IJCNN 2001 Challenge: Generalization Ability and Text Decoding

Chih-Chung Chang and Chih-Jen Lin
Department of Computer Science and Information Engineering
National Taiwan University, Taipei 106, Taiwan
E-mail: cjlin@csie.ntu.edu.tw

Abstract

In IJCNN 2001 a new event is a competition with three challenging problems. In this paper we describe our approaches on solving the first two problems: generalization ability challenge (GAC) and text decoding challenge (TDC). The main learning technique used is the support vector machine (SVM).

1 Introduction

In IJCNN 2001 a new event is a competition with three challenging problems. In this paper we describe our approaches on solving the first two problems: generalization ability challenge (GAC) and text decoding challenge (TDC). The main learning technique used is the support vector machine (SVM).

This paper is organized as follows. In Section 2 we introduce the basic concepts of support vector machines (SVM) for classification and regression. Then Sections 3 and 4 demonstrate our approaches on two problems: GAC and TDC, respectively.

2 Support Vector Machines: Classification and Regression

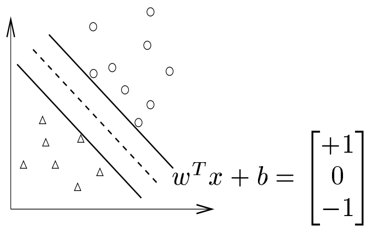

Figure 1: Separating hyperplane

The original idea of SVM [2] is to use a linear separating hyperplane to separate training data in two classes.

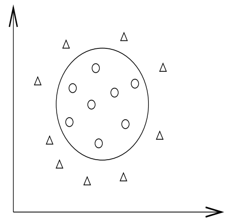

Figure 2: An example which is not linear separable

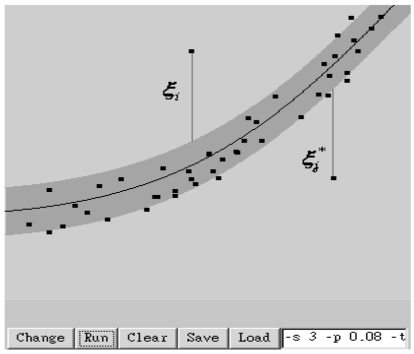

Figure 3: Support Vector Regression

Given training vectors $x_i, i = 1, \dots, l$ of length n, and a vector y defined as follows

$$y_i = \begin{cases} 1 & \text{if } x_i \text{ in class 1,} \\ -1 & \text{if } x_i \text{ in class 2,} \end{cases}$$

the support vector technique tries to find the separating hyperplane with the largest margin between two classes, measured along a line perpendicular to the hyperplane. For example, in Figure 1, two classes could be fully separated by a dotted line $w^Tx + b = 0$. We would like to decide the line with the largest margin. In other words, intuitively we think that the distance between two classes of training data should be as large as possible. That means we want to find a line with parameters w and b such that the distance between

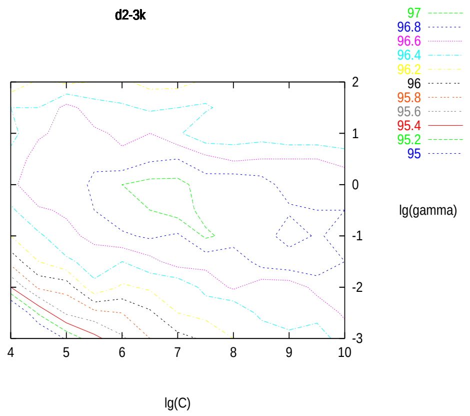

Figure 4: Cross validation using 3000 points

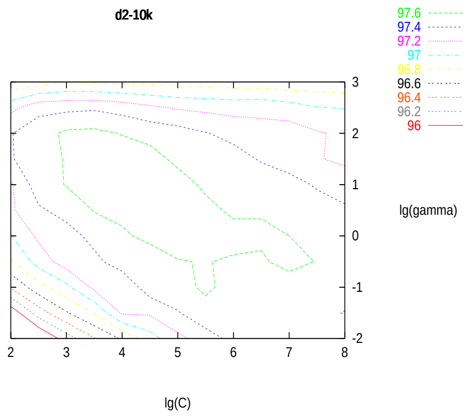

Figure 5: Cross validation using 10,000 points

$w^T x + b = \pm 1$ is maximized. As the distance between $w^T x + b = \pm 1$ is $2/\|w\|$ and maximizing $2/\|w\|$ is equivalent to minimizing $w^T w/2$, we have the following problem:

$$\min_{w,b} \quad \frac{1}{2} w^T w$$
$$y_i((w^T x_i) + b) \geq 1, i = 1, \dots, l. \qquad (1)$$

The constraint $y_i((w^T x_i) + b) \geq 1$ specify that data in the class 1 must be on the right-hand side of $w^T x + b = 0$ while data in the other class must be on the left-hand side. Note that the reason of maximizing the distance between $w^T x + b = \pm 1$ is based on Vapnik's Structural Risk Minimization.

However, practically problems may not be linear separable where an example is in Figure 2. SVM uses two methods to handle this difficulty: First, it allows training errors. Second, SVM non-linearly transforms the original input space into a higher dimensional feature space by a function ϕ:

$$\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C(\sum_{i=1}^{l} \xi_i) \qquad (2)$$
$$y_i((w^T \phi(x_i)) + b) \geq 1 - \xi_i, \qquad (3)$$
$$\xi_i \geq 0, \ i = 1, \dots, l.$$

A penalty term $C \sum_{i=1}^{l} \xi_i$ in the objective function and training errors are allowed. That is, constraints (3) allow that training data may not be on the correct side of the separating hyperplane $w^T x + b = 0$ while we minimize the training error $\sum_{i=1}^{l} \xi_i$ in the objective function. Hence if the penalty parameter C is large enough and the data is linear separable, problem (3) goes back to (1) as all ξ_i will be zero. Note that training data x is mapped into a vector in a higher (possibly infinite) dimensional space by a function $\phi(x)$. In this higher space, it is more possible that data can be linearly separated. An example by mapping a vector $[x, y, z]^T$ from R^3 to R^{10} is as follows:

$$\phi([x, y, z]^T) = (1, \sqrt{2}x, \sqrt{2}y, \sqrt{2}z, x^2, y^2, z^2, \sqrt{2}xy, \sqrt{2}xz, \sqrt{2}yz). \qquad (4)$$

Hence (2) is a problem in a high dimensional space which is not not easy. Currently the main procedure is by solving a dual formulation of (2). It needs a closed form of $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ which is usually called the kernel function. Thus we need some special functions so that $K(x_i, x_j)$ can be easily calculated. For example, using (4), $K(x_i, x_j) = (1 + x_i^T x_j)^2$. Some popular kernels are, for example, RBF kernel: $e^{-\gamma \|x_i - x_j\|^2}$ and polynomial kernel: $(x_i^T x_j/\gamma + \delta)^d$, where γ and δ are parameters.

After the dual form is solved, the decision function is written as

$$f(x) = \text{sign}(w^T \phi(x) + b).$$

In other words, for a test vector x, if $w^T \phi(x) + b > 0$, we classify it to be in the class 1. Otherwise, we think it is in the second class. Only some of $x_i, i = 1, \dots, l$ are used to construct w and b and they are important data called support vectors.

Next we briefly introduce support vector regression (SVR) which finds a function extracting the hidden relationships of the given information. Given training

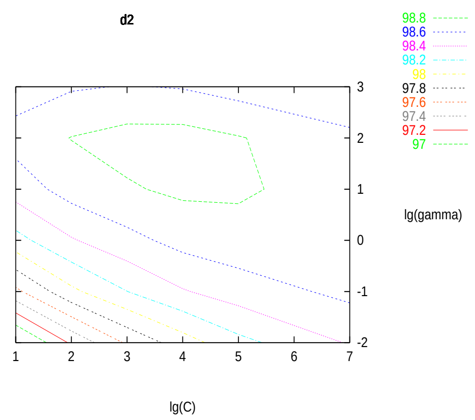

Figure 6: Cross validation using 50,000 points

data $(x_1, y_1), \ldots, (x_l, y_l)$, where x_i are input vectors and y_i are the associated output value of x_i, the support vector regression is an optimization problem [2]:

$$\begin{aligned}\min_{w,b,\xi,\xi^*} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \qquad (5)\\ \text{subject to} \quad & y_i - (w^T\phi(x_i) + b) \leq \epsilon + \xi_i,\\ & (w^T\phi(x_i) + b) - y_i \leq \epsilon + \xi_i^*,\\ & \xi_i, \xi_i^* \geq 0, i = 1, \ldots, l,\end{aligned}$$

where ξ_i is the upper training error (ξ_i^* is the lower) subject to the ϵ-insensitive tube $|y-(w^T\phi(x)+b)| \leq \epsilon$. The parameters which control the regression quality are the cost of error C, the width of tube ϵ, and the mapping function ϕ.

The constraints of (5) imply that we would like to put most data x_i in the tube $|y - (w^T\phi(x) + b)| \leq \epsilon$. This can be clearly seen from Figure 3. If x_i is not in the tube, there is an error ξ_i or ξ_i^* which we would like to minimize in the objective function. For traditional least-square regression ϵ is always zero and data are not mapped into higher dimensional spaces. Hence SVR is a more general and flexible treatment on regression problems.

3 Generalization Ability Challenge (GAC)

The GAC problem can be described as follows: Given a time series of length $T = 50,000$, the kth sample consists of four inputs: $x_1(k), \ldots, x_4(k)$ and one output $y(k)$. Then another 50,000 data with only $x_1(k), \ldots, x_4(k)$ are used for testing.

The first input $x_1(k)$ represents a binary synchronization pulse related to a natural periodicity in the system. That is, sequentially we have nine 0s, one 1, nine 0s, one 1, and so on. Other attributes, $x_2(k), \ldots, x_4(k)$, are real numbers in the range of ± 1.5. An interesting observation is that for the 50,000 training data, 90% of $y(k)$ are -1. Thus it might be possible that if we just guess all test data to be -1, there is already 90% accuracy. The difficulty is how to use learning techniques to achieve an even higher accuracy.

An important information is that $x_4(k)$ is more related to $y(k)$ than the other three. Furthermore, any past or future information may affect $y(k)$. Note that in each of these data there is an additional $x_5(k)$ which is not related to $y(k)$. However, only if $x_5(k) = 1$ we evaluate accuracy of the prediction by considering $y(k)$.

To use SVM for constructing a model, first we have to decide the attributes (i.e. features) of each data. That is, possible variables which may affect $y(k)$. In addition, for each attribute we need an encoding scheme. For example, to represent the periodicity of $x_1(k)$, we can include $x_1(k-5), \ldots, x_1(k+4)$ as 10 binary attributes of the kth data. On the other hand, we can use only one integer between 1 to 10 which indicates the the position of 1 in $x_1(k-5), \ldots, x_1(k+4)$. Based on our experience we choose the former way as it might be better for support vector machines.

We directly use $x_2(k)$ and $x_3(k)$ as they are. As $x_4(k)$ is more important, we consider some past and future elements. After conducting some cross validation tests, we decide to use $x_4(k-5), \ldots, x_4(k+4)$. Therefore, each training data consists of 22 attributes.

For learning techniques like Neural Networks or Support Vector Machines, it is recommended to scale each attribute of data into an appropriate range such as $[-1, 1]$ or $[0, 1]$. Since all raw data under our encoding scheme are already in a small region $[-1.5, 1.5]$, we do not conduct any scaling. Of course we are not very sure if this decision is right or not. We will elaborate more on this issue later.

After preparing the training data, we do the model selection by 5-fold cross validation. We consider only the RBF kernel $K(x_i, x_j) = e^{-\gamma\|x_i - x_j\|^2}$. Thus two parameters are the kernel parameter γ and the penalty parameter C in (2).

First we work on a small subset of the training data:

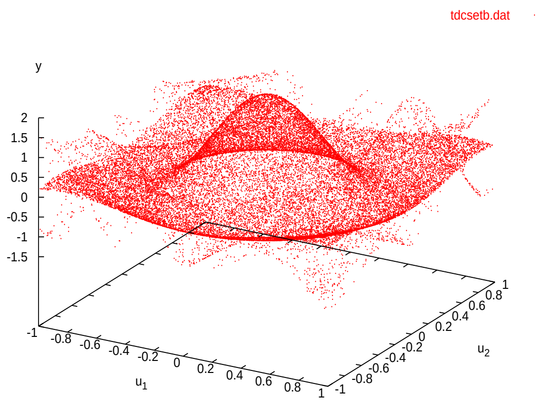

Figure 7: All data points

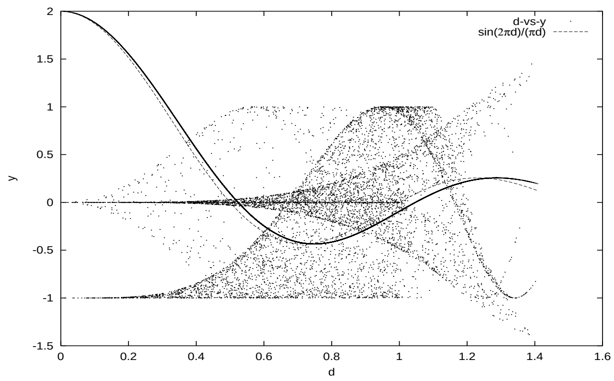

Figure 8: $d(k)$ and $y(k)$

3000 randomly selected points. The contour of cross validation accuracy is in Figure 4 where two axes are $\log_2 C$ and $\log_2 \gamma$. It can be seen that the best cross validation rates happen around $C = 2^7$ and $\gamma = 2^0$. We then work on a larger subset with 10,000 data points. Results are in Figure 5. Surprisingly we find out that parameters with the best cross validation rate are around $C = 2^4$ to 2^5 and $\gamma = 2^0$ to 2^1, a different range than that in Figure 4. This is a little suspicious as earlier we did not have this experience on other problems using SVM. Finally we do the model selection on all 50,000 training data where results are shown in Figure 6. Again we note that the best parameters move to another region.

Therefore, the experiment seems to show that the best parameters depend on the size of the training data. Though it is not clear why this happens, we conjecture that the lack of scaling might be one factor. Thus we redo the same experiments after scaling data. However,

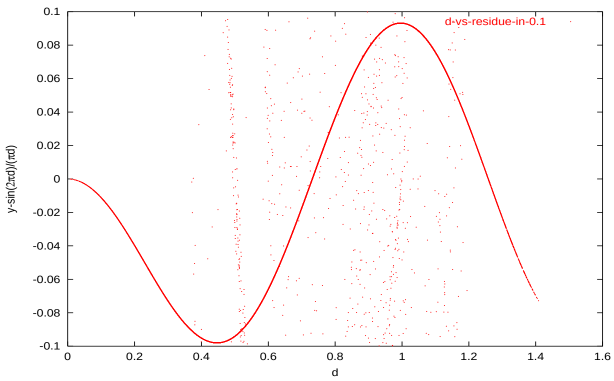

Figure 9: The function g

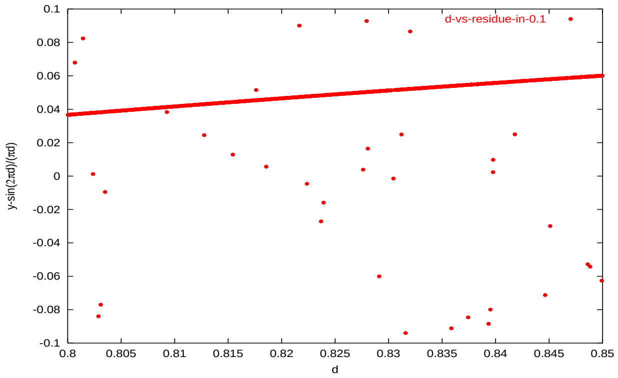

Figure 10: The function g in a small region

the same obervation still occurs.

From the results of model selection we still face a dilemma. We can just choose the best parameter obtained from the cross validation procedure on all 50,000 training data. However, as the 5-fold cross validation procedure actually means the training of 40,000 data, for the future testing on another 50,000 data, we may have to move the parameter a little bit along the trend in Figures 4, 5, and 6. Finally we take a more conservative approach by using parameters from Figure 6. Hence $C = 2^4$ and $\gamma = 2^2$ are selected. We then use this parameter set to train the 50,000 data and obtain a model. Finally we apply this model to predict 50,000 test data.

Here we provide more information about SVM training. For 50,000 training data, the number of support vectors is around 2000 to 3000. Such a small percentage of support vectors decreases the training time. For each parameter set, doing a 5-fold cross validation on 50,000 data takes less than one hour on a Pentium III-500 with 384M RAM. Our experiments were done using

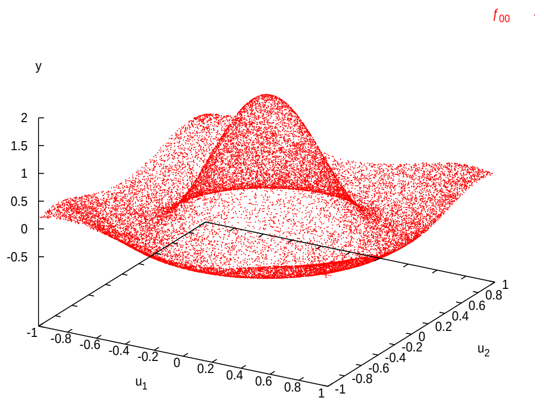

Figure 11: f_{00}

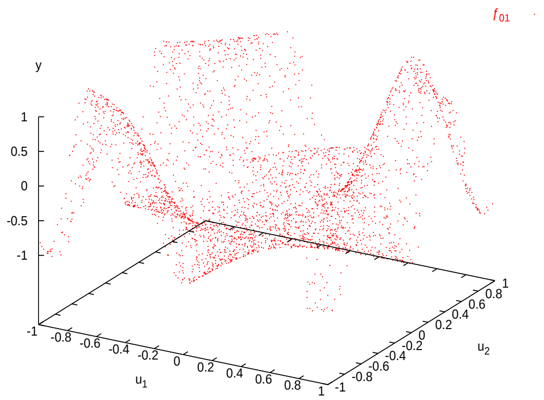

Figure 12: f_{01}

LIBSVM [1] which is one of our support vector machines software. LIBSVM is an integrated package for both support vector classification and regression.

4 Text Decoding Challenge (TDC)

The TDC problem can be described as follows: An English text is converted into a binary sequence $bit(k)$ according to the 8-bit ASCII table, with the character SPACE replaced by the character NULL. We then slide a window of length two across the entire sequence shifting by one from left to right to generate a pattern of two-bit numbers. Then four smooth functions are used to generate outputs $y(k), k = 1, \ldots, T$:

$$y(k) = f_i(u_1(k), u_2(k)), i = \mathsf{00}, \mathsf{01}, \mathsf{10}, \mathsf{11},$$

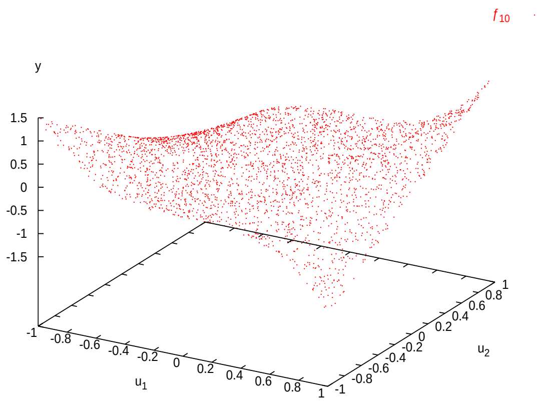

Figure 13: f_{10}

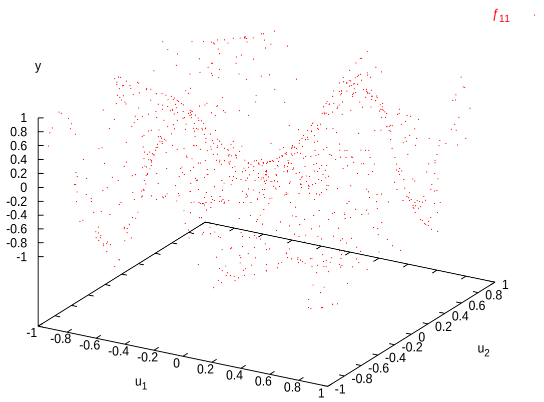

Figure 14: f_{11}

where T is the length of the sequence, and $u_1(k), u_2(k)$ are independent, identically distributed random numbers from a uniform distribution in the range (-1,+1).

We are given $T = 30,720$ data points $(u_1(k), u_2(k), y(k))$. The question is to find out $bit(k), k = 1, \ldots, T$. We also know that the fraction of 1 bits in the binary sequence is approximately 0.135. In addition, these 30,720 bits are converted from a text of understandable English.

We solve this problem by first using the fact that the fraction of 1 bits is only 0.135. Hence the probability of 00 is $(1 - 0.135)^2 \approx 0.75$. Thus the majority of points is on the function f_{00}. We then plot the whole data set $(u_1(k), u_2(k), y(k))$ in Figure 7.

We can see that most of the points are at a particular surface. In addition, for those (u_1, u_2) with the same distance to $(0,0)$, their function values are the same. In order to further analyze $f_{\mathsf{00}}(u_1, u_2)$, we define $d(k) \equiv \sqrt{u_1(k)^2 + u_2(k)^2}$ and draw a Figure 8 which shows the relation between $d(k)$ and $y(k)$. We can see a smooth function and some other points. Thus we tried to guess the analytic form of f_{00}. By drawing the following sinc function $\frac{\sin(2\pi d)}{\pi d}$ in dotted line, in Figure 8, we can see that they are nearly the same. Thus we conjecture that

$$f_{\mathsf{00}}(u_1, u_2) = \frac{\sin(2\pi d)}{\pi d} + g(u_1, u_2), \tag{6}$$

where $g(u_1, u_2)$ is an unknown smooth function. Thus we subtract all points with $\frac{\sin(2\pi d)}{\pi d}$ and draw a new Figure 9 which reveals more information about the function g. From this figure we observe that $|g(u_1, u_2)| \leq 0.1$,.

Then the problem is how to extract 00 points from the above information as we failed to guess the analytic form of $g(u_1, u_2)$. If we consider a small range of d, Figure 9 reduces to Figure 10. Clearly we can see that in a small region g is like a linear function. Thus we try a linear interpolation to find out 00 points. First we sort all data with $|y - \frac{\sin(2\pi d)}{\pi d}| < 0.1$ by their d. Then starting from data with smaller d, we gradually include points into the 00 category. To be more precise, for the kth data, we consider the previous five points which have been thought as 00 and find out the medium of their y values. Then for the next five points after k in the sorted list, we also find their medium of y. If the line passing these two mediums has the value within $\pm 10^{-4}$ of $y(k)$, we consider that the kth point should be 00.

We then use a heuristic to confirm some 0 bits. As the binary sequence is from an English text, the first bit of each character's 8 bit representation should be zero.

Now we know all 00 points. If we represent unknown bits as ?, data becomes the following sequence:
...00???00??0000?00...
The 0? is not 00 so we know it is 01. Similarly ?0 is not 00 so we know it is 10. Then the sequence becomes:
...001??100110000100...
Therefore, we have some 01 and 10 points. Now the characters in the beginning of the English text is as follows:
A C ? ? ? ? ? H ? ? ? ? ? ? ? ? ? H ? D ? ? ? ? F A B ? I ? H ? L ? L I ? ? ? ? ? ? ? ? L A ? ? ? D
We represent ? as an unknown character.

Using these 01 points we do support vector regression to find a surface. Then points near this surface might be 01. Similarly we can use the same procedure to identify 10 points. Here we use $\epsilon = 0, \gamma = 20$ and $C = 1$ for the support vector regression. A large γ means that we try to overfit the training data. Therefore, it is like that we are trying to do a surface fitting. We then assign points whose values are in ± 0.005 of the decision surface to the same category. However, when we sequentially add points to 01 and 10 categories, they should not violate existing patterns. For example, if we already know a pattern ?1 but its y is very close to the surface of 10, we will not change it to 10. A reason is that some points may be on the intersections of some of these four functions.

Now the first sentence becomes:
A C R ? S S T H E S T R E E T T H E D ? ? R ? F A B R I ? H T L Y L I T S T ? R E S L A M M E D
Finally the remaining points are likely to be 11. Then the first sentence becomes:
A C R O S S T H E S T R E E T T H E D O O R O F A B R I G H T L Y L I T S T O R E S L A M M E D

Before doing the final correction, we redo support vector regression on data in four categories. Using $\gamma = 20, \epsilon = 0, C = 100$, we obtain more fitted surfaces.

The final correction is as follows. For any bit x, if the sequence obtained so far is $\ldots axb \ldots$ but the u value at ax or xb is not very close to values of f_{ax} or f_{xb}, we think that x might be wrong. We select points which are not in the range of ± 0.01. Then we try to change x to its complement $\bar{x}$. If u values are now in the range of both $f_{a\bar{x}}$ and $f_{\bar{x}b}$, $\bar{x}$ should be better than x. For this process, only six bits are updated.

The four functions we obtained are in Figures 11, 12, 13, and 14. Their combination is the original data in Figure 7. By removing unnecessary spaces, we obtain the final draft. For example, the first sentence is
ACROSS THE STREET THE DOOR OF A BRIGHTLY LIT STORE SLAMMED
By searching for the web, we also realize the text is part of “The heart of a dog” by Mikhail Bulgakov. The whole procedure for solving this problem is by a Python script. The software used for support vector regression is also LIBSVM.

References

[1] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[2] V. Vapnik. *Statistical Learning Theory.* Wiley, New York, NY, 1998.