OPEN ACCESS Freely available online

PLOS ONE

A Voting-Based Sequential Pattern Recognition Method

Koichi Ogawara[1]*, Masahiro Fukutomi[2], Seiichi Uchida[3], Yaokai Feng[3]

1 Faculty of Systems Engineering, Wakayama University, Wakayama-shi, Wakayama, Japan, **2** Graduate School of Information Science and Electrical Engineering, Kyushu University, Nishi-ku, Fukuoka, Japan, **3** Faculty of Information Science and Electrical Engineering, Kyushu University, Nishi-ku, Fukuoka, Japan

Abstract

We propose a novel method for recognizing sequential patterns such as motion trajectory of biological objects (i.e., cells, organelle, protein molecules, etc.), human behavior motion, and meteorological data. In the proposed method, a local classifier is prepared for every point (or timing or frame) and then the whole pattern is recognized by majority voting of the recognition results of the local classifiers. The voting strategy has a strong benefit that even if an input pattern has a very large deviation from a prototype locally at several points, they do not severely influence the recognition result; they are treated just as several incorrect votes and thus will be neglected successfully through the majority voting. For regularizing the recognition result, we introduce partial-dependency to local classifiers. An important point is that this dependency is introduced to not only local classifiers at neighboring point pairs but also to those at distant point pairs. Although, the dependency makes the problem non-Markovian (i.e., higher-order Markovian), it can still be solved efficiently by using a graph cut algorithm with polynomial-order computations. The experimental results revealed that the proposed method can achieve better recognition accuracy while utilizing the above characteristics of the proposed method.

Citation: Ogawara K, Fukutomi M, Uchida S, Feng Y (2013) A Voting-Based Sequential Pattern Recognition Method. PLoS ONE 8(10): e76980. doi:10.1371/journal.pone.0076980

Editor: Rodrigo Huerta-Quintanilla, Cinvestav-Merida, Mexico

Received June 24, 2013; **Accepted** September 5, 2013; **Published** October 14, 2013

Funding: The authors have no support or funding to report.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: ogawara@sys.wakayama-u.ac.jp

Introduction

Recognition of sequential patterns, such as motion trajectory of biological objects (i.e., cells, organelle, protein molecules, etc.), human behavior motion, and meteorological data, is one of active research topics in pattern recognition. Traditional methods evaluate distance (or similarity) between an input pattern and a prototype by, for example, dynamic programming (DP) and Hidden Markov Models (HMM). The class label of the prototype with the minimum distance is determined as the class of the input pattern. It should be noted that, for recognizing a pattern whose length is N, the distance is generally defined as *accumulation* of N local distances, which are calculated individually at every point $v \in [1,N]$. We use the simple term "point" to express the address of each element of various sequential patterns. The point can be interpreted as the frame number for video and the time index for temporal patterns.

In this paper, we propose a novel method for recognizing sequential patterns, where a local classifier is prepared for every point and then the whole pattern is finally recognized based on majority voting of the class labels given by the local classifiers. Figure 1 illustrates this idea. When we consider a two-class recognition task, a two-class local classifier will be prepared at each point. Note that we can use an arbitrary classifier as the local classifier. Here, we will consider mainly a two-class problem, just for simplicity. We can deal with multi-class problems by, for example, combining the two-class classifier like SVM.

We can expect theoretically that the voting-based recognition framework of Figure 1 will have the following merits. First, if a very large deviation from a prototype happens at several points, its bad influence on the final recognition result will be negligible; this is because they are neglected as just a small number of incorrect votes among many correct votes. In contrast, the traditional distance-based methods will suffer from the large deviation. Second, by using different kinds of local classifiers at several points, the proposed method becomes a "heterogeneous" classifier and can create a complex classification boundary. Third, the proposed method can be used for some kind of application of recognizing a pattern without using the whole pattern. For example, we can realize early recognition [1,2], where the recognition result is determined at the beginning part of the pattern, just by voting the recognition results of the beginning part.

In the experimental result, it is shown that a simple majority voting method, where the votes from the local classifiers are totally independent, is not satisfactory. This is because the local classifier determines its recognition result just by observing a local part of the pattern and thus the result becomes unstable. If we have incorrect results due to the instability at many points, the final recognition result will be also incorrect even after voting.

In the proposed method, a dependency is introduced between several point pairs (u,v) for avoiding the instability. Specifically, we add a penalty when the local recognition results at a certain pair of points u and v are different. Hereafter, we call the penalty *smoothness term*. The smoothness term can be introduced between not only neighboring point pairs (i.e., $|u-v|=1$) but also some distant point pairs (i.e., $|u-v|>1$). This implies that some training is necessary to determine the point pairs where the smoothness term is to be introduced.

The proposed method is formulated as an optimal label assignment problem to determine the class label at every point $v \in [1,N]$. Let X_v denote the label variable at the point v and its value equals to 0 or 1 for two-class recognition. For the simplicity,

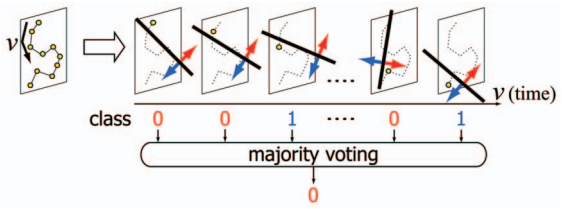


Figure 1. Sequential pattern recognition by majority voting.
doi:10.1371/journal.pone.0076980.g001

we call X_v the class label unless confusion arises. The optimized class labels $\{X_v|v=1,\ldots,N\}$ are voted to have the final result. The objective function of the label assignment problem is the sum of all the outputs of the local classifier and the smoothness term. Due to the smoothness term, it is impossible to determine the label class X_v independently. Moreover, the optimization problem becomes non-Markovian due to the smoothness term between distant point pairs and thus becomes intractable by the optimization methods (such as DP) which assume the Markovian property of the problem. Note that we call the popular first-order Markovian property Markovian and the higher-order Markovian property *non-Markovian* throughout this paper.

In this paper, we use a graph cut algorithm [3]. Graph cut can deal with a specific kind of optimization problem very efficiently, and thus has been used for image restoration [4,5], object recognition [6], shape matching [7,8], segmentation [8–12], etc. The graph cut algorithm can provide the globally optimal solution for our non-Markovian label assignment problem with few computations.

An important point of this paper is that the proposed method is a realization of *classifier ensemble* based on global optimization. When assembling outputs of local classifiers, we can introduce non-Markovian dependency as noted above. Furthermore, this dependency is controlled in a globally optimal manner. On this point, the proposed method is different from simple classifier ensemble methods [13], Boosting, and Bagging methods.

Methods

In this section, the proposed method is described in a two-step manner; first, a simplified version of the proposed method, called *independent voting*, is introduced. Then, a more sophisticated version, called *dependent voting*, is introduced.

Independent Voting

The independent voting method is a direct application of simple majority voting to sequential pattern recognition. Let $Y=y_1,\ldots,y_v,\ldots,y_N$ denote an input sequential pattern, where N is the length of Y and y_v is a d-dimensional feature vector.

In the independent voting method, a local classifier g_v is prepared for each point $v\in[1,N]$ in advance. Then for the input pattern Y, two-class recognition is performed at each v by g_v and then resulting N class labels are aggregated by the majority voting for the final recognition result. Without loss of generality, we hereafter assume that the local classifier g_v makes its classification by only using the single feature vector y_v. Any classifier, such as simple nearest neighbor classifier and quadratic classifier, can be used for the local classifier.

Figure 2 illustrates the independent voting method for $N=3$ and $d=1$. For the input pattern depicted as a "$\circ$" in this 3-dimensional pattern space, class labels are determined as 1, 0, and 1 at $v=1,2$, and 3, respectively, using local classifiers whose

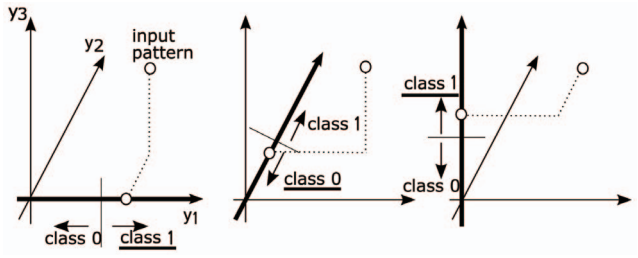


Figure 2. Recognition by the independent voting method. ($N=3, d=1$).
doi:10.1371/journal.pone.0076980.g002

classification boundary is shown on each axis. The input pattern is finally recognized as class 1 by majority voting.

The assignment of the class label for each point is formulated as a minimization problem as follows. Let $g_v(X_v)$ denote the cost of recognizing y_v as X_v by the local classifier g_v. (Precisely speaking, g_v should be denoted as $g_v(X_v|y_v)$, but we prefer the simpler notation $g_v(X_v)$.) The optimal label assignment problem is then considered as a minimization problem of the criterion $J(X)$ for the label assignment $X=(X_1,\ldots,X_v,\ldots,X_N)$

$$J(X)=\sum_{v=1}^{N} g_v(X_v). \tag{1}$$

Clearly, this minimization problem can be easily solved independently at each v, since there is no mutual dependency between two labels X_v and X_u ($u\neq v$).

The problem of the independent voting method is its instability. Since the local classifier g_v relies on the local feature y_v (i.e., a local part of Y), incorrect class labels might be assigned to some points. This instability becomes very crucial, especially when the difference between two classes is very slight and the number of incorrect votes tends to be close to the number of correct votes.

Dependent Voting

In the dependent voting method, partial dependency is newly introduced between some point pairs, for stabilizing (i.e., regularizing) the label assignment. This dependency is represented as a kind of penalty, called the smoothness term, for forcing X_u and X_v to have the same class labels. Roughly speaking, the smoothness term can eliminate short misrecognized segments. The effect of the smoothness term will be detailed later.

With the smoothness term, the criterion function (1) is extended as follows:

$$J(X)=\sum_{v=1}^{N} g_v(X_v)+\sum_{(u,v)\in E} h_{uv}(X_u,X_v), \tag{2}$$

where $h_{uv}(X_u,X_v)$ represents the smoothness term between u and v and E represents a set of point pairs having the smoothness term.

The smoothness term is designed to be larger for $X_u\neq X_v$ and smaller for $X_u=X_v$. The following equation is an example of the smoothness term:

$$h_{uv}(X_u,X_v)=\begin{cases} 0 & \text{if } X_u=X_v, \\ \theta_{uv}^{mn} & \text{otherwise,} \end{cases} \tag{3}$$

where m and $n\in\{0,1\}$ are the class labels assigned to the variables

X_u and X_v, respectively, and θ_{uv}^{01} and θ_{uv}^{10} are non-negative constants. Clearly, if θ_{uv}^{01} becomes larger, it is more difficult for u and v to have the labels 0 and 1, respectively. The value of θ can be determined automatically by some training algorithm (see Section "Training Smoothness Term") or manually.

We can introduce the smoothness term to not only neighboring point pairs (i.e., $E=\{(u,v) \mid |u-v|=1\}$) but also distant point pairs ($E=\{(u,v) \mid |u-v|>1\}$). The smoothness term for the distant point pairs is useful, for example, when the mutual dependency between the start point and the end point should be considered. Most traditional sequential pattern recognition methods, such as DP and HMM, cannot deal with such a distant dependency because the distant dependency violates the Markovian property of the problem. In the proposed method, graph cut is used instead of those traditional tools for dealing with the distant dependency, as detailed in Section "Globally Optimal Assignment Using Graph Cut".

Effect of Smoothness Terms

Figure 3 shows the classification boundary for $N=2$ and $d=1$. As the local classifier, we assume a simple nearest neighbor classifier whose prototype of class 0 is located at $(25,25)^T$ and that of class 1 is $(75,75)^T$. The cost $g_v(X_v)$ is defined by Euclidean distance as

$$g_v(0)=(y_v-25)^2,\ g_v(1)=(y_v-75)^2 \quad (v=1,2). \tag{4}$$

Figure 3 (a) shows the classification boundary by the independent voting, which is equivalent to the dependent voting with $\theta_{12}^{01}=\theta_{12}^{10}=0$. Figures 3 (b)–(e) show the boundaries by the dependent voting with different smoothness terms. These figures show that the larger the smoothness terms are, the smaller the region where the different class labels are assigned, and the larger the region in which the same class labels are assigned. Figure 3 (e) shows the dependent voting with $\theta_{12}^{01}=\theta_{12}^{10}=\infty$. Hereafter, this extreme case is called *fully-dependent voting*. Since no different label is allowed by the infinite smoothness term, all the class labels become the same. In addition, since it is equivalent to the traditional Euclidean distance-based method, that is, the nearest neighbor method, its classification boundary becomes a diagonal line.

Figure 4 shows the classification boundary of $N=3$, where the prototype of class 0 is located at $(25,25,25)^T$ and that of class 1 is $(75,75,75)^T$ and the recognition cost $g_v(X_v)$ is equal to that of $N=2$. Figures 4 (a), (b), and (c) show the classification boundaries by the independent voting, the dependent voting with a constant smoothness term, and the dependent voting with arbitrary smoothness terms, respectively. Like Figure 3, the smoothness term can control the shape of the classification boundary and a larger smoothness term results in a smoother boundary. Note that, in Figure 4 (b) and (c), positive smoothness terms are introduced not only to neighboring point pairs, but also to the distant point pairs, i.e., $h_{13}(X_1,X_3)$ and $h_{31}(X_3,X_1)$. Also note that, although these figures illustrate very low-dimensional cases, the same smoothing effect will appear in a low-dimensional subspace of a higher-dimensional feature space of more realistic sequential patterns.

Globally Optimal Assignment Using Graph Cut

Since a class label to be assigned to each point is binary, the total number of possible label assignment X is 2^N. Thus, for a larger N, it is impossible to find the globally optimal assignment by the naive exhaustive search. In addition, as already mentioned, it is also impossible to use DP because our problem does not hold Markovian property due to the smoothness terms for distant point pairs.

For our optimization problem, graph cut [3] will be one of the best choices. This is because the globally optimal solution of the criterion function (2) can be obtained with polynomial-order computations, even under the existence of the smoothness terms for distant point pairs. More precisely, if the smoothness terms satisfy the condition called sub-modularity,

$$h_{uv}(0,0)+h_{uv}(1,1) \le h_{uv}(0,1)+h_{uv}(1,0), \tag{5}$$

the globally optimal solution can be obtained and it is easily proved that the smoothness term of (3) satisfies this condition.

Training Smoothness Term

The remaining problem is how to design the smoothness terms. As emphasized above, the parameter value is very crucial factors for determining the classification boundary. Hopefully, those values are to be set up automatically for each application. Although several methods have been proposed for training the smoothness terms in some graph cut problems [4–6], no general method has been established yet. In this section, we propose a simple algorithm for training the smoothness terms. Hereafter, we assume a symmetric smoothness term, that is, $\theta_{uv}^{01}=\theta_{uv}^{10}=\theta_{uv}$.

Figure 5 shows the algorithm of training the smoothness terms. This algorithm is similar to an iterative error-correcting strategy for classical perceptrons. First, initial smoothness terms are determined by assigning the infinite smoothness term between points u,v whose local classifiers have recognition accuracies r_u and r_v being higher than a threshold. The best threshold achieving the highest recognition accuracy on the training samples is chosen among $\{0\%\sim100\%\}$. Note that if the threshold is 0%, the initial smoothness term is reduced to that of the fully-dependent method.

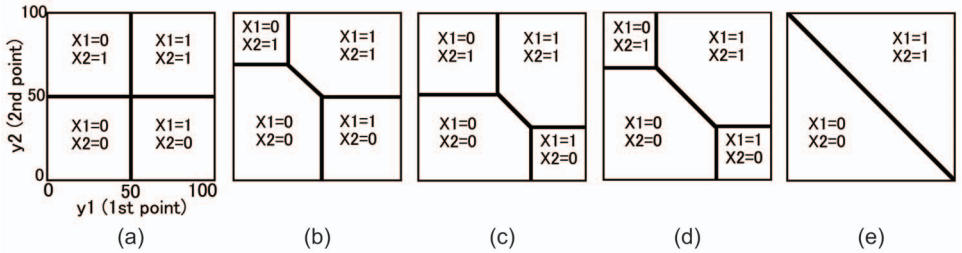


Figure 3. Example of classification boundary for $N=2$. (a) Independent voting ($\theta_{12}^{01}=\theta_{12}^{10}=0$), (b) $\theta_{12}^{01}>0$ and $\theta_{12}^{10}=0$, (c) $\theta_{12}^{01}=0$ and $\theta_{12}^{10}>0$, (d) $\theta_{12}^{01}>0$ and $\theta_{12}^{10}>0$, (e) Fully-dependent voting ($\theta_{12}^{01}=\theta_{12}^{10}=\infty$).
doi:10.1371/journal.pone.0076980.g003

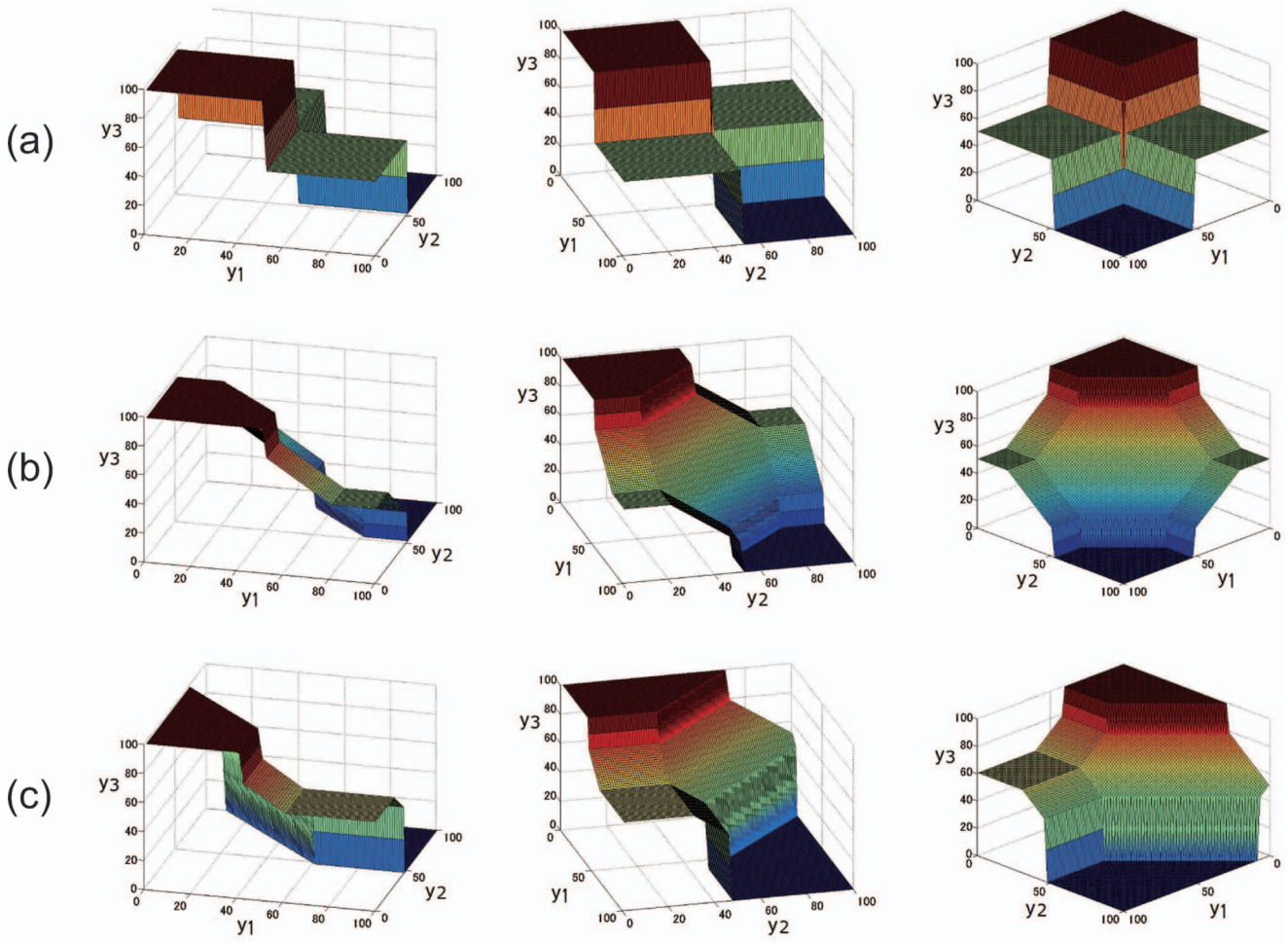


Figure 4. Example of classification boundary for $N=3$ and $d=1$ from three different view points. Different from Figure 3, the majority voting was already taken here. (a) Independent voting. (b) Dependent voting with $\theta_{uv}^{01}=\theta_{uv}^{10}$. (c) Dependent voting with $\theta_{uv}^{01}\neq\theta_{uv}^{10}$.
doi:10.1371/journal.pone.0076980.g004

Similarly, if the threshold is 100%, the initial smoothness term is reduced to that of the independent method.

Once the initial smoothness terms are determined, they are further optimized iteratively. In the optimization step, the zero smoothness term of a certain point pair is tentatively changed to the infinite smoothness term. Then, for evaluating this change, the recognition accuracy of the training samples is measured. If the accuracy is improved by the change, this change is accepted. Otherwise the change is canceled. This procedure is repeated at point pairs with zero smoothness term until no point pair improves the recognition accuracy.

1. For all u, v,

$$\theta_{uv} := \begin{cases} \infty & \text{if } r_u \geq \text{threshold and } r_v \geq \text{threshold} \\ 0 & \text{otherwise} \end{cases}$$

where threshold is chosen from {0% ~ 100%} for achieving the highest recognition accuracy, R^*, of all the training samples.

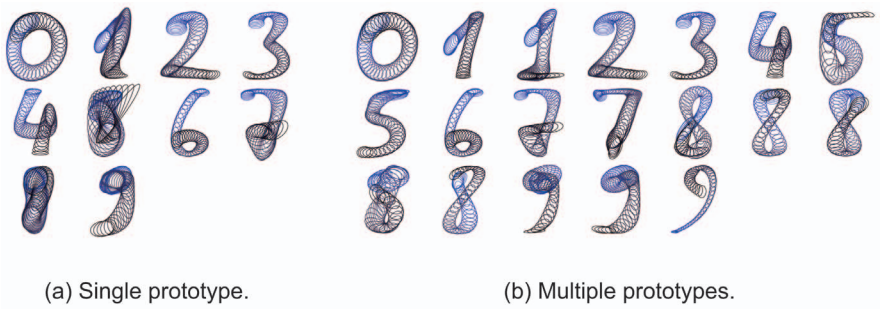
2. Repeat 3-6 until convergence.
3. Select a point pair (u, v) where $\theta_{u,v} = 0$.
4. $\theta_{u,v} := \infty$.
5. Calculate the new accuracy, R, of all the training samples.
6. If $R > R^*$ then $R^* := R$; otherwise, $\theta_{u,v} := 0$.

Figure 5. Algorithm for training smoothness terms.
doi:10.1371/journal.pone.0076980.g005

Results and Discussion

Experimental Setup

The proposed method was applied to a two-class online character recognition task for observing its performance, qualitatively and quantitatively because public datasets for rigorous evaluation are available. In online character recognition, each character pattern is represented as a temporal sequence of pen-tip position. That is, it is a kind of sequential pattern recognition and totally different from image-based character recognition (so-called OCR). In other words, the results obtained by the following online character recognition experiment will provide a good reference to any sequential pattern recognition, such as biological motion understanding, by the proposed voting-based method. Ethem Alpaydin Digit dataset (ftp://ftp.ics.uci.edu/pub/machine-learning-databases/pendigits/) that contains about 10,000 online isolated handwritten digit ("0"–"9") samples was used in the experiment. For each of the 10 classes, about 700 patterns are used as training samples, and the remaining 300 patterns are used as



(a) Single prototype. (b) Multiple prototypes.

Figure 6. Prototypes used in the experiment. Each ellipsoid illustrates the covariance for Mahalanobis distance at each point.
doi:10.1371/journal.pone.0076980.g006

test samples. As preprocessing, linear size normalization and resampling were applied. Each point was simply represented as an $x-y$ coordinate feature vector, i.e., $d=2$. The number of points N was fixed at 49 by resampling. (An odd number was better to avoid tie break of majority voting.) The local classifier was based on Mahalanobis distance, whose parameters were trained by using the training samples. In other words, the prototype of each class was represented statistically as a sequence of N Gaussian distributions. The average recognition rate was evaluated for all 45 two-class tasks of 10 digit classes.

In the later experiment, multiple prototypes were mainly used. The reason is shown in Figure 6. In the case of single prototype (a), it is observed that too large covariances were often estimated for several digits (e.g., "5" and "8"). This is simply because those digits have variations in writing order and direction. In contrast, in the case of multiple prototypes (b), covariances were estimated reasonably. The multiple prototypes were selected by k-means. After $k=1$ for all classes, k-means was performed at individual classes. Then setting $k+1\rightarrow k$ for the class with the largest clustering cost (i.e., the largest quantization error), k-means was performed again. This iteration stops if the clustering cost becomes lower than a fixed threshold at all the classes.

In the multiple prototype scenario, pre-selection of the closest prototype is performed by Mahalanobis distance. This distance is the same as the distance given by the "fully-dependent voting method", which will be introduced later. For example, for two-class recognition of "5"–"8", the closest prototype of "5" is selected among 2 prototypes of "5" for each test sample and also that of "8" among 5 prototypes of "8". Note that a two-class classifier was prepared for all the pairs of prototypes from different classes. Consequently, the total number of the classifiers are more than 45 even for 45 two-class tasks.

The computation time for recognizing all test samples at all 45 two-class tasks (i.e., for $300(\text{samples})\times 2(\text{class})\times 45(\text{classpairs})=2.7\times 10^4$ label assignments) was less than 68 seconds by using a notebook computer (CPU Intel Core i7-2960XM 2.70 GHz, Memory 16GB). Therefore, the computation time of the proposed method was about 2.5 ms. This speed is very promising because our optimal label assignment problem is non-Markovian and thus linear time algorithms such as DP optimization are not applicable.

Quantitative Evaluation

Table 1 shows the average recognition rate of the 45 two-class tasks under different conditions on the smoothness term. The fully-dependent voting method, which was introduced in Section "Effect of Smoothness Terms," is completely identical to the conventional distance-based recognition method where the discrimination is done by the comparison between $\sum_v g_v(0)$ and $\sum_v g_v(1)$. In fact, if $\sum_v g_v(0)<\sum_v g_v(1)$, all the labels $\{X_v\}$ become 0 and then the class 0 wins. In the case of "neighbor-dependent", the constant smoothness term h_0 was assigned to neighboring point pairs and zero smoothness term was assigned to the others. The case "optimally-dependent" is our main case where the smoothness terms between not only neighboring pairs but also distant point pairs were trained by the algorithm in Figure 5.

The result shows the superiority of the optimally-dependent voting method over the other methods. First, the optimally-dependent voting (98.53%) was much better than the independent voting (96.63%). This indicates the importance of the smoothness term for voting-based sequential pattern recognition. Although the smoothness term is important, it should be trained appropriately. This fact is indicated by comparing the distance-based method ("fully-dependent") to voting-based methods; among the three voting-based methods, only the optimally-dependent voting method (98.53%) was better than the fully-dependent voting method (98.12%).

It should be emphasized that the difference between "optimally-dependent" and "fully-dependent", (i.e., 98.53% and 98.12%) is not trivial. Specifically, by using the optimally-dependent voting method, we could reduce misrecognitions reduced to 78%(= 1.47/1.88). As shown in Table 1, this improvement can also be observed in the single prototype scenario, whose performance was much poorer than the multiple prototype scenario.

Table 1. Average recognition rates (%) for 45 two-class tasks.

prototype	independent	neighbor-independent	fully-dependent*	optimally-dependent
multiple	96.63	97.89	98.12	**98.53**
single	92.67	94.72	94.38	**95.95**

*Equivalent to the conventional distance-based method.
doi:10.1371/journal.pone.0076980.t001

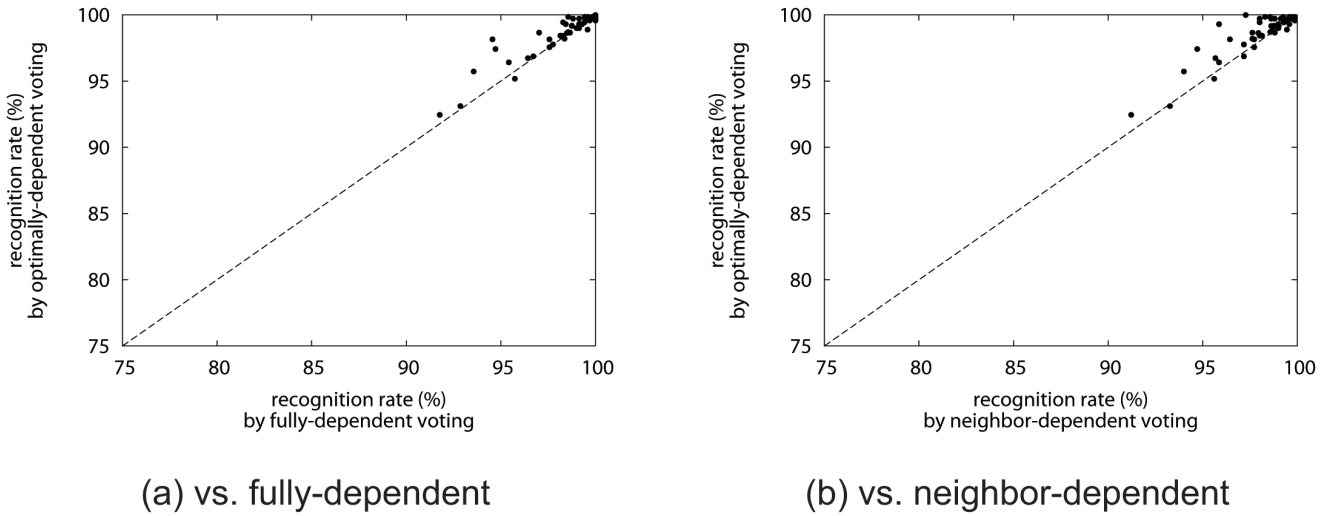


Figure 7. Recognition results of 45 two-class tasks by the optimally-dependent v.s. fully-dependent and neighbor-dependent voting methods.
doi:10.1371/journal.pone.0076980.g007

Figure 7 (a) compares the optimally-dependent voting method with the fully-dependent voting method. The figure proves that the optimally-dependent voting method outperformed in most two-class tasks. Specifically, among 45 two-class tasks, 25 tasks got better, 11 stayed the same, and 8 got worse. The class pairs with the most improved and degraded samples were "7"–"9" (improvement at 25 samples) and "0"–"7" (degradation at 5 samples). It is interesting to note that the top improved pairs are similar-shaped pairs, such as "7"–"9" (25 samples), "0"–"6" (19), "1"–"9" (15), "3"–"9" (11) and "2"–"3" (9). On the other hands, major degradations caused at "0"–"7" (5), "1"–"7" (4) and "2"–"5" (3). The reason of those improvement and degradation will be examined later. Figure 7 (b) compares the optimally-dependent voting method with the neighbor-dependent voting method and shows a similar trend over the 45 two-class tasks.

To evaluate the generalization ability of the proposed method against a small amount of training samples, the same experiment in the multiple prototype scenario was performed by exchanging the training and test samples, i.e., 300 samples for training and 700 samples for test. The average recognition rate of the 45 two-class tasks for the optimally-dependent voting method was slightly degraded from 98.53% to 97.56%, however it still achieved the best recognition rate among the four voting methods.

Qualitative evaluation

It is interesting to observe how the smoothness terms are trained automatically in the optimally-dependent method. The observation revealed the fact that non-zero smoothness terms do not exist randomly but exist around the parts where shapes of two character patterns are very different and not confusing. This will be because most of local classifiers g_v on a non-confusing part will provide correct recognition results and thus the smoothness terms around the part are effective to eliminate the remaining minor incorrect results. In contrast, local classifiers of a confusing part will not show a strong majority of a certain class; for example, about half of local classifiers g_v may provide incorrect label. If the smoothness terms are introduced to this confusing part, they have a risk that the correct labels are replaced by the incorrect label.

Figure 8 (a) visualizes the smoothness terms trained for "0"–"6" by the optimally-dependent method. In this figure, a dot at (u,v) indicates that the point pair (u,v) has a non-zero smoothness term. There are many non-zero smoothness terms around the beginning

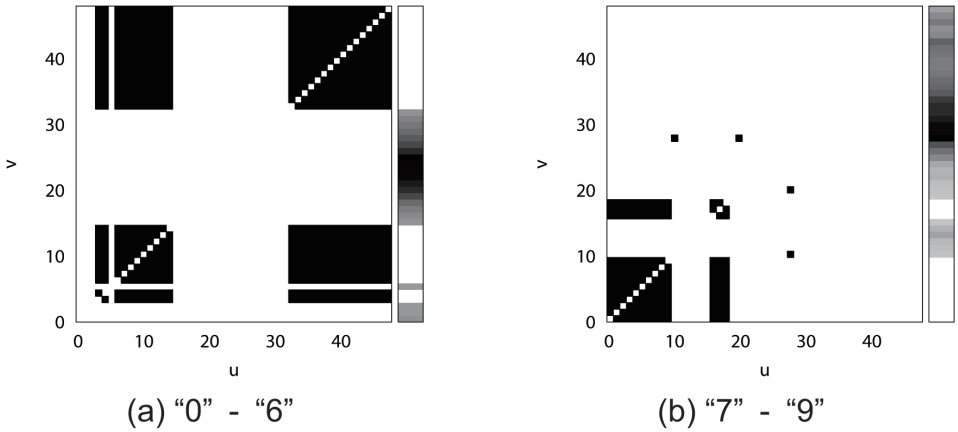


Figure 8. Training results of smoothness terms.
doi:10.1371/journal.pone.0076980.g008

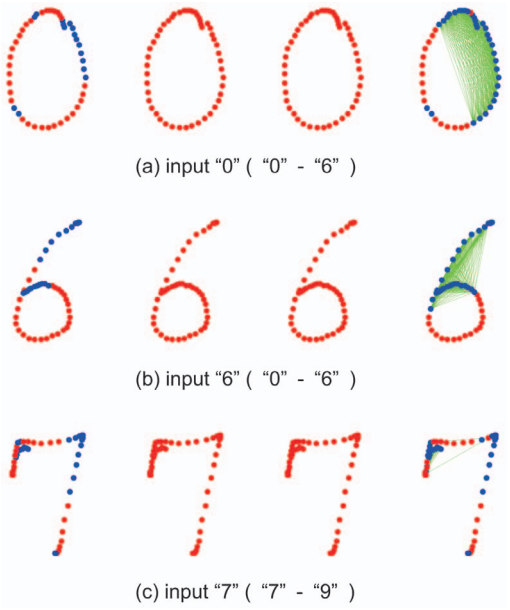


Figure 9. The effect of the distant smoothness terms. From left to right, results by independent voting, neighbor-dependent voting, fully-dependent voting, and optimally-dependent voting methods.
doi:10.1371/journal.pone.0076980.g009

part and the ending part. This can be understood from the fact that the shapes of "0" and "6" are very different in their beginning and ending parts and those parts are better to be forced to have the same label by the smoothness term. In contrast, their middle parts are confusing and thus have a risk that the local classifiers $g_v(X_v)$ around the part provide many misrecognitions. Since there is no smoothness term around this confusing part, the number of incorrect labels will not increase.

Figure 8 (b) visualizes the smoothness terms trained for "7"–"9" by the optimally-dependent method. Their very beginning parts have the non-zero smoothness terms because the parts are dissimilar. The large remaining parts (vertical stroke parts) have no smoothness term because those parts are confusing. Consequently, the smoothness terms for "7"–"9" are rather sparse than "0"–"6".

This tendency can be observed from several examples at the rightmost patterns of Figure 9. In this figure, the points with correct labels are depicted by blue circles and those with incorrect labels are by red circles. A point pair with a non-zero smoothness term is connected by a link (green line). The first example, Figure 9 (a), shows the case of recognizing a sample of "0" in "0"–"6" classification task. As shown in Figure 8 (a), there are many non-zero smoothness terms around the beginning and ending parts and no term around the middle part. Without the smoothness term (i.e., the result of the independent method), its confusing middle part is totally misrecognized as "6". Since this middle part is large, the fully-dependent method results in misrecognition. In contrast, the optimally-dependent method successfully utilizes the correct local recognitions around the beginning and the ending part. The correct recognitions in those parts were locally propagated by the trained smoothness terms, and finally they became the majority. Figure 9 (b) shows the case of recognizing a sample of "6" in the same "0"–"6" classification task. Like Figure 9 (a), the smoothness terms worked for the correct recognition.

Figure 9 (c) shows a result of recognizing a sample of "7" in "7"–"9" classification task. As we observed in Figure 8 (b), this case has a set of non-zero smoothness terms at the beginning part. These smoothness terms forced the local classifiers at the beginning part to have the same recognition result and the correct local classifiers became the majority. Meanwhile, other methods such as the independent voting method, neighbor-dependent voting method and the fully dependent voting method failed in this case.

Conclusions

In this paper, we have introduced the majority voting strategy to sequential pattern recognition and applied it to online character recognition. Specifically speaking, a two-class classification is done at each point $v \in [1,N]$ by using a local classifier and the recognition result of the entire pattern is determined by using majority voting of results of local classifiers at all points. This can be formulated as an optimal class label assignment problem where the class labels at all points, $X_1,\ldots,X_v,\ldots,X_N$ ($X_v \in \{0,1\}$), are considered as control variables and the sum of classification costs by local classifiers is considered as a criterion function to be minimized.

We also introduced partial dependency to the local classifiers. If this partial dependency is introduced to a pair of classifiers at u and v, a penalty, called smoothness term, is imposed if those local classifiers have different class labels X_u and X_v. Consequently, the label X_v cannot be determined at each v independently; the problem becomes a constrained optimization problem. Moreover, since the smoothness term is introduced to not only neighboring point pairs but also distant point pairs, the traditional DP optimization, which is often used for sequential pattern recognition problems with the Markovian property, cannot be used in our problem. Thus, we have employed graph cut as a more suitable optimization method; using graph cut, we can have the globally optimal solution with polynomial-order computation.

We have the following future work. First, we consider another and more sophisticated method for training the smoothness terms. For example, the steepest decent method may be effective for updating the smoothness terms in a direction of decreasing misrecognition. Second, we extend the proposed method to deal with multi-class problems. Instead of combining two-class classifiers (e.g., 1-vs-others approach), it can be realized by using recent multi-label graph cut methods [14–16]. Third, we must deal with nonlinear time warping because it is a very common distortion in sequential patterns. Traditionally, the warping problem is solved by DP; however, we can no longer use DP under the smoothness terms between distant point pairs. Fortunately, in [7], it has been shown that graph cut is promising for the warping problem. Thus it is worth trying to combine the label assignment problem and the time warping problem, and then solve it by graph cut. Fourth, the method should be evaluated on a larger-class problem, which will reveal more non-Markovian dependencies between various pairs of characters with different shapes.

Author Contributions

Conceived and designed the experiments: KO MF SU YF. Performed the experiments: KO MF. Analyzed the data: KO MF SU YF. Contributed reagents/materials/analysis tools: KO MF SU YF. Wrote the paper: KO MF SU YF.

References

1. Uchida S, Amamoto K (2008) Early recognition of sequential patterns by classifier combination. In: Proc. ICPR, ThAT4.6.
2. Ishiguro K, Sawada H, Sakano H (2010) Multi-class boosting for early classification of sequences. In: Proc. BMVC. 24.1–24.10.
3. Kolmogorov V, Zabih R (2004) What energy functions can be minimized via graph cuts? IEEE Trans on PAMI 26: 147–159.
4. Kumar S, Hebert M (2003) Discriminative fields for modeling spatial dependencies in natural images. In: Proc. NIPS.
5. Cremers D, Grady L (2006) Statistical priors for efficient combinatorial optimization via graph cuts. In: Proc. ECCV. 263–274.
6. Anguelov D, Taskar B, Chatalbashev V, Koller D, Gupta D, et al. (2005) Discriminative learning of markov random fields for segmentation of 3d scan data. In: Proc. CVPR. 169–176.
7. Schmidt FR, Toppe E, Cremers D, Boykov Y (2007) Efficient shape matching via graph cuts. In: Proc. CVPR. 1–6.
8. Schmidt FR, Toppe E, Cremers D (2009) Efficient planar graph cuts with applications in computer vision. In: Proc. CVPR. 1–6.
9. Boykov Y, Jolly MP (2001) Interactive graph cuts for optimal boundary region segmentation of objects in n-d images. In: Proc. ICCV. 105–112.
10. Price B, Morse B, Cohen S (2010) Geodesic graph cut for interactive image segmentation. In: Proc. CVPR. 1–8.
11. Schoenemann T, Kahl F, Cremers D (2009) Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In: Proc. ICCV. 17–23.
12. Liu X, Veksler O, Samarabandu J (2008) Graph cut with ordering constraints on labels and its applications. In: Proc. CVPR. 1–8.
13. Kittler J, Hatef M, Duin R, Matas J (1998) On combining classifiers. IEEE Trans on PAMI 20: 226–239.
14. Boykov Y, Veksler O, Zabin R (2001) Fast approximate energy minimization via graph cuts. IEEE Trans on PAMI 23: 1222–1239.
15. Schlesinger D, Flach B (2006) Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-FI06-01, Dresden University of Technology.
16. Kolmogorov V, Rother C (2007) Minimizing non-submodular functions with graph cuts - a review. IEEE Trans on Pattern Analysis and Machine Intelligence 29: 1274–1279.