An Output Grouping Based Approach to Multiclass Classification Using Support Vector Machines

Xuan Zhao, Steven Guan and Ka Lok Man

Abstract Support Vector Machine (SVM) classifiers are binary classifiers in nature, which have to be coupled/assembled to solve multi-class problems. One-Versus-Rest (1-v-r) is a fast and accurate method for SVM multiclass classification. This paper investigates the effect of output grouping on multiclass classification with SVM and offers an even faster version of 1-v-r based on our output grouping algorithm.

Keywords Multiclass classification · Support vector machine · Decomposition method · Output grouping

1 Introduction

Support Vector Machine (SVM) is a popular supervised machine learning algorithm for solving problems in classification and regression. An SVM takes a set of training samples (known observations belonging to a certain class/category) and yields a set of decision functions to determine which class a test sample (unknown observation) belongs to [1].

However a novel SVM classifier only solves two-class problems. To solve a multi-class problem, multiple binary SVM classifiers are coupled or assembled as a multi-class SVM. Such a decomposition method plays an important role on the multi-class SVM's generalization performance and time consumption. One-Versus-One (1-v-1), One-Versus-Rest (1-v-r) and ECOC are three of the most commonly

X. Zhao (✉) · S. Guan · K.L. Man
Xi'an Jiaotong-Liverpool University, Suzhou, China
e-mail: xuan.zhao@xjtlu.edu.cn

S. Guan
e-mail: steven.guan@xjtlu.edu.cn

K.L. Man
e-mail: ka.man@xjtlu.edu.cn

J.J. (Jong Hyuk) Park et al. (eds.), *Advanced Multimedia and Ubiquitous Engineering*, Lecture Notes in Electrical Engineering 393,
DOI 10.1007/978-981-10-1536-6_51

used ensemble methods [2, 3]. Empirical result shows 1-v-r is the fastest method in predicting unknown samples [4–6] with good performance.

This paper and investigates the effect of output (class) grouping, which were previously researched in neural networks [7–9], in multiclass SVM classifiers. Finally we offer an even faster version of 1-v-r with the help of the output-grouping algorithm.

2 Background

Output grouping is a learning strategy being applied in neural networks to improve the accuracy and reduce training cost. The strategy is based on observation of class correlation. In previous study in neural networks, strong correlation between two classes i, j implies high similarity. Then i, j can be trained together for better precision [8].

2.1 One-Versus-Rest (1-v-R)

The idea of 1-v-R classification is to build N classifiers for each of the classes. In the training process all the training data is used for every one of the N classifiers. If a classifier is built for class i then the classifier is trained with the samples that belong to class i against the rest of the samples from the training data set.

Once the model is built, an unknown sample goes through all the N classifiers. Each classifier decides whether or not this sample belongs to the corresponding class. Ideally one and only one classifier should respond 'yes' to the unknown sample and the corresponding class will be the class that the sample belongs to, otherwise there is either a clash or a total miss. Under such a circumstance the unknown sample goes to the class with the most training samples.

2.2 Output Grouping

For SVM classifiers, if samples from two classes are 'mixed' together then it is difficult for SVM to draw a line or plane to distinguish the two classes. In the case of 1-v-r, suppose a class i needs to be distinguished from the remaining classes (the 'rest'). If another class j among the rest is highly similar to i then it will also be difficult to tell i and the rest apart, even if other classes among the rest (excluding class j) are easy to be separated from i. As a result, such interaction, correlation or similarity between classes i and j would contribute to a complex but less accurate model.

We tested this hypothesis using the UCI Iris dataset. We duplicated all samples from class 3 and labelled them as class 4, train a linear SVM model using 1-v-r

Table 1 Iris 1-v-r SVM classification performance

Iris	Baseline	3 duplicated as 4	3 and 4 grouped
Model size	112	336	256
Accuracy (%)	94.40	65.86	72.57

method, ran a prediction on training set and ran a 10-fold cross prediction with classifier and the data set. We then merged samples from classes 3 and 4 and labelled them as a new class (grouping), trained and validated an additional 3-versus-4 classifier and merged the results from both classifiers (Table 1).

If the classifier cannot tell 3 and 4 apart, then the best chance of classification error between 3 and 4 should be 50 % (random guessing). Then the cross validated accuracy should be around 71 % instead of 65.86 %. In other words, the strong similarity between class 3 and 4 should not but has affected the performance of other classification tasks. While with class 3 and 4 grouped, we seem to eliminate such undesirable interference.

2.3 *Output-Grouping-One-Versus-Rest (OG) Algorithm*

We propose an algorithm, OG, which aggregates outputs (classes) by their 'similarity' in groups, then treats the groups as new classes and performs 1-v-r learning.

The algorithm has six components: (1) similarity scorer, (2) class grouper, (3) 1-v-r trainer, (4) intra-group classifier trainer, (5) 1-v-r predictor and (6) intra-group predictor (Figs. 1 and 2; Table 2).

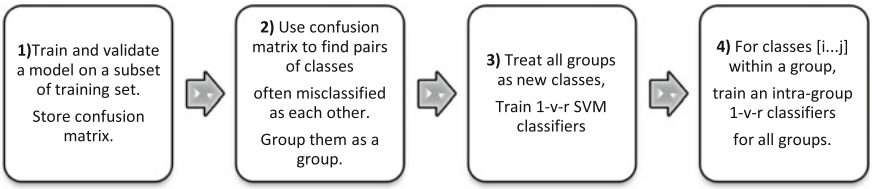

Fig. 1 Training components (*1–4*) of OG

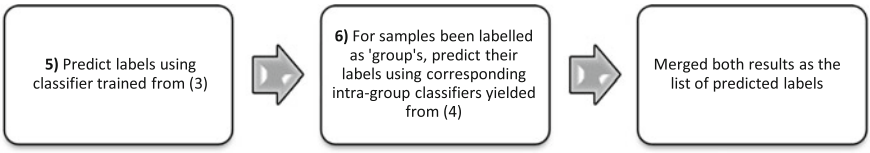

Fig. 2 Prediction components (*5* and *6*) of OG

Table 2 Grouping algorithm of OG

Input `L_sig`, list of pair of classes that often been misclassified as each other, sorted by the descending order of mutual misclassification rate. Pairs with misclassification rate lower than the average (or average multiplied by a regularisation parameter) are excluded from the list.
Output `G: []`, list of groups containing grouped classes

```
while len(L_sig) > 0:
       first_itm = L_sig.pop(0)
       c1, c2 = first_itm[0], first_itm[1]
       group= set().add(c1).add(c2)
       idx = 0
       while 0 <= idx < len(L_sig):
              itm = L_sig[idx]
              if itm[0] or itm[1] in group:
                     pitm = L_sig.pop(idx)
                     group.add(pitm[0]) .add(pitm[1])
                     idx = 0
              else:  idx += 1
       G.append(group)
```

OG attempts to group "hard-to-tell-apart" classes together so that it should be able to reduce the model complexity, improved the accuracy and further increase the prediction speed. However the speed-up is at the cost of an additional training/prediction, grouping and result merging.

3 Experiments

The OG was evaluated on seven different data sets: UCI Iris Plant, UCI Glass Identification, UCI Vowel, Statlog Handwritten Letters, Statlog Satimage, USPS and MNIST [11]. The values in each data set have been scaled to [0, 1].

Each data set was trained and tested with five multiclass SVM algorithms: OG, OGPG, 1-v-r, 1-v-1 and ECOC [12]. OG is the implementation of our proposed algorithm under the *scikit-learn* framework, while 1-v-r, 1-v-1 and ECOC are the existing implementations in *scikit-learn* [13]. All the algorithms use SVC [2] to train and test the SVM models.

To further address the effectiveness of grouping, we introduce a variant of OG, OG with pre-computed group (OGPG), by supplying the algorithm pre-computed groups.

The experiment for each method ran 50 times for the averaged measurements. Each time the model was trained by randomly-spit 2/3 of the data set and validated on the remaining 1/3 (Table 3).

Table 3 Experiment data sets, parameters and pre-computed groups (OGPG only)

	Size	Classes	Features	Kernel	*C*	Groups
Iris	150	3	4	RBF	100	(2, 3)
Glass	214	6	9	Poly	100	(1–3, 5)
Vowel	10	11	10	RBF	1000	(4, 5, 10)
Letter	20,000	26	16	Poly	100	(3, 4, 6)
Satimage	7435	6	36	Poly	100	(4, 6, 8–10, 16)
USPS	7291	10	256	Poly	100	(5, 8, 10)
MNIST	10,000	10	780	Poly	100	(7, 9)

4 Results

The accuracy is close for all algorithms except for ECOC in some occasions across all the datasets. OG and OGPG perform slightly better than 1-v-r algorithm on smaller datasets (*iris*, *glass* and *vowel*). However we cannot conclude which algorithm is the best in terms of accuracy (Table 4).

On large data sets (*satimage*, *usps* and *mnist*) OG and OGPG slightly reduced the training time and significantly reduced the prediction time by 0.8–29.2 % and 5.4–30.6 % respectively, compared with 1-v-r method. OGPG is the fastest in terms of prediction, seconded by OG.

Table 4 Averaged accuracy, model size and time consumption of OG, 1-v-r, 1-v-1 and ECOC

		Accuracy (%), StDev/mean	# Support vectors	Training CPU time (ms)	Prediction CPU time (ms)
Iris	OG	**97.25** (2.98)	14	10.8	1.6 (59.06)
	OGPG	**97.25** (3.00)	**13**	10.1	1.4 (180.84)
	1-v-r	97 (2.90)	23	7.2	**0.5** (64.41)
	1-v-1	**97.25** (3.00)	18	6.3	1 (120.51)
	ECOC	90.25 (13.16)	22	**5.2**	2.4 (279.18)
Glass	OG	67.01 (6.72)	238	22	25 (49.64)
	OGPG	67.17 (7.18)	239	22	24 (36.87)
	1-v-r	66.55 (6.57)	241	**17**	**17** (97.77)
	1-v-1	**68.49** (7.40)	**204**	35	62 (39.62)
	ECOC	65.96 (7.34)	511	21	50 (261.80)
Vowel	OG	96.29 (1.97)	**348**	71	71 (28.14)
	OGPG	96 (1.78)	365	74	**58** (58.31)
	1-v-r	95.76 (1.93)	375	**69**	**58** (40.03)
	1-v-1	**96.53** (1.71)	683	135	387 (19.96)
	ECOC	92.71 (2.80)	1507	461	161 (102.60)

(continued)

Table 4 (continued)

		Accuracy (%), StDev/mean	# Support vectors	Training CPU time (ms)	Prediction CPU time (ms)
Satimage	OG	89.09 (0.90)	1739	2107	264 (10.06)
	OGPG	89.09 (0.89)	1731	2037	**259** (7.19)
	1-v-r	**89.16** (0.90)	1810	2983	373 (7.71)
	1-v-1	88.58 (0.93)	**1077**	**560**	488 (6.04)
	ECOC	88.36 (3.28)	3903	7156	799 (19.34)
Letter	OG	94.46 (0.34)	10776	28328	3288 (4.29)
	OGPG	**94.48** (0.33)	**10504**	28164	**3136** (4.95)
	1-v-r	**94.48** (0.34)	10872	28535	3316 (4.40)
	1-v-1	94.8 (0.33)	17813	**4077**	13559 (4.05)
	ECOC	91.62 (0.43)	94667	461447	30190 (12.01)
Usps	OG	98.02 (0.29)	**2086**	19591	4564 (8.91)
	OGPG	98.05 (0.30)	2160	19575	**4230** (6.84)
	1-v-r	97.99 (0.30)	2118	19886	4495 (6.17)
	1-v-1	**98.15** (0.31)	3204	**4666**	10882 (3.46)
	ECOC	97.88 (0.32)	7126	80958	21593 (19.36)
Mnist	OG	96.31 (0.35)	**4866**	50274	21472 (5.36)
	OGPG	96.3 (0.34)	5057	47507	**19809** (2.10)
	1-v-r	**96.35** (0.32)	5018	50759	22781 (2.64)
	1-v-1	96.08 (0.39)	6466	**13038**	55331 (1.79)
	ECOC	95.73 (0.47)	16928	181370	79754 (8.33)

With standard deviation divided by average in the bracket
Bold are the best measured performance among five methods for each data set

On smaller data sets, OG and OGPG's worse speed performance were caused by the fact that the overhead of additional grouping and merging dominates the time consumption of SVM training and prediction.

Even though OG performs an additional training/prediction to determine the grouping on another 33–50 % of the training set, its training time is still less than 1-v-r. The model size is also slightly decreased for all data sets except for *usps* and *mnist*.

5 Discussions and Conclusions

To our disappointment neither OG nor OGPG did improve the accuracy. Our speculation during the experiment shows that the accuracy tends to be *saturated* at a certain point when the number of training samples is *large enough* or the parameters of SVM are *fairly well* optimised.

We may conclude that OG and OGPG improve the speed performance of 1-v-r method even at the cost of performing additional classification tasks.

Our findings show that output grouping is effective at reducing the training complexity of 1-v-r multi-class classification tasks using support vector machines. Nevertheless output grouping can be used to boost the prediction speed of multi-class SVMs with no extra cost. We are now investigating the cause of the *saturated* accuracy and working on combining input and output grouping methods [8, 9, 14–17] for better classification accuracy.

References

1. Chang C-C, Lin C-J (2011) Libsvm. ACM Trans Intell Syst Technol 2(3):1–27
2. svm@scikit-learn.org (2016). Available: http://scikit-learn.org/stable/modules/svm.html
3. Hsu C-W, Lin C-J (2002) A comparison of methods for multi-class support vector machines. IEEE Trans Neural Networks 13(2):415–425
4. Mehra N, Gupta S (2013) Survey on multiclass classification methods. Int J Comput Sci Inf Technol 4(4):572–576
5. Rifkin R (2008) Multiclass classification
6. Tewari A, Bartlett PL (2007) On the Consistency of multiclass classification methods. J Mach Learn Res 8:1007–1025
7. Guan S-U, Li S (2002) Parallel growing and training of neural networks using output parallelism. IEEE Trans Neural Netw 13(3):542–550
8. Yang S, Guan S-U, Guo SJ, Zhao LF, Li WF, Xue HX (2013) Neural Network output partitioning based on correlation. J Clean Energy Technol 1(4):342–345
9. Yang S, Guan SU, Li WF, Zhao LF (2013) Low-interference output partitioning for neural network training. J Clean Energy Technol 1(4)
10. Bordes A, Ertekin S, Weston J, Bottou L (2005) Fast kernel classifiers with online and active learning. J Mach Learn Res 6:1579–1619
11. LIBSVM data: classification, regression, and multi-label. Available https://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/. Accessed on 03 Feb 2016
12. Dietterich TG, Bakiri G (1994) Solving multiclass learning problems via error-correcting output codes. J Artif Intell Res 2:263–286
13. multiclass@scikit-learn.org (2016). Available: http://scikit-learn.org/stable/modules/multiclass.html
14. Guo SJ, Guan S-U, Yang S, Li WF, Zhao LF, Song JH (2013) Input partitioning based on correlation for neural network learning. J Clean Energy Technol 1(4):335–338
15. Guo S, Guan S-U, Li W, Man KL, Liu F, Qin AK (2013) Input space partitioning for neural network learning. Int J Appl Evol Comput 4(2):56–66
16. Guo S, Guan SU, Li W, Zhao L, Song J, Cao M (2012) Promotion-based input partitioning of neural network. In: International conference on computer, communication, automation and control 2012 (CCAC 2012)
17. Guo S, Guan SU, Li W, Zhao L, Song J, Cao M (2014) Promotion-based input partitioning of neural network. In: Proceedings of the 9th international symposium on linear drives for industry applications, vol 3, pp 179–186