Handwritten Character Recognition

Saurabh Mathur

December 10, 2010

1 Introduction

Touchpad based devices like phones and tablets are now ubiquitous and growing even more in popularity. Due to their form factors, however, otherwise standard means of input like keyboards are less effective in these devices. Infact using scribbling to recognize handwriting is a viable alternative. In this project we investigated a method of recognizing handwritten characters to allow automatic recognition of characters. We used a gaussian mixture model for modelling the feature distributions.

2 Previous Work

Most of the published literature in this field is about a decade old partly because touch input devices were limited to specialized usage and were not commonly used, thus limiting the impact of any research in this area. [4] and [5] offer a good summary of all the techniques that have been tried for online and offline handwriting recognition.

Among the approaches taken towards handwriting recognition one is to first segment the given words into characters and then recognize each of the characters. The online problem where timestamp is given for each point is similar to speech recognition and thus ideas from that field have been applied to handwriting recognition mainly by modelleing either the words or the characters using markov models [2]. Our approach is based on that taken by [3] as a first step.

3 Character Features

Most touch devices can convert a scribble on the touch screen to a series of x and y coordinates with timestamps. For the scope of this work we only considered the offline part of this data, which is a sequence of coordinates without the timestamp therefore assuming uniform speed. This also makes the problem less writer specific by ignoring differences in writing speed. However, since the input is otherwise unconstrained it is desriable for our features to be independent of the length, orientation and scale of the input. We call a series of coordinates generated without lifting the pen a stroke. Each character can be made from one of more strokes. For cursive handwriting every word can also have arbitary number of strokes.

Since our input is a series of points our features are defined at these points. At each point we use

1. The horizontal and the vertical components of the gradient defined between every two consecutive points.
2. The sine and the cosine of the angle made by the gradient with the horizontal axis. This feature is scale invariant.
3. The gaussian curvature defined as the angle between the two segments joined by a point. As can be seen in figure 1 it is both scale and rotation invariant.

These features allow us to capture local details. To recognize a stroke we need this information defined at several points. Thus we define a frame as these features defined on a window of consecutive points

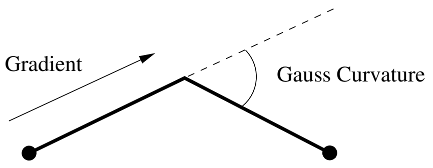

Figure 1: Features computed at a point.

of a fixed length. Since using all points on a stroke may lead to overfitting, as we expect the number of points to vary with scale and writing style, we only use a subset of frames defined on a stroke. By using local extremas on a stroke as the center of these windows we expect to cover the most important features within each stroke. To formalize, each character sequence $\vec{C}$ can contain any number of strokes $\vec{S}$. And each stroke has some features $F_1, F_2...$ defined on it. Thus our problem is to find the character that has the maximum a posteriori probability given a set of features.

$$\vec{C}^* = \arg\max_{\vec{C}} Pr(\vec{C}|\vec{S}) \tag{1}$$

This is the same as maximizing

$$\vec{C}^* = \arg\max_{\vec{C}} Pr(\vec{S}|\vec{C})Pr(\vec{C}) \tag{2}$$

$Pr(C)$ is the prior of the character or the character sequence under consideration. For instance, we can use the frequency of each word in the english language as its prior. For our experiments we assume each character to be equally likely. Thus we try to maximize

$$\vec{C}^* = \arg\max_{\vec{C}} Pr(F_1, F_2, ...|\vec{C}) \tag{3}$$

where we simply replaced S by its constituent features. Assuming independence of features and ignoring the order between them

$$\vec{C}^* = \arg\max_{\vec{C}} \prod_i Pr(F_i|\vec{C}) \tag{4}$$

We used our approach to distinguish discretely written individual characters however our approach can be extended to cursive writing as well if the boundaries between each character in a word is already known. We can also remove the constraint on segmentation if the handwriting is modelled as a Hidden Markov process.

4 Gaussian Mixture Model

In order to find $Pr(F_i|\vec{C})$ we model our features as a mixture of gaussians. The features can then be projected on the gaussians which have certain prior for each character learned during training.

$$Pr(F_i|\vec{C}) = \sum_{j=0}^{K} Pr(F_i|G_j)Pr(G_j|\vec{C}) \tag{5}$$

We train the parameters of our gaussians mixture model using EM algorithm and by using the labelling of the training set estimate the $Pr(G_j|\vec{C})$ as

$$Pr(G_j|\vec{C}) = \frac{\sum_{i=0}^{F} 1\{F_i \exists \vec{C}\} Pr(G_j|F_i)}{\sum_{j=0} N \sum_{i=0}^{F} 1\{F_i \exists \vec{C}\} Pr(G_j'|F_i)} \tag{6}$$

where F is all the features in the training set, N is the number of gaussians and $1\{F_i \exists \vec{C}\}$ is the indicator function of features coming from character $\vec{C}$. During the testing phase we can then rank the characters by

$$\prod_i^{F_s} \sum_j^{K} (Pr(F_i|G_j)Pr(G_j|\vec{C})) \tag{7}$$

5 Results

We used the UJIpenchars2 dataset from the UCI Machine Learning repository for our experiments. This is a dateset of about 11k samples of handwritten characters from 11 writers. Characters include the both upper and lower case English letters, digits, 16 other ASCII characters and 14 spanish non ASCII characters. An example character is shown in figure 2.

In our experiments we found that this approach is insufficient to predict a character with very high accuracy. At best this can be a preprocessing step to a more detailed prediction based on markov models

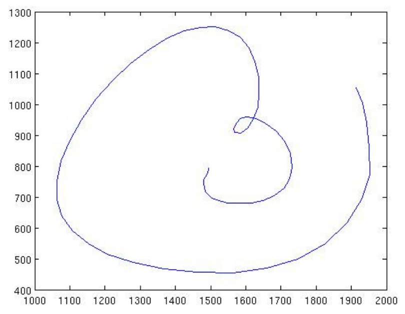

Figure 2: An example character in our dataset.

and further restricted by dictionary search on words or characters. In order to quantify the accuracy of this approach we choose as a metric our performance measure as the average position of the actual label in the sorted list of labels predicted by our algorithm. In the plots ahead accuracy is the inverse of the average of this index for all characters.

We used hold out cross validation for these results. EM was initialized using k-means to avoid singularities and get faster convergence[1]. In figure a we show the improvement in accuracy with increasing features.

In figure 3 we show the accuracy with increasing number of gaussians used to model the feature set. The accuracy increases upto a certain point but past that it leads to overfitting and the convergence also suffers. Figure 4 plots the accuracy with window size for 10 gaussians. Figure 5 plots the accuracy with increasing features. The first iteration only included gradient projections, the second included the sines and cosines and the last also included the curvature. The number of gaussians were fixed at 10 and the window size at 15.

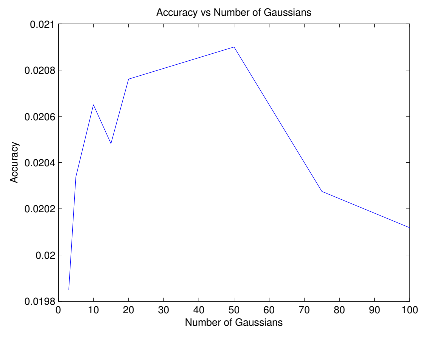

Figure 3: About 100 gaussians is our optimal point.

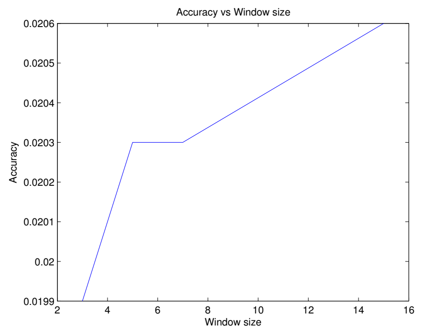

Figure 4: Accuracy increases with window size.

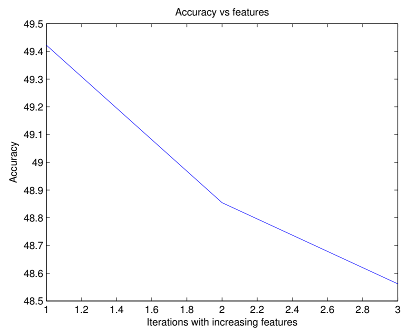

Figure 5: Accuracy increases with more features.

6 Future Work

In our experiments we ignored the sequence or the temporal information among the strokes within a character. Using that information could give a good boost to our results. Also using language dictionaries would limit the search space once we move to word recognitions.

References

[1] S. Calinon, F. Guenter, and A. Billard. On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37(2):286–298, 2007.

[2] Jianying Hu, M.K. Brown, and W. Turin. Hmm based online handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(10):1039 –1045, October 1996.

[3] K.S. Nathan, H.S.M. Beigi, Jayashree Subrahmonia, G.J. Clary, and H. Maruyama. Real-time on-line unconstrained handwriting recognition using statistical methods. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 4, pages 2619 –2622 vol.4, May 1995.

[4] R. Plamondon and S.N. Srihari. Online and offline handwriting recognition: a comprehensive survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):63 –84, jan. 2000.

[5] C. C. Tappert, C. Y. Suen, and T. Wakahara. The state of the art in online handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(8):787–808, 1990.