In this section, we briefly review the concept of ensemble learning and several established methods to solve the face recognition problem.

2.1. Ensemble Learning

Ensemble Learning refers to the class of algorithms that combines predictions from multiple statistical models to form one final prediction. Several weak models can be combined into an strong learner. There are mainly three varieties of ensemble methods: vary the selection of training data, vary the choice of models, and vary how results from each members are combined.

Bootstrap aggregation method, or bagging in short, decreases variance by creating a number of subsets of training data with replacement and train each classifier on different subset, hence generating an ensemble of models that have similar structure but distinct bias[2]. The final prediction is made based on the average of the ensemble members' results. For example, random forest employs a large number of unpruned decision trees that have high variance and low bias. Other alternative techniques to create dataset involve choosing a random subset of input features, or disabling replacement while sampling.

Another method known as boosting is a sequential ensemble method that gradually adds ensemble members which could correct the mistake of previous models. However, as boosting is sensitive to noise, the performance of boosting methods is dependent on the characteristics of the dataset and can possibly create ensembles that are less accurate than a single classifier, especially when using neural network[9].

Stacking, or stacked generalization, introduced by David H. Wolpert, deduces the biases of the low-level machine learning models and uses another high level model whose inputs are the guesses of the original models that are trained on part of the training set, and whose output is the ground-truth [11]. Essentially, the predictions from low level models are used as features for the high level generalizer.

As for the optimal number of classifiers in the ensemble, it has been shown that the first few additions produces most of the reduction in error [9][4].

As a popular method to increase prediction accuracy, the ensemble method of deep neural network has been exploited extensively. Szegedy *et al.* explores the possibility of combining Inception network with residual components. An ensemble of three residual and one Inception-v4 networks was tested and achieved 3.08% top-5 error on the ImageNet Classification challenge. It's worth noting that the ensemble network shows only 0.5% improvement than best performing network, ResNet-v2[10]. Minetto *et al.* presents Hydra, an ensemble of CNN for geospatial land classification. A coarsely optimized ResNet made up of the Hydra's body and provides a good starting point for further optimization. The initial weights are then fine-tuned with different data augmentation technique, crop styles, and weighted ground-truth to form an ensemble of CNNs. The author reports that the training time is significantly reduced while maintaining the classification accuracy[8]. The Hydra ensemble network proves its competency in FMOW and NWPU-RESISC45 by achieving similar to best reported result while keeping relatively low training time.

2.2. Face Recognition Technique

2.2.1 Convolution Neural Network

Convolutional Neural Network is a special type of deep neural network which has been largely used in analyzing visual imagery because of its shift-invariant property. It works by extracting local features from the image and classify based on the generated feature vector. Here we introduce three state-of-the-art CNN architecture that has shown great result on ImageNet classfication and will be adapted in our project.

In 2015, He *et al.* proposed ResNet for image recognition. The network features a residual learning framework and a substantially larger number of hidden layers. Instead of learning unreferenced functions, each layers contains residual networks that tries to optimize learning residual functions with reference to the layer input[6]. An ensemble of these residual nets achieves 3.57% error on the ImageNet test.

In 2017, Google Researchers presents a new CNN model MobileNet in order to deal with the vision task with high efficiency. It implements depth-wise separable convolutions to build light weight neural network [7]. Only after one year, they improve the MobileNet by adding the residual connection and droping the ReLU activation to prevent information loss. MobileNetV2 has shown a very good result with ImageNet dataset.

ShuffleNet further reduces the computation cost required by utilizing two new operations: pointwise group convolution and channel shuffle. Zhang *et al.* argues that pointwise group convolution is computation efficient, nevertheless would stem information flow between groups. As a result, a channel shuffle operation is proposed [12]. Experiments on ImageNet classification demonstrate superior performance of ShuffleNet over other architectures as of early 2018.

2.2.2 Eigenfaces

Eigenface technique use Principal Components Analysis(PCA) to reduce the dimensionality of the images into representations of much lower dimension hyperspaces. For classification purposes, the unknown face image is normalized and projected on the eigenspace to transform into vector of features. We could calculate the Euclidean distance, or Mahalanobis distance, between the vector of features to each vector existing in our dataset and select the highest similarity result. Statistical classifiers, such as KNN, Neural Network, or SVM, could then be applied to classifies the

eigenface features and generate calssification.

2.2.3 ICA & LDA

Independent Component Analysis(ICA) method attempts to find the basis along which the projected data are most statistically independent. Bartlett *et al.* uses ICA to generate statistically independent basis images and reports better performance on PCA(Eigenfaces) method[1]. Linear Discriminant Analysis (LDA) finds the vectors in the underlying space that has the strongest discriminatory power and best discriminate among classes. Kamran *et al.* applies LDA to different aspects of human faces in both the spatial and wavelet domain and achieves superior result than using PCA along[3].

3. Data

The dataset we use for this project comes from homework2, part2 of 11-785: Introduction to Deep Learning. The dataset contains around 82,0000 face images of 2300 people, both male and female. There are around 400 face images for each person and each image is shoot at different angle, in different environment, at various light conditions, and across a concentrated period of the person's life. Images in this dataset have the size of 32 by 32. The images have been pre-treated with a pinkish shade so that no external pre-trained models can be used on the dataset. The whole dataset will be divided into three parts: training set, validation set and test set. The test set has 4600 images.

Figure 1. Example Data of One Person

Model	Train Accuracy	Test Accuracy
MobileNetV2	78.6%	62.7%
ResNet34	82.7%	67.3%
ShuffleNet	83.5%	55.7%
Ensemble Net1	99.8%	77.5%
Ensemble Net2	95.8%	77.2%
Ensemble Net3	99.2%	74.0%

Table 1. Train and Test accuracy of Each Model

4. Methods

We first train three state-of-the-art individual CNNs for face classification. We then test their performance on the provided Kaggle competition. We found that all of these networks suffer from overfitting problem and have relatively low test accuracy. We then employ ensemble learning technique to these three networks, which combines the output of these three networks. Details are discussed below.

4.1. Train three networks for the face classification problem

In our method, we use three state-of-the-art architectures for face classification problem, which are ShuffleNet, ResNet34 and MobileNetV2. We use these three networks as our backbones and make modifications on these networks. In ResNet34, we omit the original 7 by 7 convolution kernel and the maxpooling layer at the beginning of the network. This is because the size of our input image is 32 by 32, which is much smaller than the size of those images used in original paper. We don't down-sample the image for the first few layers because we want to keep the large scale features. We use 1 by 1 convolution kernel in the first few blocks of ResNet34 to retrieve useful information as much as possible. Also, we set the number of neurons in the embedding layer to be 512 for each of the network architectures for convenience of concatenating embeddings. The detailed architecture of these three networks are shown in Figure 2. Each network is trained from scratch to make sure it is compatible for subsequent ensemble learning method. At last, we got relatively low performance of the individual networks, as shown in Table 1.

4.2. Make Embeddings for Each Image

Each network deduce a 2300-way classification from the 512-dimension embedding of each image. That is, the size of last layer is 2300, which is the number of classes. In the ensemble learning method, we try to make three embeddings for each input image with each network producing one embedding. We then concatenate these three 512-length embeddings into a one-dimension vector. This vector is a new representation for this image. To do this, we delete the last classification layer for each architecture while keeping

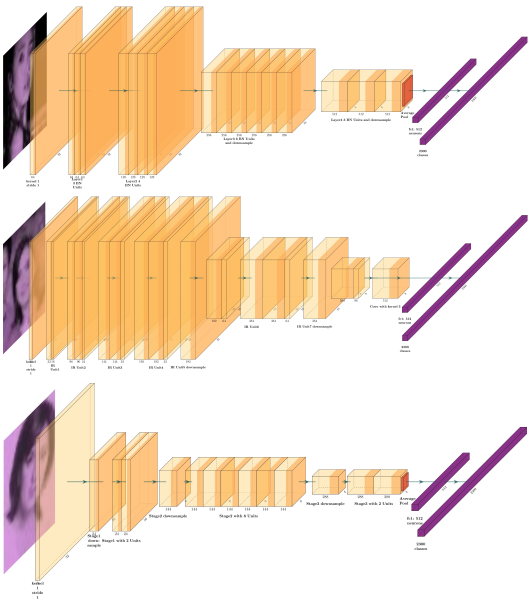

Figure 2. The architecture of three state-of-the-art networks for image classification. From the top to the bottom are: ResNet34, MobileNetV2, ShuffleNet. The size of embedding layer is 512 for each networks for the convenience of future ensemble learning work. Each architecture is slightly modified for our 32 by 32, single-channel input images.

other parts. Then we use each training image as the input to the trained model for three networks, extract and concatenate three embeddings into one single vector. The label for each image remains the same. This process is shown in Figure 3.

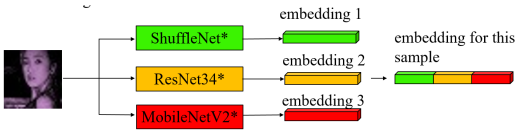

Figure 3. The pipeline for making embeddings for each input image. * means that the last layer of this model is deleted.

4.3. Train a Multilayer Perceptron on the Embeddings

Once each training image is represented by a new one-dimension vector, we can train a multilayer perceptron to do classification. The architecture of this multilayer perceptron is quite simple with only two hidden layers. Other complicated models are unnecessary here. This is because the one-dimension vector combines the prediction from three state-of-the-art models and the features of the input image are well represented in the vector. The task for an extra MLP is to work on those high dimensional representation of the image and somehow gives reasonable weights for high dimensional features. In this case, an MLP with two hidden layers will suffice. The architecture of this MLP is shown in figure 4.

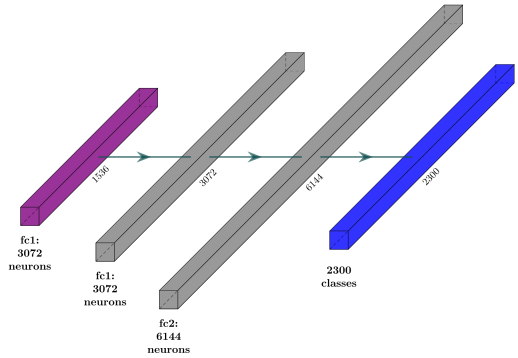

Figure 4. The detailed architecture of the MLP. Notice that this is a rather simple architecture with only two hidden layers.

A MLP is trained on the embeddings of training images and optimization method like dropout and batch normalization is applied to improve its performance.

4.4. Test the performance on Kaggle competition

Same pipeline is applied for test images. Each image is put through three networks to make embedding and this embedding is used as input for the MLP. The output from MLP is used as the label of the input image. The overall pipeline is shown in figure 5.

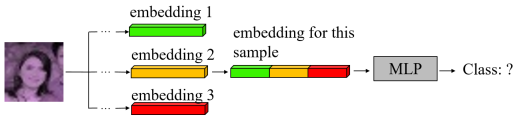

Figure 5. The overall pipeline for a test image.

5. Results

By ensemble learning, we improve the test accuracy of around 10 to 20% compared to each individual baseline model. Also, we have the train accuracy almost close to 100% which indicates the ensemble learning could map the input to label in a nearly perfect manner. From table 1, we can see that the train accuracy of MobileNetV2, ResNet34, ShulffleNet is 78.6%, 82.7%, and 83.5%. This level of train accuracy means the models cannot fully interpret our data so that it will not maximize the test performance. In the contrary, our three Ensemble Nets all reaches train accuracy over 95%. Especially for Ensemble Net1 and Ensemble Net3, the train accuracy is above 99%. Because of the improvement in train accuracy, the test accuracy of Ensemble Net is 77.5%, 77.2%, and 74% which are 10 20% higher than three baseline models.

For the Ensemble Net itself, we also try using different architecture to maximize the performance. Initially, we use

the simplest three layers multilayer perceptron (MLP) as our classifier which already gives a very strong result. However, we find our model still have some overfitting problem: the difference between train accuracy and test accuracy is 20%. Based on that, we want to apply dropout to reduce the amount of overfit. Even though the dropout successfully reduces the gap between train accuracy and test accuracy, the model can only reach a relatively low train accuracy compared to previous model so that the final test accuracy does not change much. So, at third time, we add one more layer and reduce the dropout rate in order to maintain our train accuracy, but the test result shows a decreasing test accuracy. We also tried adding a CNN in our ensemble net, but the train loss decreases very slow and it takes very long to converge. In conclusion, the simplest 3-layer MLP would be best for our Ensemble Net in our experiment.

6. Discussion

From our train accuracy and test accuracy result, ensemble learning method can improve the performance a lot. In this section, we will analyze the reason for this improvement in performance and give some potentially direction for our future work.

Because of its complexity, deep neural networks are often called "black box" model which people could hardly understand what exactly details are going on this "black box". However, instead of being a real "black box", we could interpret deep neural network in a very intuitive way. During training, we want our model to separate data as clear as possible which means we need to draw a clear boundary for our dataset. If we naively use simple regression to do this, we will not be able to draw a very clear separate line (even though we could use the "kernel function"). What deep neural network does is: each layer will transform the "inseparable" data into a more separable manner. The more layer we have, the easier our last layer can deal with. For our baseline model, all three models use residual or skip connection to increase the effective layer for training. However, it is still very hard for each model to perfectly get a fully separable data. Especially for MobileNetV2, our training accuracy is only 78 % which does not fit very well for the dataset. In this case, we ensemble the embedding of all three datasets and train it with a deeper MLP classifier. This could succeed because the MLP can automatically adjust weights to choose the most separable manner across our three embeddings. Besides that, there are large variance in these models. Ensemble method combines the prediction of these models and achieve better performance by averaging the variance from different models. Figure 6 shows why ensemble method could outperform individual networks.

Another issue with our baseline model is the overfitting problem. Although there is still overfitting problem in our ensemble model, it is already somehow fixing a part of the

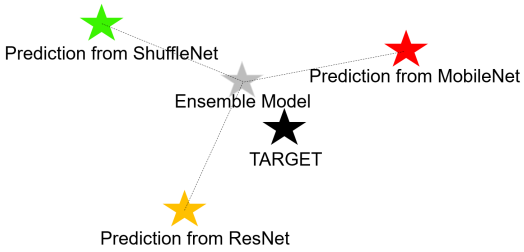

Figure 6. The mechanism of ensemble learning method

issue. ShuffleNet, for example, has a relative high training accuracy with a low-test accuracy which indicates a big overfitting problem. One of the main reasons, we would like to use ensemble learning is to ignore the overfitting part of each model. Different models will definitely have different part for overfitting. If we can know that which part is more "important" and which part is actually "useless details", then we can overcome this problem. This task is very easy for a simple MLP which could adjust its weight to each embedding value. Our model still could be improved in some ways. For baseline model, we didn't perfectly tune each hyperparameters because of the limitation of our computation resources. If properly tuned, our baseline model could achieve a test accuracy of around 70 %, so that our ensemble model could be at least higher than 80 % accuracy. Also, the architecture of our ensemble net could be more complicated to extract more useful ensemble data (for example adding CNN layer). Besides, some data preprocessing, such as data augmentation, could be done to improve our train and test accuracy.

7. Conclusion & Future Work

For image classification, the existing CNN architecture cannot have a very perfect result for our dataset. Even for the most recent model like MobileNetV2, ShuffleNet, the performance can only reach around 60 % accuracy individually. In reality, we definitely would like to get a higher accuracy for the application purpose. To solve that problem, our ensemble learning method has a great improvement for the test accuracy. We trained three model individually with a relatively low accuracy and high overfitting problem. Then, we drop the last linear layer of each model, and produces an embedding of 512 size for each image. After concatenating all three models' embeddings, we apply another simple three-layer MLP for the final classification. The results are much better than each individual model: For all three ensemble nets, their train accuracy is all above 90 % and test accuracy is above 70 %. The highest train accuracy is 99.8 % with a test accuracy of 77.5 %. These accuracies are much acceptable for some industrial face classification use purpose. The reason for our success by ensemble learning is that the final MLP layer has selec-

tively choose the most "important" information underlying the embedding instead of only depending on one individual results. These not only help the model to learn the most important information, but also reduces some overfitting problems. Even though we had made a huge improvement in the accuracy, there are still many ways to improve our model to achieve an even better result. One way of doing that is fine tuning of our baseline model. Due to the computation resources limitation, we cannot perfectly tune our baseline model. It is very possible for each model to reach around 70 % test accuracy individually. Second way to improve is that we can do some preprocessing to our dataset like data augmentation which could also help improve the accuracy. These two improvements are very intuitive, but we can also do some more complicated trail: change the architecture of baseline modes, add CNN or RNN layers to our model. In all, ensemble learning could improve the performance of face classification a lot, and it would be very useful in the future face classification task in industry.

8. Acknowledgement

We want to sincerely thank Professor Amir Barati Farimani and TA Weikun Zhen for advising us on this project and providing constructive suggestions.

References

[1] M. S. Bartlett, J. R. Movellan, and T. J. Sejnowski. Face recognition by independent component analysis. *Trans. Neur. Netw.*, 13(6):1450–1464, Nov. 2002.

[2] L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, Aug. 1996.

[3] K. Etemad and R. Chellappa. Discriminant analysis for recognition of human face images. In J. Bigün, G. Chollet, and G. Borgefors, editors, *Audio- and Video-based Biometric Person Authentication*, pages 125–142, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[4] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, ICML'96, pages 148–156, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[5] L. Hansen and P. Salamon. Neural network ensembles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12:993 – 1001, 11 1990.

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

[7] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[8] R. Minetto, M. Pamplona Segundo, and S. Sarkar. Hydra: An ensemble of convolutional neural networks for geospatial land classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):6530–6541, Sep 2019.

[9] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, Aug 1999.

[10] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning, 2016.

[11] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[12] X. Zhang, X. Zhou, M. Lin, and J. Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices, 2017.