# CLASSIFICATION OF HANDWRITTEN DIGITS BY THE SET OF PARTIAL LINEAR AND QUADRATIC MODELS

*Youngsik Kim*

CS229 Final Project
Stanford University
youngsik@stanford.edu

## ABSTRACT

There have been many algorithms to categorize a set of images. They are successful in some sense, but they usually are vulnerable to small variations, such as rotation, translation, skewness, etc. Handwritten digits also have many variations, but we can promptly recognize what the digit is without much effort. When we think about the moment writing a digit, we can easily find that we are actually drawing several lines, arcs, or ellipses. In this project, we focused on the constituents of a digit, which are line segments and ellipses, and their relative positions. This project has two major steps. The first step is to find a set the best line segment and ellipse models that can represent a digit using locally weighted linear regression model. The second step is to generate features from the set of models, and to categorize digits using existing classification methods. The result is noticeable for several digits, but not as good as the best performing algorithm, yet.

## 1. INTRODUCTION

When we write a digit, we don't think about pixels but the entire topology. More specifically, we just draw lines and ellipses or arcs, which are parts of ellipses. The main goal of this project is to show that it is possible to categorize a digit with the set of simple 1st order or 2nd order models. We will use the locally weighted linear regression method to find the best locally fitted linear or quadratic model. And group the similar models together to represent them with a small number of features. The last step will be using Naive Bayesian method for the pairs of the features to emphasize the association between features. In this project, we are going to use MNIST Database of Handwritten Digits. [1] Expected goal of this project is to get the decent error rate and a good classifier that can tolerate lots of variations.

## 2. FEATURE EXTRACTION

Based on the usual experience, we can assume that the constituents of a digit are line segments, arcs, and ellipses. Since an arc is a part of an ellipse, we can divide the feature extraction as two parts: finding a linear model for the best line segment and a quadratic model for the best arcs or ellipses. Basically we used the locally weighted linear regression by solving a normal equation.

### 2.1. Linear Model

Any line can be written as,

$$ax_1 + bx_2 + c = 0 \tag{1}$$

It is hard for a digit to have a line segment whose extension passes the origin, which is the upper-left part of an image, let's assume that c cannot be 0. In order to form a normal equation, we can rewrite the equation as below:

$$ax_1 + bx_2 = \theta^T X = y \tag{2}$$

where $\theta = [a \;\; b]^T$, $X = [x_1 \;\; x_2]^T$, and $y = [1 \;\; 1]^T$
To minimize MSE, we know the answer is,

$$\theta = (X^T X)^{-1} X^T y \tag{3}$$

### 2.2. Quadratic Model

General form of a quadratic curve is,

$$a{x_1}^2 + 2bx_1x_2 + c{x_2}^2 + 2dx_1 + 2fx_2 + g = 0 \tag{4}$$

We can also assume that g is not zero, since there will be no ellipse that passes the origin. Then we can obtain the same expression as equation 3 after defining $X = [x_1 \;\; x_2]^T$, and $\theta = [a \;\; 2b \;\; c \;\; 2d \;\; 2f]^T$.
Though we can get the parameter vector $\theta$ from the normal equation, it cannot be always an ellipse. In order to be a matrix, these three conditions [2] have to be satisfied.

$$\triangle = \begin{vmatrix} a & b & c \\ b & c & f \\ d & f & g \end{vmatrix} \neq 0 \tag{5}$$

$$J = \begin{vmatrix} a & b \\ b & c \end{vmatrix} > 0 \tag{6}$$

$$\frac{\triangle}{a+c} < 0 \tag{7}$$

When we obtain the parameter vector $\theta$, we can also test whether it is a closed ellipse or not. Also, when it is open, the direction of missing arc is also important. Unless we know it, there is no way to distinguish 3 from 8.

### 2.3. Weight Matrix

Now the remaining question is how we can find a set of points which gives us the best linear or quadratic model. If we use all the points in modeling, we may find a good model for simple digits such as 0 and 1. But for the other complex digits, there cannot be a model that includes all the points. On the other hand, if we use too few points, the model may not represent a real constituent of the digit. Thus, we have to use the locally weighted linear regression method with a weight matrix $W$,

$$\theta = (X^T W X)^{-1} X^T W y \tag{8}$$

It is clear that the model we get depends on the weight matrix. Therefore, We need to apply various weight matrices to find the best model. In order to minimze the inteference from unnecessary points, we can define a weight matrix $W$ as,

$$W_{x_1,x_2} = \begin{cases} 1 & \text{, if } (x_1-p_1)^2 + (x_2-p_2)^2 < r^2 \\ 0 & \text{, otherwise} \end{cases} \tag{9}$$

where $(p_1, p_2)$ is a center, and $r$ is a radius of a circle. We may use the famous weight function that is similar to Gaussian distribution, but the number of points that have significant effect on the model grows very fast as $\sigma$ increases. This weight function is better for controlling the number of effective points than Gaussian-like weight functions.

Obviously, there are two variables that affects $W$: the center and radius of a circle. Again, to minimize the interference, the top/bottom/left/right-most points are chosen to be the candidate for center of a circle. For example, when we choose a random point from a digit 9, the points close to both a circle and a line are not good candidiates to guess a model. Then we sweep the radius $r$ from 2 to the width of a digit image.

### 2.4. Set Of Best Models

Once we have found the best model among all centers and radii, we can keep doing this process after removing the points which are close to the found models. Because we are reducing the number of remaining points, this repetition finishes either by too few number of remaining points or by absolutely bad fitting score of the best model. Figure 1 shows the steps of finding best models, and the reconstructed digt based on the models.

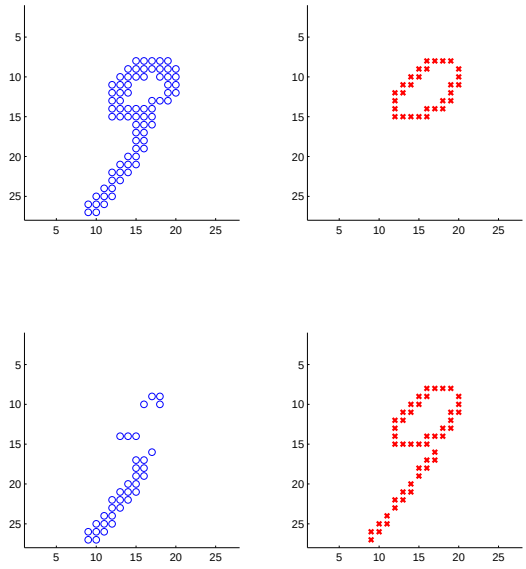

Figure 1: Finding best models. Left column shows the remaining points, while right column represents the reconstructed digit. Red crosses are for the best model found by the iteration.

### 2.5. Feature Representation

Though we have a lot of parameter vectors $\theta$, we can group them together with several features. Lines can be grouped depending on whether they are more horizontal or vertical. We are also able to group ellipses by their center and the direction of the empty arc. Most of the time, zeros or ones can be represented by a single model as in figure 2,

However, the other numbers need at least two models, and the relationship between models as well as the existence of the models becomes very important. As in figure 3, the existence of certain models cannot tell two models at all. Therefore, we can define a feature as a pair of two models and their relative positions.

If there is a pair of two ellipses, their relative radii are encoded as a feature. If an ellipse and a line segment are a pair of models, the contact point or the closest point is found, and the relative positions are encoded based on the pont of contact. When we have two line segments, we can handle them as the same manner by finding the closest point or the contact point.

Table below shows the entire set of features. Total 216 features are used, in this project, but one of the important thing is that a digit can be represented by a sparse vector since 3 or 4 models are enough to cover all the points in a

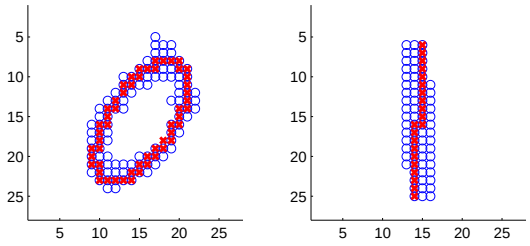

Figure 2: Simple digits such as 0 and 1 can be represented a single model.

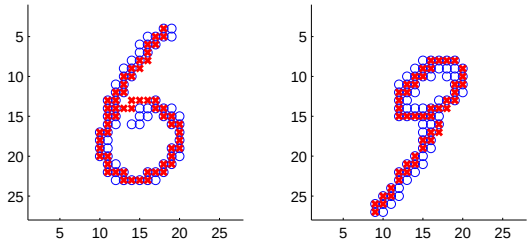

Figure 3: Existence of an ellipse and a line segment is the same for 6 and 9. Relative position has to be considered.

digit. Average L1-norm of the feature vectors is 1.41 in the dataset we used.

## 3. CLASSIFICATION

We used Naive Bayesian method with laplace smoothing. The main reason of using Naive Bayesian is because we have binary feature vectors and only positive examples for each digit. To apply the other methods, we need many meaningful negative examples, too. For example, to categorize digit zero, there have to be a meaningful set of examples which are not zeros. They can be training set of other digits, or randomly generated feature vectors. The former might have an issue of over-fitting, and the latter might generate meaningless data that can harm the performance of classification. Laplacian smoooothing is also useful since the feature vectors are very sparse, and there can be features that has never been found by training set. Actually there were 116 features that have never been found in the training set.

The only concern is the assumption of this method which assumes the conditional independence between the features given the class. Of course, it is hard to say that the lines and ellipses representing a digit are independent from each other if the digit is given. Though we used pairs of models as features rather than the linear or quadratic models, we still have doubt in independence of features.

| Digit | 1st Answer | 2nd Answer |
|---|---|---|
| 0 | 15.7% | 7.5% |
| 1 | 3.1% | 3.1% |
| 2 | 49.7% | 24.0% |
| 3 | 35.0% | 21.6% |
| 4 | 9.7% | 3.9% |
| 5 | 31.9% | 19.7% |
| 6 | 22.7% | 11.7% |
| 7 | 23.7% | 6.5% |
| 8 | 29.5% | 16.7% |
| 9 | 16.8% | 10.2% |
| Avg. | 23.8% | 12.5% |

Table 1: Error rate of categorization.

## 4. EVALUATION

### 4.1. Dataset and Test Method

MNIST handwritten digit dataset [1], which is a subset of NIST handwritten dataset, is used. It has its own training examples and test examples. However, in this project, we used only 1000 images per digit in the training set. We computed error rates by taking average of 10 different trials. 700 training and 300 test examples were randomlly chosen for every iteration. Features are extracted in the same manner for both training and test examples.

### 4.2. Result

Average error rate is about 23%. This error rate is relatively bad when we compare it with the published result in [1]. If a digit is simple as 0 or 1, the error rate becomes lower. Also, we can explain the low error rate of digit 4 and the high error rates of digit 2 and 3 as the difficulty of estimating quadratic models and get a good feature from them. The good thing about this result is that almost 90% of classification results were correct with 2nd guess. It shows that this algorithm at least is heading a good direction. If we can eliminate the interference of the incorrect first choice later, it still have a chance to perform better.

## 5. CONCLUSION

In this project, we've tested a new algorithm to categorize handwritten digits which is based on the intuition for the

constituents of digits. It has some drawbacks, such as exhaustive search for the best models, and mediocre performance. However, if we can find the best models using parallel processing, and devise good features from them, it is doable. On the other hand, this algorithm has two advantages over other algorithms. First, the concept of constituents of handwritten digits also can be applied many kinds of handwritings, such as various set of alphabets in many languages, since we usually don't think about more complex models when we write something. Second, the feature vector is relatively tolerant to the variations.

The relatively high error rates might have come from the inappropriate method of generating features, or from the errors when finding the best models. For example, when a stroke in a digit is too thick, it is hard to remove all the relevant points because there is a risk of deleting other points which are important to find the next best model from the remaining points. We can try some techniques in vision area to perfectly extract lines and ellipses, but it will be a future work. Also, using other techniques but Naive Bayesian for classification would be a good direction for a future work.

## 6. REFERENCES

[1] http://yann.lecun.com/exdb/mnist/

[2] http://mathworld.wolfram.com/Ellipse.html