COMPUTATIONAL ANALYSIS OF TONGUE IMAGE
FOR HEALTH DIAGNOSIS

A Thesis
presented to
the Faculty of the Graduate School
at the University of Missouri-Columbia

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
MENG ZHANG
Dr. Dong Xu, Thesis Supervisor
May 2017
The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

**COMPUTATIONAL ANALYSIS OF TONGUE IMAGE FOR HEALTH DIAGNOSIS**

presented by **Meng Zhang**, a candidate for the degree of **Master of Science**, and hereby certify that, in their opinion, it is worthy of acceptance.

---

**Professor Xu Dong**

**Professor Duan Ye**

**Professor Li Hao**
ACKNOWLEDGEMENTS

To begin with, I would like to express my sincere appreciation to my adviser, Dr. Dong Xu, for his outstanding guidance and valuable suggestions. I was inspired by him in the whole process while doing this thesis research. Without his encouragement and support, I would not have been able to complete this research smoothly and efficiently.

Furthermore, many thanks to Dr. Ye Duan, for providing me a nice place to study in, such a great lab, and his constructive suggestions always gave me a lot of help. Thanks to my committee member, Prof. Hao Li, for reviewing and giving suggestions on my research.

In addition, thanks to Dr. Liping Tu and Xu Wang, for offering lots of help on my thesis research. All the labels of tongue images are provided by Dr. Liping Tu. Tongue image segmentation algorithms are implemented by Xu Wang.

Finally, I would like to convey my special thanks to my family, for their support. Without their encouragement, understanding and trust, I won’t have opportunities to study and chase my dream here.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>ii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>ix</td>
</tr>
<tr>
<td>CHAPTER 1  INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>CHAPTER 2  TONGUE RECOGNITION</td>
<td>4</td>
</tr>
<tr>
<td>2.1  Introduction</td>
<td>4</td>
</tr>
<tr>
<td>2.2  Naïve Bayes Classification Method</td>
<td>5</td>
</tr>
<tr>
<td>2.2.1  Training Set and Cross Validation</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2  Features</td>
<td>8</td>
</tr>
<tr>
<td>2.2.3  Color Features</td>
<td>8</td>
</tr>
<tr>
<td>2.2.4  Color Smoothness Features</td>
<td>9</td>
</tr>
<tr>
<td>2.2.5  Feature Extraction</td>
<td>10</td>
</tr>
<tr>
<td>2.2.6  Distribution of features</td>
<td>12</td>
</tr>
<tr>
<td>2.2.7  Result</td>
<td>15</td>
</tr>
<tr>
<td>2.3  SVM Classification Method</td>
<td>16</td>
</tr>
<tr>
<td>2.3.1  Training Set and Feature Extraction</td>
<td>16</td>
</tr>
<tr>
<td>2.3.2  Result</td>
<td>17</td>
</tr>
<tr>
<td>2.4  Comparison and Analysis</td>
<td>19</td>
</tr>
<tr>
<td>2.5  Tongue Shape Symmetry</td>
<td>21</td>
</tr>
<tr>
<td>2.5.1  Image Rotation and Getting Contour</td>
<td>21</td>
</tr>
<tr>
<td>2.5.2  Correlation</td>
<td>24</td>
</tr>
<tr>
<td>2.5.3  Result</td>
<td>25</td>
</tr>
<tr>
<td>2.6  Conclusion</td>
<td>26</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 2-1 Different tongue images with different color scales .................................................. 9
Figure 2-2 Original image and mask image from segmentation algorithm ...................................... 11
Figure 2-3 Combined new image and cropped image ......................................................................... 12
Figure 2-4 Color feature histograms on tongue and non-tongue class ............................................ 13
Figure 2-5 Color smoothness feature histograms on tongue and non-tongue class .......................... 13
Figure 2-6 Color feature distributions on tongue and non-tongue class ........................................... 14
Figure 2-7 Color smoothness feature distributions on tongue and non-tongue class ..................... 15
Figure 2-8 Misclassified non-tongue image samples ........................................................................ 19
Figure 2-9 Misclassified tongue image samples .............................................................................. 19
Figure 2-10 Original imported image and segmented mask ............................................................. 22
Figure 2-11 positive15 degree and negative15 degree rotation mask images ............................... 23
Figure 2-12 Corrected image ROI and contour ............................................................................... 24
Figure 2-13 Correlation on left and right part of contour ............................................................... 25
Figure 3-1 Segmented tongue image partitions ................................................................................. 30
Figure 3-2 One tongue image under 7 color spaces ....................................................................... 31
Figure 3-3 (a) Image taken in stable Light Source Box ....................................................................... 34
(b) Common fluorescent lamp ....................................................................................................... 34
Figure 3-4 (a) Minority samples K Nearest Neighbors ..................................................................... 37
(b) A new sample created by KNN distances .................................................................................. 37
Figure 3-5 X-rite Color Checker ..................................................................................................... 40
Figure 3-6 Iphone5S 85 images training set fitting results on Standard Color Values and Input Color Values ......................................................................................................................... 41
Figure 3-7 Shanghai 257 images training set fitting results on Standard Color Values and Input Color Values ......................................................................................................................... 41
Figure 3-8 Original image and corrected image example .................................................................. 42
Figure 3-9 Five classes tongue images ............................................................................................. 45
Figure 4-1 Tongue Scan function screenshot .................................................................................. 53
Figure 4-2 Image uploading failed screenshot ................................................................. 55
Figure 4-3 Questionnaire and Result pages screenshots ........................................... 56
Figure 4-4 Instructions on Hot-Cold status and Physical type screenshots .................. 57
Figure 4-5 Ask A Doctor function screenshots .................................................................. 58
Figure 4-6 Customer checkout pages on ‘Ask A Doctor’ function screenshots ............... 59
Figure 4-7 Doctor Analysis System webpage screenshot ................................................. 60
Figure 4-8 Register and Login screenshots ....................................................................... 61
Figure 4-9 Password forgotten screenshots ...................................................................... 63
Figure 4-10 Setting page contents screenshots .................................................................. 64
Figure 4-11 Resources webpage contents screenshots ...................................................... 65
LIST OF TABLES

Table 2-1 Recognition tongue features on Naïve Bayes Classifier results…………………………15
(a) Confusion matrix ........................................................................................................................................15
(b) Classification results.....................................................................................................................................16
Table 2-2 Recognition tongue features on SVM Classifier results .................................................17
(a) Confusion matrix ........................................................................................................................................17
(b) Classification results.....................................................................................................................................18
Table 2-3 Recognition correlation on SVM Classifier results.......................................................... 26
(a) Confusion matrix ........................................................................................................................................26
(b) Classification results.....................................................................................................................................26
Table 3-1 85 images training set SVM classification confusion matrix..................................................35
Table 3-2 85 images training set SVM classification results ...............................................................35
Table 3-3 257 images training set SVM classification confusion matrix...................................................36
Table 3-4 257 images training set SVM classification results ...............................................................36
Table 3-5 SMOTE application training set confusion matrix .................................................................38
Table 3-6 SMOTE application training set classification results .........................................................38
Table 3-7 Combined training set confusion matrix .................................................................................39
Table 3-8 Combined training set classification results .............................................................................39
Table 3-9 Pearson correlation results .........................................................................................................40
Table 3-10 Corrected combined training set confusion matrix ...............................................................42
Table 3-11 Corrected combined training set classification result .............................................................42
Table 3-13 SMOTE corrected combined training set classification results ............................................43
Table 3-14 Five classes original and SMOTE results .............................................................................46
(a) Confusion matrix ........................................................................................................................................46
(b) Classification results.....................................................................................................................................46
Table 3-15 Normal class classifier original and SMOTE results .............................................................47
(a) Confusion matrix ................................................................. 55
(b) SMOTE confusion matrix ......................................................... 47
(c) Classification results ................................................................. 47
(d) SMOTE Classification results ..................................................... 47

Table 3-16 Qi-deficiency class classifier original and SMOTE results ...................... 48
(a) Confusion matrix ................................................................. 48
(b) SMOTE confusion matrix ......................................................... 48
(c) Classification results ................................................................. 48
(d) SMOTE classification results ..................................................... 48

Table 3-17 Phlegm-dampness class classifier original and SMOTE results ................. 48
(a) Confusion matrix ................................................................. 47
(b) SMOTE confusion matrix ......................................................... 48
(c) Classification results ................................................................. 49
(d) SMOTE classification result ....................................................... 49

Table 3-18 Qi-deficiency and Phlegm-dampness class classifier original and SMOTE results ... 48
(a) Confusion matrix ................................................................. 48
(b) SMOTE confusion matrix ......................................................... 49
(c) Classification results ................................................................. 49
(d) SMOTE classification results ..................................................... 50

Table 3-19 Blood-stasis class classifier original and SMOTE results ................................ 50
(a) Confusion matrix ................................................................. 49
(b) SMOTE confusion matrix ......................................................... 50
(c) Classification results ................................................................. 50
(d) SMOTE classification results ..................................................... 50
Tongue, the primary taste organ in the mouth, can reflect the whole body’s health conditions based on the Traditional Chinese Medical (TCM) theories. Watching the tongue is one of the most common, essential and reliable methods for the TCM doctor to make diagnoses.

In this thesis, a new health system is introduced based on tongue image analysis. The technologies adopted in this system ranged from tongue image processing algorithms to machine learning applications. The tongue image algorithms used in this work include image segmentation, tongue recognition and tongue image classification. Image segmentation was used to get rid of other unrelated parts, such as lip, face and neck, while keeping the tongue only. Then two recognition methods were applied to check whether the segmented result is a tongue or not. For different tongue patterns, the Support Vector Machine is applied to train a machine learning model and make predictions to classify the tongue into different labeled groups.

An app named ‘iTongue’ is designed to monitor the body status by taking and processing tongue images in smart phones. The app provides a user-friendly, fast and powerful health tool based on TCM theories. The whole system is implemented in a web-based environment. An advanced portal was developed to connect the users and the TCM
doctors. The users will not only obtain the analysis label of tongue images, but also get some life style recommendations based on the tongue image analysis. This portal helps the user understand more about his body status and guide him to adopt a more suitable diet and improve exercise.
CHAPTER 1 INTRODUCTION

In ancient times, without the developed electronic technologies as support, most diagnoses were made by virtue of the doctor’s experiences, based on some visible measurement and pulse taking on patients. Especially for the TCM doctors, watching the tongue is one of the essential ways to acquire health information on patients [1]. Similarly, with fingerprints, everybody’s tongue is not exactly the same. Different tongue patterns convey unique and personal body information of individuals, and distinct parts of the tongue also reflect distinct organ conditions of the human body [2].

First of all, in order to make the whole tongue analysis process more rigorous, tongue recognition function is added after image segmentation. Two approaches are applied to determine whether an input segmented image is a tongue image or not. The first one is based on the features of tongue images, since tongue color is unique and generally distinguished from other objects. Tongue color features are extracted from two color spaces- RGB and CIELAB* (B* means it is different with the channel B from RGB). Then based on these color features, images can be classified into two groups: tongue and non-tongue. Two methods are applied to the classification and both of them are model based. One is Naïve Bayes Classifier, which is a statistical classification method based on Bayes’ theorem; the other is SVM, which is a machine learning method based on statistical learning theory. The comparison of the results on these two methods is illustrated at the end of chapter 2. The other approach is based on the shape of tongues,
since most of qualified segmented tongue images are supposed to be symmetric. The symmetry axis of the segmented input image can be found by the axis rotations. Once the symmetry axis is available, the correlated value of the left and right part can be calculated by the distance comparison. This correlated value demonstrates whether the shape is symmetric or not. These two approaches are combined together to restrict the input segmented images into tongue images.

After the tongue recognition, two classification functions are introduced based on two different TCM standards. One is the Hot-Cold standard, through which the classification function will classify tongue images into three groups: Hot, Hot-Cold Balanced and Cold. The other is physical type standard, by which images can be classified into five groups: Normal, Qi-deficiency, Phlegm-dampness, Qi-deficiency and Phlegm-dampness, Blood-stasis. In both of the two classification functions, the machine learning method SVM is employed to do the model learning and prediction. In preprocessing, the tongue image is first partitioned into five separated parts and 7 color spaces are applied, which includes 21 color channels as the color features. The features on the left, right and bottom parts combined, express the tongue body color, and the features on the middle part demonstrate the tongue coating color. When classification functions are finished, two prediction labels are generated.

A smart phone app is developed with many functional modules. One of the primary functions embedded with tongue recognition and classification functions, the “Tongue Scan” function. With the app installation on smart devices, such as cell phones (IOS or Android) and tablets, a user can use “Tongue Scan” function to take tongue images. Exceptionally, if the image is not in good quality or it is not identified as a
tongue image, the function will notify the user to retake or reload. When the user uploads the tongue image via “Tongue Scan” function, two analysis labels will be returned. One label is based on hot-cold status analysis with food recommendations; the other is on physical analysis.

Another function of the app is called “Ask A Doctor”. If the user prefers a more sophisticated and professional diagnosis, he can select as many as four tongue images and send them to real doctors, since an advanced portal is provided to connect the users with real TCM doctors. A comprehensive report will be sent back to user’s email address as soon as the doctors finish their analysis. Within in the report, not only is the risk analysis provided, but also various lifestyles are recommended including food and sports.

The significance of this app is to make more people realize the changes of the tongue pattern and then pay attention to the body changes.
CHAPTER 2  TONGUE RECOGNITION

2.1 Introduction

Tongue recognition is used to examine whether an input segmented image is a tongue or not. This function is essential, since it can make the entire “Tongue Scan” function much more reliable and powerful. If the images are not filtered, all the non-tongue images can also go through the classification function, and then the new labels based on color features are generated. The whole process for non-tongue images is meaningless. This noise will weaken the whole process and definitely does not bring any valuable information. In order to improve the quality and efficiency of the entire computational process, the tongue recognition part included is necessary.

In this chapter, two approaches are adopted to restrict whether an image is a tongue or not. In 2.2, 2.3 and 2.4, one approach is introduced, which includes two methods on tongue features classification, Naïve Bayes (2.2) and SVM (2.3). Both of the two methods are based on tongue color features and color smooth consistency features, and they are model based methods. The comparison (2.4) and analysis was done on the results of two methods. In 2.5, another approach is introduced regarding the tongue shape. Suppose all the tongue images are symmetrical, the tongue can be divided into two parts by the symmetric axis: left and right. Correlation on the left and right part can tell whether this object is symmetrical or not. The machine learning method applications of
the first approach on tongue features were combined with the shape information of the second approach together to recognize tongues.

In the result section of this chapter, the confusion matrix is given to show the results and provide some discussion on each approach.

2.2 Naïve Bayes Classification Method

Naïve Bayes is a traditional classification method that is based on the independence assumption and Bayesian Theorem. This potential probabilistic model typically adopted independent features. The basis of a Bayesian classifier is probabilistic reasoning. Although all the conditions are uncertain, the reasoning and decisions can be made based on the probabilities.

One of the benefits of this classifier is that the essential parameters (mean value and variance) can be estimated via training a small group of samples. The other benefit is that the whole covariance matrix does not need to be taken care of, since the variables are under independent assumption.

Based on Bayes theorem, \( p(C|F_1, ... F_n) \) is defined as the posterior to be solved, independent variable \( C \) may contain many labels. In this chapter of tongue recognition, two labels are defined: tongue label and non-tongue label. \( C \) is conditional dependent on features: \( F_1, F_2, ..., F_n \). Here, only 12 features are considered totally (\( n=12 \)), 6 of which are color features and the other 6 are color smoothness consistency features. Based on Bayesian Theorem, the equation:

\[
\text{posterior} = \frac{\text{prior} \cdot \text{likelihood}}{\text{evidence}}
\]
\[
p(C|F_1, ...F_{12}) = \frac{p(C) \cdot p(F_1, ...F_{12}|C)}{p(F_1, ...F_{12})}
\]

\(p(C)\) is prior probability; \(p(F_1, ...F_{12}|C)\) expresses the likelihood and \(p(F_1, ...F_{12})\) represents the evidence. Especially,

\[
p(C|F_1, ...F_{12}) \propto p(C) \cdot p(F_1, ...F_{12}|C)
\]

\[
\propto p(C) \cdot p(F_1|C) \cdot p(F_2, ...F_{12}|C, F_1)
\]

\[
\propto p(C) \cdot p(F_1|C) \cdot p(F_2|C, F_1) \cdot p(F_3, ...F_{12}|C, F_1, F_2)
\]

\[
\propto p(C) \cdot p(F_1|C) \cdot p(F_2|C, F_1) \cdot ... \cdot p(F_{12}|C, F_1, F_2, ..., F_{12})
\]

Due to the premise of independence of each feature, so \(F_i\) is conditional independent with \(F_j\) \((i \neq j)\). It means \(p(F_i|C, F_j) = p(F_i|C)\). Therefore:

\[
p(C|F_1, ...F_{12}) \propto p(C) \cdot p(F_1|C) \cdot p(F_2|C) \cdot ... \cdot p(F_{12}|C)
\]

\[
\propto p(C) \cdot \prod_{i=1}^{12} p(F_i|C)
\]

So the equation can be deduced to: \(p(C|F_1, ...F_{12}) = \frac{1}{Z} \cdot p(C) \cdot \prod_{i=1}^{12} p(F_i|C)\), in which the ‘\(Z\)’ is just a constant factor depending on \(F_1, ..., F_{12}\).

In order to obtain the posterior value of each input image, prior \(p(C)\) needs to be calculated first. It is not a distribution but a value. It expresses all classes in proportion in the whole sample space. Here \(p(C)\) means the ratio of class \(C\) in the whole. Secondly, distribution along each feature \(F_i\) in each class must be generated. The 12 features’ histograms of tongue and non-tongue classes were shown in Figure 2-4, 2-5, 2-6 and 2-7. When the essential parameters of each feature distribution are available, the
probability density functions of each distribution can be generated. When a new segmented image comes to the model, the posterior probability value of the corresponding feature can be calculated under this distribution in the equation:

\[
P(x = v|C) = \frac{1}{\sqrt{2\pi \sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}.
\]

Finally, the result labels can be obtained by probabilities comparison between the tongue class and non-tongue class. The class in larger probability can be viewed as the prediction label of this image.

This method is implemented in Matlab and the results are shown in Table 2-1.

### 2.2.1 Training Set and Cross Validation

There are 854 images in the training set, which includes 427 tongue images and 427 non-tongue images. Among them, 170 tongue images are collected by iphone5S (85 are taken in flash mode and the other 85 are in non-flash mode) at the University of Missouri Life Science Building Lab110 under fluorescent lamp light source condition. The other 257 tongue images are collected in Shanghai TCM University. The 427 non-tongue images include fruits, natural scenes and other objects, and the colors are various.

In order to test the performance of the methods, the 10 folds cross validation is employed. The order of 857 images is randomly permutated and divided into 10 folds, 9 of which are used as training set and the residue one fold as testing. In order to make it more general, this process is repeated 100 times to get the average accuracy. Each time the order of the images is unique and random.
In this project, the labels for tongue images are defined as ones and the non-tongue images are zeros.

2.2.2 Features

The color and color consistency of tongue images are unique and distinguished with other irrelevant object images, such as elbow, leg, bread, pomegranate, and so on. Tongue color ranges from pale reddish to red, sometimes a little yellow on coating or a little purple on body; the degree of color changing on tongue surface is not very smooth as normal body skin or fruit surface, since there is lots of fungi form papilla on the tongue body. So a model can be learned to capture the pattern of tongue to distinguish tongue images and non-tongue images.

2.2.3 Color Features

Two color spaces RGB and LAB* are employed.

In RGB color space, there are 3 color channels: Red (R), Green (G) and Blue (B). The RGB color space is the most common and has wide usage in research and applications. It is used for the sensing, representation, and display of images in electronic systems, and it is an additive color model in which red, green and blue are added together in various ways to reproduce a broad array of colors.

In LAB* color space, there are 3 color channels; they are Luminosity (L), A and B*. The space itself is a three-dimensional real number space that contains infinite possible representations of colors. However, in practice, the space is usually mapped onto a three-dimensional integer space for device-independent digital representation, and for
these reasons, the L, A, and B* values are usually absolute, with a pre-defined range. The lightness, L, represents the darkest black at L = 0, and the brightest white at L = 100. The color channels, A and B*, represents true neutral gray values at A = 0 and B* = 0. The red/green opponent colors are represented along A axis, with green at negative A values and red at positive A values. The yellow/blue opponent colors are represented along the B* axis, with blue at negative B* values and yellow at positive B* values. The scaling and limits of the A and B* axis depend on the specific implementation of LAB* color, as described below, but they often run in the range of ±100 or −128 to +127.

The colors of tongue images are ranged from pinky white to reddish, some of the tongue images are shown with the yellow coating or the purple body color. Figure 2-1 shows the different colors of tongue images.

![Tongue Images](image)

(a) White tongue  (b) Pink tongue  (c) Reddish tongue

*Figure 2-1 Different tongue images with different color scales*

### 2.2.4 Color Smoothness Features
In order to demonstrate the color smoothness on each tongue image, the standard deviation of color modification is applied which could show whether the color consistency on an object deviates severely or just a little from the mean value of color features. In statistics, the standard deviation (SD, also represented by the Greek letter sigma \( \sigma \) or the Latin letter \( s \)) is a measure that is used to quantify the amount of variation or dispersion of a set of data values. A low standard deviation indicates that the data points tend to be close to the mean value of the set, while a high standard deviation indicates that the data points are spread out over a wider range of values.

2.2.5 Feature Extraction

In the segmentation part, the input is just a regular image, the output of segmentation is a binary image, called “mask”, in which the object segmented is in white while all the other parts are in black [3][4]. The format of the output image is PPM, which is a lossless image format. The size of the binary image is the same with the original image. Figure 2-2 shows the details.
After the segmentation, the color feature extraction is followed. The inputs of this algorithm are two images, one is the original image, and the other is the output binary image from segmentation. These two images are combined together to get a new image, which can be cropped to an object fully taken image, Figure 2-3 showed this new image and the cropped one.
First of all, the color features on the object part are extracted by pixels. For each pixel, there are six features, which correspond to the six color channels, respectively. Secondly, the mean values of all the pixels (color features) and the standard deviation values (color smoothness features) of each color channel are calculated. It means, for each tongue, there are 12 features: mean values and the corresponding standard deviation values of R, G, B, L, A, B* in each channel.

2.2.6 Distribution of features

The histograms of color features R, G, B, L, A and B* in 427 tongue images and 427 non-tongue images are shown below in Figure 2-4. The color blue stands for the Tongue group and color yellow stands for Non-tongue group. The x-axis expresses the feature values and y-axis represents the histogram values of each feature interval.
The histograms of color smoothness features R, G, B, L, A and B* in 427 tongue images and 427 non-tongue images are shown below in Figure 2-5.
From the histograms generated, in these two classes, each feature can be viewed as normal distribution. So it is assumed that each of them follows Gaussian distribution. Figure 2-6 shows the Gaussian distributions on color features in tongue class and non-tongue class, and Figure 2-7 shows color smoothness features in tongue and non-tongue class. The x-axis represents the corresponding values of R, G, B, L, A and B*, and the y-axis represents the probability density function values. In Figure 2-6 and 2-7, color red, green, blue, pink, yellow, aquamarine curves express feature R, G, B, L, A, B* respectively on tongue class, and all black curves represent those features on non-tongue class. For each sub image, comparison can be made between the same features on tongue and non-tongue images.

*Figure 2-6 Color feature distributions on tongue and non-tongue class*
Figure 2-7 Color smoothness feature distributions on tongue and non-tongue class

2.2.7 Result

The Naïve Bayes method is implemented in WEKA3.7 under WINDOWS 10 i5-3210M @ 2.5GHZ environment. The total running time is 0.26 seconds.

Table 2-1 Recognition tongue features on Naïve Bayes Classifier results

(a) Confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as Tongue images</th>
<th>Classified as Non-tongue images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Tongue images</td>
<td>397</td>
<td>30</td>
</tr>
<tr>
<td>Actual Non-tongue images</td>
<td>10</td>
<td>417</td>
</tr>
</tbody>
</table>
(b) Classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.953</td>
<td>0.047</td>
<td>0.954</td>
<td>0.953</td>
<td>0.953</td>
<td>0.907</td>
</tr>
</tbody>
</table>

From results of the confusion matrix in table (a), for non-tongue images, there are 417 classified correctly and 10 misclassified. So the weighted average FPR (false positive rate) is: 0.047. For tongue images, there are 397 images are classified correctly and 30 are incorrectly. The weighted average TPR (true positive rate) is 0.953. The accuracy can reach 95.3% and it performs well in this images set.

2.3 SVM Classification Method

2.3.1 Training Set and Feature Extraction

The SVM based tongue recognition applied in this chapter is efficient and in low computational cost, especially for the non-linear classification problems. It is also a model based machine learning method. So the features and labels are necessary to learn a model.

The features imported for SVM based methods are the same as applied to Naïve Bayes classification method: 12 features in total, 6 of them are from color features and the other 6 are from color smoothness features. Tongue labels are ones and non-tongues are zeros. The training set and testing set are also the same with the previous method.
SVM is applied by the third party implementation package “Libsvm”, which provides five SVM types and four kernel functions. In this chapter, C-SVC is chosen as the SVM type, and for the kernels, both linear kernel and polynomial kernel are tried.

When SVM is adopted on the training set of 854 images, a model is trained with 10 folds cross validation and the results are obtained with confusion matrix and accuracy in Table 2-2. This trained model could be used to predict new input images later for tongue recognition.

2.3.2 Result

This method is implemented in SVM of WEKA3.7 under WINDOWS 10 i5-3210M @ 2.5GHZ environment. Running time is 7.16 seconds without parameters turning. The SVM Classifier applied is from the third party functions “Libsvm” with C-SVC and Linear kernel type. Parameters of cost and gamma are default.

<table>
<thead>
<tr>
<th></th>
<th>Classified as Tongue images</th>
<th>Classified as Non-tongue images</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual Tongue images</strong></td>
<td>403</td>
<td>24</td>
</tr>
<tr>
<td><strong>Actual Non-tongue images</strong></td>
<td>5</td>
<td>422</td>
</tr>
</tbody>
</table>
(b) Classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.966</td>
<td>0.034</td>
<td>0.967</td>
<td>0.966</td>
<td>0.966</td>
<td>0.933</td>
</tr>
</tbody>
</table>

From results of the confusion matrix in table (a), there are 403 tongue images classified into the tongue group correctly and 24 misclassified to the non-tongue group. For the non-tongue group, there are 422 non-tongue images correctly classified and 5 misclassified. The weighted average TPR is 0.966 and FPR is 0.034.

Different kernel types are adjusted to make sure the non-tongue images are classified as accurately as possible. In this tongue recognition problem, it is preferable that fewer non-tongue images go through the algorithm, since it is meaningless to generate a Hot-Cold status label on a non-tongue image. So in the case of similar accuracy, the fewer images misclassified in the non-tongue group, the better performance of the classifier.

In the non-tongue image group, several images are classified to the tongue image group, such as the samples shown below in Figure 2-8. The reason why they are misclassified is that color features are very close to the tongue color range, such as red dress, yellow persimmon, and pink stuffs.
In the tongue image group, images classified to the non-tongue image group have similar features. Samples are shown below in Feature 2-9. The reason why they are misclassified is that color features on tongues are far away from the tongue color range, such as sever white coating and thick yellow coating.

2.4 Comparison and Analysis

From the results of two methods mentioned above: Naïve Bayes and SVM, both of the F-Measure scores can reach more than 0.95 with using the same training set and cross validation fold numbers. They both are powerful and have great performance on this tongue recognition problem.
Even faster than SVM, the Naïve Bayes classifier has its own disadvantages. First of all, Naïve Bayes has one presumption that all of any two features should be independent with each other. In fact, the practical problems may not meet this assumption for there are some inner relationships with the color channels in pixels. For example, a new color space was introduced: XYZ. Their values can be obtained by a linear transformation of the gamma corrected value of the RGB normalized color space \{ri, gi, bi\}. CIE LAB* color space is a nonlinear transformation of the CIE XYZ color space. Although the relationship of RGB and LAB* is not as direct as RGB and XYZ, The LAB* are indeed transferred from RGB. Furthermore, even if any two features are assumed to be independent of each other, it is not sure how strictly they meet the premise of Naïve Bayes. Although the features on Gaussian distributions can be generated, it is uncertain how much the whole dataset obey the distributions.

For the SVM method, there is no need to worry about the premise, but this method has its own disadvantages. First of all, it is much more time consuming on model training. By comparing the time cost on each method, SVM can reach as 27.5 times as Naïve Bayes does. In addition, the parameters are hard to tune. Except for the types and kernel functions, many other parameters may be under concern, such as cost and gamma. If the parameter tuning part is included, the time consuming would be rather high.

After the comparison and analysis on these two methods, the results that came from Naïve Bayes Classifier and SVM classification method were compared. Both of their results are generally good. For SVM classification, after the comparison of trying on different kernel types, linear kernel type was chosen. Although there are 24 tongue images that were identified to non-tongue group, the misclassified number of non-tongue
images was just 5. The features of tongue images may not be perfect, for they are somehow influenced by light source condition.

By the comparison and analysis performed above, SVM is preferred on this tongue recognition problem. First, although it is more time consuming, it is still within acceptable time frame. For practical usage in the app, the model could predict the input images by SVM in real-time. Furthermore, for the kernel types chosen and parameter tuning, different kernels provided in WEKA were tried and linear kernel combined with other parameters as default were chosen; the results are already good enough. Moreover, and the most important, the results SVM yielded had better performance than the result from Naïve Bayes Classifier.

Given to the limited features on tongue images, there are some objects that may have similar colors or color smoothness features with tongue images, so tongue shape could be added to improve the tongue recognition.

2.5 Tongue Shape Symmetry

2.5.1 Image Rotation and Getting Contour

Due to the fact that most of tongue images are symmetrical, an image can be judged by its symmetric degree to improve the prediction of whether it belongs to a tongue image group or non-tongue image group.

The input of this method is an image with its segmented mask, as mentioned above in 2.2. First of all, shrinking both of the image and its mask to the one-fifth sizes is
necessary, since most of the time an image imported from iPhones can reach more than 2500px on its length. Resizing the image can make the later correlation part much easier and more efficient.

Secondly, the mask image is rotated by its center point. The tongue image may be a little askew when taken by someone holding the phone, but the degree needed to be corrected would not be too much. If the degree is larger than 15, it can be detected by our eyes and the tongue image will be retaken. So a small correction is needed from negative 15 degree to positive 15 degree with the step length of 0.1 degree to find the symmetry axis during this interval. Figure 2-11 shows the positive 15 degree rotation and negative 15 degree rotations of the mask image. Before the rotation, the center point \( P_c = (x_c, y_c) \) needs to be determined. In order to keep the maximum information of this rectangle mask, the mask image is enlarged to a square shape with its larger side. Otherwise, when the image is rotated, some parts of the image, such as the corners, may be out of the image range. So \( x_c \) and \( y_c \) are one twice of the square side. Figure 2-10 shows the original image and mask image. In this image, the symmetry axis can be seen a little tilted to left.

![Figure 2-10 Original imported image and segmented mask](image)

22
Furthermore, for each rotation, the interesting region and the corresponding contour are exacted once. And for each generated interested region (ROI), the symmetry axis can be seen as a line connecting the midpoints of two sides, which are the top and bottom boundary of the interested region. So this interested region of the rotated image can be divided into a left part and right part by a symmetry axis. For the gray level binary image, each part can be reviewed as an array, and the non-zero array elements numbers are countable. By this means, the minimum difference between these two arrays can be calculated within the rotation degree range. In Figure 2-12, the left image shows the image with the minimum difference between the left and right part of the interested region during rotations, and right image shows the contour of the left image.
2.5.2 Correlation

Finally, when the image with minimum difference is available, the corresponding contour can be generated easily. The contour distances are calculated to get the distance between one contour pixel and symmetry axis. By the comparison of the left and right contour distances row by row, a correlation score is obtained, which is applied to illustrate the difference of two contours. If these two contours are in high correlation, the image can be viewed as symmetric, so it would be a tongue image in high probability. Otherwise the image would not be a tongue. Figure 2-13 shows the two parts’ correlation on the contour image. The red line means the symmetry axis on contour image. Blue points on the left contour and green points on the right contour correspond to each other of every row by the white dotted line, which is perpendicular to the symmetry axis.
For each training sample, a correlation score representing the correlation degree of left and right contour is added as a new feature of training samples. This feature restricts the images that have the similar color features with tongue images to be non-tongue images. So this correlation part is essential on tongue recognition.

Both Naïve Bayes and SVM classifier are applied on the new features that are combined by the color features, color smoothness features and the correlation feature. Results are shown below in table 2-3.

2.5.3 Result
Table 2-3 Recognition correlation on SVM Classifier results

(a) Confusion matrix

<table>
<thead>
<tr>
<th>Actual Tongue images</th>
<th>Classified as Tongue images</th>
<th>Classified as Non-tongue images</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>407</td>
<td>20</td>
</tr>
<tr>
<td>Actual Non-Tongue images</td>
<td>5</td>
<td>422</td>
</tr>
</tbody>
</table>

(b) Classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.971</td>
<td>0.029</td>
<td>0.971</td>
<td>0.971</td>
<td>0.971</td>
<td>0.942</td>
</tr>
</tbody>
</table>

Comparing the result without correlation feature in table 2-2, there are 4 tongue images that are rectified from the misclassified group. F-Measure result of recognition features including correlation is improved by 0.5%. The weighted average TPR increased from 0.966 to 0.971 and the FPR decreased from 0.034 to 0.029. Although the improvement is small, there is indeed a difference in this training set.

Feature extraction is implemented in in-house codes on Linux environment with OPENCV library configuration. Classification is implemented on Weka3.7. SVM Classifier is implemented from the third party function “Libsvm”.

2.6 Conclusion

SVM classification method and the shape symmetry recognition are both applied to do the tongue recognition. These two parts are independent. Both of them are essential
for tongue recognition. If the input images have symmetrical shape on its object segmented contour, they can be restricted by color and color smoothness features via SVM classification method. While if the imported images have the similar color features with tongue images, the shape symmetry recognition can be limited. So these two parts are combined together to judge whether the images are tongue or not.
CHAPTER 3  TONGUE CLASSIFICATION

3.1  Introduction

Based on the Traditional Chinese Medical theory, different tongue illustrates different health status. Not only does the tongue’s appearance convey valuable information, such as tongue body color and coating color, texture, shape, but also the different regions of tongue represent many organ functions of human body. For example, the tongue tip reflects the health status of lung and heart, and two sides of tongue reflect the health status of liver.

In this chapter, two SVM classifiers are applied to two training sets based on two different TCM rules. One rule is based on Hot-cold status, so the classifier is for three classes (Hot, Cold, Hot-Cold Balanced) (3.3); and the other rule is based on physical status, the classifier is for five classes (Normal, Qi-deficiency, Phlegm-Dampness, Qi-deficiency and Phlegm-Dampness, Blood-stasis) (3.5). At this point, the class Hot or Cold does not mean the temperature of body but the whole body balance status according to TCM theory.

Before the classification, preprocessing is needed on these two tongue image training sets. (3.2) Partition each tongue into five parts and combined some of them to do the color feature extraction. 10 folds cross validation is also employed with SVM classifiers in this chapter.
After the classifications, a method SMOTE is introduced in this chapter to improve the data imbalanced problem and light source correction is also employed to improve the classification results. Results and discussions are provided at the end of this chapter.

3.2 Preprocessing

After segmentation, the mask image is generated, which is a binary image. When the mask and original image are overlapped together, the segmented image is created, which is used for classification process. Before the classification, the image partition and feature extraction are necessary.

Image partition is based on the one-fifth rule which is introduced by Ratchadaporn Kanawong [5], and each tongue image can be divided into five parts. Two perpendicular red lines shown in Figure 3-1 means partition based on one-fifth of tongue width and two horizontal red lines means one-fifth of tongue length.

As shown in Figure 3-1 right, the bottom part of tongue was named as the tongue tip part (red region); the left and right parts represent tongue sides (blue regions); the top part presents tongue root part (green region); the middle part reflects tongue coating part (white region). In the coating region, color features can be extracted as coating color features of the whole tongue; and the color features on left, right and bottom parts can be combined together and defined as the tongue body color features.
3.2.1 Color Feature Extraction

For the Hot-cold status three-class classification problem, it only focuses on the tongue body color features (the bottom, left and right parts). In physical type five-class classification problem, the tongue body color features and the tongue coating color features (middle part) were extracted, since physical type also depends on tongue coating color features.

7 color spaces are applied to illustrate tongue color features. They are RGB, HIS, YCrCb, CIE Lab, XYZ, LUV and YUV [5]. For each color space, there are 3 color channels. So there are 21 color channel values, it is also called 21 dimension color features, to express each divided part of tongue images. Figure 3-2 shows one tongue image’s performances under different color spaces.
Microscopically, for each pixel on tongue divided part of a tongue image, there are 21 dimension color features. The mean value function is applied to all pixels of each partition: the left part, right part, bottom part and middle part individually.
For three-class training set, there are 3 regions (left, right and bottom) needed to do the feature extraction. For each region, there are 21 averaged color features. So for one sample image, there are 63 dimension color features in total.

For five-class training set, there are 4 regions (left, right, bottom and middle) and 84 averaged color features for each region. So for one sample image, there are 84 dimension color features in total.

### 3.2.2 Color Smoothness Feature Extraction

In order to enhance the color features, the color smoothness features are introduced to represent the color changing of each region along with different color channels. From 3.2.1, there are 21 averaged color features on each region. After that, the standard deviation of color features is easily available by regions. The dimensions are the same as color features. So for each sample image of three-class training set, there are 63 dimension color smoothness features and for each of five-class training set, there are 84 dimension color smoothness features in total.

In sum, for each tongue image with 3 regions (three-class), there are 126 dimension features including the 63 color features and 63 color smoothness features. And for each with 4 regions (five-class), there are 168 dimension features including 84 color features and 84 color smoothness features.

The color smoothness feature extraction is implemented in in-house script under Linux environment. The OPENCV libraries are also applied, which are the same as the color features extraction process in 3.2.1.
A third party classification tool “Libsvm” is applied to do the model training and 10 folds cross validation on the training set.

3.3 Three Classes Classification

3.3.1 Training Set

In this classification problem, the goal is to divide tongue images into three classes: Hot, Hot-cold Balanced and Cold. Two different training sets used for the different light source conditions determined distinguished labeling standards. Each data set is collected under its unique light source condition, however, an image in one light source condition is labeled by Hot may be labeled by Cold in another light source condition. So these two dataset cannot simply be mixed together without doing light source correction (3.3.4). One training set includes 85 images captured with iPhone 5S back camera under fluorescent lamp environment in Digital Biology Lab 110, in which 35 are in Cold, 32 in Hot-Cold Balanced and 18 in Hot. The other is 257 images captured with iphone5S back camera under the stable light source box environment in Shanghai TCM University.

The stable light source box means it can make the light source rather stable no matter what the surrounding light sources are. Figure 3-3 shows the comparison of an image taken in stable light source box (a) with one taken under normal fluorescent lamp (b). The image of 3-3 (b) can be easily influenced by the surrounding light conditions or the collar cloth color and the skin color.
Within 257 images from Shanghai, 209 are in Cold, 4 are in Hot and 44 are in Hot-Cold Balanced.

The labels are analyzed and provided by Dr. Liping Tu, associate professor at Shanghai TCM University. “1” is the label for Hot class, “0” is the label for Hot-Cold Blanced class and “-1” for Cold class. As Figure 2-1 in Chapter one shown, (a) is in Cold, (b) is in Hot-cold Balanced and (c) is in Hot.

3.3.2 SVM Classification

For the 85 images training set, SVM is applied with C-SVC polynomial kernel type and 10 folds cross validation. The results are shown in Table 3-1 and 3-2. For the images actual in Cold, there are 21 images are classified correctly; there are 10 images are misclassified to Hot-cold Balanced and 4 images to Hot. For the images actual in Hot-
cold Balanced, there are 16 classified correctly. For the images actual in Hot, there are 11 images are classified correctly. The averaged TPR is 0.565 and the FPR is 0.247. The SVM classifier performance on this data set looks not good enough, since one of the reasons is that the training set is relatively small and may not be representative for the generality and universality. Even if the samples are close enough to the future system application data that would be taken by cell phone camera from customers, this training set is still not acceptable.

### Table 3-1 85 images training set SVM classification confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in Cold</td>
<td>21</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>Actual in Hot-cold Balanced</td>
<td>12</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>Actual in Hot</td>
<td>2</td>
<td>5</td>
<td>11</td>
</tr>
</tbody>
</table>

### Table 3-2 85 images training set SVM classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.565</td>
<td>0.247</td>
<td>0.564</td>
<td>0.565</td>
<td>0.564</td>
<td>0.316</td>
</tr>
</tbody>
</table>

In order to make the training set more general, another training set is introduced, a relative larger scale, 257 images training set. While there is another problem on this data set, since it is imbalanced (209 - Cold, 44 - Hot-cold Balanced and 4 – Hot). SVM classifier is applied with C-SVC polynomial kernel type and cross validation is 10 folds.
Although the averaged TPR 0.743 looks good as shown in Table 3-3, the FPR is as high as 0.471. So the result is unacceptable with this data imbalanced problem unsolved.

### Table 3-3 257 images training set SVM classification confusion matrix

<table>
<thead>
<tr>
<th>Actual in Cold</th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>172</td>
<td>36</td>
<td>1</td>
</tr>
<tr>
<td>Actual in Hot-cold Balanced</td>
<td>24</td>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>Actual in Hot</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 3-4 257 images training set SVM classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.743</td>
<td>0.471</td>
<td>0.764</td>
<td>0.743</td>
<td>0.752</td>
<td>0.251</td>
</tr>
</tbody>
</table>

#### 3.3.3 SMOTE

Randomly undersampling or oversampling is a typical strategy to cope with imbalanced training set [6]. Undersampling could reduce the size of majority samples to balance the whole training set, but the eliminated samples would cause information lost. While the oversampling strategy increases the size of minority samples by obtaining basic information and copying, it may cause training set redundancy and easily get over fitting problems. Here one method called Synthetic Minority Oversampling Technique (SMOTE) is applied [7]. Figure 3-4 shows how function SMOTE works. (a) In minority
samples (star group), SMOTE finds every sample’s K nearest neighbors by calculating the Euclidean distance. (b) A new sample is created synthetically according their neighbor’s information. This process could go over and over again to increase the size of minority samples.

257 images training set is applied on WEKA software to be balanced with SMOTE. After 7 times iteration, the new training set tends to be balanced with 209 in Cold, 176 in Hot-Cold Balanced and 128 in Hot. Table 3-5 and 3-6 show the results of this new training set. The weighted average TPR is 0.844 and the FPR is 0.092. This results are much better than the ones without SMOTE applied.
Table 3-5 SMOTE application on 257 images training set confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in Cold</td>
<td>170</td>
<td>35</td>
<td>4</td>
</tr>
<tr>
<td>Actual in Hot-cold Balanced</td>
<td>35</td>
<td>140</td>
<td>1</td>
</tr>
<tr>
<td>Actual in Hot</td>
<td>3</td>
<td>2</td>
<td>123</td>
</tr>
</tbody>
</table>

Table 3-6 SMOTE application training set classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.844</td>
<td>0.092</td>
<td>0.844</td>
<td>0.844</td>
<td>0.844</td>
<td>0.752</td>
</tr>
</tbody>
</table>

In order to make full use of these two training sets, the light source condition problems are inevitably to be solved. Because simply mixing images from different light environment together would make the model unstable and unbelievable. For the purpose of measuring the performances of light source correction, the results of mixing images together without applying SMOTE and light source correction were generated as a comparison. In the whole training set, there are 246 images in Cold, 70 images in Hot-cold Balanced and 26 images in Hot. SVM classifier is employed with polynomial kernel type. Results are shown in Table 3-7 and 3-8, and the accuracy can only reach 65% and weighted average FPR is 0.392. So in next section, a method is employed to do the light source correction for different light source condition systems.
### Table 3-7 Combined 85 images training set with 257 images training set confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actual in Cold</strong></td>
<td>188</td>
<td>52</td>
<td>6</td>
</tr>
<tr>
<td><strong>Actual in Hot-cold Balanced</strong></td>
<td>39</td>
<td>23</td>
<td>8</td>
</tr>
<tr>
<td><strong>Actual in Hot</strong></td>
<td>7</td>
<td>6</td>
<td>13</td>
</tr>
</tbody>
</table>

### Table 3-8 Combined 85 images training set with 257 images training set classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.655</td>
<td>0.392</td>
<td>0.673</td>
<td>0.655</td>
<td>0.663</td>
<td>0.255</td>
</tr>
</tbody>
</table>

#### 3.3.4 Light Source Correction

In order to change the light source from one to another, the X-rite Color Checker is used which is a professional tool for photographers and filmmakers, shown in Figure 3-5. The Color Checker has 24 color squares in standard value on the card which provides a quantitative and reliable reference for color checking in any light environment. Photos under two light source conditions were taken and the MATLAB Imatest 4.3 MCR v9.0 is applied to extract the features on RGB color space, which includes the standard values of the color checker and the practical values of input photos taken under different light environments.
In order to check the data relationship quality, Pearson’s Correlation \( (P) \), a measure of the strength of the association between the two variables, is applied on each group of data. These results demonstrate that the input values and the standard values are in high correlation relationships, which are shown in Table 3-9.

<table>
<thead>
<tr>
<th>Correlation</th>
<th>Shanghai 257 images training set</th>
<th>Iphone5S 85 images training set</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.9645</td>
<td>0.9946</td>
</tr>
<tr>
<td>G</td>
<td>0.9807</td>
<td>0.9834</td>
</tr>
<tr>
<td>R</td>
<td>0.9819</td>
<td>0.9773</td>
</tr>
</tbody>
</table>

After data quality checking, the fitting functions between the standard values and the input practical values are calculated. Figure 3-10 and 3-11 (a), (b) and (c) show the
fitting results. X-axis represents the standard color values and the y-axis expresses the input color values.

Figure 3-6 Iphone5S 85 images training set fitting results on Standard Color Values and Input Color Values

Figure 3-7 Shanghai 257 images training set fitting results on Standard Color Values and Input Color Values

After being fitted by the linear functions, the data of two training sets were transformed to the standardized data in RGB color space respectively. Then the images were generated, one of which is shown in Figure 3-8 as an example. After creating the new images, the feature extraction process was repeated to get the new features from 21 color channels on these two training sets. As new features are extracted, these two sets can be combined together as one (244 in Cold, 70 in Hot-cold Balanced and 26 in Hot). 10 folds cross validation and SVM classifier with C-SVC polynomial kernel type are applied. Results are shown in Table 3-10 and 3-11 and the accuracy is 71%. When
SMOTE function is applied to this corrected combined training set, there are 244 samples in Cold, 152 in Hot-cold Balanced and 176 in Hot. A better result can be generated, which is shown in Table 3-12 and 3-13. The weighted average TPR is 0.773 and FPR is 0.133.

![Original image and corrected image example](image)

**Figure 3-8 Original image and corrected image example**

**Table 3-10 Corrected combined training set confusion matrix**

<table>
<thead>
<tr>
<th>Actual in Cold</th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in Cold</td>
<td>203</td>
<td>34</td>
<td>7</td>
</tr>
<tr>
<td>Actual in Hot-cold Balanced</td>
<td>36</td>
<td>31</td>
<td>9</td>
</tr>
<tr>
<td>Actual in Hot</td>
<td>8</td>
<td>5</td>
<td>9</td>
</tr>
</tbody>
</table>

**Table 3-11 Corrected combined training set classification result**

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.711</td>
<td>0.356</td>
<td>0.708</td>
<td>0.711</td>
<td>0.709</td>
<td>0.357</td>
</tr>
</tbody>
</table>
Table 3-12 SMOTE corrected combined training set confusion matrix

<table>
<thead>
<tr>
<th>Classified as</th>
<th>Classified as Cold</th>
<th>Classified as Hot-cold Balanced</th>
<th>Classified as Hot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in Cold</td>
<td>202</td>
<td>33</td>
<td>9</td>
</tr>
<tr>
<td>Actual in Hot-cold Balanced</td>
<td>64</td>
<td>80</td>
<td>8</td>
</tr>
<tr>
<td>Actual in Hot</td>
<td>9</td>
<td>7</td>
<td>160</td>
</tr>
</tbody>
</table>

Table 3-13 SMOTE corrected combined training set classification results

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.773</td>
<td>0.133</td>
<td>0.769</td>
<td>0.773</td>
<td>0.767</td>
<td>0.646</td>
</tr>
</tbody>
</table>

3.4 Discussion

From the results in Table 3-3 and 3-4, the 257 images training set did not perform well. Although the accuracy is 74%, if the class Cold is removed and only focus on the other two, the accuracy is just 40%.

By the comparison with Table 3-5 and 3-6, SMOTE could reduce the influences caused by data imbalanced problem and improve the performance of the training set to some extent.

When two data sets are combined to one, the light source correction was performed first. The result accuracy is improved from 65% (Table 3-7 and 3-8) to 71%
(Table 3-10 and 3-11). Although the improvement is limited, just 6%, light source correction is necessary for two different environment training sets combination. It makes the model trained more powerful and believable. Then the SMOTE was performed secondly on the combined corrected data set. The result accuracy is improved from 71% to 77% (Table 3-12 and 3-13).

Although the results are not perfect and there may be some other issues existing, such as the data quality, currently they are acceptable for the 3-class classification in the system training.

3.5 Five Classes Classification

3.5.1 Training Set

In this classification problem, the goal is to divide tongue images into five groups based on TCM physical rule: Normal, Qi-deficiency, Qi-deficiency and Phlegm dampness, Phlegm dampness and Blood stasis.

For the five classes training set, there are 221 images captured with iPhone 5S back camera under natural light source environment in Shanghai TCM University.

The labels are also analyzed and provided from Dr. Liping Tu. “1” is the label for Normal class, “2” is the label for Qi-deficiency class, “3” for Qi-deficiency and Phlegm dampness class, “4” for Phlegm dampness class and “5” for Blood stasis class. Figure 3-9 shows the tongue images in different classes.
In this training set, there are 38 samples on Normal class, 60 samples on Qi-deficiency class, 42 samples on Qi-deficiency and Phlegm dampness class, 61 samples on Phlegm dampness class and 20 samples on Blood stasis class.

3.5.2 Algorithms

For five classes feature extraction, not only is the tongue body color features (left, right and bottom part) applied, but also the tongue coating color features (middle part) is considered. The features employed are 21 color channels on color features and color smoothness features, which are the same as three-class classification in 3.3. Thus, there are 42 dimension features on each part (21-color features and 21-color smoothness features) and 168 dimension features on each training image (4 parts in total).

10 folds cross validation and SVM classifier with C-SVC polynomial kernel type are employed. Results are shown below in Table 3-14. For five classes, the weighted average TPR is 0.412 and FPR is 0.173. The accuracy is as low as 41 %.
There may be two main issues that cause the low accuracy. One is related to the SVM classifier: SVM doesn’t perform well on multiple classes problems. Another is related to this training set: the number of each class is small and the data set is a little imbalanced.

In order to solve the first issue, the previous one classifier of five classes is transformed to five classifiers of two classes. Thus, there are five classifiers in total. For
each classifier, it predicts an input sample whether it belongs to the class (like Qi-deficiency) or not.

Given to the imbalanced training set on each classifier, SMOTE is also applied to preprocess the data before classification. Comparison results are shown in Tables below.

3.5.3 Result

<table>
<thead>
<tr>
<th>Table 3-15 Normal class classifier original and SMOTE results</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Confusion matrix</td>
</tr>
<tr>
<td>Actual in Normal</td>
</tr>
<tr>
<td>Actual in Normal</td>
</tr>
<tr>
<td>Actual in Non-Normal</td>
</tr>
<tr>
<td>(b) SMOTE confusion matrix</td>
</tr>
<tr>
<td>Actual in Normal</td>
</tr>
<tr>
<td>Actual in Normal</td>
</tr>
<tr>
<td>Actual in Non-Normal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(c) Classification results</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP Rate</td>
</tr>
<tr>
<td>0.787</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(d) SMOTE Classification results</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP Rate</td>
</tr>
<tr>
<td>0.842</td>
</tr>
</tbody>
</table>
Table 3-16 Qi-deficiency class classifier original and SMOTE results

(a) Confusion matrix

\[
\begin{array}{c|cc}
\text{Actual as QD} & \text{Classified as QD} & \text{Classified as Non-QD} \\
\hline
145 & 16 \\
45 & 15 \\
\end{array}
\]

(b) SMOTE confusion matrix

\[
\begin{array}{c|cc}
\text{Actual as QD} & \text{Classified as QD} & \text{Classified as QD} \\
\hline
126 & 35 \\
35 & 85 \\
\end{array}
\]

(c) Classification results

\[
\begin{array}{cccccc}
\text{TP Rate} & \text{FP Rate} & \text{Precision} & \text{Recall} & \text{F-Measure} & \text{MCC} \\
0.724 & 0.573 & 0.687 & 0.724 & 0.691 & 0.193 \\
\end{array}
\]

(d) SMOTE classification results

\[
\begin{array}{cccccc}
\text{TP Rate} & \text{FP Rate} & \text{Precision} & \text{Recall} & \text{F-Measure} & \text{MCC} \\
0.751 & 0.260 & 0.751 & 0.751 & 0.751 & 0.491 \\
\end{array}
\]

Table 3-17 Phlegm-dampness class classifier original and SMOTE results

(a) Confusion matrix

\[
\begin{array}{c|cc}
\text{Actual in PD} & \text{Classified as PD} & \text{Classified as Non-PD} \\
\hline
153 & 26 \\
27 & 15 \\
\end{array}
\]

(b) SMOTE confusion matrix

\[
\begin{array}{c|cc}
\text{Actual in PD} & \text{Classified as PD} & \text{Classified as Non-PD} \\
\hline
155 & 24 \\
30 & 138 \\
\end{array}
\]
Table 3-18 Qi-deficiency and Phlegm-dampness class classifier original and SMOTE results

(a) Confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as QD&amp;PD</th>
<th>Classified as Non-QD&amp;PD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in QD&amp;PD</td>
<td>125</td>
<td>35</td>
</tr>
<tr>
<td>Actual in Non-QD&amp;PD</td>
<td>38</td>
<td>23</td>
</tr>
</tbody>
</table>

(b) SMOTE confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as QD&amp;PD</th>
<th>Classified as Non-QD&amp;PD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in QD&amp;PD</td>
<td>117</td>
<td>43</td>
</tr>
<tr>
<td>Actual in Non-QD&amp;PD</td>
<td>37</td>
<td>85</td>
</tr>
</tbody>
</table>

(c) Classification results

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>0.760</td>
<td>0.548</td>
<td>0.758</td>
<td>0.760</td>
<td>0.759</td>
<td>0.214</td>
</tr>
<tr>
<td>SMOTE</td>
<td>0.844</td>
<td>0.157</td>
<td>0.845</td>
<td>0.844</td>
<td>0.844</td>
<td>0.689</td>
</tr>
</tbody>
</table>
### Table 3-19 Blood-stasis class classifier original and SMOTE results

#### (a) Confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as BS</th>
<th>Classified as Non-BS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in BS</td>
<td>179</td>
<td>22</td>
</tr>
<tr>
<td>Actual in Non-BS</td>
<td>13</td>
<td>7</td>
</tr>
</tbody>
</table>

#### (b) SMOTE confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>Classified as BS</th>
<th>Classified as Non-BS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual in BS</td>
<td>183</td>
<td>18</td>
</tr>
<tr>
<td>Actual in Non-BS</td>
<td>16</td>
<td>144</td>
</tr>
</tbody>
</table>

#### (c) Classification results

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.842</td>
<td>0.601</td>
<td>0.870</td>
<td>0.842</td>
<td>0.854</td>
<td>0.204</td>
</tr>
</tbody>
</table>

#### (d) SMOTE classification results

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.906</td>
<td>0.095</td>
<td>0.906</td>
<td>0.906</td>
<td>0.906</td>
<td>0.809</td>
</tr>
</tbody>
</table>

### 3.5.4 Discussion
Comparing with the one-against all strategy, this one against one strategy with SMOTE could improve the prediction power to some extent. From the results in Table 3-14 to 3-19, the weighted average TPR is just improved 3%-12%, but FPR is significantly decreased by 30%-50%. Although the results are not perfect enough, they are acceptable as a function in the whole health system.

3.6 Conclusion

Imbalanced data is a normal problem in machine learning area. In this chapter, the classical SMOTE method is applied to cope with this problem, and obviously the results are better.

Due to the performance of SVM is outstanding on two-class classification problem, one classifier of five classes is transformed to five classifiers and each of them is a two-class classification problem. This made the SVM classifier model learning process much easier and the results are much better.

The number of each class in the training set is only tens of tongue images. If the number of samples is large enough, like hundreds of images, more application methods and analysis can be tried to improve the accuracy.
CHAPTER 4  TONGUE ANALYSIS SYSTEM

4.1 Introduction

Based on the proposed work on tongue image processing, a tongue health monitor app has been developed, iTongue. It is a user friendly app which could run on both IOS and Android smart phone systems.

In this app, the tongue image taking, tongue segmentation, recognition and classification are all carefully designed and operated automatically one by one. Furthermore, an optional questionnaire is provided, which would help tongue analysis more correctly. The customer can not only get the tongue labels of Hot-cold status and physical status, but also the recommendation food for reference.

In addition, an advance portal is provided to connect customers to the real TCM doctors. This advance feature would make more people focus on their body status conveniently and efficiently. When the doctor finished the analysis, there would be a report sent as the attachment to the customer’s email address.

Also, register and login system, analysis records and physical fitness test and so on.

For all functions of this customer version app, MYSQL is applied as the database; the HTML and JavaScript are employed to implement the front side and PHP on the back side [8]. The interaction of database and front side can be implemented by PHP.
All of the functions are implemented in HBuilder software and test with Iphone5s on ios 7.0 and ZTE3.0 on android 4.4.

4.2 Tongue Scan Function

Although it is not necessary to login or register when this function is used, customer information like weight, height, age and gender is important for the analysis, especially for the real doctors’ analysis. So a notification is designed to show on the screen to reminder the users, it is shown in Figure 4-1.

*Figure 4-1 Tongue Scan function screenshot*
If the user wants to register, clicking the link would take him to the register page. While if the user just wants to have a simple try, clicking “Later” and the functions are ready to use.

The app provides two ways for users to upload their tongue images. One is to choose the existed images from cell phone’s camera roll; the other is to call the camera function of phone to take one. The tongue image information would be sent to the server side in either way.

On server side, the app will check whether the image can go through the segmentation and the recognition functions. When a user uploaded the tongue image and got a notification to say “Processing failed”, like the Figure 4-2 shown below, the user can click the link in the last line and find the reason why the tongue image is not passed to the server side and then retake or re-choose image. There are several reasons of processing failed: there may not be a tongue included in this uploading image, or there is some problem with the light source, too strong or too weak and so on.
When the image can go through the segmentation and the recognition functions, the app will take the user to the optional questionnaire page. There are 6 questions in this section, users can skip all the questions and the classification functions would follow on the server side to generate analysis results. Figure 4-3 shows the questionnaire and result screenshots.
Actually, there are some logic relationships between the tongue image classification result and the questionnaire answers. Based on the recommendations of Dr. Jiatuo Xu, different weights are assigned to both parts, which are then combined together to represent the final result, which is shown on the result page in Figure 4-3. If there is some conflict between them, the questionnaire answers are chosen as the primary results.

Take the above result page as an example, there are two labels included: Cold and Normal. If the users want to understand more on these two statuses, he can click the words “Hot-Cold Status” and “Physical Type” to see the brief explanations, shown in Figure 4-4.
Based on different labels user got, the app provides some food recommendations. For Cold status, users are suggested to eat food in warm characters to warm the body up and oppositely, users in Hot are recommended with food in cold characters to cool down the body.

4.3 Asking Doctor Function

Users can get connection with the real doctor from the result page in an advance portal shown in Figure 4-3 above or from the main dashboard “Ask a Doctor” function.

This function requires users to login and use their own accounts, as the analysis report would be back to the email account registered. Except for the account email, the name, age and gender information is also necessary. Figure 4-5 shows the details.
In order to make sure the tongue images sent are qualified for doctors’ analysis, users can choose as maximum as four images to send. When clicking record rows, the result lists with details are shown. Leaving descriptions or messages is available for users, which would be wrapped together with the tongue images and sent to doctors. Furthermore, the app provides two report languages to choose, English and Chinese.

In the checkout process, a third part checkout system named “stripe” is employed, which would verify the card information of the customers. If the transaction can be finished successfully, stripe would send a long string token into database. Then a notification would be pushed back to screen. Figure 4-6 is shown below. In the whole
process, the app won’t access any confidential card details of users. When successful, the app sends two emails, one is to the doctor, and the other is to the user.

For the users, it sends the order confirmation, which includes the compliments and order details.

For the doctors, it sends the customer’s information. At this point, the app sends the user ID instead of the user’s name. It also sends age, gender, tongue images, description, the preference of report language and the analysis link. The doctor can do the tongue analysis by clicking the link. Just some clicks make it simple, convenient and efficient for the doctors to finish the analysis process. Snapshot is shown in Figure 4-7. The analysis page provides single and multiple choices for doctors. For example, if the user is in both Qi-deficiency and Blood-stasis and the food suitable for them are not the...
combined food recommendation, doctor can get rid of the inappropriate items depending on individual particularity or difference. So does the sports choices.

Figure 4-7 Doctor Analysis System webpage screenshot
After doctor’s analysis, the data is saved into database and the report is ready to be generated, as a PDF file, to be reviewed for doctors who choose to send it or re-evaluate the tongue images. Finally, the report would be sent to user’s registered email address as an attachment.

4.4 Login and Register Functions

The register and login system is designed including forgotten password requests. It is shown in Figure 4-8. The app reminds users to register during the navigation on the app several times.

![Register and Login screenshots](image)
For the first time users, the app gives a status on the main dashboard to say “Login required” to remind the users. By clicking the words, users will be guided into the login and register pages. If the user does not want to log in at this time, the app sends a notification when he tried to use the tongue scan function. Such as Figure 4-1, the tongue scan function screenshot shows.

When the users choose “Later”, the app sends his cell phone Universally Unique Identifier, short for UUID, to the database and creates a new customer record. Usually in the login and register tables, email address is as the primary key. Later when he registered, the app will replace his phone UUID with the email address. The reason why it is designed in this way is that after the user’s registration, all the records are kept and ready to browse and send to doctors, even for the pre-registration records. Although the user can use the tongue scan function without any difficulties, he will get the same notification reminds once he using this function.

When the user chooses “Register Now”, the user is guided to the register page, shown in Figure 4-8 above.

If the password is forgotten, the user can click the blue link under register button in Figure 4-8 above. On the password saving process, hash function is applied and salts are added on the hashed password. Once the app receives the password reset request, it will generate a new 8 digit password randomly; replacing the old one with this new one in the database and send this new password to user’s email address. When the user uses this password to login, he can change it to his own one in the “Account Info” page.
Details are shown in Figure 4-9. In addition on the account info page, users can change his personal information like weight, height, and others.

![Image](image_url)

*Figure 4-9 Password forgotten screenshots*

### 4.5 Other functions

On the setting page, the app gives a brief introduction, valuable significance and key authors on this product in “About”. Furthermore, a suggestion function in the app is provided for users to contact the team by leaving their email address. Additionally, writing a review or rating the app is available on “Review” page. Figure 4-10 shows the details. English is the only language currently available. Simplified and Traditional Chinese language options are under development.
Under the “Resources” link on main dashboard, the app provides a tutorial of the app, videos and articles related with TCM, and the standard physical fitness test.

The tutorial tells users how to use this app comprehensively and meticulously.

The app provides some articles related to tongue common sense, relationships between disposition and body status and some other enlightening topics of TCM. Some of them are originals from the cooperating doctors, the others are reproduced by the editors from the open websites.

The videos are published on video websites: Youtube and Youku, and the players are embedded on the app. The contents are related to the introduction of TCM. These videos are presented by Dr. Yanzhu Hong, TCM professor at Xiamen University. Although the knowledge of TCM is complicated and difficult to understand, the lectures given by Dr. Hong are concise, accurate, vivid, and interesting.
Physical fitness test is a function providing 67 classical TCM questions, which can be divided into 9 groups. They include Normal, Qi-deficiency, Yin-deficiency, Yang-deficiency, Phlegm-dampness, Dampness-heat, Blood-stasis, Qi-stagnation and Special- Constitution type. Each group represents one physical body type, which contains 7-9 questions. Based on different body feeling, 5 level answers are provided. Each answer has its own weight. Finally, depending on the weight of each group and each answer, the app gives the labels of user’s physical body type. The user can pause the test at any time. A notification pushed on screen to ask user whether he wants to continue from the breakpoint last time or start over from the beginning, as this questionnaire may cost some time to answer. Figure 4-11 shows the details of resources page.

Figure 4-11 Resources webpage contents screenshots

4.6 Conclusion
This app is concise, powerful and easy to navigate. First of all, it responds fast on all functions. For the tongue scan function, it spends approximately 10 seconds from uploading tongue image to getting the analysis labels. Because the segmentation algorithm consumes about 1 to 3 seconds, recognition result for 6-8 seconds and the classification result for 1-2 seconds. Furthermore, it is designed intelligently. It not only provides the users some lifestyle reference but also a way to connect the users to the real doctors. For the questionnaire, customers can stop any time and come back to the break point next time. Finally, the app is user friendly. It sends the notifications to remind customers their non-registered status and keep all the records for each non-registered user. The customers can leave message and rate the app if they have some good suggestions or they are facing some difficulties.
CHAPTER 5 DISCUSSION AND CONCLUSION

The methods applied include traditional statistic methods and classical machine learning methods. These two methods were compared in chapter 2 on the tongue recognition algorithms. SVM is applied on chapter 3, three class and five class tongue classification algorithms.

The training sets of the algorithms are not perfect currently. They are imbalanced and images are taken under different light source conditions. These issues were solved by the SMOTE preprocessing and the light source correction algorithms featured in chapter 3.

The app is designed and implemented based on the tongue images researches, which established the significance of TCM applications on tongue. This app can be installed on both IOS and Android platform. Users can enjoy the benefits by the app on their smart phones.

Main functions are illustrated in chapter 4 and they are easy to navigate for users. Tongue scan function includes tongue recognition and classification functions. Although there is room for algorithms to be optimized, the current time consuming can be accepted. Users can either choose tongue scan function to track the body changes or choose to send tongue images to the real doctors. The former one is quick and convenient, since tongue scan function is not restricted by the time and location. The latter one is subject to the
doctors, but the results are more personal and specific. The users can choose depending on their own requirements.

This app has been tested frequently, so as to avoid confusing customers regarding the app’s functions. Some instructions are added on the words link. Although some users are interested in the TCM tongue analysis but they may need more information to understand the TCM concepts better. How the tongue is related to our body status, and how it determines the dispositions, there are some brief explanations given. Some videos are provided to illustrate the TCM theories in understandable lectures for ordinary people. Some articles are provided periodically to spread the knowledge on relationships between tongue and organs.
REFERENCE


APPENDIX A – SOURCE CODE FOR FEATURE EXTRACTION

#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <math.h>
#include "types_c.h"
#include <stdio.h>
#include <ctime>
#include <string.h>
#include <vector>
#include <numeric>
#include <highgui.h>
#include <fstream>
#include <opencv2/opencv.hpp>
#include <stdlib.h>
#include <string>
#include <time.h>
#include <sstream>
using namespace cv;
using namespace std;

int main()
{
    //init environment
    string labelFileName="/u1/home/matlabUSR/mznrb/eclipse/workspace/TongueRecognition/label_test257_nonTongue.txt";
    string tongueImagePath="/u1/home/matlabUSR/mznrb/eclipse/workspace/TongueRecognition/Non-tongue257/";
    string maskPath="/u1/home/matlabUSR/mznrb/eclipse/workspace/TongueRecognition/Non-tongue257/";
    int totalFigureNum=257;
    int FeaturesNum=12;
    vector<vector<int> > Tongue;
    //init TongueBody features vectors
    for(int i=0;i<FeaturesNum;i++)
    {
        vector<int> tmpVec;
        tmpVec.clear();
        tmpVec.push_back(0);
        Tongue.push_back(tmpVec);
    }
    //read label from file
    ifstream file;
    file.open(labelFileName.c_str(),ios::in);
    if(!file)
    {
        cout<<"file not founded"<<endl;
        int label[1000];
        int pos = 0;
        while(!file.eof()) //check file end
        {
            file>>label[pos];
            pos++;
            if(pos>=1000)
                break;
        }
        file.close();
    }
    //get images
    string tongueImageFileName;
    string maskFileName;
    string imageIndex;
    string maskIndex;
    char filename[10000];
    for(int p=0;p<totalFigureNum;p++)
    {
        cout<<"number "<<p+1<<endl;
        //tongue image
        stringstream outimage;
        outimage << p+1;
        imageIndex = outimage.str();
        tongueImageFileName = tongueImagePath + "NonTonImg" + imageIndex +".jpg";
        sprintf(filename,tongueImageFileName.c_str(),p);
        IplImage* itimage=cvLoadImage(filename,1);
        //mask
        stringstream outmask;
        outmask << p+1;
        maskIndex = outmask.str();
        maskFileName = maskPath+"NonTonImg" + maskIndex +"_mask.ppm";
        sprintf(filename,maskFileName.c_str(),p);
        IplImage* imimage=cvLoadImage(filename,1);
        Mat timage=cv::Mat(itimage);
        Mat mimage=cv::Mat(imimage);
        if (!timage.data)                              // Check for invalid input
        {
            std::cout << "Could not open or find the image" << std::endl;
            return -1;
        }
        if (!mimage.data)                              // Check for invalid input
        {
            std::cout << "Could not open or find the mask" << std::endl;
            return -1;
        }
        //normalization
        //calc features
        //save result
    }
}
int trows = timage.rows;
int tcols = timage.cols;
cv::Size t = timage.size();
trows = t.height;
tcols = t.width;
std::cout << trows << tcols << endl;

Mat mimage1 = mimage(Range(0, trows), Range(0, tcols));
IplImage *im1 = cvCloneImage(&IplImage(mimage1));
int m1rows = mimage1.rows;
int m1cols = mimage1.cols;
cv::Size m1 = mimage1.size();
m1rows = m1.height;
m1cols = m1.width;
std::cout << m1rows << m1cols << endl;

//get nimage (mask dot product tongue image)
Mat mimage3 = mimage1 / 255;
IplImage *mi3 = cvCloneImage(&IplImage(mimage3));
Mat nimage = timage.clone();
IplImage *ni = cvCloneImage(&IplImage(nimage));
cv::Mul(timage, mi3, ni);
nimage = cv::Mat(ni);

//Recognition features extraction
vector<vector<int>> TongueBodyFeatures;
for (int i = 0; i < (FeaturesNum / 2); i++) {
    vector<int> tmpVec;
    tmpVec.clear();
    tmpVec.push_back(0);
    TongueBodyFeatures.push_back(tmpVec);
}

Mat nlab;
cvtColor(nimage, nlab, CV_BGR2Lab);
IplImage *lab = cvCloneImage(&IplImage(nlab);
int im1b, im1g, im1r;
int Bval, Gval, Rval, Lval, Aval, BBval;
for (int i = 0; i < m1rows; i++) {
    for (int j = 0; j < m1cols; j++) {
        im1b = ((uchar*)ni->imageData + ni->widthStep * i)[j * ni->nChannels];
        im1g = ((uchar*)ni->imageData + ni->widthStep * i)[j * ni->nChannels + 1];
        im1r = ((uchar*)ni->imageData + ni->widthStep * i)[j * ni->nChannels + 2];
        if (im1r > 200 && im1g > 200 && im1b > 200) {
            // R G B
            B = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels];
            G = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels + 1];
            R = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels + 2];
            L = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels];
            A = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels + 1];
            BB = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels + 2];
            Bval = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels];
            Gval = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels + 1];
            Rval = ((uchar*)(ni->imageData + ni->widthStep * i))[j * ni->nChannels + 2];
            Lval = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels];
            Aval = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels + 1];
            BBval = ((uchar*)(lab->imageData + lab->widthStep * i))[j * lab->nChannels + 2];
            TongueBodyFeatures[0].insert(TongueBodyFeatures[0].end(), Rval);
            TongueBodyFeatures[1].insert(TongueBodyFeatures[1].end(), Gval);
            TongueBodyFeatures[2].insert(TongueBodyFeatures[2].end(), Rval);
            TongueBodyFeatures[3].insert(TongueBodyFeatures[3].end(), Lval);
            TongueBodyFeatures[4].insert(TongueBodyFeatures[4].end(), Aval);
            TongueBodyFeatures[5].insert(TongueBodyFeatures[5].end(), BBval);
        }
    }
}

//mean values
vector<double> aver_vec;
for (int i = 0; i < (FeaturesNum / 2); i++) {
    aver_vec.push_back(accumulate(TongueBodyFeatures[i].begin(), TongueBodyFeatures[i].end(), 0.0) / TongueBodyFeatures[i].size());
}

//standard deviation values
vector<double> std_vec;
for (int i = 0; i < (FeaturesNum / 2); i++) {
    vector<double> diff(TongueBodyFeatures[i].size());
    std::transform(TongueBodyFeatures[i].begin(), TongueBodyFeatures[i].end(), diff.begin(), std::bind2nd(std::minus<double>(), aver_vec[i]));
    std_vec.push_back(sqrt(std::inner_product(diff.begin(), diff.end(), diff.begin(), 0.0) / TongueBodyFeatures[i].size()));
}

}
TongueBodyFeatures.clear();
vector<vector<int> >=(TongueBodyFeatures).swap(TongueBodyFeatures);
for(int i=0;i<(FeaturesNum/2);i++){
    Tongue[i*2].insert(Tongue[i*2].end(),aver_vec[i]);
    Tongue[i*2+1].insert(Tongue[i*2+1].end(),std_vec[i]);
} aver_vec.clear();
std_vec.clear();
//release memory
mimage1.release();nimage.release();
lab.release();timage.release();
mimage.release();mimage3.release();
cvReleaseImage(&imimage);
vector<double>(aver_vec).swap(aver_vec);
vector<double>(std_vec).swap(std_vec);
imageIndex.clear();
maskIndex.clear();
tongueImageFileName.clear();
maskFileName.clear();
memset(filename, 0, 10000); } //end images
tonguePath.~string();
maskPath.~string();
for(int i=0;i<FeaturesNum;i++){}
Tongue[i].erase(Tongue[i].begin(),Tongue[i].begin()+n()+1);
}
ofstream ocout;
string str1,str2,str3;
string outStr;
//mean values combined with standard deviation values
ocout.open("Recognition_feature_257_nonTongue.txt");
for(int i=0;i<Tongue[0].size();i++){
    //get label value
    std::stringstream out1;
    out1 << label[i];
    str1 = out1.str();
    outStr=str1 + " ";
    //get feature values
    for(int j=0;j<(FeaturesNum/2);j++)
    {
        //get feature values
        std::stringstream out2,out3;
        out2 << j+1;
        str2 = out2.str();
        out3 << Tongue[2*j][i];
        str3 = out3.str();
        outStr=outStr + str2 + ":" + str3 + ";
    }
    for(int j=0;j<(FeaturesNum/2);j++)
    {
        //get feature values
        std::stringstream out2,out3;
        out2 << j+1+(FeaturesNum/2);
        str2 = out2.str();
        out3 << Tongue[2*j+1][i];
        str3 = out3.str();
        outStr=outStr + str2 + ":" + str3 + ";
    }
    ocout<<outStr<<endl;
}
ocout.close();
outStr.clear();
str1.clear();
str2.clear();
str3.clear();
cout<<"Done!"<<endl;
return 1;
APPENDIX B – SOURCE CODE FOR RECOGNITION CORRELATION

/* RecognitionCorrelationFeatureExtraction.cpp
* Created on: Sep 26, 2015
* Author: Meng Zhang
*/

#include <iostream>
#include <fstream>
#include <sstream>
#include <time.h>
#include <stdio.h>
#include <stdio.h>
#include <iostream>
#include "opencv2/core/core.hpp"
#include "opencv2/features2d/features2d.hpp"
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/nonfree/nonfree.hpp"
#include "opencv2/calib3d/calib3d.hpp"
#include "opencv2/imgproc/imgproc.hpp"

#ifndef _CRT_SECURE_NO_WARNINGS
#define _CRT_SECURE_NO_WARNINGS
#endif

using namespace cv;
using namespace std;

// rotation function
void rotate(cv::Mat& src, double angle, cv::Mat& dst)
{
    int len = std::max(src.cols, src.rows);
    cv::Point2f pt(len/2., len/2.);
    cv::Mat r = cv::getRotationMatrix2D(pt, angle, 1.0);
    cv::warpAffine(src, dst, r, cv::Size(len, len));
}

double different(Mat oriMask) {
    Mat gray;
    Mat oriMaskGray;
    int minNum = 10000000000;
    cv::Mat oriImage;
    vector<vector<Point> > contours;
    vector<Vec4i> hierarchy;
    cv::Mat cropContour;
    cv::Mat minGray;
    // rotation degree i from -15 degree to 15 degree
    for(double i = -15; i <= 15; i = i+0.1 )
    {  
        cv::Mat tmp;
        rotate(oriMask, i, tmp);
        cv::Mat maskTmp( 1000, 1000, oriMask.type(), tmp.at<uchar>(0, 0) );
        Mat dst_roi = maskTmp(cv::Rect(150, 150, tmp.cols, tmp.rows));
        dst_roi.copyTo(tmp);
        tmp = maskTmp;
        cvtColor(tmp, gray,

    return minNum;
}

int main()
{  
    cv::Rect ROI(0, 0, 1000, 1000);
    cv::Mat image;
    cv::cvtColor(image, gray, CV_BGR2GRAY);
    cv::cvtColor(oriImage, oriMaskGray, CV_BGR2GRAY);
    Canny(gray, gray, 100, 200, 3);
    // find contours
    findContours(gray, contours, hierarchy, CV_RETR_LIST,
    CV_CHAIN_APPROX_SIMPLE, Point(0, 0) );
    int max = 0, min = oriMask.cols,
    maxY = 0, minY = oriMask.rows;
    for(int j=0; j<contours[0].size(); j++)
    {  
        Point p = contours[0][j];
        max = max(maxX, p.x);
        min = min(minX, p.x);
        maxY = max(maxY, p.y);
        minY = min(minY, p.y);
    }
    cv::Rect myROI(minX, minY, maxX-
    minX, maxY-minY);
    // *******************left part*******************
    cv::Mat croppedGray =
    oriMaskGray(myROI);
    cv::Mat cropGray = gray(myROI);
    cv::Rect leftROI(0, 0, (maxX-
    minX)/2, maxY-
    minY);
    cv::Mat image_left =
    croppedGray( leftROI );
    cv::Mat image_right =
    croppedGray(右ROI );
    int num =
    abs(countNonZero(image_left) -
    countNonZero(image_right));
    if(num < minNum){
        minNum = num;
        croppedGray.copyTo(minGray);
        cropGray.copyTo(cropContour);
    }
    }*/
Scalar intensity =
cropContour.at<uchar>(i, j);
if(intensity.val[0] > 100){
    numLeft += abs(j - mid);
    break;
}
for(int j = cropContour.cols-1; j >= cropContour.cols/2; j--){
    Scalar intensity =
cropContour.at<uchar>(i, j);
    if(intensity.val[0] > 100){
        numRight += abs(j - mid);
        break;
    }
}
vec1.push_back(numLeft);
vec2.push_back(numRight);
float *arr1 = &vec1[0];
Mat p(1,vec1.size(),CV_32F,arr1);
float *arr2 = &vec2[0];
Mat p1(1,vec2.size(),CV_32F,arr2);
Mat result1;
matchTemplate(p,p1,result1,CV_TM_CCORR_NORMED);
cout<<result1.at<float>(0,0)<<endl;
return result1.at<float>(0,0);
}
int main(){
    try{
        string str,temp;
        vector<double> scores;
        for (int i = 1; i <=257; i++)
        {
            //get images
            str = "/u1/home/matlabUSR/mznrb/eclipse/workspace/TongueRecognition/Non-tongue257/";
            std::stringstream out;
            out << i;
            temp = out.str();
            std::cout<<temp;
            str = str + "NonTonImg" +
temp + ".jpg";
            Mat oriImage= imread(str,
CV_LOAD_IMAGE_COLOR);
            if (!oriImage.data) // Check for invalid input
            {
                std::cout << "Could not open or find the image" << std::endl;
                return -1;
            }
            //get masks
            str = "/u1/home/matlabUSR/mznrb/eclipse/workspace/TongueRecognition/Non-tongue257/";
            str = str + "NonTonImg" +
temp + ".mask.ppm";
            Mat mask = imread(str,
CV_LOAD_IMAGE_COLOR);
            if (!mask.data) // Check for invalid input
            {
                std::cout << "Could not open or find the image" << std::endl;
                return -1;
            }
            //shrink the image size and mask size to one-fifth
            int rows = mask.rows/5;
            int cols = mask.cols/5;
            resize(mask, mask,
Size(cols, rows), 0, 0, INTER_CUBIC);
            resize(oriImage, oriImage,
Size(cols, rows), 0, 0, INTER_CUBIC);
            double result1=different(mask);
            scores.push_back(result1);
            ofstream ocout;
            string str1;
            ocout.open("CorrelationFeature_NonTongue257.txt");
            for(int i=0;i<scores.size();i++)
            {
                std::stringstream out1;
                out1 << scores[i];
                str1 = out1.str();
                ocout<<str1<<endl;
            }
            ocout.close();
            str1.clear();
            std::cout<<"Done!"<<std::endl;
            return 1;
        }
    } //catch opencv exception
    catch( cv::Exception& e )
    {
        const char* err_msg = e.what();
        std::cout << "exception caught: " << err_msg
<< std::endl;
    }
}