Multi-task learning based on Convolutional Neural Network for Age and Gender Classification

Hengtong Kang, Haotian Jiang, Bojia Li
Department of ECE, University of Florida,
Gainesville, Florida, United States

***Abstract*—This paper addressed the problem of age and gender classification, which is a subarea of face recognition. This paper firstly provides an introduction of the motivation of doing gender and age classification and the current work in this field. Then it discusses the face detection method used for preprocessing the images, which is used for extract face from whole input images, before classification. It then discusses the two convolutional neural networks (CNN) used to classify gender and age, which are LeNet and AlexNet. Finally, we proposed a multi-task model, which make the task share convolutional layers, which is used for deducing parameters, to jointly train age and gender together.**

***Index Terms*—age and gender classification, face detection, LeNet, AlexNet**

I. Introduction

Gender classification from face images has long been heated concern in biometric research. [1] It is considered as a soft trait in biometrics research. Accurate classification of gender can help improve the results of gait and keystroke recognition. Also, gender classification is often used as the first step in the online identity verification process.

Automated age estimation from facial images is also an important problem studied in several fields such as computer forensics, human computer interaction, biometrics, entertainment, pattern recognition, and computer vision. [2] In biometrics, estimation of age can contribute improving the identification results of face and fingerprint recognition. [3] An age estimation system may be used to prevent vending machines from selling products, e.g., alcohol, tobacco, to adolescents. [1]The fact that people's preferences change depending on their ages also yields a number of potential applications of automated age estimation. [2]

II. Related Work

Convolutional networks (ConvNets) have recently enjoyed a great success in large-scale image and video recognition (Krizhevsky et al., 2012; Zeiler & Fergus, 2013; Sermanet et al., 2014; Simonyan & Zisserman, 2014) which has become possible due to the large public image repositories, such as ImageNet (Deng et al., 2009), and high-performance computing systems, such as GPUs or large-scale distributed clusters (Dean et al., 2012). In particular, an important role in the advance of deep visual recognition architectures has been played by the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al., 2014), which has served as a testbed for a few generations of large-scale image classification systems, from high-dimensional shallow feature encodings (Perronnin et al., 2010) (the winner of ILSVRC-2011) to deep ConvNets (Krizhevsky et al., 2012) (the winner of ILSVRC-2012). [1]

The areas of age and gender classification have been studied for decades. Various different approaches have been taken over the years to tackle this problem, with varying levels of success. Some of the recent age classification approaches are surveyed in detail in [3]. Very early attempts [10] focused on the identification of manually tuned facial features and used differences in these features' dimensions and ratios as signs of varying age. The features of interest in this approach included the size of the eyes, mouth, ears, and the distances between them. While some early attempts have shown fairly high accuracies on constrained input images (near ideal lighting, angle, and visibility), few [2] have attempted to address the difficulties that arise from real-world variations in picture quality/clarity.

A holistic overview of methods applied to gender classification can be found in [14]. As early as 1990, neural networks were considered for the purposes of gender classification in [4] (which has perhaps one of the most interesting paper names in all of computer vision). Later on in the early 2000s, [16] used support vector machines (SVMs) and found they could achieve very low error rates on gender prediction of "thumbnail images" of subjects which were of very low resolution. Yet again though, none of these attempts seemed to acknowledge that the constrained settings of their training and test data hindered their systems from achieving equally impressive performance numbers in realworld applications where images can be subject to altered lighting, tilt, focus, occlusion, etc. Virtually all of these papers, and their corresponding methods, tackle either age classification (in some cases regression) or gender classification, but usually not both. But in 2015, [12] broke this norm by developing one methodology and architecture to address both age and gender. Furthermore, the authors address the undeniable reality that images taken in real-world settings are not perfectly aligned, lit, or centered. To that end, they train on images from a wide range of angles, lighting conditions, etc., and they oversample the input images to the classifier to consider various regions in the image for classification.

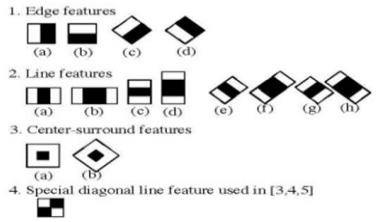

Fig. 1. Haar-like features

Their focus on using deep convolutional neural networks (CNNs), follows a pattern in the computer vision community as CNNs are shown more and more to provide unparalleled performance for other types of image classification. The first application of CNNs was the LeNet-5, as described in [11]. However deeper architectures in the early 1990s were infeasible due to the state of hardware performance and cost. In recent years, with the dawn of never-before seen fast and cheap compute, [9] revived the interest in CNNs showing that deep architectures are now both feasible and effective, and [19] continued to increase the depth of such networks to show even better performance. Therefore the authors of [12] leveraged these advances to build a powerful network that showed state-of-the-art performance.

They advocate for a relatively shallow network, however, in order to prevent over-fitting the relatively small dataset they were operating on. Deeper networks, although generally more expressive, also have a greater tendency to fit noise in the data. So while [19] shows improved performance with deeper architectures training on millions of images, [12] shows improvements for shallower architectures for their use case.

We aim to show that I can build off of this prior work, particularly the work in [12], to develop a system that leverages the inherent inter-relationships between age and gender to link these architectures in such a way as to improve overall performance.

III. METHOD

A. Face Detection

In our project, Viola and Jones method is used for face detection, which mainly includes three parts. The first one is to gain Haar-like features used for the feature extraction and then get an Integral Image [4]. The second one is classifier using AdaBoost, a machine learning algorithm, to select a small number of important visual features from a very large set of potential features [4] [5]. The third one is Cascade classifier which can efficiently combine many features to discard background image [5].

1) Haar -Like Features and Integral Image

Haar feature is used to determine if there is any feature present in the given image [5]. Each feature returns a single value which is subtracting sum value of the pixels in the black region from the sum value of pixels in the white region. If the difference is larger than threshold, then feature is considered as present. Some of the Haar-like features are given in Fig. 1:

Integral image is used to speed up feature detection, that is, in an image, every pixel value is sum of pixel value above it and to its left [6]. Sum pixel value of any rectangular in an image can be calculated by operation of four coordinate of vertex of the rectangle. Integral image is shown in the Fig. 2. Sum of pixel value of D equals to (x4,y4) – (x2,y2) – (x3,y3) + (x1,y1).

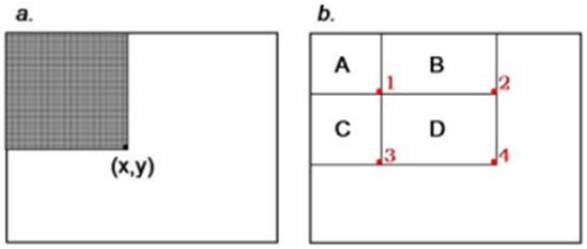

Fig. 2. Integral image

2) Adaboost

AdaBoost is used for choosing optimal Haar-like features out of 160,000 and adjusting threshold value. AdaBoost linearly combines many weak classifier to make a robust classifier. [7] A weak classifier is mathematically described as:

$$h(x)=\begin{cases}1 & pf(x)<p\theta \\ 0 & others\end{cases} \quad (1)$$

Where x is 24*24 pixel sub window, f(x) is given image, p is polarity and θ is threshold value. Each weak classifier can determine whether this sub window is face or not. [8]

AdaBoost Algorithm is shown as below [9]:

a) For training data set $D=\{(x_1,y_1),(x_2,y_2),...,(x_m,y_m)\}$, where i=1,2,3,...,m, $y_i=1/0$ represents positive and negative examples respectively, m is the number of examples.

b) Initialize weights $w_{1,i}$, when $y_i=1$, $w_{1,i}=\frac{1}{2n}$;when $y_i=0$, $w_{1,i}=\frac{1}{2l}$, where n and l are the number of positive and negative examples respectively.

c) For t=1,2,3,...,T iteration, following operations are carried out:

(1) Normalize the weights:

$$w_{t,i}=\frac{w_{t,i}}{\sum_{j=1}^{m}w_{t,j}} \quad (2)$$

(2) For each feature, calculate weighted classification error:

$$\varepsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right| \quad (3)$$

(3) Select the classifier h_t with the lowest ε_t as the best weak classifier.

(4) Combine many weak classifiers into a strong classifier, which is described as:

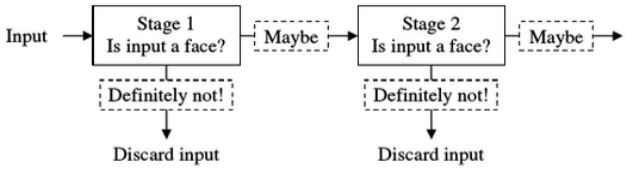

Fig. 3. Cascade classifier

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} a_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} a_t \\ 0 & others \end{cases} \quad (4)$$

According to AdaBoost algorithm, increasing weight value of examples wrongly classified and reducing weight value of examples correctly classified can improve accuracy of correct classification of examples previously wrongly classified.

3) Cascade Classifier

Cascade classifier is used for retaining face and discarding background image. If any one filter in the classifier fails to pass an image area, the area is directly classified as non-face. [8] Otherwise, the area will be seen as face with a square mark. Flowchart is shown in the Fig. 3.

Both face in color image and gray image can be successfully detected and extracted. Results are shown in Fig. 4 and Fig. 5.

B. Classification

In this section, we design a shallow CNN neural network to classify genders. The network architecture is designed as shown in Fig. 6.

1) Gender Classification

The size of input image is 128*128. We added three convolutional layers with kernel 7*7, 5*5 and 3*3, each number is 32, 64 and 64. After each convolutional layer, we apply LRN(Local Response Normalization) to normalize output of first two conv layers. Above the convolutional layers, we stack two fully connected layers. On the top of the second fully connected layer, we stack a softmax classifier.

As for hyper-parameters, we set initial learning rate 0.01 and decay factor as 0.96 after 5000 steps-training. Since we only use small dataset, about 2000 images, we only train our model 2000 times to debug. We also use regularization term to avoid over fitting. The coefficient is 1e-4.

2) Age Classification

We use LeNet as our network to do age classification. LeNet is a rather simple CNN which only contains 5 layers. It was invented by Yann Lechun in 1998. [10]

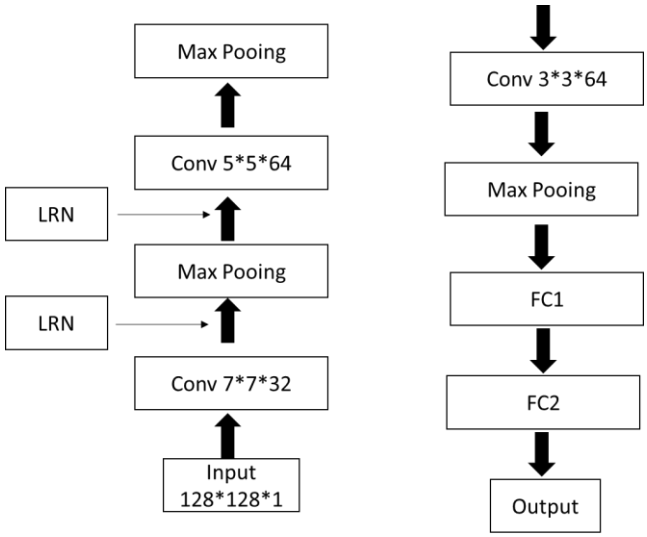

Fig. 3. AlexNet

The LeNet we use has 6 layers. The input image size is 128*128. The first layer is a convolution layer whose function is generating feature maps. To generate a feature map, the filter needs to run over the entire image. At each location, the filter should compute a linear combination of all the pixel values under it and place the result in the center pixel of the filter. The first layer generate 6 feature maps, so it has 6 different filters. The size of filter we use is 5*5, so each filter has 26 parameters including 1 bias. And the size of the feature map is 124*124.

We can reduce the parameters we use by sharing the same parameters in each feature map, which means that the parameters of a certain filter does not change when it is running on the images. Since there are 6 filters, the sum of number of parameters in this layer is 156. These parameters will be learned by doing backproagation later. Also, we can compute the number of links we have in this layer which is 2398656. We can see that convolutional layers can significantly reduce the number of parameters compared to fully connected layers, which is a remarkable advantage of CNN.

The second layer is a max-pooling layer. This layer generate 6 feature maps whose sizes are 62*62. Each pixel in the feature map is connected to a 2*2 area in the feature map of the previous layer. We compute the sum of the pixels in that area and multiplies a weight. The product is then added to a bias, and the sum is put into a sigmoid function. The result is the value in the feature map in this layer. Since, we have 6 feature maps in the previous layer and 2 parameters for each feature map in this

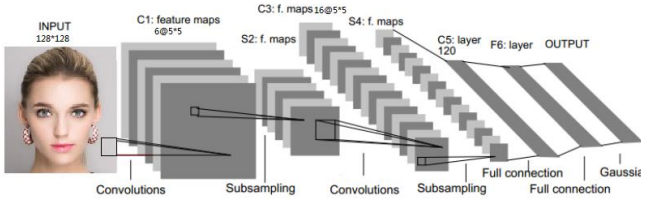

Fig. 4 LeNet

layer, the number of all parameters in the second layer is 12. The function of max-pooling layer is to reduce the spatial size of the feature map, which can help further reduce parameters in the future operations.

The third layer of the network is again a convolutional layer. Here we generate 16 feature maps by using 16 filters. The filter size here is 5*5, so there are 416 parameters in this layer. The fourth layer is a max-pooling layer, and the pooling is also conducted in each 2*2 area. The fifth layer is a convolutional layer which uses 5*5 filter and generates 120 feature maps. The sixth and seventh layer are fully-connected. The pixel values in the feature maps in the fifth layer are fed into these two layers.

In this way, we can show the architecture of the network for age classification in Fig. 4.

C. *Multi-Task Learning*

From previous models, we could find out that the convolutional layers could be shared, which is a good way to reduce parameters. Therefore, we proposed a new method to train the two task jointly.

As the input images are of different sizes, after preprocessing, we have cropped each image into 128*128*1, which means we also reduced the dimension as 1, grayscale image. So the input dimension is 128*128*1.

There are 3 convolutional layers, followed by 3 fully connected layer that are parallel with each other.

1. Conv1- 32 filters of size 1x7x7 are convolved with stride 2 and padding 2, resulting in an output volume size of 32x64x64. This is followed by a ReLU, maxpooling pooling which reduces the size to 32x32x32, and a local-response normalization (LRN).
2. Conv2- 64 filters of size 64x5x5 are convolved with stride 2 and padding 2, resulting in an output volume size of 64x32x32. This is followed by a ReLU, maxpooling pooling which reduces the size to 64x16x16, and a local-response normalization (LRN).
3. Conv3- 64 filters of size 64x3x3 are convolved with stride 2 and padding 2, resulting in an output volume size of 64x16x16. This is followed by a ReLU, maxpooling pooling which reduces the size to 64x8x8, and a local-response normalization (LRN).

The two fully connected layers have same dimension:

1. FC6- 512 neurons fully connected to the 64x8x8 output of Conv3, followed by a ReLU layer and dropout layer.
2. FC7- 100 neurons fully connected to the 1x512 output of FC6 followed by a ReLU layer and dropout layer.
3. FC8- 2 or 4 neurons fully connected to the 1x512 output of FC7, yielding the un-normalized class scores for either gender or age, respectively.

And finally there is a softmax layer that sits on top of FC8, which gives the loss and final class probabilities.

Since the two task may not converge at the same time, we added two parameters to eliminate this impact, which may cause the weight in the first converge change abruptly. And we made the two parameters trainable. The loss function is shown in e.q.5.

$$Loss_{whole} = p \cdot loss_{age} + q \cdot loss_{gender} \tag{5}$$

Where the parameters p and q are trainable. We could train these parameters using back-propagation algorithm.

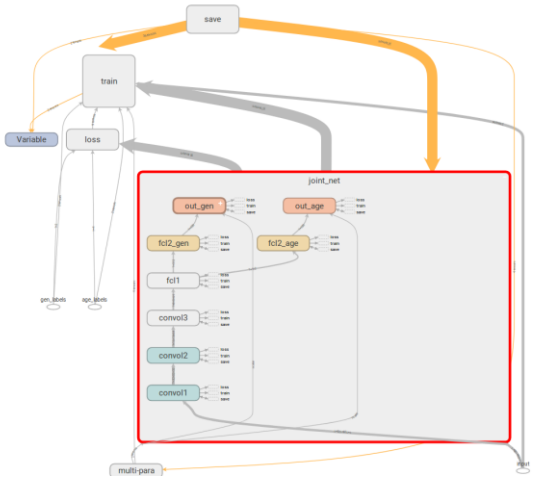

Fig 5 Network Architecture

D. *Technical Details*

We add local response normalization layers (LRN) which can improve the generalization ability after the pooling layers. LRN can make the different filters compete with each other to get large activation, which can eliminate the phenomenon that many similar kernels record the same input area and preserve only the prominent activation. So, the local response normalized activation $b^i_{x,y}$ is given by:

$$b^i_{x,y} = \frac{a^i_{x,y}}{k+\alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^i_{x,y})^2} \tag{6}$$

In the equation, $a^i_{x,y}$ is the activation of a neuron, and k, n, α, β are all hyper-parameters.

We use the Softmax function to compute the loss of the training process and predict the class probabilities during classification. Unlike some traditional loss functions, Softmax regard the final output of the fully-connected layer as the probability of different classes. The probabilities are unnormalized and are in log form. So, we can get the softmax function like this:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{7}$$

In the formula, z_j is the score of class j. We can minimize the negative log of the scores to maximize the log likelihood of the current class. The negative log can be expressed as:

$$L_i = -\log(\frac{e^{f_{yi}}}{\sum_j e^{fj}}) \quad (8)$$

Softmax can normalize the real scores from f by their exponentiated sum, which can make the sum of its scores to be 1. This can make the corresponding classes of the scores to be regarded as a real class based on their probabilities. As we know, softmax is a form of the cross-entropy loss function, which can be expressed as:

$$\mathrm{H(p,q)} = -\sum p(x) log q(x) \quad (9)$$

In the formula, p indicates a distribution, and q indicates a approximate distribution of it. So, softmax serves a function of minimizing the cross-entropy between the probabilities of the classes and the real distribution. In the real distribution, all real classes are labeled 1 while other things are labeled 0.

When training the classifiers, we want to minimize the loss we calculated from softmax. And we choose to use the most common way which is stochastic gradient descent (SGD). The gradient of the loss function is computed as the maximum value decreasing speed in all directions, which is one of its derivatives. The directions here are the derivatives regarding different variables in the loss function. After computing the gradient, we can move toward the direction corresponding that gradient for a certain step-size so as to decrease the loss. Then, we can compute the gradient and move towards the selected direction again. After many iterations, we will come to the 'valley' of the loss function, and this valley will be the global minimum of the loss function if we are lucky enough. This process can be expressed using the following function:

$$\mathrm{w} = \mathrm{w} - \eta \nabla_w \mathrm{L} \quad (10)$$

In this formula, η is the step-size, w is the weight vector, and $\nabla_w \mathrm{L}$ is the gradient. If we use this method to optimize the loss function, we will surely get a good result. However, it seems that computing the gradient based on the entire training set can cost so much computational resources. So, we can develop a way of computing using a sample collected from the training set which is less expensive. This method is actually called mini-batch. It can also achieve a minimum but will require larger number of iterations. Since the process of computing gradient become much quicker in this way, having more iterations actually will not change the fact that mini-batch can be performed really fast. SGD is a modified version of mini-batch, because it uses random batch for each iteration. This method is even faster, but we need to use smaller step-size and larger iterations to guarantee a good result.

IV. Experiment

A. Face Detection

Fig. 6. Detected and extracted face in color image

We used Viola Jones algorithm to detect the faces in input image. After detecting the faces, we will resize the face area into 128*128, which is standard input size of our neural network.

B. Gender Classification

We test the performance of AlexNet on small data, which includes about 6000 training images, 1000 testing and 800 validation data. We plot the training loss and accurate rate in figure 10.

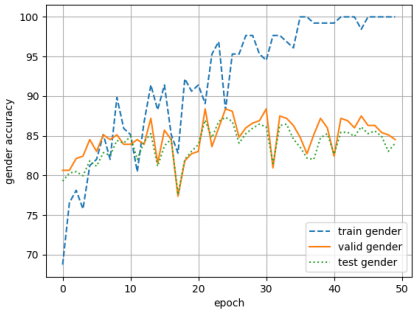

Fig 7 Accuracy of Gender Classification

From figure 10, we can find that the test accurate rate could reach to 85%. Since this is trained on small dataset, the model over-fitted obviously.

C. Age Classification

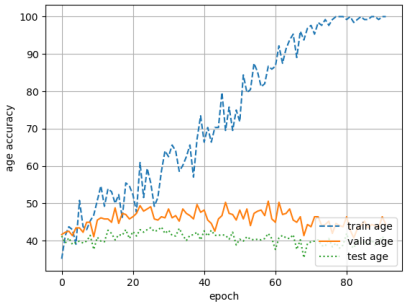

Fig 8 Accuracy of Age Classification

From figure 11, we can find out that the test accurate rate is as low as that to 40%, which is quite normal, since our input image it not purely faces, but also include many irrelevant information, like background and other objects. It is also obvious to see that this model also overfitted.

D. Jointly Training

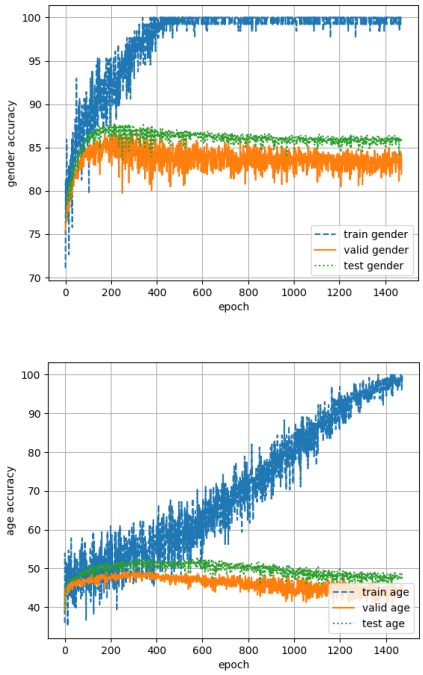

Fig 9 Accuracy of Age and Gender Classification Jointly

about 30,000 images. We can see that the result is better than that using small dataset, especially age classification. The accurate rate of age is large than 50% and for gender, it is larger than 85%.

E. Final Result

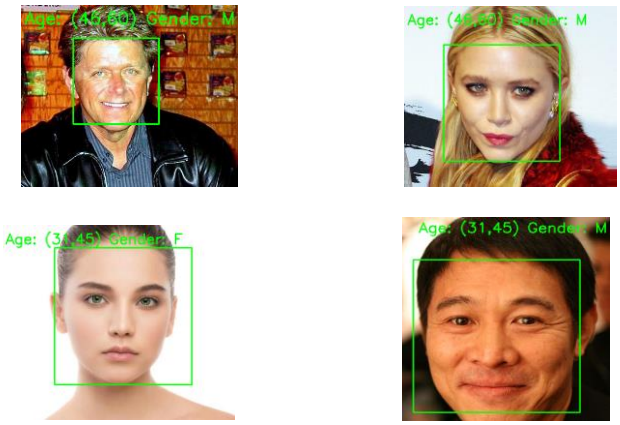

Figure 10 Results

From Figure 9, we can find out that after jointly training, the result is not as good as that trained separately. There are several reasons, the two task can't converge at the same step should be the main reason.

Here is our final result of our project in figure 14.In our result, F represents female and M represents male. Our algorithm is not fine grain age classification, and we only divide the images into four intervals. So, our model only predicts the possible age interval.

From figure 10, we can see that the result is not great. Our model will make mistakes some time. Actually, the result is reasonable since the accuracy of age is really bad. And the base line is about 50%.

The result above is based on image. And our model could also process video and display age and gender in real-time. You can access our code at https://github.com/htkang369/age_gender_classification.git

V. Data Set

We choose IMDB-WIKI as our dataset, which includes 62,328 photos in 20,284 subjects. Since the row images are not standard and some of them are empty. So, we need to clean such unused images.

As for training data, we resize all the images to 128*128 and delete all the empty images. As for labels, we use one hot encoding to encode labels for age and gender. The age intervals are [0, 30], [31, 45], [46, 60], [60, 100] because the images are collected from celebrities, the age distribution is shown in figure 13.

To accelerate data process, we store all the images as matrix in pkl format files, which are easy to be loaded in Python.

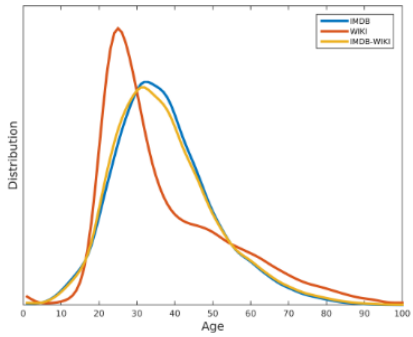

Figure 11 Age Distribution

Figure 12

VI. Conclusion

Using large dataset to train same model will get much better result than using small data, especially when facing large models.

We could train the two tasks separately as well as training them jointly. After jointly training, the result is better than that trained separately. However, the converging time takes longer because of the unbalanced converge time of different task.

VII. Future Work

Even we added two trainable parameters to balance the unbalanced converge time of the two tasks, the performance is not as good as my expectation. Therefore, we could add restrictions on the tow parameters.

Our project could not detect and get result in real-time, it could be described as pseudo-real-time. In this perspective, we could improve its real-time performance.

Our detection method is great for detecting front profile while not good at side profiles. So, we could apply more fantastic models to detect face images, like Mask-RCNN or Faster-RCNN or Yolo.

Our training samples are not pure faces. Therefore, during our training process, we added more redundant features for our models, which may have bad impact on classification task. In the future, we could first create our pure face image and train them.

As for the low accuracy of age, I think it is mainly based on unbalanced distribution of age, which can be seen in figure 16.

VIII. Current Progress and Project Management

A. Current Status

Now, we have finished dataset preprocessing, face detection in an image based on Viola Jones and establishing two shallow CNN networks to classify age and gender in small dataset.

B. Team Coordination

Hengtong Kang: Design and implemented multi-task learning model and wrote code and results. Constructed AlexNet and wrote related codes; preprocessed the image data; completed the data preprocessing, gender classification and future plan part of the slide and report; synthesis the final slide. Finished the final video face classification task.

Haotian Jiang: Constructed LeNet and wrote related codes; completed age classification part of the slides and report; synthesized the final report.

Bojia Li: Completed face detection and wrote related codes; completed face detection part of the slides and report. Wrote raw code for face detection in video. Wrote final paper.

References

[1] Hengtong Kang, Haotian Jiang, Bojia Li, “Convolutional Neural Networks for Age and Gender Classification-Proposal,” unpublished.

[2] M. F. Aydogdu, V. Celik and M. F. Demirci, "Comparison of Three Different CNN Architectures for Age Classification," 2017 IEEE 11th International Conference on Semantic Computing (ICSC), San Diego, CA, 2017, pp. 372-377.

[3] G. Ozbulak, Y. Aytar and H. K. Ekenel, "How Transferable Are CNN-Based Features for Age and Gender Classification?," 2016 International Conference of the Biometrics Special Interest Group (BIOSIG), Darmstadt, 2016, pp. 1-6.

[4] K. Kadir, M. K. Kamaruddin, H. Nasir, S. I. Safie and Z. A. K. Bakti, "A comparative study between LBP and Haar-like features for Face Detection using OpenCV," 2014 4th International Conference on Engineering Technology and Technopreneuship (ICE2T), Kuala Lumpur, 2014, pp. 335-339.

[5] M. Nehru and S. Padmavathi, "Illumination invariant face detection using viola jones algorithm," 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, 2017, pp. 1-4.

[6] M. D. Putro, T. B. Adji and B. Winduratna, "Adult image classifiers based on face detection using Viola-Jones method," 2015 1st International Conference on Wireless and Telematics (ICWT), Manado, 2015, pp. 1-6.

[7] M. Da'san, A. Alqudah and O. Debeir, "Face detection using Viola and Jones method and neural networks," 2015 International Conference on Information and Communication Technology Research (ICTRC), Abu Dhabi, 2015, pp. 40-43.

[8] G. C. Luh, "Face detection using combination of skin color pixel detection and Viola-Jones face detector," 2014 International Conference on Machine Learning and Cybernetics, Lanzhou, 2014, pp. 364-370.

[9] M. Yu, L. Yun, Z. Chen and F. Cheng, "Research on video face detection based on AdaBoost algorithm training classifier," 2017 First International Conference on Electronics Instrumentation & Information Systems (EIIS), Harbin, China, 2017, pp. 1-6.

[10] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov 1998.

[11] G. Levi and T. Hassner. Age and gender classification using convolutional neural networks. In *IEEE Conf. on Computer Vision and Pattern Recognition(CVPR)workshops*,June 2015.

[12] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua. A convolutional neural network cascade for face detection. In *Computer Vision and Pattern Recognition(CVPR),2015IEEEConference on*, pages 5325–5334, June 2015.

[13] E. Makinen and R. Raisamo. Evaluation of gender classification methods with automatically detected and aligned faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):541–547, March 2008.

[14] B. Moghaddam and M.-H. Yang. Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):707–711, May 2002.

[15] B. Moghaddam and M.-H. Yang. Learning gender with support faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):707–711, May 2002.

[16] D. T. Nguyen, S. R. Cho, T. D. Pham, and K. R. Park. Human age estimation method robust to camera sensor and/or face movement. *Sensors*, 15(9):21898, 2015.

[17] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.