Multi-Category Classification by Soft-Max Combination of Binary Classifiers

Kaibo Duan[1], S. Sathiya Keerthi[1], Wei Chu[1], Shirish Krishnaj Shevade[2], and Aun Neow Poo[1]

[1] Control Division, Department of Mechanical Engineering
National University of Singapore, Singapore 119260
{engp9286, mpessk, engp9354, mpepooan}@nus.edu.sg
[2] Department of Computer Science and Automation
Indian Institute of Science, Bangalore 560 012
shirish@csa.iisc.ernet.in

Abstract. In this paper, we propose a multi-category classification method that combines binary classifiers through soft-max function. Posteriori probabilities are also obtained. Both, one-versus-all and one-versus-one classifiers can be used in the combination. Empirical comparison shows that the proposed method is competitive with other implementations of one-versus-all and one-versus-one methods in terms of both classification accuracy and posteriori probability estimate.

1 Introduction

Multi-category classification is a central problem in machine learning. While binary classification methods are relatively well-developed, how to effectively extend these binary classification methods for multi-category classification is an important but still on-going research issue. For methods such as support vector machines (SVMs) [6] the most common approaches to multi-category classification are binary-classifier-based methods, such as "one-versus-all" (1va) and "one-versus-one" (1v1), that make direct use of binary classifiers to tackle multi-category classification tasks. For SVMs, some researchers also proposed "all-together" approaches [6] [7] that solve the multi-category classification problem in one step by considering all the examples from all classes together at once. However, the training speed of "all-together" methods is usually slow.

One-versus-all method is usually implemented using a "Winner-Takes-All" (WTA) strategy. It constructs M binary classifier models where M is the number of classess. The ith binary classifier is trained with all the examples from ith class ω_i with positive labels (typically $+1$), and the examples from all other classes with negative labels (typically -1). For an example $\mathbf{x}$, WTA strategy assigns it to the class with the largest decision function value.

One-versus-one method is usually implemented using a "Max-Wins" voting (MWV) strategy. This method constructs one binary classifier for every pair of distinct classes and so, all together it constructs $M(M-1)/2$ binary classifiers.

The binary classifier C_{ij} is trained with examples from ith class ω_i and jth class ω_j only, where examples from class ω_i take positive labels while examples from class ω_j take negative labels. For an example $\mathbf{x}$, if classifier C_{ij} says $\mathbf{x}$ is in class ω_i, then the vote for class ω_i is added by one. Otherwise, the vote for class ω_j is increased by one. After each of the $M(M-1)/2$ binary classifiers makes its vote, MWV strategy assigns $\mathbf{x}$ to the class with the largest number of votes.

For binary classifiers with probabilistic outputs, such as kernel logistic regression (kLOGREG) [5], a "pairwise coupling" (PWC) procedure was proposed in [2] to implement the one-versus-one method. The central idea is to couple the $M(M-1)/2$ pairwise class probability estimates to obtain estimates of posteriori probabilities for the M classes. PWC assigns an example $\mathbf{x}$ to the class with the largest posteriori probability among the M classes.

In this paper, we present a multi-category classification method by combining one-versus-all or one-versus-one binary classifiers, through a soft-max function. Posteriori probabilities obtained from the combination are used to do multi-category classification.

In section 2 we present various designs of soft-max combining functions for one-versus-all and one-versus-one binary classifiers. Practical implementation issues associated with these designs are given in section 3. Numerical experiments are given in section 4 where the proposed methods are compared with other implementations of one-versus-all and one-versus-one methods with different binary classification techniques. Finally, results are analyzed and conclusions are made in section 5.

2 Soft-Max Combination of Binary Classifiers

In this section, we present soft-max combination of one-versus-all and one-versus-one binary classifiers. The relation of the methods with some previous work is also briefly discussed.

2.1 Soft-Max Combination of One-Versus-All Classifiers

Suppose there are M classes and l labelled training data $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l)$, where $\mathbf{x}_i \in \mathbf{R}^m$ is the ith training example and $y_i \in \{1, \cdots, M\}$ is the class label of $\mathbf{x}_i$. For an example $\mathbf{x}_i$, let us denote the output (decision function value) of the kth binary classifier (class ω_k versus the rest) as r_k^i; r_k^i is expected to be large if $\mathbf{x}_i$ is in class ω_k and small otherwise.

After M one-versus-all binary classifiers are constructed, we can obtain the posteriori probabilities through a soft-max function

$$P_k^i = \mathrm{Prob}(\omega_k|\mathbf{x}_i) = \frac{e^{w_k r_k^i + w_{ko}}}{z^i} \; , \tag{1}$$

where $z^i = \sum_{k=1}^{M} e^{w_k r_k^i + w_{ko}}$ is a normalization term that ensures that $\sum_{k=1}^{M} P_k^i = 1$. The parameters of the soft-max function, $(w_1, w_{1o}), \cdots, (w_M, w_{Mo})$, can be designed by minimizing a penalized negative log-likelihood (NLL) function , i.e.,

$$\min \quad E = \frac{1}{2}\|\mathbf{w}\|^2 - C\sum_{i=1}^{l} \log P_{y_i}^i \tag{2}$$
$$\text{subject to} \quad w_k, w_{ko} > 0\ ,\ k = 1, \cdots, M \tag{3}$$

where $\|\mathbf{w}\|^2 = \sum_{k=1}^{M}(w_k^2 + w_{ko}^2)$ and C is a positive regularization parameter. Note that positiveness constraints are placed on weight factor w_k and bias factor w_{ko}. We place positiveness constraints on w_k because we assume that r_k^i is large if $\mathbf{x}_i$ is in class ω_k and small otherwise. We place positiveness constraints on w_{ko} simply to reduce redundancy, since adding a same constant to all w_{ko} does not change the posteriori probability estimates in (1).

The above constrained optimization problem can be transformed to an unconstrained one by using the following substitute variables

$$s_k = \log(w_k) \text{ and } s_{ko} = \log(w_{ko})\ ,\ k = 1, \cdots, M\ . \tag{4}$$

The unconstrained optimization problem can be solved using gradient based methods, such as BFGS [3]. The first-order derivatives of E with respect to the substitute vairables can be computed using the following formulas

$$\frac{\partial E}{\partial s_k} = \frac{\partial E}{\partial w_k}\frac{\partial w_k}{\partial s_k} = \left(w_k + C\sum_{y_i=k}\left(P_k^i - 1\right) r_k^i + C\sum_{y_i\neq k} P_k^i r_k^i \right) w_k\ , \tag{5}$$

$$\frac{\partial E}{\partial s_{ko}} = \frac{\partial E}{\partial w_{ko}}\frac{\partial w_{ko}}{\partial ds_{ko}} = \left(w_{ko} + C\sum_{y_i=k}\left(P_k^i - 1\right) + C\sum_{y_i\neq k} P_k^i \right) w_{ko}\ . \tag{6}$$

2.2 Soft-Max Combination of One-Versus-One Classifiers

Following the same idea as in the previous subsection, posteriori probabilities can also be obtained by soft-max combination of one-versus-one binary classifiers. For an example $\mathbf{x}_i$, let us denote the outputs of one-versus-one classifier C_{kt} as r_{kt}^i. Obviously we have $r_{tk}^i = -r_{kt}^i$. The following soft-max function is used to combine the one-versus-one binary classifiers

$$P_k^i = \text{Prob}(\omega_k|\mathbf{x}_i) = \frac{e^{\sum_{t\neq k} w_{kt} r_{kt}^i + w_{ko}}}{z^i}\ , \tag{7}$$

where $z^i = \sum_{k=1}^{M} e^{\sum_{t\neq k} w_{kt} r_{kt}^i + w_{ko}}$ is a normalization term. The soft-max function parameters can be determined by solving the following optimization problem

$$\min \quad E = \frac{1}{2}\|\mathbf{w}\|^2 - C\sum_{i=1}^{l} \log P_{y_i}^i \tag{8}$$
$$\text{subject to} \quad w_{kt}, w_{ko} > 0\ ,\ k, t = 1, \cdots, M \text{ and } t \neq k \tag{9}$$

where $\|\mathbf{w}\|^2 = \sum_{k=1}^{M}(\sum_{t\neq k} w_{kt}^2 + w_{ko}^2)$ and C is a positive regularization parameter. Note that, as in soft-max combination of one-versus-all classifiers, positiveness constraints are placed on w_{kt} and w_{ko} for the same reason.

As before, we can transform the above constrained optimization problem to an unconstrained one by using the following substitute variables

$$s_{kt} = \log(w_{kt}) \quad \text{and} \quad s_{ko} = \log(w_{ko}) \ , \ k, t = 1, \cdots, M \text{ and } t \neq k \tag{10}$$

The first-order derivatives of E with respect to the substitute variables are

$$\frac{\partial E}{\partial s_{kt}} = \frac{\partial E}{\partial w_{kt}} \frac{\partial w_{kt}}{\partial s_{kt}} = \left(w_{kt} + C \sum_{y_i = k} \left(P_k^i - 1\right) r_{kt}^i + C \sum_{y_i \neq k} P_k^i r_{kt}^i \right) w_{kt} \ , \tag{11}$$

$$\frac{\partial E}{\partial s_{ko}} = \frac{\partial E}{\partial w_{ko}} \frac{\partial w_{ko}}{\partial s_{ko}} = \left(w_{ko} + C \sum_{y_i = k} \left(P_k^i - 1\right) + C \sum_{y_i \neq k} P_k^i \right) w_{ko} \ . \tag{12}$$

The proposed soft-max combination method can be used with any binary classification technique with non-probabilistic outputs. In our numerical study, SVMs are mainly used as the binary classification method.

2.3 Relation to Previous Work

For binary classification, Platt [4] proposed a method to map the outputs of SVMs into probabilities by fitting a sigmoid function after the SVMs are designed. The following parametric model is used by Platt to fit the posteriori probability

$$\text{Prob}(\omega_1 | \mathbf{x}_i) = \frac{1}{1 + e^{A f_i + B}} \ , \tag{13}$$

where f_i is the output of the SVMs associated with $\mathbf{x}_i$. The parameters A and B are determined by minimizing the NLL function of the validation data. We refer to the combination of SVM plus a sigmoid function post-fitted using Platt's method as *PSVM*.

Let us look at our soft-max combination of one-versus-all classifiers for the case $M = 2$. In this case, we have $r_1^i = f_i$, $r_2^i = -r_1^i$, and

$$\text{Prob}(\omega_1 | \mathbf{x}_i) = \frac{1}{1 + e^{-(w_1 + w_2) f_i + (w_{2o} - w_{1o})}} \ . \tag{14}$$

In soft-max combination of one-versus-one classifiers, in the case $M = 2$, we have $r_{12}^i = f_i$, $r_{21}^i = -r_{12}^i$, and

$$\text{Prob}(\omega_1 | \mathbf{x}_i) = \frac{1}{1 + e^{-(w_{12} + w_{21}) f_i + (w_{2o} - w_{1o})}} \ . \tag{15}$$

Note that (14) and (15) are exactly in the same form as (13). Therefore, our soft-max combination methods can be viewed as natural extensions of Platt's sigmoid-fitting idea to multi-category classification. To design A and B of the sigmoid function (13), Platt used some simple ideas from Parzen windows design because there were only two parameters. We employ penalized likelihood for our designs because there are many parameters in (1) and (7).

3 Practical Issues in the Soft-Max Function Design

In this section we discuss two important practical issues in the soft-max function design, i.e., how to get the training examples for the soft-max function design and how to tune the regularization parameter C.

3.1 Training Examples for the Soft-Max function Design

We use 5-fold cross-validation (CV) to get unbiased training data (r_k^i or r_{kt}^i) for soft-max function design. The original training data ($\mathbf{x}$) are partitioned into 5 almost equal folds with each fold contains almost equal percentage of examples of one particular class.

Let us first consider the soft-max combination of one-versus-all classifiers. Take one k. The r_k^i of examples ($\mathbf{x}_i$) in the left-out fold is determined by a binary classifier trained on all other examples except those in the left-out fold, where training examples from class ω_k take positive labels and other examples take negative labels.

Now consider the soft-max combination of one-versus-one classifiers. Take one k and t. The r_{kt}^i of examples ($\mathbf{x}_i$) in the left-out fold from class ω_k and class ω_t are determined by a binary classifier trained on examples of only class ω_k and class ω_t, in all the other folds, where examples from ω_k take positive labels and those from class ω_t take negative labels. For the examples ($\mathbf{x}_i$) not from either class ω_k or class ω_t, the r_{kt}^i are determined by a binary classifier trained on the full examples, in all folds, from class ω_k and class ω_t, where examples from class ω_k and class ω_t respectively take positive and negative labels.

The cross-validation determination of unbiased training data mentioned above adds almost no extra cost to the overall design process since it is also used to tune the hyperparameters of the binary classifiers.

3.2 Regularization Parameter *C*

Let us now discuss the choice of parameter C for use in (2) and (8). Since designing the soft-max combining function is a relatively small optimization problem with only $2M$ or M^2 variables, we can use k-fold cross-validation to tune this regularization parameter with little extra computation cost.

We can do k-fold cross-validation at various values of C and then choose one C value based on measurements, of the multicategory classifier's performance, estimated from cross-validation. In our study, we do 5-fold cross-validation and try the set of C values: $C = \{2^{-15}, 2^{-14}, \cdots, 2^{15}\}$. Let us denote the cross-validation estimates of error rate and NLL with $cvErr$ and $cvNLL$. In order to get a good classification accuracy as well as good posteriori probability estimates, instead of choosing the C associated with the least $cvErr$, we do the following. First, let us denote the least $cvErr$ with Err_Least and define the C set: $SC_{LeastErr} = \{C : cvErr(C) \leq 1.05 Err\_Least\}$. Second, let us define the least cvNLL with $C \in SC_{LeastErr}$ and as NLL_Least. Based on NLL_Least, the relaxed C set is defined: $SC_{LeastNLL} = \{C : cvNLL(C) \leq 1.05 NLL\_Least, C \in SC_{LeastErr}\}$.

The smallest $C \in SC_{LeastNLL}$ is chosen as the final regularization parameter value.

3.3 Simplified Soft-max Function Design

We may simplify the soft-max function design by minimizing only the second term in (2) and (8), i.e., omit the use of regularization; equivalently, set $C = +\infty$. The only change in solving the optimization problem is that, the gradient expression of (5), (6), (11) and (12) are appropriately modified. In the next section, we will evaluate this simplified design against the one designed with regularizer, as well as against standard multi-category methods.

4 Numerical Study

In this section, we numerically study the performance of the soft-max combination methods and compare it with other implementations of one-versus-all and one-versus-one methods. For the basic binary classifier the following three methods were used: (a) the support vector machine (SVM)[6]; (b) support vector machine with Platt's posterior probabilities (PSVM)[4]; and (c) kernel logistic regression (kLOGREG) [5]. SVM, PSVM and kLOGREG all are kernel-based classification methods. The Gaussian kernel, $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2}$ was employed. The following multi-category classification methods were studied:

- **Soft-max combination of SVM one-versus-all classifiers:** The soft max functions may be designed with the regularizer or without it (the simplified design); we refer to the corresponding methods as *SMC1va* and *SSMC1va* (the first "S" denotes "simplified").
- **Soft-max combination of SVM one-versus-one classifiers:** The method in which the combination function is designed with the regularizer is referred as *SMC1va*; the method which did not use the regularizer (the simplified design) is referred as *SSMC1v1*.
- **Winner-Takes-All implementation of one-versus-all classifiers:** Using SVM, PSVM and kLOGREG for binary classification with WTA we obtain 3 methods: *WTA_SVM*, *WTA_PSVM* and *WTA_KLR*. For WTA_PSVM and WTA_KLR, we can obtain posteriori probability estimates crudely by simple normalization, i.e., $\text{Prob}(\omega_k|\mathbf{x}) = p_k / \sum_{t=1}^{M} p_t$, where p_k is the probabilistic output of kth binary classifier (class ω_k versus the rest).
- **Max-Wins voting implementation of one-versus-one classifiers:** Using SVM, PSVM and kLOGREG for binary classification with MWV we obtain 3 methods: *MWV_SVM*, *MWV_PSVM* and *MWV_KLR*.
- **Pairwise coupling implementation of one-versus-one classifiers:** We refer to the PWC implementations with PSVM and kLOGREG methods respectively as *PWC_PSVM* and *PWC_KLR*.

Each binary classifier (whether it is SVM, PSVM or kLOGREG) requires the selection of two hyperparameters (a regularization parameter C and kernel

Table 1. Basic information about the datasets and training sizes used in the numerical study

Dataset	#Classes	#Inputs	#Training Examples	#Total Examples
ABE	3	16	560	2,323
DNA	3	180	500	3,186
Satellite Image (SAT)	6	36	1,500	6,435
Image Segmentation (SEG)	7	19	500	2,310
Waveform (WAV)	3	21	300	5,000

parameter σ^2). Every multi-category classification method mentioned above involves several binary classifiers. We take the C and σ^2 of each of the binary classifiers within a multi-category method to be the same. The two hyperparameters are tuned using 5-fold cross-validation estimation of the multi-category classification performance. We select the optimal hyperparameter pair by a two-step grid search. First we do a coarse grid search using the following sets of values: $C = \{10^{-3}, \cdots, 10^3\}$ and $\sigma^2 = \{10^{-3}, \cdots, 10^3\}$. Thus 49 combinations of C and σ^2 are tried in this step. An optimal pair (C_o, σ_o^2) is selected from this coarse grid search. In the second step, a fine grid search is conducted around (C_o, σ_o^2), where $C = \{0.2C_o, 0.4C_o, \cdots, 0.8C_o, C_o, 2C_o, 4C_o, \cdots, 8C_o\}$ and $\sigma^2 = \{0.2\sigma_o^2, 0.4\sigma_o^2, \cdots, 0.8\sigma_o^2, \sigma_o^2, 2\sigma_o^2, 4\sigma_o^2, \cdots, 8\sigma_o^2\}$. All together, 81 combinations of C and σ^2 are tried in this step. The final optimal hyperparameter pair is selected from this fine search. In grid search, especially in the fine search step, it is quite often the case that there are several pairs of hyperparameters that give the same cross validational classification accuracy. So, some principles have to be used to select one pair of C and σ^2 from these short-listed combinations. For the methods with posteriori probability estimates, where cross-validation estimated error rate (Err) and NLL are both available, the following strategies are applied sequentially until we find one unique parameter pair: (a) select the pair with smallest Err value; (b) select the pair with smallest NLL value; (c) select the pair with smaller C and larger σ^2; (d) select the pair with smallest 8-neighbour average Err value; (e) select the pair with smallest 8-neighbour average NLL value; (f) select the pair with smallest C value. For the methods without posteriori probability estimates, only step (a), (c), (d) and (f) are sequentially applied.

When the training size is large enough, probably all the methods may perform as well as Bayes-optimal algorithm. So, to clearly see differences in the performance of various methods, reasonably sparse training sets at which there is still room for performance improvement should be used. For this purpose, we run all the methods on 5 datasets with training sets that are somewhat smaller than those which are usually used. The training set sizes are chosen based on the suggestions in [1]. The 5 standard datasets used are: *ABE*, *DNA*, *Satellite Image* (SAT), *Image Segmentation* (SEG) and *Waveform* (WAV). *ABE* is a dataset that we extract from the dataset *Letter* by using only the classes corresponding to the characters "A", "B" and "E". All the continuous inputs of these datasets are normalized to have zero mean and unit standard deviation. Table 1 summa-

Table 2. The mean and standard deviation of test error rate (in percentage) over 20 partitions, of the methods based on the one-versus-all binary classifiers

	SMC1va	SSMC1va	WTA_SVM	WTA_PSVM	WTA_KLR
ABE	0.91±0.37	0.95±0.30	0.91±0.33	0.90±0.35	0.90±0.34
DNA	7.66±0.73	7.54±0.64	7.80±0.74	7.73±1.00	7.74±0.73
SAT	10.07±0.44	10.16±0.49	10.02±0.41	10.13±0.43	10.28±0.47
SEG	6.01±0.80	5.91±0.94	6.56±0.88	6.18±0.96	5.72±0.80
WAV	15.17±0.77	15.33±0.76	15.29±0.74	15.40±0.99	14.82±0.59

Table 3. The mean and standard deviation of test error rate (in percentage) over 20 partitions, of the methods based on one-versus-one binary classifiers

	SMC1v1	SSMC1v1	MWV_SVM	MWV_PSVM	MWV_KLR	PWC_PSVM	PWC_KLR
ABE	1.20±.0061	0.99±0.40	1.01±0.41	0.98±0.38	0.97±0.36	0.87±0.28	0.93±0.37
DNA	7.81±.0096	7.96±0.77	7.65±0.93	7.89±0.93	7.81±0.70	7.65±0.77	7.56±0.73
SAT	10.31±.0064	10.03±0.37	10.37±0.71	10.18±0.50	10.23±0.43	9.98±0.41	10.21±0.39
SEG	5.92±.0156	5.66±0.93	5.41±1.02	5.38±0.96	4.85±0.69	5.42±0.90	4.82±0.68
WAV	14.47±.0087	14.55±1.08	16.01±1.06	14.62±0.82	14.63±0.66	14.66±0.76	14.66±0.67

Table 4. mean and standard deviation of test NLL value over 20 partitions, of the methods with posteriori probability estimates

	SMC1va	SSMC1va	SMC1v1	SSMC1v1	WTA_PSVM	WTA_KLR	PWC_PSVM	PWC_KLR
ABE	.0452±.0378	.0392±.0226	.0661±.0413	.0460±.0368	.0316±.0096	.0257±.0068	.0361±.0057	.0306±.0142
DNA	.2250±.0346	.2153±.0123	.2295±.0386	.2154±.0212	.2326±.0235	.2315±.0290	.2105±.0166	.2165±.0195
SAT	.3129±.0103	.3017±.0116	.2911±.0134	.2763±.0110	.3231±.0093	.2887±.0462	.2976±.0078	.2915±.0144
SEG	.2080±.0496	.2221±.0413	.2353±.0985	.2723±.0782	.2392±.0226	.2098±.0294	.2923±.0200	.1752±.0134
WAV	.3808±.0452	.3664±.0198	.3584±.0522	.3448±.0231	.3617±.0229	.4353±.0698	.3553±.0154	.4132±.0436

rizes the basic information about the datasets and the training set sizes used. For each dataset, we randomly partition the whole dataset into a training set and a test set 20 times by stratified sampling. For each partition of one dataset, after each multi-category classifier is designed using the training set, it is tested on the test set. Then the mean and the standard deviation of the test set error rate of each method is computed using the results of the 20 runs. These values are reported in Table 2-3 for the five datasets. For the methods with posteriori probability estimates, the mean and the standard deviation values of the test set NLL are reported in Table 4.

5 Results and Conclusions

Let us now analyze the results of our numerical study. First, it is easy to see from Tables 2–4 that all the methods included in the numerical study give competitive performance. However, some methods show overall betterness over other methods. We do a finer analysis and comparison to see this.

Suppose we want to compare method 1 against method 2 on a given dataset. The pairwise *t-test* is conducted to analyze if the (test set) error of method 1 is greater than that of method 2. Assuming normality of the populations and using the hypothesis that the mean errors of the two methods are same, we compute the p-value, which is the probability that the mean error of method 1 is greater than that of method 2. Thus, a large p-value (say > 0.9) indicates that method 1 is clearly worse than method 2, while a small value (say < 0.1) indicates that method 1 is clearly better than method 2. Since there are five datasets, we compute a set of five p-values, using the following order for the datasets: ABE, DNA, SAT, SEG and WAV. Similar to (test set) error, we can also conduct the above analysis for (test set) NLL. Unless mentioned otherwise, all p-values will refer to the (test set) error.

For the WTA implementions of one-versus-all methods, WTA_KLR has a slightly better classification accuracy than WTA_SVM and WTA_PSVM. The sets of p-values of WTA_KLR against WTA_SVM and WTA_PSVM are $\{0.4005, 0.3121, 0.9995, 2.5044\times10^{-5}, 1.1164\times10^{-4}\}$ and $\{0.4818, 0.5433, 0.9557, 0.0015, 0.0078\}$. The set of p-values of WTA_PSVM against WTA_SVM is $\{0.4225, 0.3360, 0.9323, 0.0092, 0.7166\}$; these two methods give close performance.

For the MWV implementation of one-versus-one methods, MWV_KLR and MWV_PSVM have slightly better classification accuracy than MWV_SVM. The sets of p-values of MWV_KLR and MWV_PSVM against MWV_SVM are $\{0.3278, 0.9300, 0.1887, 0.0133, 1.0282\times10^{-5}\}$ and $\{0.2574, 0.8499, 0.0749, 0.3617, 1.4880\times 10^{-7}\}$. The classification accuracies of MWV_KLR and MWV_PSVM are close; the set of p-values of MWV_KLR against MWV_PSVM is $\{0.4846, 0.3713, 0.7442, 0.\ 0082, 0.5160\}$.

For soft-max combination of one-versus-all classifiers, SSMC1va and SMC1va give close classification accuracy. But SSMC1va has a better probability estimate. The set of p-values of SSMC1va against SMC1va, from the pairwise t-test on (test set) NLL, is $\{0.2242, 0.1088, 9.9571^{-6}\ 0.9070, 0.0610\ \}$. It should be noted that the design of the combination function for SSMC1va is also much simpler than that of SMC1va.

For soft-max combination of one-versus-one classifiers, SSMC1v1 is better than SMC1v1, both, in terms of classification accuracy as well as probability estimate. The sets of p-values from pairwise t-test on (test set) error and (test set) NLL, of SSMC1v1 against SMC1v1, are $\{0.0088, 0.1808, 0.0141, 0.1830, 0.6939\ \}$ and $\{\ 0.0053, 0.0652, 1.0274 \times 10^{-4}\ 0.9233, 0.1437\ \}$.

Overall, the two PWC implementations, PWC_PSVM and PWC_KLR, are the best. The variance of PWC_KLR is smaller than that of PWC_PSVM for (test set) error while the variance of PWC_PSVM is smaller than that of PWC_KLR for (test set) NLL. The variances of PWC_PSVM and PWC_KLR are also smaller than those of other methods. Given below are the p-values, from the t-test of (test set) error on five datasets, of PWC_PSVM and PWC_KLR against the rest of the methods:

	PWC_PSVM					PWC_KLR				
	ABE	DNA	SAT	SEG	WAV	ABE	DNA	SAT	SEG	WAV
WTA_SVM	0.1613	0.1470	0.2604	1.5271e-6	1.4349e-4	0.6248	9.2922e-20	6.0812e-25	2.0127e-16	3.8722e-26
WTA_PSVM	0.2994	0.3629	0.0183	3.5621e-4	0.0016	0.8127	0.2274	0.7764	5.8390e-7	6.3759e-4
WTA_KLR	0.3001	0.2742	7.6409e-5	0.0570	0.1989	0.6858	0.0849	0.1823	1.1105e-6	0.0822
MWV_SVM	0.0124	0.4869	0.0054	0.5111	3.0141e-7	0.1892	0.1950	0.1432	0.0040	1.0058e-5
MWV_PSVM	0.0421	0.0375	0.0077	0.6686	0.6191	0.2977	0.0673	0.6182	0.0019	0.5838
MWV_KLR	0.0525	0.1572	2.2732e-4	0.9960	0.5539	0.2319	0.0027	0.3458	0.3473	0.6276
SMC1va	0.2785	0.4609	0.1310	0.0012	0.0011	0.6116	0.2159	0.9468	1.2190e-7	0.0045
SSMC1va	0.0656	0.7975	0.0026	0.0016	0.0014	0.4209	0.5673	0.7105	7.0694e-7	0.0041
SMC1v1	0.0011	0.1920	0.0120	0.0277	0.8451	0.0012	0.0924	0.1992	0.0016	0.8008
SSMC1v1	0.0410	0.3818	0.2655	0.1022	0.6799	0.1786	0.1423	0.9896	8.5907e-5	0.6631

To conclude, we can say the following. WTA_KLR seems to be the best WTA implementation among all one-versus-all classifiers. MWV_KLR seems to be the best MWV implementation among all one-versus-one classifiers. The two PWC implementations, PWC_PSVM and PWC_KLR are the best overall. The proposed soft-max combination methods with simplified combination function design, SSMC1va and SSMC1v1, are better than their 'regularized' counterparts, SMC1va and SMC1v1; they are also much simpler to design. The proposed soft-max combination methods provide new good ways of doing multicategory classification with binary classification methods, and more importantly, new ways of obtaining posteriori probability estimates from binary classifiers whose outputs are not probabilistic values.

References

1. Bauer, E., R. Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting and variants. Machine Learning, 36 (1999) 105–142.
2. Hastie, T., Tibshirani, R.: Classification by Pairwise Coupling. In: Jordan, M.I., Kearns, M.J., Solla, A.S. (eds.): Advances in Neural Information Processing Systems, Vol. 10. MIT Press (1998)
3. Liu, D.C., Nocedal, J.: On the Limited Memory Method for Large Scale Optimization. Mathematical Programming B, 45 (1989) 503-528
4. Platt, J.: Probabilistic Outputs for Support Vector Machines and Comparison to Regularized Likelihood Methods. In: Smola, A.J., Bartlett, P., Schölkopf, B., Schuurmans, D. (eds.): Advances in Large Margin Classifiers. MIT Press (1999) 61–74
5. Roth, V.: Probabilistic Discriminant Kernel Classifiers for Multi-class Problems. In: Radig, B., Florczyk, S. (eds.): Pattern Recognition-DAGM'01. Springer (2001) 246–253
6. Vapnik, V.: Statistical Learning Theory. Wiley, New York, NY (1998)
7. Weston, J., Watkins, C.: Multi-class Support Vector Machines. Technical Report CSD-TR-98-04, Royal Holoway (1998)