Project A

Problem Description

Give you a picture of a person, could you tell me whether he/she is smiling? Please make your computer to learn smile detection.

Is she smiling now? Let your computer answer this question.

How to solve such a problem?

This is a typical classification problem. In this project, we hope you can solve it using traditional machine leaning approaches.

1, Dataset

Here we use GENKI-4K dataset to train and test your algorithms. GENKI-4K contains 4,000 face images spanning a wide range of subjects, facial appearance, illumination, geographical locations, imaging conditions, and camera models. All images are labeled for both Smile content (1=smile, 0=non-smile) and Head Pose (yaw, pitch, and roll parameters, in radians).

Download it: http://mplab.ucsd.edu/wordpress/wp-content/uploads/genki4k.tar

2, Definition

Train/Test set: Divide the dataset into train set and test set. Images in train set are used to train a model, and images in test set are used to test and evaluate a model.

Positive samples: images labeled as smile.

Negative samples: images labeled as non-smile.

10-fold cross validation: Divide datasets into 10 folds, 9 for train and 1 for test. Do it for 10 times, and compute the average.

F1 score: Metric to measure your model. For classification tasks, the terms *true positives*, *true negatives*, *false positives*, and *false negative* compare the results of the classifier under test with trusted external judgments. The terms positive and negative refer to the classifier's prediction (sometimes known as the expectation), and the terms true and false refer to whether that prediction corresponds to the external judgment (sometimes known as the observation). Let us define an experiment from positive instances and negative instances for some condition. The four outcomes can be formulated in a 2×2 contingency table or confusion matrix, as follows:

		Ground truth	
		Positive	Negative
Predicted results	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True Negative (TN)

$$F_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

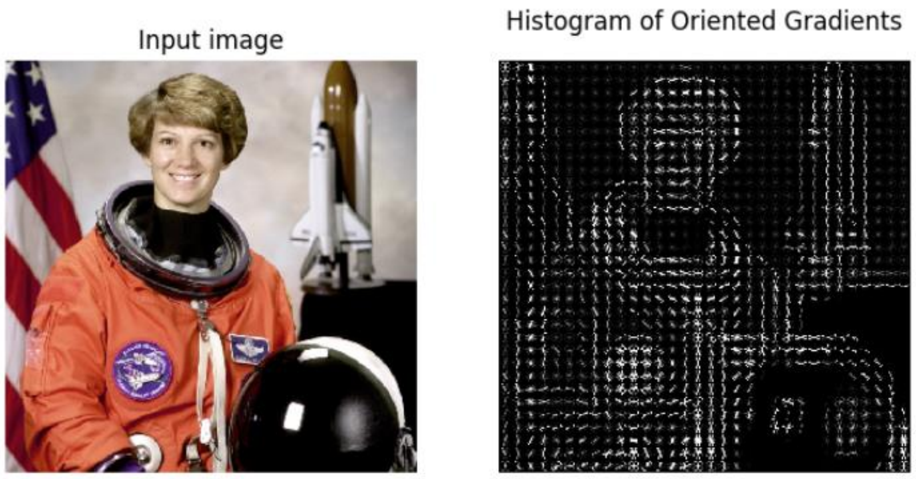
3, Process of smile detection

Give you an image, first, find the face. Next classify whether it is smile or non-smile.

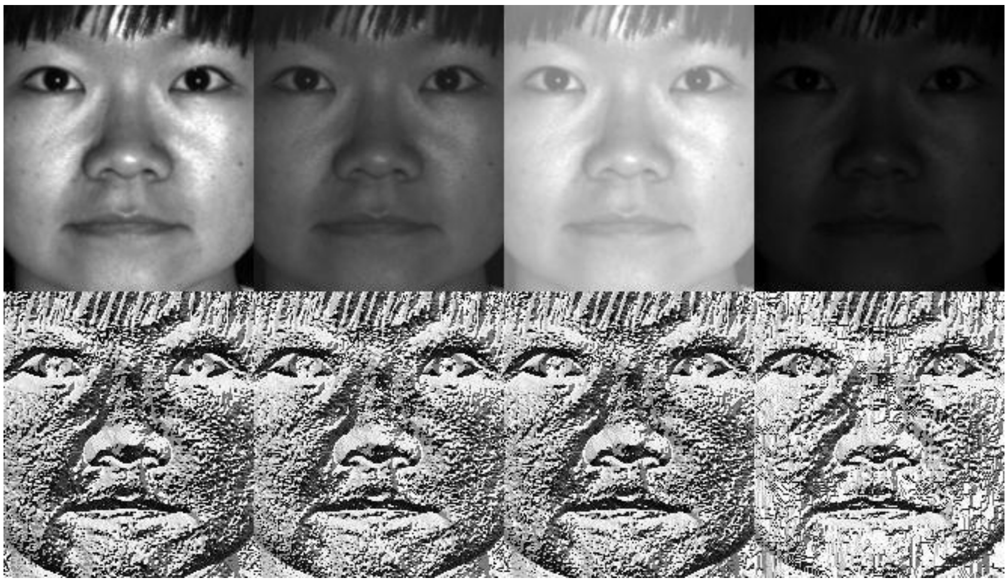
For the first step, you can directly use packages, for example, face detector in opencv/dlib.

For the next step, you should train a model by yourself. The input of the model should be the faces you extract from images, and the output should be predicted results. Classification usually contains two steps, feature extraction and classification. Here we introduce two kinds of feature, HOG and LBP, and one classification method, SVM. You needn’t learn exact computing process and theory of them, but you need understand what their main idea and how they should be used.

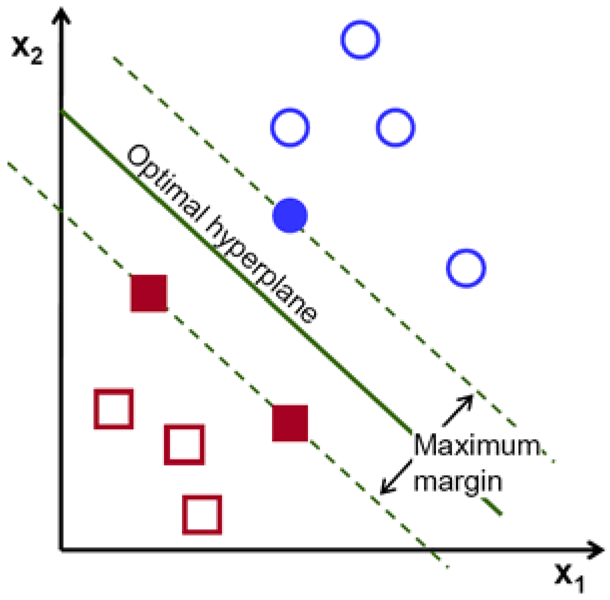
HOG, histogram of oriented gradients. The essential thought behind the HOG descriptor is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. The HOG descriptor has a few key advantages over other descriptors. The following is an example for HOG.



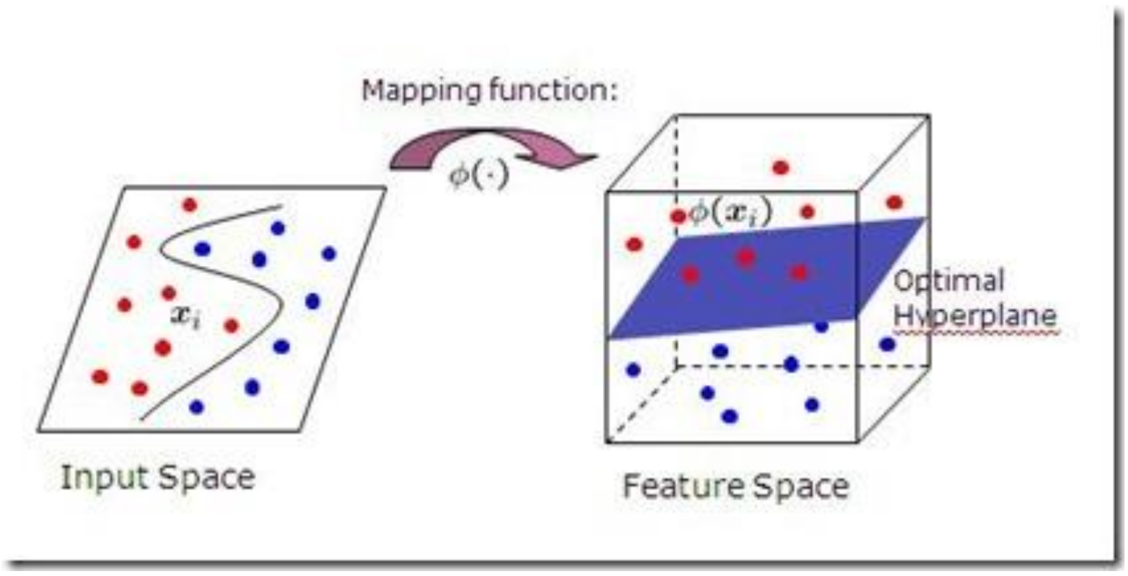
LBP, Local binary patterns. LBP has been shown powerful and is often used in problems about human face. The simple LBP records the contrast information between the pixel and the surrounding pixels. LBP is used to describe local texture features. The following is an example.



SVM, Support vector machine. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible, as the following figure shows.



New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces, as the following figure shows.



Your tasks in this project

1, Face detection. (5')

Using any face detector tool to detect faces in those images.

2, Smile classification. (15')

Using face as input, train a model by yourself, and test your model. Choose the best one for classification. Here, you can choose HOG or LBP or both as features, and you could also use other features if you like, for example, SIFT. You can use SVM as classification algorithm, and you could also use other algorithms if you like. We encourage different ways to solve the same problems. We also encourage some groups to compare different ways. Smile classification is the most important one for this project. We will test your model.

3, Your own real-time Smile Detection application using camera (5')

After you have trained your own model, we hope you can use it. For example, could you please open your camera on your computer and detect whether you are smiling real-time?

Note:

1) No any deep learning model (CNN);

2) Choose or design any feature and classification method by yourself;

3) Encourage novel idea.

All codes, files and report should be in one .zip or .rar file.