Fingerprint Identification using SVM

Team member: Xuan Xu, SUNetID: xuanx

I. Introduction

Fingerprint is considered as a dominant biometric trait due to its uniqueness and low cost. Fingerprint identification is the process of comparing instances of friction ridge skin impressions, from human fingers or toes, to determine whether these impressions could have come from the same individual. Fingerprint identification is applied to many areas in our life, such as mobile phone and PC unlock, crime detection and so on. However existing traditional solutions of fingerprint identification has many problems. For example, the requirement of the quality of fingerprint image received by sensor of electronic device must be high. If you touch the sensor with water, the system always cannot figure it out and failed to recognize it.

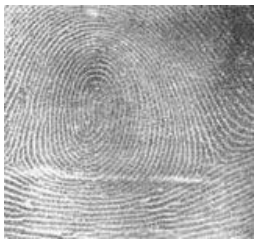
Machine learning technique such as SVM presents a non-traditional solution for fingerprint identification problem. It has the potential to achieve higher accuracy and performance. The idea is to build feature vectors based on pixel values of fingerprint and use it to train the SVM to understand the certain pattern hiding behind it. Then detect the owner of fingerprints through pattern match. Experimental results indicate that one versus all SVM is a promising approach for fingerprint identification. The baseline can reach accuracy as high as 90.24%.

II. Related Work

The traditional solutions of fingerprint identification problem is to do classification based on the general ridge patterns of several or all fingers, such as the presence or absence of circular patterns. There are several classification systems, such as the Roscher system, the Juan Vucetich system, and the Henry Classification System. Usually the classification systems categorize fingerprints into several patterns. They are Arch, Loop, Whorl, Tented Arch and Right Loop in general. There are also some special patterns such as Double Loop. Figure1 is some examples of these patterns. Then fingerprint identification expert or expert computer systems identify the fingerprints by comparing the position and variance of these patterns.

Arch

Loop



Whorl

Tented Arch

Figure1. Fingerprint Patterns

However the pattern classification systems require quite high quality fingerprint image collected. It's hard to filter out some noise such as scar or water. SVM can overcome this type of challenges by intentionally include some noise into the training set.

III. Dataset and Features

The fingerprints are collected from 5 different volunteers. Since it easy to differentiate fingerprint from different finger just by finger size, to make the most challenge case and simplify the data set collection process, only thumb fingerprints are collected. So in the data set, there are totally 378 fingerprints of 5 different thumbs. The number of each thumb is arbitrarily different, vary from 72 to 84. To include some noise in the dataset, one volunteer is piano performer whose skin is rough due to long time practice. The fingerprint quality collected from this volunteer is pretty low. There are many arbitrarily small dots located in the fingerprints.

To converse the fingerprints to data, scan all fingerprints and extract the pixel value of them. The RBG value contains no more information than Gray value. So to reduce the data size without any information loss, convert RBG value Gray value is necessary. Each fingerprint will match to one corresponding matrix of pixel value.

Before using pixel value of fingerprints as features, three problems have to be solved and some preprocess should be done.

1. Blank parts around fingerprints:
   The blank parts around fingerprints contain no information. Too much blank parts will cause two potential risks. 1st is the larger redundant pixel values of 255 which make feature size too big and slow down both training and predicting process. 2nd is introduce unnecessary noise which will make SVM get confused. For example, SVM may differentiate fingerprints by simply look at the size of blanks part, which is not valid in fingerprint identification process. To solve this

problem, a simple way is to trim those columns and rows of all 255 in the fingerprint matrices and reform smaller matrices.

2. Darkness variance:
   The darkness of fingerprint varies due to the thumb pressure and ink type. One same thumb can have fingerprints of infinite darkness variance. So it should be treat as noise of fingerprint identification process. To filter it out, do these steps of normalization is a good solution:

   Iterate for all fingerprint matrices $X_i,\ i \in \{1, 378\}$
   (i) $X_i = X_i/255$
   (ii) $\mu_i = sum(sum(X_i))/(row \times column)$
   (iii) $X_i = X_i - \mu_i$

   This normalization will also help fit data into RBF kernel, which will be mentioned later. To visualize the post normalized fingerprints, just multiply 255 to X_i. Figure2 shows the result that above 2 problems are solved.

Before After

Figure2. Pre/Post-process comparison

3. Feature size difference:
   Since the sizes of all fingerprints are different, the dimensions of all matrices of pixel value are different from each other. The width of collected 378 fingerprints varies from 140 to 238 pixels. The height varies from 238 to 324. The unified feature size of all examples is mandatory. So all fingerprints are cut into $\mathbf{row} \times \mathbf{column}$ pieces. Here **row** and **column** are same numbers for all fingerprints. Then get the average number of all pixel values in one piece, which is one feature of the fingerprint. Thus, each example has $\mathbf{row} \times \mathbf{column}$ features. Experimental result shows $\mathbf{65 \times 105}$ give the best performance.

This series of pre-processing work finalized the feature of examples. For label part, something also needs to be done to make this multi-class classification problem to a binary classification problem support by SVM. Since there are only five types of fingerprints collected, the label $Y \in R^5$. Each element in label indicates the probability that it belong to corresponding category. For example, if $y = [1,0,0,0,0]$, it means this fingerprint come from the first volunteer. So the label of all examples is a matrix of 378 rows and 5 columns.

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_{378} \end{bmatrix} = \begin{bmatrix} y_1^1 & \cdots & y_1^5 \\ \vdots & \ddots & \vdots \\ y_{378}^1 & \cdots & y_{378}^5 \end{bmatrix}$$

And for sure, only one element in one row can be 1, others should be 0.

IV. METHODS

The model selected for the fingerprint identification problem is SVM with kernel of Radial Basis Function. The cost function of SVM can be written as

$$J(\alpha) = \frac{1}{m} \sum_{i=1}^{m} L(\sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)}), y^{(i)})$$

In this function, $K(x^{(i)}, x^{(j)})$ is the Radial Basis Function kernel which is

$$K(x^{(i)}, x^{(j)}) = exp(-\frac{1}{2\tau^2} \left\| x^{(i)} - x^{(j)} \right\|_2^2)$$

The kernel project features into a higher dimension, so that we can find the boundary of different classes clearly. Take the Radial Basis Function as an example. It described the similarity between two cases. If the distance between $x^{(i)}$ and $x^{(j)}$ is huge, then $K(x^{(i)}, x^{(j)}) \approx 0$. If $x^{(i)} = x^{(j)}$, $K(x^{(i)}, x^{(j)}) = 1$. Remember in section III, it is mentioned that data normalization is necessary to make it fit RBF kernel. The RBF kernel as the Fourier transform of the Gaussian distribution with mean zero and variance τ^2. Normalization can guarantee that all feature values have a zero mean.

The second key part is α, which is the object that would be trained to fit the patterns of examples. From the function it's easy to tell that α is the weight of similarity between two examples. The α after training should be like this: if the similarity between $x^{(i)}$ and $x^{(j)}$ is high, the corresponding α should be greater than 0(the value is the bigger the better) which means bonus for the pattern match, otherwise α should be a negative number which means penalty for mismatch.

$y^{(i)}$ in the cost function is the label of ith example. The general SVM is used for binary classification problems. So $y^{(i)}$ here is different from what discussed in section III. In binary classification $y^{(i)} \in \{-1,1\}$. Usually 0 is the decision boundary to indicate which class the examples belong to. But sometimes it varies for some specific problem, little greater/less than 0 is allowed.

L is the loss term of one example.

$$L(\sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)}), y^{(i)}) = [1 - y^{(i)} \sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)})]_+$$

where $[t]_+ = \max\{t, 0\}$ is the positive part function. $\sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)})$ is the probability that $x^{(i)}$ get a positive result. So if $y^{(i)} = 1$ and the probability greater

than 1 or $y^{(i)} = -1$ and probability less than -1 which means the prediction is **strong** correct, the loss term is zero. Otherwise it will give a penalty larger than 0 to cost function.

To minimize cost function $J(\alpha)$, stochastic gradient descent for kernel supervised learning problem is a good option for performance wise. The gradient of the individual loss term is

$$\nabla_\alpha[\mathrm{L}]_+ = \begin{cases} -y^{(i)} \sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)}) & \alpha < 1 \\ 0 & \text{otherwise.} \end{cases}$$

In SVM training, index $\mathrm{i} \in \{1, \ldots, \mathrm{m}\}$ is uniformly chosen at random for a total of $200 \times \mathrm{m}$ steps. Update α in step t

$$\alpha := \alpha - \eta_t \nabla_\alpha[\mathrm{L}]_+$$

η_t is stepsize of each iteration. The common choice of stepsize is to use $\eta_t = 1/\sqrt{t}$. But the experimental result shows $\eta_t = t^{-0.35}$ give the best accuracy for this problem.

To avoid high variance issue, it's better to add regularization part into cost function

$$\frac{\lambda}{2} \sum_{i,j} \alpha_i \alpha_j K(x^{(i)}, x^{(j)})$$

The goal of regularization is to make the pattern that fit by SVM more general instead of try too hard to fit one special example in training set. So the individual gradient should also include this regularization term.

$$\nabla_\alpha[L]_+ = \begin{cases} -y^{(i)} \sum_{j=1}^{m} \alpha_j K(x^{(i)}, x^{(j)}) + \lambda K^{(i)} \alpha_i & \alpha < 1 \\ \lambda K^{(i)} \alpha_i & otherwise. \end{cases}$$

where $K^{(i)} = \begin{bmatrix} K(x^{(i)}, x^{(1)}) \\ \vdots \\ K(x^{(i)}, x^{(m)}) \end{bmatrix}$

As mentioned above, SVM is used for binary classification problems. To make it as a solution for fingerprint identification, the most intuitive method is build a one versus all SVM for this multi-class classification problem. Since the class number of collected data is 5. The training object switch from vector $\alpha \in \mathrm{R}^{\mathrm{m}}$ to matrix $\mathrm{A} \in \mathrm{R}^{\mathrm{m} \times 5}$. $\mathrm{A_k}$, the kth row of matrix A is the $\alpha^{\mathrm{k}} \in \mathrm{R}^{\mathrm{m}}$ of kth class.

$$A = \begin{bmatrix} \alpha^1 \\ \alpha^2 \\ \alpha^3 \\ \alpha^4 \\ \alpha^5 \end{bmatrix}, \alpha^k \in R^m$$

In other word, instead of training one vector $\alpha \in \mathrm{R}^{\mathrm{m}}$, 5 vectors $\alpha^{\mathrm{k}} \in \mathrm{R}^{\mathrm{m}}$ (k ∈ [1,2,3,4,5]) needs to be trained separately. Before training process of α^{k} starts, the labels of training examples should be converted from a 5 dimension vector to a binary value $y^{(i)} \in \{-1,1\}$. The conversion process is simple. For α^{k} (k ∈ [1,2,3,4,5]) training,

$$y^{(i)} = 2 \times 1\{y_k^i = 1\} - 1$$

The intuition of this conversion is that for kth class, if example belongs to it, $y^{(i)} = 1$. Otherwise $y^{(i)} = 0$.

Now A = $[\alpha^1, \alpha^2, \alpha^3, \alpha^4, \alpha^5]^{\mathrm{T}}$ including 5 trained SVM α correspond to 5 classes. Pick a testing example not included in the training set and calculate the probability that it belong to each class.

$$P(test\ example\ i \in class\ k) = \sum_{j=1}^{m} \alpha_j^k K(x^{(i)}, x^{(j)})$$

The class which test example belong to should have the highest *P* correspondingly.

V. Results

There are many parameters that can affect the performance of SVM. The first pair of key parameters are **row** and **column** mentioned in section III when do data preprocessing. This pair of parameters decides the feature selection directly. The feature size is **row** × **column**. And each feature is formed by how many pixel dots. The process is like convert a high definition image to a lower definition image. So that many constraints of that convert process have to be followed. As mentioned in section III, the matrices' dimensions vary from 238×140 to 324×238. The upper bounds are 238 and 140 for **row** and **column** separately. Otherwise there will be some features of small fingerprints are not available(NaN) which would hurt the prediction accuracy significantly, lower than 30%. So 4 pairs of **row** and **column** are picked within upper boundary to compare the result and see its effect to prediction accuracy. Figure3 show the result of these 4 pairs of **row** and **column** with different λ.

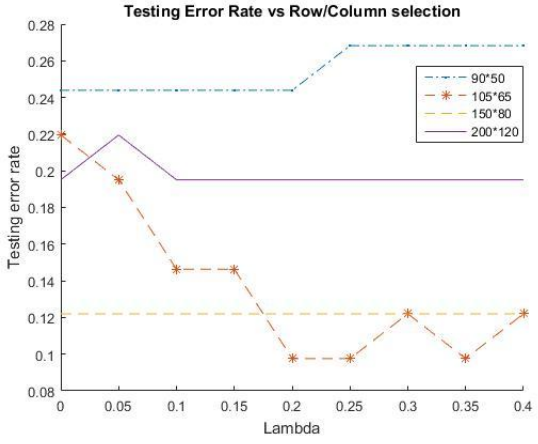


Figure3. Row/Column selection comparison

The result is base on 90% collected data are in training set and the left 10% are in testing set. And $\tau = 14$. The best prediction accuracy is given when row and column are 105 and 65 separately. Either larger or smaller row×column will lead to lower accuracy. This can be explained intuitively. For big row×column cases, eg. 200×120, it will be problematic when dealing with small fingerprint of 238×140. Look at the row dimension, split 238 dots into 200 pieces, the result is that 162 pieces include only one dot for each. The other 38 pieces include 2 dot for each. This unbalanced feature selection made those 38 pieces as outliers and the information at those spots are missing compare to other 162 pieces. Same problem happens to the column dimension. Thus the cases of big row×column perform badly to small fingerprints. Then for the small row×column cases, it easier to understand. Think about compress a high definition image to a lower definition image. There must be some part of the low definition image blurred and cannot be recognized clearly. Even more, the image is just like be full of mosaic. Under this circumstance, SVM also cannot make correct prediction based on the distorted data. In contrast with the big row × column cases, its prediction accuracy for big fingerprints is worse.

The other critical element affect the performance of SVM significantly is the size of training set. Experimental result show that SVM cannot understand the pattern clearly with training set less than 30 examples for each class. Figure4 shows the Testing error rate varies with training examples and Lambda. It indicated that with appropriate λ, the prediction accuracy increases with the size of training data set.

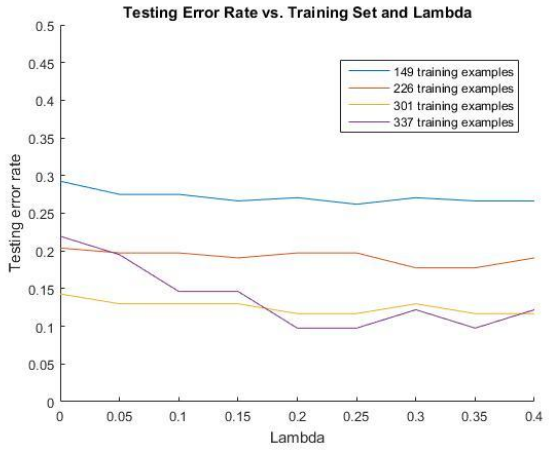


Figure4. Training set size vary comparison

The best result of 90.24 accuracy reached by using $\lambda = 0.25$, training set covers 90% collected data, and row×column equal to 105×65.

VI. Conclusion

The work done is just to do multi-class classification in only 5 classes. The appearance examples from 6^{th} class should be considered. Thus it's necessary to anneal a threshold for the output of each α^k ($k \in [1,2,3,4,5]$). If the output is not larger than any threshold, that means it come from 6^{th} class.

Based on SVM model, it can come to 90.24% correct detection for fingerprint. However there is still plenty space to improve the design. Firstly 90.24% accuracy is not enough in some application like crime detect. Since all fingerprints collected have the noise coming from the original scanned position, the helper unit which is used to adjust scanned fingerprint to make them with zero angle leaning needs to be added in the pre-process. Secondly the training time needs to be improved further. Currently each fingerprint is cut into small boxes to be aggregated as a feature. In this way, it is easy to boost the feature size to 6500 to 10000 with fingerprints of 100 dpi which results in minutes run time. This is where Principal Component Analysis comes to the scene. PCA can compress the large feature size to smallest set but with good quality. Thirdly new models need to be added to be compared with SVM model. The one that is also popular in fingerprint detection is Generalization model. Generalization model will use Bayes Rule to vote for the most possible case. In the end these two models could be integrated to achieve the task.

References

[1] Richard O. Duda, Peter E. Hart, David G. Stork, "Pattern Classification 2nd ed.", Wiley-Interscience, 1973.

[2] Tom M. Mitchell, "Machine Learning", McGraw-Hill Science/Engineering/Math, 1997.

[3] Yuan Yao, Gian Luca Marcialis, Massibmiliano Pontil, Paolo Frasconi, and Fabio Roli, "A New Machine Learning Approach to Fingerprint Classification" Dept. of Mathematics, City University of Hong Kong, DIEE University of Cagliari, Italy, DII, University of Siena, Italy, DSI, University of Florence, Italy.

[4] Ali Ismail Awad, "Machine Learning Techniques for Fingerprint Identification: A Short Review" EE Dept, Al Azhar University.