APPLIED MACHINE LEARNING SYSTEM ELEC0132 20/21 REPORT

SN: 20056373

ABSTRACT

This report discusses four specific classification tasks. It contains two binary classification tasks A1, A2 (gender classification and smiling classification) on *CelebFaces Attributes Dataset (CelebA)* and two multi-class classification tasks B1, B2 (face shapes recognition and eye recognition) on *Cartoon Set*. Solutions are found among SVM, KNN, random forest, convolutional neural network (CNN) models with task-oriented preprocessed method, achieving accuracies of 94 %, 89%, 100% and 100%. The report also shows that different tasks using different data preprocessing methods may result in better or worse results. The performance of using the face detector on some static images is poor. In the eye recognition task, extract left eye images and restore the eye colors behind the glasses in advance can help the recognition rate reach 100% with simplest model.

Index Terms—Image classification/recognition, SVM, KNN, random forest, CNN, facial detector, eye colors restoration

1. INTRODUCTION

Classification problems are common task goals in machine learning. Classfical machine learning models such as SVM with different kernels, KNN, random forest is often used for classification. At the same time, practice in recent years has showed that CNN are very effective when dealing with image problem. The models used in this report are selected from the above four.

In this report, A1 and A2 use the *CelebFaces Attributes Dataset (CelebA)*[1], and B1 and B2 use the *Cartoon Set*[2].

Task A1: The goal of this task is to perform gender classification on a dataset of celebrity head images. The data will take two forms as input samples for different models. One is to use the original image as the input of the CNN. The other is to use pre-training face detector[3] to converts the original image into face landmarks as input, and uses cross-validation for SVM ,KNN and random forest hyperparameters tuning to select the best performing model. Finally, compare the classification effects of the two types of models.

Task A2: the goal of the task is smiling classification on the celebrity head images. As in the A1 scheme, two forms are used as different inputs. One is the original image as the input of CNN, and the other is to transform the original image into face landmarks as the model input. The performance of SVM, KNN and random forest with different hyper parameters is compared.

Task B1: the goal of this task is to recognize the face shape of cartoon character. The cartoon set contains five kinds of face types. Because there is a great loss in converting the images into face landmarks, the image is directly used as the model input in this task, only CNN is used in this task.

Task B2: the goal of this task is to recognize eye color of the cartoon character. Since the position distribution of the images is uniform, only the local image of the left eye is cut out for recognition. In the data preprocessing stage, the samples with opaque sunglasses are removed and the color of the eyes with colored glasses is restored, Finally, the processed image is used as input and a simple linear SVM is used for training and testing.

This report is organized into 6 sections. Section 1 briefly introduces the tasks targets and methodologies as well as models. In section 2, prior knowledge, relevant work and potential solutions for the tasks are shortly discussed. In section 3, the description of models for each task are introduced and the rational behind it. In section 4, the implementation of the models will be shown in detail, including the whole pipeline. Experiment results will be presented in section 5, along with the analysis. Summary of the report will be given in section 6, the future improvements are also discussed in this section

2. LITERATURE SURVEY

2.1 SVM, KNN and Random Forest

The principle of SVM classification is to find the best hyperplane that can separate the positive and negative samples of the training set in the feature space. SVM is a classic algorithm for solving linear classification problems. After using the kernel method to map the data to the high-dimensional space, SVM can solve some non-linear linear classification problems. Commonly used kernels have linear kernel, polynomial kernel, radial basis function (rbf) kernel, sigmoid. In the process of finding the hyperplane, the hyperplane can be allowed to wrongly classify some data points. The degree of this error is controlled by the penalty parameter, denoted as C. In practical applications, different cores and Different hyperparameters have different performance on different data and require a tuning process.

K-Nearest Neighbor (KNN) is a simple classification algorithm. Its main idea is to find the K samples closest to the

Link to Source code
Link to download face detector 'shape_predictor_68_face_landmarks.dat'
Link to download dataset of the report (Used as training set and validation set in this report)
Link to download additional dataset of the report (Used as test set)

data point. These K samples come from a pre-labeled data set. These K samples are equivalent to K voters who vote on the category to which the data point belongs. The category with more votes is the category predicted by the data point by the model.

Random forest is built on the basis of decision trees. A random forest consists of multiple decision trees. Each decision tree is trained on the bootstrap training set. Information entropy or gini index is the standard for generating decision tree.

Ratna Astuti Nugrahaeni[4] compares the classification effects of the above three methods on seven facial expressions. The experiment performed data prediction on the facial feature points, that is, using other pre-processed models to extract the feature points of the face in the image for training. When using a small data set training (ranging from 165-520 traning data), the testing accuracy of SVM, KNN, and random forests are 80%, 76.97% and 75.15%, respectively. When large data sets for training, the accuracy rate increases to 90%, 98.85% and 98.85%.

Similar to work of Ratna Astuti Nugrahaeni[4], Min Tanga[5] proposed a SVM model based on curvelet transform and particle swarm optimization (PSO) for facial expression recognition. The model extracts facial feature points from the image as training data, and then a SVM is used for classification. The paper shows that using wavelet transform to process images can have more reasonable facial feature points, thereby improving the accuracy and robustness of model recognition.

Michal Uricar[6] shows an experiment of prediction of age, gender and smile from deep features. The clipped face images are passes through the VGG-16 deep neural network for feature extraction, and then the obtained features are classified using non-linear SVM. Experiments show that the prediction accuracy of using SVM to classify image features processed by VGG-16 is higher than that of direct use of deep CNN predictor on gender tasks or age tasks

Saurabh Agrawal[7] uses histograms in different color spaces as classification data for model training, which can reduce the huge changes in the data caused by changes in perspective, translation or rotation. Such results are better than those in specific dataset. The classification accuracy of the method is higher than that of directly using the image as the input as the training data of the SVM. It is more efficient and avoids the problems caused by the excessively high dimensionality of the training data.

Ibrahim A. Adeyanju[8] used different kernel methods in the experiment, rbf, linear, quadratic and polynomial SVM to solve the facial emotion recognition problem, and compare the accuracy of input images with different resolutions to the model. It is pointed out that on average, the larger the input image size, the higher the accuracy of model prediction, but the training time of the model does not necessarily take longer to train with the larger size. The experimental results showed that when the model's input is the entire image, the rbf kernel performed poorly, and the quadratic kernel performed the best, reaching 99%.

2.2 Convolutional Neural Network (CNN)

CNN is widely used in the field of computer vision. In order to adapt to image problems of different scenes, the more complex structure and deeper CNN are proposed to improve the accuracy of the network. On the other hand, there are researches discussing the balance the performance of model and computational resources.

Lucy Nwosu[9] and his team were designing a two-channel convolutional neural network to solve facial expression recognition, and extract the images of eyes and mouth through the feature detector. In the network, the eye image passed through the first channel and the mouth image passed through the second channel before the output of the two channels joined. By training the two-channel convolutional neural network, the recognition rates of 97.71% and 95.72% are achieved on the JAFFEE dataset and CK+ database, respectively.

In order to find a minimized CNN model to solve the gender recognition problem, Grigory Antipov[10] proposed several steps to simplify the CNN model, from reducing data input to reducing convolutional filters, and then to determine the size of the fully-connected layer. The optimized model is smaller ten to twenty times than the state-of-the-art model at the time without losing significant accuracy.

The EfficientNet proposed by Mingxing Tan[11] also provides an efficient way to scale CNN to achieve a good trade-off between model depth, width and input resolution. The experiment also showed that using this technique to optimize the state-of-the-art model at the time, the model size can be smaller and the training speed can be faster under the same accuracy.

2.3 Hyperparameters Tuning

In the process of training models, it is inevitable to tune hyperparameters. James Bergstra and Yoshua Bengio[12] proposed that the method of using random search is better than grid search when computing resources are limited. This conclusion should also help us find the optimal hyperparameters of CNN in A1, A2 and B1.

3. DESCRIPTION OF MODELS

In order to solve these problems, two schemes to find the classification models are proposed:

One is the traditional machine learning classification model, including SVM, KNN and random forest. This kind of model has been proved to be able to solve many classification problems in the past[45678]. However, this kind of model is better at solving the low dimension classification model[7]. Therefore, when it is applied to image classification, it is not efficient to directly to use the original image as input (178*218 * 3 for *CelebA* and 500 * 500 * 3 for *Carton Set*)

because the dimension is too high, so it is necessary to find a method to reduce the dimension of the original image before using these models as classifiers.

The other is to use CNN as the classifier. The advantage is that the original image can be directly used as the model input, all the information of the image are captured by the model. Though the CNN, the original image are presented as a series of abstract features with different types of convolutional kernel which is adaptively learned at the training stage. However, its disadvantage is that the model has high degree of freedom and needs to carefully tune the hyperparameters to improve the fitting effect of the data set, which involves massive computation and is time-consuming.

3.1. Task A1, A2: gender and emotion classifications

3.1.1. SVM, KNN, random forest models

The first scheme uses the classical classification model, SVM, KNN and random forest to classify. This scheme aims to find the simplest model to classify the image features which are close to the essence while ignoring the other information that is irrelevant.

In the image recognition model, if the original image is taken as the whole input, the model is inevitably influenced by the environment (illumination, background etc.) in the image. Therefore, in the process of model training, the dimension space of these elements is included in the learning process, which may result in potential overfitting problem or time-consuming process.

Therefore, it can help to simplify the complexity of the model by transforming the image data with high dimension into representative reduced dimension data before using SVM, KNN and random forest model. The method selected here is to process the original image with a pre-trained face detector[3], and transform each image from high-dimensional RGB space to 68 * 2-dimensional space composed of only 68 facial landmarks, which greatly reduces the difficulty of data fitting and simplifies the complexity of the model. The 68-point landmarks effectively represent facial features in the form of landmarks. Intuitively, most of the redundant information is filtered out and the effective information is extracted.

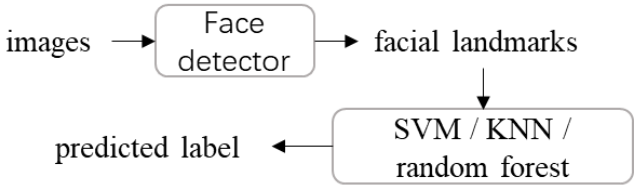

Figure 1 Pipeline of SVM, KNN, random forest models. Face detector3 is used to convert the images to facial landmarks.

It is also important to find the model that have the best performance among the three types of models (SVM, KNN and random forest). For each type of model, the hyperparameters at the training stage can greatly affect the model performance on specific data distribution, so a process of hyperparameters tuning is necessary.

3.1.2.CNN models

In the second scheme, CNN is used as classifier. CNN is widely used to deal with image recognition problems, such as VGG[13], RestNet[14], etc. However, in the case of the task, the dataset is relatively small and the quality of the sample is high (the classified object is relatively clear and take up the majority of the image), it is unnecessary to use large-scale CNN as solution which can potentially lead to over fitting and training difficulty. The CNN model abstracts the image into potential image features through multi-layer convolution layer and pooling layer. And another advantage is that it does not rely on any other independent model and easily generalize to all the image in the dataset.

Before the output layer and after the convolutional layers and pooling layers, a fully-connected layer is used to scores the abstract features extracted automatically from the original image.

The output layer is a two-unit dense layer with SoftMax activation. The function of this layer is basically summarizing the scores from the previous dense layer and gives the scores that represents the probabilities for each classes of the image (male/female, smiling/not smiling).

Hyperparameters tuning is also important in building CNN classifier. A balance between the model performance and the training cost should be carefully considered.

3.2. Task B1: face shape recognition

Task methodologies in B1 is similar to that in A1 and A2. What is different is A1, A2 use *CelebA* as dataset to implement binary recognition models while B1 uses *Cartoon Set* as dataset to implement multi-class recognition models.

However, B1 is not going to use SVM, KNN or random forest as classifier, because in practice, converting face image into facial landmarks sometimes fail when the faces are not clearly shown in the image, and it happened in a large proportion of images in *Cartoon Set* for some reasons (detailly discussed in section 4).

Another solution is using CNN models. But before building the CNN models, the image should be resized to be small enough to simplify the model as well as saving the training cost.

3.3. Task B2: eye color recognition

B2 is to recognize the eye colors of cartoon character from *Cartoon Set*. In order to reduce the data dimension, remove interfering elements and simplify the model com-

plexity, preprocessing the image to extract the eye area of the image is prior to the model selection.

In addition, it is important to identify cartoon characters that wear opaque glasses, which can infer no information of the eye color.

And for the cartoon characters that wear colored-glasses, it is necessary to restore the eye color behind the glasses.

To find the characters that wear opaque glasses and restore the eye color behind the glasses, the assumptions are as followed:

Figure 2 Cartoon character without glasses (left), cartoon character with opaque sunglasses (middle), cartoon character with colored-glasses (right)

1) the presented pixel color is denoted as $pixel\_value$, the color of the glasses and the eye image in original color are denoted as $glass\_color$ and $original\_color$, the transparency of the glass is denoted as α.

$$pixel\_value = \alpha \times glass\_color + (1-\alpha) \times original\_color$$

2) for brightest pixel in eye image:

$$original\_color = (255,255,255)$$

for darkest pixel in eye image:

$$original\_color = (0,0,0)$$

In this assumption it is inferred that:
1) the color of the glass $glass\_color$ and the transparency of the glass α can be solved if the brightest pixel and darkest pixel in the eye image.
2) the eye image in original color can be calculated if $glass\_color$ and α are known.
3) if the brightest pixel is nearly dark, the eye is covered by the opaque glasses.

After removing the eyes covered by the opaque glasses and recovered the color of the eye image, the data is highly linearly separatable, a simple linear SVM is able to recognize the eye color.

4. IMPLEMENTATION

4.1. Dataset

4.1.1. Dataset Description

Celeba: The dataset contains 5,000 RGB images with 178*218 resolution each. Each image is a head image of a celebrity, most of the images are clear front faces, some are side faces and some are in inferior quality either under low luminance or have obvious distraction from other objects. The celebrities are in different environments and postures. Each image is accurately labeled with the gender and the emotion (smiling or not).

Figure 4 Clear front face of celebrity (left), side face of celebrity (middle), image with blurred details under low luminance (right)

Celeba_Test: This dataset is an extension of the *Celeba* dataset and contains 1,000 images. The final result of the model will be presented as the accuracy of this test dataset.

Cartoon_Set: The dataset contains 10,000 RGB images, each with a resolution of 500*500. Each image is a randomly generated cartoon head image. Each cartoon character head image has a fixed facial position and a unique combination of different personality including gender, skin color, hairstyle, eye color, face shape, eyebrows, beard and randomly generated glasses. Each image is correctly labeled with one of the five face shapes and one of the five eye colors.
Cartoon_Set_Test: The dataset is an extension of the *Cartoon_Set*, which includes 2,500 pictures, final scores of the models will be show as the accuracy tested on this dataset.

4.1.1. Dataset Separation

Celeba is used as the training set and validation set for tasks A1 and A2, the splitting ratio is 8:2.

Celeba_Test is used as testing set for tasks A1 and A2.

Cartoon_Set is used as the training set and validation set for tasks B1 and B2, the splitting ratio is 8:2.

Cartoon_Set_Test is used as testing set for tasks B1 and B2.

4.2. Coding environment and External libraries and files

4.2.1. Coding Environment

Operating system: windows 10
Language: Python 3.6

4.2.2. External libraries and files

External file:
shape_predictor_68_face_landmarks.dat:

A pre-trained face detector[3] used with dlib library to estimate the 68 facial landmarks[15].

External libraries:
Pandas. verision 1.1.3

A python open source library used in data analysis and manipulation.

Numpy version. 1.85.5

A fundamental package for scientific computing in Python.

Tensorflow version 2.3.1

Tensorflow is a powerful machine learning open source platform, which provides highly encapsuled machine learning tools.

matplotlib version 2.3.0

matplotlib is a library that provides visualization tools in Python.

Kerastuner verision 1.0.2

A efficient hyperparameter tuner for Keras. Tensorflow 2.3 or higher version is needed.

Scikit-learn verision 0.23.2

Scikit-learn is a python module for machine learning.

OpenCV verision 4.4.0

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library.

Dlib verision 19.8.1

A C++ toolkit that provides machine learning algorithms and tools.

4.3. Task A1, A2: gender and emotion classifications

4.3.1. SVM, KNN, random forest models

4.3.1.1. Data preprocessing

In this step, the facial landmarks should be firstly extracted from the original images to reduce the data dimension. The face detector[3], 'shape_predicotr_68_face_landmarks.dat', from dlib is used to extract.

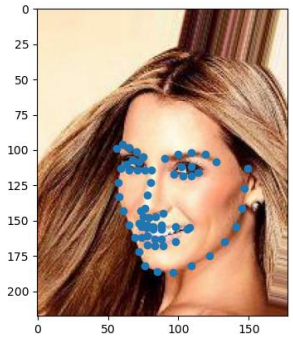

Figure 5 Facial landmarks of the celebrity

However, in the process of extraction, face detector[3] failed to detect facial feature points of 168 images (3.36% loss). Most of these images were caused by the incomplete display of facial information or the influence of ambient light.

Figure 6 Example that failed to detect facial landmarks because of low luminance (left) and example that failed to detect facial landmarks because of only one side of face available

The shape of landmarks is [4832,68,2], where 4832 is the number of successful samples extracted to landmarks, 68 is the number of facial points, 2 is the position of a facial point. The landmarks are then flattened into 136 features, after which the samples are rescaled and mean centered into distribution with zero mean and variation of one.

```
from sklearn.preprocessing import StandardScaler
Scaler = StandardScaler().fit(landmarks.reshape(-1,68*2))
landmarks = Scaler.transform(landmarks.reshape(-1,68*2))
```

Figure 7 Code to rescale and reshape landmarks

Noticed at the testing stage:

1. Only images that can be detected by the face detector[3] can run the test.
2. The testing landmarks should be rescale by the same scaler as previous one, which means shift factor and scale factor remain the same.

4.3.1.2. Hyperparameters tuning

The candidate models in this scheme are SVM, KNN, and random forest, each of them has their own hyper parameters. The Scikit-learn library provides powerful tool to build model and doing grid search of Hyperparameters.

Points to find hyperparameters are as followed:

1. Determine the scope of the hyperparameters

```
param_grid=[
    {'kernel': ['linear'],'C': [1e-2, 1, 10, 100, 1000]},
    {'kernel': ['poly'], 'gamma': [1e-2, 1e-3, 1e-4],'C': [1e-2, 1, 10, 100, 1000],'degree': [1, 3, 5]},
    {'kernel': ['sigmoid'], 'gamma': [1e-2, 1e-3, 1e-4],'C': [1e-2, 1, 10, 100, 1000],'degree': [1, 3, 5]},
    {'kernel': ['rbf'], 'gamma': [1e-2, 1e-3, 1e-4],'C': [1e-2, 1, 10, 100, 1000],'degree': [1, 3, 5]},
]
```

Figure 8 Hyperparameter scope for SVM

```
param_grid = {'n_neighbors':np.arange(1,101,5)}
```

Figure 9 Hyperparameter scope for KNN, number of neighbors from 1 to 101, increase by 5 every time

```
param_grid = {'n_estimators':np.arange(1,101,5)}
```

Figure 10 Hyperparameter scope for random forest, number of estimators from 1 to 101, increase by 5 every time

2. Doing grid search on the scoper, using GridSearchCV() function from Scikit-learn library.
3. Doing cross validation (fold-10)[16] and use the mean validation accuracy as the score for specific hyperparameter.

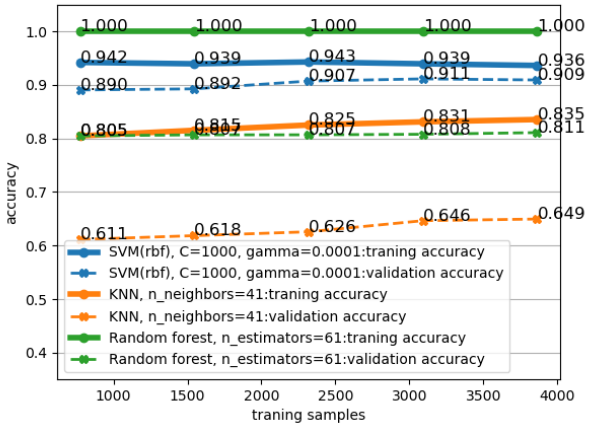

Figure 11 Training accuracies and validation accuracies of tuned SVM, KNN and Random Forest models versus the training samples in task A1

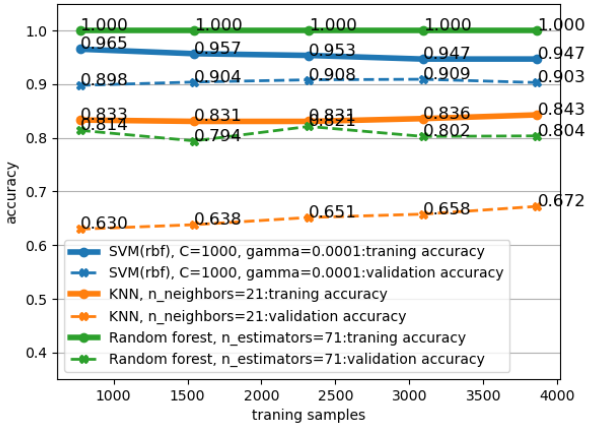

Figure 12 Training accuracies and validation accuracies of tuned SVM, KNN and Random Forest models versus the training samples in task A2

Figure 11 and Figure 12 show that for both task A1 and task A2, the optimal models are SVM with rbf kernel, achieving accuracy of 90.9% and 90.3% on validation set.

The scores for random forests for A1 and A2 are also acceptable with accuracy of 81.1 and 80.4. However the training accuracies for both tasks are 100% all the time in the training process, which indicates overfitting occurs.

4.3.2. CNN models

The input of CNN is the whole image. At the stage of hyperparameters tuning, no data preprocess is used in order to tune the hyperparameters as soon as possible. After the optimal hyperparameters has been found, additional data preprocess will be used to enrich the training data to avoid over-fitting to retrain the CNN model based on the hyperparameters found.

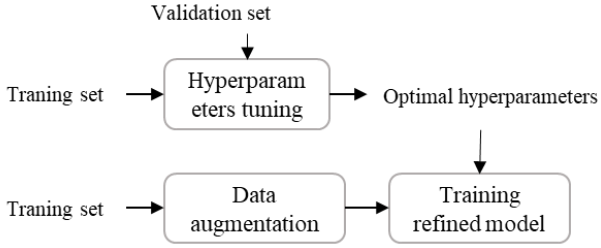

Figure 13 Pipeline of training optimal CNN model

4.3.2.1. Hyperparameters tuning

The basic CNN model can be divided into 3 part

1. Multiple combinations of a convolutional layer and a maxpooling layer
2. One fully-connected layer that voted on the flattened output of the previous part, denoted as voting layer for convenience.
3. An output layer with softmax activation to represent probability of each class

The fixed hyperparameters:

a. Each convolutional layer has kernels sized 3 by 3 and each maxpooling layer has kernels sized 2 by 2.
b. Use L2 regularization on part 2 of CNN model to avoid alleviate over-fitting.
c. Number of minibatch is 64 for empirical reason.
d. Learning rate is 1e-3 for empirical reason.

Hyperparameters tuning is implemented with kerastuner library, which provide efficient tuning tools.

Changeable hyperparameters:

a. number of convolutional layers (as well as number of-maxpooling layers) in scope from 32 to 128 with step 32
b. units for each convolutional layer in scope from 1 to 5
c. units for voting layer in scope from 32 to 256 with step 32

Training criteria:

a. Random search on the hyperparameters, maximum trial is 20.
b. Maximum number of the training epochs is set as 50
c. Early stop when the loss of the validation does not improve within 5 epochs
d. The hyperparameters that tuned the model with highest validation accuracy is regarded as the optimal hyperparameters.

4.3.2.2. Refined model

Based on the optimal hyperparameters, model is retrained with augmented training images and the model is trained until the validation loss had not improved for 15 epochs.

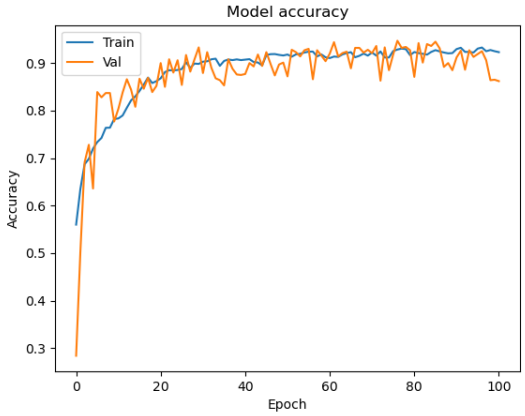

Figure 14 CNN training accuracy and validation accuracy versus number of epochs in task A1

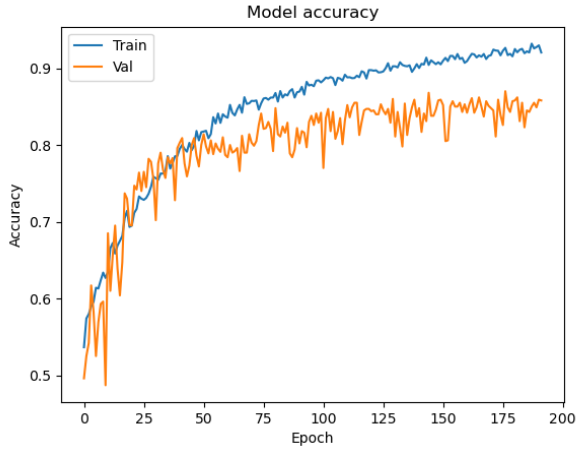

Figure 15 CNN training accuracy and validation accuracy versus number of epochs in task A2

4.4. Task B1: face shape recognition

4.4.1. CNN models

It is basically the same as in A1, A2, but the input image should be resized to resolution of 200 by 200 to decrease training difficulty. The output layer should have 5 units to represent the probability of 5 class.

4.4.1.1. Hyperparameters tuning

The learning rate is different from that in A1 and A2, lower learning rate 1e-4 is used because learning rate of 1e-3 does not converge at the very early stage.

4.4.1.2. Refined model

No data augmentation is used in B1 since that dataset is large enough and the image is uniform. Data augmentation may not improve the quality of the model. Like in A1, A2, the model is retraining with the optimal hyperparameters until validation loss had not improved for 15 epochs.

Like at the Hyperparameters tuning stage, learning rate of 1e-4 is used.

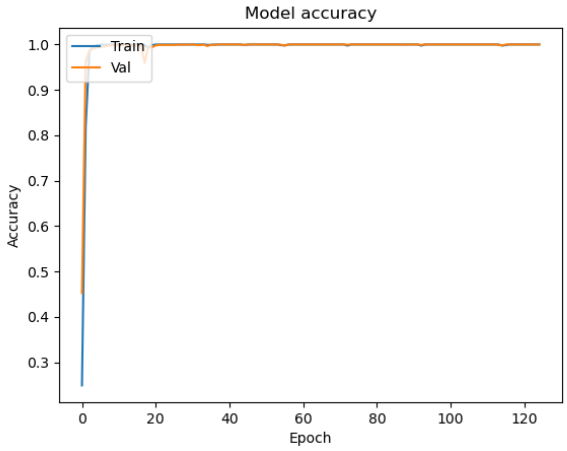

Figure 16 CNN training accuracy and validation accuracy versus number of epochs in task B1

4.5. Task B2: eye color recognition

In B2, not the whole image of the cartoon character is used as model input. What is concerned is the image of the left eye, especially which is close to the nature of the eye color.

The first step to cut out the left eye image from the whole image.

The next step is to detect the brightest pixel and darkest pixel in the eye image. If the brightest pixel is nearly dark, it is regarded that the opaque sunglasses is covering on the eyes.

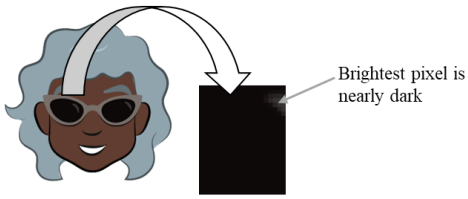

Figure 17 Left eye image of cartoon character wearing opaque sunglasses, all the pixels are dark

With the pixel value of the brightest pixel and darkest pixel in the raw eye image, the original color of the eye image can be recovered by calculating the transparency of the glasses and color of the glasses. If the cartoon character do not wear glasses, the calculated transparency is zero, the color of the eye image remains.

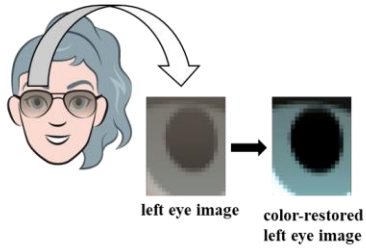

Figure 18 Pipeline from cut off the left image to restore the color of the left eye image

Finally, each color-restored left eye images is flattened as input of a SVM with linear kernel.
No hyperparameters needed because the restored images are linearly separatable. One shot is needed to obtain 100% accuracy on both training set and validation set.

5. EXPERIMENTAL RESULTS AND ANALYSIS

The following is the optimal models and their performance on different dataset. Noted that models that involves face detector are not selected since face detector sometimes failed to work, reducing the robustness of the models.

Task	Model	Train Acc	Val Acc	Test Acc
A1	CNN	0.939	0.945	0.939
A2	CNN	0.937	0.870	0.887
B1	CNN	1.0	1.0	1.0
B2	SVM	1.0	1.0	1.0

Although models in tasks A1 and A2 both use CNN, the same hyperparameters tuning methods, data preprocessing and model training method, they behave differently with a discrepancy of more than 5 percentage of the accuracies on the validation samples and test samples. Training accuracy nearly the same on both models may indicates that both the networks have struggled to fit the distribution of the training samples at this point. The reason for the great difference between them is that the variance of the distribution of the potential features extracted from the same datasets may differ from task to task. The features learned by the CNN model in task A1 are closer to the nature of the gender, so the accuracy of the training samples varies little from that of the validation/test samples. There are two assumption about why model in A2 fail to work as well as model in A1, either the CNN model is not suitable for revealing the nature of smile or too many redundant information indicating nothing about smile.

For B1, the training set, validation set and test set have reached 100% accuracy. This is because the distribution of the cartoon set data set is more consistent, and the model can learn their features better.

Model in B2 also achieved 100% accuracy. It has the similar reason as in task B1. After data preprocessing, the variance of the distribution of the samples that belongs to the same category is smaller, and the model is therefore easier to distinguish between different categorical samples.

6. CONCLUSION

This report using different machine learning models (SVM, KNN, random forest and CNN) to solve different recognition tasks, namely gender classification (A1), smile recognition (A2), face shape recognition (B1) and eye colors recognition (B2). Face features points are extracted from the images before using SVM with rbf kernel model in task A1 and task A2, with a minor loss in the dataset, the models achieve accuracies of 91% and 90%. However, face detector works poorly in task B1 where parts of the cartoon characters of the dataset wear glasses and have unclear facial features. On the other hand, CNN models have higher robustness in solving image recognition tasks A1, A2 and B1, which do not rely on any other pre-trained model. The CNN models also reach high accuracies of 94%, 89% and 100% for tasks A1, A2 and B1, respectively. For the eye colors recognition task (B2), data preprocessing which clip out the left eye images and recovers the eye color behind the colored glasses simplify the complexity of the problem and make the sample linearly separatable.

For future work, a more robust face detector can be used to increase the rate of success of extracting facial feature points. And B2 has shown the importance of data preprocess. Expressing the samples in a reasonable manner, either filtering out the redundant information or making the form of the samples more consistent, can make a difference.

It is also possible to increase the accuracy in the recognition tasks by adopting some other deep learning models, such as graph neural network[17] or attention based neural network[18].

12. REFERENCES

[1] S. Yang, P. Luo, C.C. Loy, and X. Tang, "From facial parts responses to face detection: A Deep Learning

Approach", in IEEE International Conference on Computer Vision (ICCV), 2015

[2] Google, “Cartoon Set: An Image Dataset of Random Cartoons,” https://google.github.io/cartoonset/.

[3] C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 faces In-the-wild challenge: Database and results. Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation "In-The-Wild". 2016

[4] Nugrahaeni, Ratna Astuti, and Kusprasapta Mutijarsa. "Comparative analysis of machine learning KNN, SVM, and random forests algorithm for facial expression classification." 2016 International Seminar on Application for Technology of Information and Communication (ISemantic). IEEE, 2016.

[5] Tang, Min, and Feng Chen. "Facial expression recognition and its application based on curvelet transform and PSO-SVM." Optik 124.22 (2013): 5401-5406.

[6] Uricár, Michal, et al. "Structured output svm prediction of apparent age, gender and smile from deep features." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2016.

[7] Agrawal, Saurabh, et al. "Content based color image classification using SVM." 2011 Eighth International Conference on Information Technology: New Generations. IEEE, 2011.

[8] Adeyanju, Ibrahim A., Elijah O. Omidiora, and Omobolaji F. Oyedokun. "Performance evaluation of different support vector machine kernels for face emotion recognition." 2015 SAI Intelligent Systems Conference (IntelliSys). IEEE, 2015.

[9] Nwosu, Lucy, et al. "Deep convolutional neural network for facial expression recognition using facial parts." 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech). IEEE, 2017.

[10] Antipov, Grigory, Sid-Ahmed Berrani, and Jean-Luc Dugelay. "Minimalistic CNN-based ensemble model for gender prediction from face images." Pattern recognition letters 70 (2016): 59-65.

[11] Tan, Mingxing, and Quoc V. Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." arXiv preprint arXiv:1905.11946 (2019).

[12] Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." The Journal of Machine Learning Research 13.1 (2012): 281-305.

[13] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[14] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[15] i·bug, “i·bug - resources - Facial point annotations”, https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/

[16] Browne, Michael W. "Cross-validation methods." Journal of mathematical psychology 44.1 (2000): 108-132.

[17] Scarselli, Franco, et al. "The graph neural network model." IEEE Transactions on Neural Networks 20.1 (2008): 61-80.

[18] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017): 5998-6008.