Relative Pattern Recognition for Noisy Handwritten Numeral Recognition

Hansheng Lei

Computer and Information Sciences department, the Univ of Texas at Brownsville
Brownsville, TX 78520
hansheng.lei@utb.edu

Venu Govindaraju

CUBS, Center for Unified Biometrics and Sensors, State University of New York at Buffalo
Amherst, NY 14260
govind@buffalo.edu

Abstract

A novel framework for pattern recognition, namely Relative Pattern Recognition (RPR), is presented in this paper. RPR is based on the relative feature representation/extraction. While traditional feature extraction determines features to discriminate all the classes in a given domain, RPR extracts relative features to separate only two classes at a time. The feature determination process becomes less costly in practice. To demonstrate the feasibility of RPR, an implement method of RPR using Recursive Feature Elimination in Support Vector Machines (SVMs) is described. Experimental results on public datasets, such as Iris, Isolet and MNIST show that RPR is effective and efficient with advantages over standard SVM. In addition, simulated experimental results show that RPR has robustness in partial and strong noisy handwritten numeral recognition.

Keywords: Relative Feature Representation, Relative Feature Extraction, Relative Pattern Recognition, Recursive Feature Elimination

1. Introduction

Pattern recognition (PR) remains one of the most classic and core research problems in computer science, despite the emergence of "new" research areas such as data mining. A standard PR process has three major steps: i) preprocessing, ii)feature representation/extraction, and iii) classifier learning. To extract discriminative features, handcrafted experiments have to be conducted with expert knowledge. Given numerous existing tools for classifier learning (such as Support Vector Machines, Artificial Neural Networks, etc.), the most costly part in developing a PR system lies in the feature extraction step.

Feature extraction is essentially a function $f(x)$, which maps the preprocessed data point x to a feature space. With an ideal mapping function, the mapped points in the feature space are close to each other if they are from the same class and far away from each other if from different classes. However, many PR problems, such as handwriting recognition, face recognition and shape recognition, are multi-class. To determine such a mapping function that discriminates one class from *all other* classes is expensive in practice. We can call features extracted in this traditionally way *absolute features*. In other side, a multi-class classifier can be built on the combination of binary classifiers as in state-of-the-art Support Vector Machines (SVMs). Instead of extracting the absolute features, why not extract the *relative features* which distinguish pairwise classes? It is easier to extract relative features than absolute features because the former involves only two classes.

Feature extraction implicitly assumes some kind of feature representation. Vector and structural string are two of the most common feature representation forms [10]. In vector representation, the features are extracted uniformly and their length are same. Structural representation is usually utilized for sequential patterns where the length varies. In examining the history of pattern recognition research in the past decades, one will see that novel feature representation leads to radical novelty in classifier learning. The feature presentation in vector space had been broadly applied and implicitly assumed in Statistic pattern recognition [15]. The structural feature representation was initialized in the early 1980s which eventually led to the birth of the burgeoning subfield of Syntactic pattern recognition [3].

In the paper, we present an innovative feature representation and feature extraction method, namely, *Relative Feature Representation/Extraction*. The pattern recognition based on relative features is called Relative Pattern Recognition (RPR) accordingly. To illustrate the basic idea, consider a white apple (denoted *A*), a red apple (*B*)

and a while pear (*C*). To distinguish objects *A* and *B*, the color is a discriminate feature. But for *A* and *C*, the shape is the feature while the color is not. That means, the features are always relative. RPR is built on the relativity of features as shown in Figure 1. The benefits of RPR include: i) faster evaluation, since only the discriminative features are used; ii) possible partial recognition. E.g., even if the objects *A* and *B* are partially visible, the color is sufficient to distinguish them; iii) higher scalability. The feature extraction process is independently on each pair of classes, which enables the independent change/updating on relatives features when objects evolve or the domain expands.

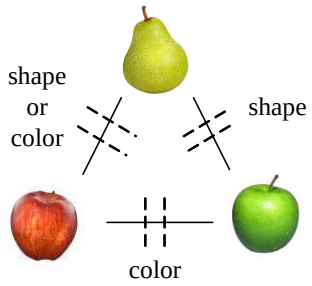

Figure 1. Features are relative. To separate a red apple and a white apple, color is the feature; to separate a white apple and a white pearl, the shape is the feature. To distinguish the white pearl and the red apple, either shape or color is sufficient.

The rest of paper is organized as follows: in Section 2, a general framework of RPR training and evaluation is presented. In Section 3, an implementation method of RPR using Recursive Feature Elimination (RFE) and SVM is described. The application of RPR in partial object recognition is also discussed. Experimental results are reported in Section 4. Finally, Section 5 draws conclusions.

2. Relative Pattern Recognition

To extract relative features, efforts are focused on each pair of classes independent of other pairs. The conventional methods which are used to determine features are applicable here. Without the need to extract features which distinguish one class from all other classes, the relative feature determination task becomes less costly practice. In addition, the feature determination and extraction are inherently parallel, which enables higher scalability than regular feature extraction.

Given a training set $X = \{x_1, x_2, ..., x_n\}$ and associated labels $Y = \{y_1, y_2, ..., y_n\}$ [1] , the RPR works as follows:

[1] we only consider supervised classification here

Algorithm 1. RPR framework - Training

Inputs: Training samples: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, class labels: $\mathbf{Y} = [y_1, y_2, \cdots, y_n]$ and the number of classes: M.

Outputs: One-Vs-One Classifier Model: OVO_{ij}; Feature extracting functions: F_{ij} ($i = 1, 2, ..., M$, $j = i+1, i+2, ..., M$).

1: **for** i=1 to M **do**
2: **for** j=i+1 to M **do**
3: $C_i = \mathbf{X}(:, find(\mathbf{Y} == i))$; /*data of class i*/
4: $C_j = \mathbf{X}(:, find(\mathbf{Y} == j))$; /*data of class j*/
5: $F_{ij} \leftarrow$ Feature determination for class C_i and C_j;
6: $OVO_{ij} \leftarrow$ Training binary classifier on C_i and C_j ;
7: **end for**
8: **end for**/*end of algorithm*/

The RPR framework is simple and very generic. In line 5, any feature determination process (either manual work with prior knowledge, semi-automatic with interactive experiments or automatic feature mining) can be engaged. Once feature determination method is formed, it becomes a feature mapping function. Each pair of classes will have a feature mapping function (denoted as F_{ij}) independent of one another. In line 6, any classifier can be utilized as long as it works for binary case. All classification techniques distinguishes at least 2 classes, such as SVM, ANN, k-NN and decision trees.

Once all the feature mapping F_{ij} are determined and the OVO model is trained, the evaluation (testing) of RPR can be implemented as follows.

Algorithm 2. RPR framework - Evaluation

Input: Evaluation samples $\mathbf{X_o} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m]$; OVO Model OVO_{ij} and feature extracting functions F_{ij} ($i = 1, 2, ..., M$, $j = i+1, i+2, ..., M$).

Output: Labels $\mathbf{Y_o} = [y_1, y_2, \cdots, y_m]$.

1: **for** $k = 1$ to m **do**
2: $\mathbf{x} = \mathbf{X_o}(:, k)$; /*one test sample*/
3: Votes=zeros(1,M); /*votes for each class*/
4: **for** $i = 1$ to M **do**
5: **for** $j = i+1$ to M **do**
6: $\mathbf{x}'$=$F_{ij}(\mathbf{x})$; /*Relative feature extraction*/
7: $Label$=$\leftarrow$ $OVO_{ij}(\mathbf{x}')$; /* binary classifier evaluation*/
8: **if** $Label == i$ **then**
9: $Votes(i)$=$Votes(i)$+1;
10: **else**
11: $Votes(j)$=$Votes(j)$+1;
12: **end if**
13: **end for**
14: **end for**

15: $\mathbf{Y_o}(k) = find(Votes == max(Votes))$; /*the one with max votes wins*/
16: **end for**/*end of algorithm*/

The RPR evaluation is implemented by combining the votes of every binary classifier. The one with maximum votes wins. This idea has been applied extensively in multi-class SVMs. RPR can be implemented seamlessly in the framework of One-Vs-One (OVO) multi-class SVMs if coupled with Recursive Feature Selection (RFE) [5]. Before we describe the implementation method of RPR using RFE and SVM in Section 3, we briefly analyze the performance of RPR.

2.1. RPR performance analysis

The overall performance RPR has direction relation with the performance of those binary classifiers. The total number of binary classifiers is $M(M-1)/2$ for a M-class problem. Suppose the performance of individual classifier is p_{ij} in term of accuracy (0%- 100%). Now consider the votes to class i, the votes come from $M-1$ classifiers: OVO_{ij}, $j = 1, ..., M, j \neq i$. Assuming the votes are evenly distributed to every class without the loss of generality, the average ratio of "correct" votes for class i is $\frac{\sum_{j \neq i} p_{ij}}{M-1}$. Overall, the correct ratio (performance) for RPR is $\frac{\sum_{i=1}^{M} \sum_{j \neq i} p_{ij}}{M(M-1)}$, which is the sum average of the performance of all individual classifiers.

3. SVM and Recursive Feature Selection

SVM was originally designed for binary classification problems and has been shown to yield state-of-the-art performance in many pattern analysis applications [1]. There are two types of approaches to extend binary SVM to multi-class case: (i) consider all data in a single optimization function [2] or (ii) decompose the multiple classes into a series of binary SVMs, such as "One-Verse-All" (OVA) [15] and "One-Verse-One" (OVO) [8]. Although there are variances of OVO [13] and other sophisticated approaches for multi-class SVM, OVO is one of the most suited method in practice [7, 14].

OVO is constructed by training binary SVMs between pairwise classes. Thus, OVO model consists of $\frac{M(M-1)}{2}$ binary SVMs for the M-class problem. Each of the $\frac{M(M-1)}{2}$ SVMs casts one vote for its favored class, and finally the class with most votes wins [8].

Feature selection in kernel machines has been extensively addressed [4]. Among many feature selection methods, RFE directly utilizes the training results of SVM and has been successfully applied in applications such as gene selection[5].

RFE is an iterative procedure to remove non-discriminative features [5] in binary classification. The framework of RFE consists of the following steps: 1) Train the classifier; 2) Rank the features based on their contribution to classification and 3) Remove the feature with the lowest ranking. Goto step 1) until the desired number of features is reached. The following is pseudo code of RFE using SVM.

Algorithm 3. SVM-RFE
Inputs: Training samples $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$ and class labels $\mathbf{Y} = [y_1, y_2, \cdots, y_n]$.
Outputs: feature ranked list $\mathbf{r}$.

1: $\mathbf{s} = [1, 2, \cdots, d]$; /*surviving features*/
2: $\mathbf{r} = []$; /*feature ranking list*/
3: **while** $\mathbf{s} \neq []$ **do**
4: $\mathbf{X} = \mathbf{X}_0(\mathbf{s}, :)$; /*only use surviving features of every sample*/
5: Train linear SVM on $\mathbf{X}$ and $\mathbf{Y}$ and obtain $\mathbf{w}$;
6: $c_i = w_i^2, \forall i$ /*weight of the i-th feature in $\mathbf{s}$ */
7: $f = argmin_i(c_i)$; /*index of the lowest ranking*/
8: $\mathbf{r} = [\mathbf{s}(f), \mathbf{r}]$; /*update list*/
9: $\mathbf{s} = \mathbf{s}(1 : f-1, f+1 : length(\mathbf{s}))$; /*eliminate the lowest ranked feature*/
10: **end while**. /*end of algorithm*/

Note that the ranking criterion for the discriminability of features is based on $\mathbf{w}$, the normal vector of the separating hyper-plane (line 6). The idea here is that one a feature is discriminative if it significantly influences the width of the margin of the SVM, $\frac{2}{\|\mathbf{w}\|^2} = \frac{2}{\sum_{i=1}^{n} w_i^2}$.

RFE requires iterations. If we have d features and need to choose the top d' features, the number of iterations is $d - d'$ if only one feature is eliminated at a time. More than one feature can be removed at a time empirically by modifying line 7-9 in the algorithm above.

We can incorporate RFE into standard SVM (either linear or non-linear) to eliminate the non-discriminative features, which leads a way for relative feature extraction. We consider applications such as handwriting recognition and face detection where the features can be the pixels of the input image. Selecting the most discriminative pixels for recognition while maintaining the performance is the objective. This is feasible because not all features are discriminative. For example, to distinguish digit '4' and '9', the upper part of the digit (closed or open) is sufficient to tell them apart. Some features are discriminative for this pair of classes, but may be not useful for another pair. We do not have to use a fixed number of features for every binary SVM classifier. Since OVO is based on binary SVM, the technique of RFE for two-class problem is naturally applicable to multi-class.

3.1. Partial Object Recognition

Partial object recognition is inherently a hard problem in pattern recognition (PR). Compared to research works such as face recognition and handwriting recogni-

Figure 2. Handwritten digit '0', '9' and the selected 76 relative features that separate them.

tion, fewer works have addressed partial recognition. One common way for partial recognition is to define and extract the local invariant features [11]. However, the extraction method of such features is ad hoc.

Partial object recognition is possible using the SVM and RFE in the framework of RPR. Intuitively, RFE selects a portion of features for each binary classifier. The starting features can be just the pixels in the image. As an example, Figure 2 shows the relative features (pixels) to distinguish handwritten digit '0' and '9'. Given an input test image, the relative feature extraction in this case is to take the pixels in the positions as feature vector, indicated by the white pixels in the third image in Figure 2. Now, suppose a part of the input image is visually blocked. As long as the critical part is visible, the relative recognition will not be affected. Even some critical part is blocked, the remaining part can still have separation ability to some extent. It is true that the separation part for one pair of classes may not be the separation part of another pair. But the overall performance is balanced over all binary classifiers, as discussed in the section 2.1. Therefore, RPR provides robustness to a some degree in partial object recognition.

4. Experiments

The objective of the experiments is to evaluate the performance of RPR and its application in partial object recognition. Three datasets were used in our experiments as described in Table 1. Iris is a classical dataset for testing classification, available from the UCI KDD archives [6]. The second dataset, Isolet was generated from spoken letters [6]. The third dataset MNIST contains 10 classes of handwritten digits (0-9) [9]. There are 60,000 samples for training and 10,000 samples for testing. The digits have been normalized and centered in a fixed-size image (28×28). It is a benchmark database for machine learning and pattern recognition experimental studies.

The OSU SVM Classifier Matlab Toolbox [12] was chosen as the base classifier. On each dataset, we trained the multi-class OVO SVM with the RBF_kernel. The regularizing parameters C and σ were determined by cross validation on the training set. The C and σ that give the highest accuracy were selected. The RPR based on RFE and SVM was trained with exactly the same parameters

Table 1. The multi-class datasets used in the experiments.

Name	# of Training Samples	# of Testing Samples	# of Classes	# of Attributes
Iris	105	45	3	4
Isolet	6238	1559	26	617
MNIST	60000	10000	10	784

Table 2. The results of RPR on the Iris. F is the number of top ranked features chosen for each pair of classes. $C = 500$ and $\sigma = 200$ for both OVO SVM and RPR.

F	Accuracy (secs)	RFE	Training (secs)	Testing (secs)
4	100%	$\emptyset$	0	0
3	100%	0.016	0	0
2	100%	0.016	0	0
1	**100%**	0.016	0	0

(C and σ) and conditions as OVO SVM except that we varied one additional parameters F (the number of top ranked features). We compared the performance of standard OVO SVM and RPR in three aspects: classification accuracy, training and testing speed.

4.1 RPR using RFE and SVM

We used the RFE in standard OVO SVM to implement the Relative Pattern Recognition. In the experiments, we varied F (the number of selected features for each pair of classes). Since the feature elimination procedure required many iterations, we set the number of features eliminated each time empirically. The classification accuracy, RFE time, training time and testing time were recorded for each F. The experimental results are shown in Table 2, 3 and 4. The first row shows the results of standard OVO SVM where no feature is eliminated. The execution time of RFE was recorded separately from the regular SVM training time. The total training time of RPR is therefore the sum of RFE (the second column) and SVM training (the third column).

On the Iris dataset, only RFE required extra time in addition to regular SVM training, since the dataset is very small. It is clear that only one feature is sufficient to distinguish a pair of Iris classes ($F = 1$, the accuracy is still 100%), as shown in Table 2.

The results on the MNIST are shown in Table 3. The classification accuracy steadily decreased as F decreased, but not significantly. In addition, we can see that RFE is computationally expensive because of its iterations. To eliminate 684 features (let $F = 100$), the number of iterations was 34 which took 13.7 hours (49432 secs). However, the compensation is that RPR speeds up the testing phase since the relative feature vectors are shorter

Table 3. The results of RPR on the MNIST. F is the number of top ranked features chosen for each pair of classes. $C = 5$ and $\sigma = 20$ for both OVO SVM and RPR.

F	Accuracy	RFE (secs)	Training (secs)	Testing (secs)
784	97.90%	$\emptyset$	4471.6	**278.9**
500	97.87%	35306	3179.7	251.1
300	97.82%	45120	2535.3	243.6
200	97.74%	47280	1094.4	221.0
150	97.40%	49344	648.0	114.1
100	96.62%	**49432**	398.7	**110.6**
50	94.56%	49440	286.2	47.1
25	89.86%	49446	208.8	25.5
10	75.60%	52126	171.9	13.5

Table 4. The results of RPR on the Isolet. F is the number of top ranked features chosen for each pair of classes. $C = 10$ and $\sigma = 200$ for both OVO SVM and RPR.

F	Accuracy	RFE (secs)	Training (secs)	Testing (secs)
617	96.92%	$\emptyset$	249.4	36.7
400	96.98%	758.4	80.2	22.3
300	96.86%	868.6	50.0	15.7
200	96.60%	981.1	30.0	12.4
150	96.73 %	1065.3	23.6	11.5
100	96.28 %	1164.0	12.8	5.3
50	96.09 %	1284.8	4.6	3.1
25	95.2 %	1301.8	3.9	3.0
10	93.14 %	1314.7	2.6	2.8

than original feature vectors. When $F = 784$, i.e, no feature elimination, the test time is 278.9 secs; when $F = 100$, the testing time is 110.6 secs. The speedup ratio is $278.9/110.6 \simeq 2.5$. From the results on the Isolet as shown in Table 4, we have similar observations.

4.2. RPR for Partial Object Recognition

The MNIST dataset was used to evaluate RPR in partial object recognition. The test handwritten digits are randomly removed a $k \times k$ block to simulate partial objects. To remove a block in a image, we located a block first and then set all its pixel values to 1, which makes block visually white. Figure 3 (a) shows examples of partial digit artificially generated for the experiments.

Note that we only generated partial test digits instead of training digits. All the training digits were untouched, because in reality, the training samples can be collected and cleaned manually, but test samples are supposed to be processed blindly without knowing that they are partial are not.

We compared the classification accuracy using standard OVO SVM and RPR on the testing of partial digits by varying the size of block k . Figure 4 (a) shows the results. When k increases, that is, bigger part is invisible, both SVM and RPR degrades. However, the decreasing

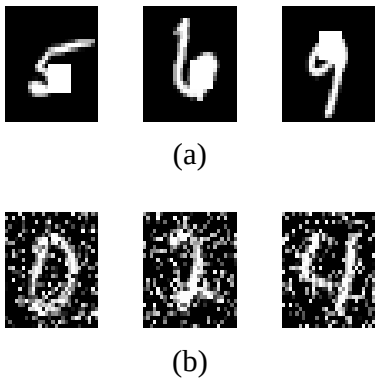

Figure 3. (a) Simulated partial handwritten digits; (b)Simulated noisy handwritten digits.

of performance by SVM is much faster than RPR. That demonstrates that the RPR is more robust than SVM in partial recognition. Of course, we can not say that RPR is totally invariant to partial objects because of the inherent incompleteness of some objects. The incompleteness of information just makes it is not possible to distinguish some fatally blocked digits. Figure 5 shows examples of miss recognition by both SVM and RPR. The bottom label is the genuine label of the digit, while the label on image's left is the miss- recognized label.

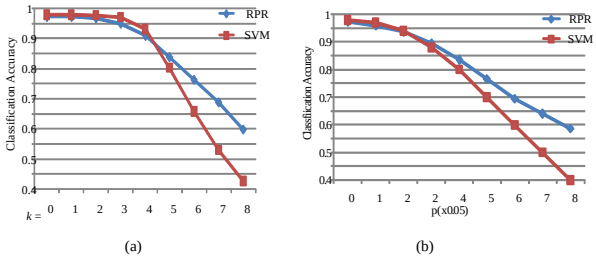

Figure 4. (a) The classification accuracy comparison between RPR and standard SVM on partial digits while k varies from 0 to 8; (b)Comparison on noisy digits while p varies from 0 to 0.40.

Experiments were also conducted to observe the robustness of RPR for strong noisy objects. Intuitively, if RPR works well for partial recognition, it should be able to work well too for noisy objects. The noisy part has some probability to be ignored by the relative feature extraction that only consider the difference between pairwise classes. To simulate noisy digits, we randomly set a portion of the pixels to 1. The portion is measured as p,which means $100 * p\%$ pixels are set to white. Figure 3 (a) shows examples of noisy digits. Again, only test samples were added noise.

The classification accuracy of RPR and OVO SVM

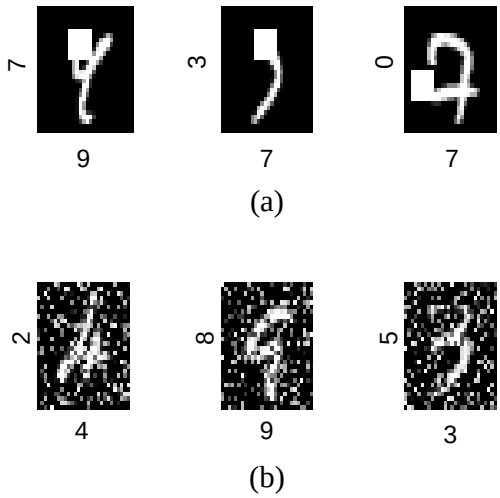

Figure 5. Partial and Noisy digits that are miss-recognized by both RPR and SVM. The bottom label is the genuine label of the digit, while the label on image's left is the miss- recognized label.

was compared with results as shown in Figure 4 (b). When p increases, RPR degrades, but in lower speed than OVO SVM. RPR demonstrates its robustness in noisy data to some degree.

5. Conclusion Remarks

Relative Pattern Recognition provides a general framework where current feature extraction approaches and learning techniques can still be utilized. It paves new potential ways for pattern recognition to face the challenges of costly feature determination, poor scalability and low robustness in the situation of partial/noisy object classification. The experiments results, though quite preliminary, are encouraging. RPR speeds up classification testing and also has robustness in partial/noisy object recognition.

The feature selection techniques play an important role in RPR. Though feature selection has been well researched, how many features among a given bunch of feature candidates are sufficient and necessary to distinguish two classes is still an open problem. In our experiment, we empirically set the size of feature subset, which is not optimal. It is necessary to address this issue in future work. In addition, we are aware that the experiments on partial objects were simulated. More experiments on partial 2D and 3D objects are needed to further validate the RPR in terms of scalability and robustness in partial object recognition.

References

[1] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

[2] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, vol. 2:265–292, 2001.

[3] L. Goldfarb. Pattern representation and the future of pattern recognition: a program for action. *ICPR 2004 satellite workshop, Cambridge, UK*, 2004.

[4] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research Special Issue on Variable and Feature Selection*, vol. 3:1157–1182, 2003.

[5] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, vol. 46:389–422, 2002.

[6] S. Hettich and S. Bay. The UCI KDD archive. *http://kdd.ics.uci.edu*, 1999.

[7] C. Hsu and C. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002.

[8] U. Kreßel. Pairwise classification and support vector machines. In *Advances in Kernel Methods: Support Vector Learnings*, pages 255–268, Cambridge, MA, 1999. MIT Press.

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, vol. 86:2278–2324, 1998.

[10] O. G. Lev Goldfarb and D. Korkin. What is a structural representation? *Faculty of Computer Science, U.N.B., Technical Report TR00-137*, 2000.

[11] D. G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, pages 1150–1157, 1999.

[12] J. Ma, Y. Zhao, and S. Ahalt. OSU svm classifier matlab toolbox. *http://www.kernel-machines.org/*, 2002.

[13] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In *Advances in Neural Information Processing Systems*, volume 12, pages 547–553, 2000.

[14] R. Rifin and A. Klautau. In defense of one vs-all-classification. *Journal of Machine Learning Research*, vol. 5:101–141, 2004.

[15] V. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.