Partial Discriminative Training of Neural Networks for Classification of Overlapping Classes

Cheng-Lin Liu

National Laboratory of Pattern Recognition (NLPR)
Institute of Automation, Chinese Academy of Sciences
95 Zongguancun East Road, Beijing 100190, P.R. China
Email: liucl@nlpr.ia.ac.cn

Abstract. In applications such as character recognition, some classes are heavily overlapped but are not necessarily to be separated. For classification of such overlapping classes, either discriminating between them or merging them into a metaclass does not satisfy. Merging the overlapping classes into a metaclass implies that within-metaclass substitution is considered as correct classification. For such classification problems, I propose a partial discriminative training (PDT) scheme for neural networks, in which, a training pattern of an overlapping class is used as a positive sample of its labeled class, and neither positive nor negative sample for its allied classes (classes overlapping with the labeled class). In experiments of handwritten letter recognition using neural networks and support vector machines, the PDT scheme mostly outperforms cross-training (a scheme for multi-labeled classification), ordinary discriminative training and metaclass classification.

1 Introduction

In some pattern recognition applications, some patterns from different classes have very similar characteristics. In the feature space, such patterns of different classes correspond to co-incident or very close points, residing in an overlapping region. We call such classes as overlapping classes. A typical application is handwritten character recognition, where some classes such as letters 'O', 'o' and numeral '0' have identical shape, and it is neither possible nor necessary to separate them. Some other classes, such as upper-case letters 'A', 'M' and 'N', have many samples written in lower-case shapes (see Fig. 1). Thus, the upper-case letter and its corresponding lower-case have partially overlapping regions in the feature space. For such overlapping classes and all the pairs of upper-case/lower-case letters, it is not necessary to separate them at character level because it is easy to disambiguate according to the context.

Generally, there are two ways to deal with the overlapping classification problem. One way is to simply merge the overlapping classes into a metaclass and ignore the boundary between the classes. In metaclass classification, the substitution between overlapping classes (within a metaclass) is considered as correct.

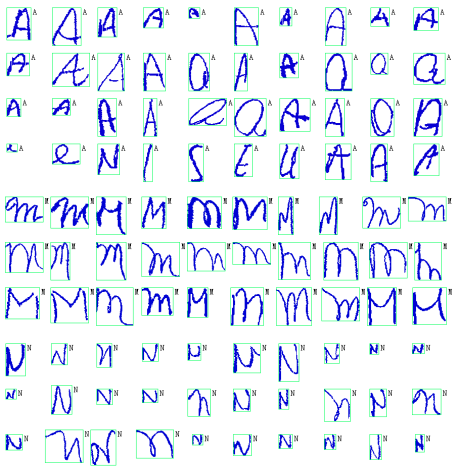

Fig. 1. Many samples of "AMN" are written in lower-case shape.

In English letter recognition, the 52 letters can be merged into 26 case-insensitive classes. The 52-class letter classifier can also be evaluated at metaclass level by ignoring the substitution between upper and lower cases. Another way is to separate the overlapping classes by refining the classification boundary in the feature space, using multi-stage classifier or combining multiple classifiers [1, 2]. Nevertheless, such attempt of discrimination is not necessary in the context of character recognition. The accuracy of overlapping classes separation is also limited by the inherent feature space overlap.

The problem of overlapping classification is similar to multi-labeled classification [3, 4], where a pattern may belong to multiple classes. If we enhance the class label of a pattern from an overlapping class such that it belongs to the labeled class as well as the allied classes (those overlapping with the labeled class), the overlapping classification problem becomes a multi-labeled one. In evaluation, the classification of a pattern to any of its allied classes is considered correct.

For overlapping classification ignoring within-metaclass substitution, I propose a new scheme for training neural networks and support vector machines (SVMs). In my training scheme, called partial discriminative training (PDT), the pattern of an overlapping class is used as a positive sample of its labeled class, and neither positive nor negative sample of the allied classes. By contrast, in ordinary discriminative training of neural networks and SVMs, the pattern of a class is used as negative sample of all the other classes, and in multi-labeled classification (cross-training [3]), the pattern is used as positive sample of its allied classes.

To evaluate the performance of the proposed PDT method, I experimented on the C-Cube handwritten letter database [5, 6] using neural networks and SVMs for classification. The results show that PDT mostly outperforms cross-training,

ordinary discriminative training, and metaclass classification when evaluated at metaclass level.

In the rest of this paper, Section 2 briefly reviews the related works; Section 3 describes the proposed PDT scheme for neural networks and SVMs; Section 4 presents the experimental results, and Section 5 offers concluding remarks.

2 Related Works

Statistical classifiers [7] and artificial neural networks [8] have been popularly applied to pattern recognition. A parametric statistical classifier, which estimates the probability density function of each class without considering the boundary between classes, is ready for classification of overlapping classes. The overlap between classes will not affect the estimation of parameters of parametric statistical classifiers. In training neural networks, the connected weights are iteratively adjusted by optimizing an objective of minimum squared error or cross-entropy between class outputs and desired targets [8]. The overlap between classes will affect the complexity of decision boundary. I will show in Section 3 that the training objective of neural networks can be decomposed into multiple binary (two-class) classification problems.

The support vector machine (SVM) [9] is an emerging classifier for solving difficult classification problems. Multi-class classification is usually accomplished by combining multiple binary SVMs encoded as one-versus-all, pairwise, or other ways. The binary SVM is trained (coefficients of kernel functions estimated) by maximizing the margin between two classes. The overlap between two classes also affects the boundary of the trained SVM.

Both neural networks and SVMs, as discriminative classifiers, attempt to separate different classes in the feature space. For overlapping classes, the decision boundary tends to be complicated. If we ignore the substitution between overlapping classes, as for handwritten letter recognition, the overlapping classes can be merged into a metaclass and then the ordinary discriminative classifiers can be applied to this reduced class set problem. Koerich [10] designed several neural network classifiers for recognizing 52 letters, 26 upper-case letters, 26 lower-case letters and 26 metaclasses, respectively, and showed that the metaclass classifier outperforms the 52-class classifier (evaluated at metaclass level) and the combination of upper-case and lower-case classifiers.

Blumenstein et al. [11] merged 52 letters into 36 metaclasses: all upper-case letters except "ABDEGHNQRT" are merged with their lower-case letters, and use a neural network for 36-class classification. Camastra et al. [6] use one-versus-all SVM classifiers for classifying handwritten letters in 52 classes, 26 classes and adaptively merged classes according to the overlap degree between upper and lower cases. Using classifiers of 52 classes, 38 classes and 26 classes, they obtained test accuracies (evaluated at 26-metaclass level) of 89.20%, 90.05% and 89.61%, respectively.

Multi-labeled classification methods have not been applied to overlapping classes problems, but I will test it in this case. Multi-labeled classification is

generally transformed to multiple binary classification tasks, and different methods differ in the way of attaching binary labels to the training samples [4]. An effective method, called cross-training [3], uses each multi-labeled sample as the positive sample of each class it belongs to and not as negative sample for any of the labeled classes. For example, if a sample belongs to classes 'A' and 'a', it is used as positive sample when training the binary classifiers for 'A' and 'a', and as negative samples for the binary classifiers of other classes.

3 Partial Discriminative Training

Before describing the proposed partial discriminative training (PDT) scheme, I briefly review the training objectives of neural network classifiers.

3.1 Training of Neural Networks

Assume to classify a pattern (represented by a feature vector $\mathbf{x}$) to one of M classes $\{\omega_1, \ldots, \omega_M\}$. There are N training samples $(\mathbf{x}^n, c^n)$ (c^n is the class label of sample $\mathbf{x}^n$), $n = 1, \ldots, N$, for training a multi-class classifier. On an input pattern $\mathbf{x}$, the classifier outputs (sigmoidal) confidence values $y_k(\mathbf{x}, W)$ (W denotes the set of parameters) for classes $k = 1, \ldots, M$. The objective of neural network training is to minimize the squared error (SE):

$$\min_W SE = \min_W \sum_{n=1}^{N} \sum_{k=1}^{M} [y_k(\mathbf{x}^n, W) - t_k^n]^2, \tag{1}$$

where t_k^n denotes the target output:

$$t_k^n = \delta(c^n, k) = \begin{cases} 1, k = c^n, \\ 0, \text{otherwise.} \end{cases} \tag{2}$$

The SE in Eq. (1) can be re-written as

$$SE = \sum_{k=1}^{M} \sum_{n=1}^{N} [y_k(\mathbf{x}^n, W) - t_k^n]^2 = \sum_{k=1}^{M} E_k, \tag{3}$$

where $E_k = \sum_{n=1}^{N} [y_k(\mathbf{x}^n, W) - t_k^n]^2$ is the squared error of a binary classifier for class ω_k versus the others. Thus, the training of the multi-class neural network is equivalent to the training of multiple binary one-versus-all classifiers. Accordingly, the class output $y_k(\mathbf{x}, W)$ functions as the discriminant for separating class ω_k from the others.

The cross-entropy (CE) objective for neural networks can be similarly decomposed into multiple binary classifiers:

$$\begin{aligned} CE &= -\sum_{n=1}^{N} \sum_{k=1}^{M} [t_k^n \log y_k + (1 - t_k^n) \log(1 - y_k)]^2 \\ &= -\sum_{k=1}^{M} \sum_{n=1}^{N} [t_k^n \log y_k + (1 - t_k^n) \log(1 - y_k)]^2 \\ &= \sum_{k=1}^{M} CE_k. \end{aligned} \tag{4}$$

For multi-labeled classification, each sample $\mathbf{x}^n$ is labeled to belong to a subset of classes C^n. For training neural networks in such case, the objective is the same as Eq. (1), (3) or (4) except that the target output is changed to

$$t_k^n = \begin{cases} 1, k \in C^n, \\ 0, \text{otherwise.} \end{cases} \tag{5}$$

Due to the class modularity of objective functions SE and CE, for either single-labeled or multi-labeled classification, we can either train a multi-class classifier or multiple binary one-versus-all classifiers.

The overlapping classification problem is different from multi-labeled classification in that the training samples have single class labels, but I enhance the label of each sample with the allied classes (those overlapping with the labeled class) to convert the problem to be multi-labeled.

3.2 Partial Discriminative Training (PDT)

For overlapping classification with classifiers trained with single-labeled samples, the boundary between overlapping classes will by complicated by discriminative training to maximize the separation between overlapping classes. The over-complicated boundary will deteriorate the generalized classification performance and also affect the boundary between metaclasses. On the other hand, simply merging the overlapping classes into a metaclass will complicate the distribution of the metaclass.

If the substitution between overlapping classes is to be ignored, the training objective of neural networks, squared error (SE) or cross-entropy (CE), can be modified to ignore the error of the allied classes of each training sample. Denote the allied classes of ω_k as a set $\Lambda(k)$ (e.g., in alphanumeric recognition, the allied classes of 'O' are "o0"), the squared error of Eq. (3) is modified as

$$SE = \sum_{k=1}^{M} \sum_{n=1, k \notin \Lambda(c^n)}^{N} [y_k(\mathbf{x}^n, W) - t_k^n]^2. \tag{6}$$

This implies, the training pattern $\mathbf{x}^n$ is not used as negative sample for the allied classes of c^n. Note that the relation of alliance is symmetric, i.e., $k \in \Lambda(c) \Leftrightarrow c \in \Lambda(k)$

Excluding a training pattern from the negative samples of the allied classes of the label prevents the classifier from over-fitting the boundary between the labeled class and its allied classes (which are overlapping with the labeled class). Still, the number of classes remains unchanged (the structure of the multi-class classifier does not change), unlike in metaclass merging, the number of classes is reduced. Remaining the number of classes has the benefit that the classifier outputs confidence scores to each of the overlapping classes. If two allied classes are partially overlapped, a sample from the un-overlapped region can be classified to its class unambiguously. By class merging, however, the boundary between allied classes are totally ignored.

The PDT scheme can be applied to all types of binary classifiers, with multiple binary classifiers combined to perform multi-class classification. For multi-class classification using one-versus-all SVMs, when training an SVM for a class ω_k, if ω_k is an allied class of a sample from a different class, this sample is excluded from the negative samples of ω_k.

3.3 Specific Classifiers

I have applied the PDT scheme to five types of neural networks and SVMs with two types of kernel functions. The neural classifiers are single-layer neural network (SLNN), multi-layer perceptron (MLP), radial basis function (RBF) network [8], polynomial network classifier (PNC) [12, 13], and class-specific feature polynomial classifier (CFPC) [14]. Two one-versus-all SVM classifiers use a polynomial kernel and an RBF kernel, respectively.

The neural classifiers have a common nature that each class output is the sigmoidal (logistic) function of the weighted sum of values of the previous layer. In SLNN, the input feature values are directly linked to the output layer. The MLP that I use has one layerof hidden units and all the connecting weights are trained by back-propagation. The RBF network has one hidden layer of Gaussian kernel units, and in training, the Gaussian centers and variance values are initialized by clustering and are optimized together with the weights by error minimization. The PNC is a single-layer network with the polynomials of feature values as inputs. For reducing the number of polynomial terns, I use a PNC with the binomial terms of the principal components [13]. Unlike the PNC that uses a class-independent principal subspace, the CFPC uses class-specific subspaces as well as the residuals of subspace projection [14].

For saving the computation of projection onto class-specific subspaces, the CFPC is trained class by class [14], i.e., the binary one-versus-all classifiers are trained separately. The other four neural networks are trained for all classes simultaneously. The weights of the neural networks are trained by minimizing the squared error criterion by stochastic gradient descent.

The one-versus-all SVM classifier has multiple binary SVMs each separating one class from the others. In my implementation of SVMs using two types of kernel functions, the pattern vectors are appropriately scaled for the polynomial kernel, with the scaling factor estimated from the lengths of the sample vectors. For the Gaussian (RBF) kernel, the kernel width σ^2 is estimated from the variance of the sample vectors. I call the SVM classifier using polynomial kernel SVM-poly and the one using Gaussian kernel SVM-rbf. In partial discriminative training (PDT) of a binary SVM for class ω_k, the only change is to remove from negative samples the ones of the allied classes of ω_k.

4 Experimental Results

I evaluated the partial discriminative training (PDT) scheme and related methods with different classifiers on a public database of handwritten letters, C-Cube

[5, 6][1]. This database contains 57,293 samples of 52 English letters, partitioned into 38,160 training samples and 19,133 test samples. The samples were segmented from handwritten words, so the character shapes are very cursive and the number of samples per class is seriously imbalanced. In addition to confusion between upper-case and lower-case letters, the confusion between different case-insensitive letters is also considerable. By k-NN classification based on vector quantization, the authors ordered the overlap degree of upper/lower cases of each letter for merging the cases of selected letters. The database provides binary images as well as extracted feature values (34D) of the samples. Since my intention is to evaluate classifiers, I do not improve the features, but use the given features in the database.

I consider three numbers of classes as those in [5, 6]: 52 case-sensitive letters, 38 classes by merging the upper/lower cases of 14 letters ("CXOWYZMKJUNFVA"), and 26 case-insensitive letters. In the cases of 38 classes and 26 letters, each merged upper-case letter is allied with its lower-case and vice versa. In all cases, I set the number of hidden units of MLP as 100, the number of hidden units of RBF network as 150. The PNC uses linear and binomial terms of the original features without dimensionality reduction. The CFPC uses 25D class-specific subspaces. The SVM-poly uses 4-th order polynomial kernel, and the SVM-rbf uses an RBF kernel with kernel width fixed at 0.5 times the average within-class variance.

First, I trained four multi-class neural networks (SLNN, MLP, RBF, and PNC) with three training schemes optimizing the squared error criterion: ordinary discriminative training, PDT, and cross-training (enhancing the label of each sample with its allied classes). The accuracies on test samples are shown in Table 1, where each row gives the accuracies evaluated at a number of metaclasses (52, 38 or 26, within-metaclass substitution is ignored), and each column corresponds to a number of metaclasses in training. By ordinary discriminative training, the number of classes is reduced by class merging, whereas by PDT and cross-training, the number of classes remains unchanged but the samples are attached allied classes or multi-labeled. Each classifier can be evaluated at a reduced number of classes by ignoring within-metaclass substitution. At each row (evaluated at a number of metaclasses), the highest accuracy is highlighted in bold face, and the accuracies of merged metaclass training and PDT are boxed.

Apparently, the ordinary all-class discriminative training (3rd column of Table 1) gives the highest accuracy for 52-class classification. This is reasonable because all the classes are aimed to be separated in this case, while PDT ignores the separation between allied classes. When evaluated at reduced number of classes, however, merged metaclass training (4th and 5th columns) and PDT (6th and 7th columns) may give higher accuracies than all-class training. In seven of eight cases (two class numbers 38 and 26 combined with four classifiers), PDT gives higher accuracies than all-class training and merged metaclass training. The inferior performance of cross-training can be explained that the

[1] Downloadable at http://ccc.idiap.ch/

Table 1. Test accuracies (%) of multi-class neural networks on the C-Cube Letter database. Each row gives the accuracies evaluated at a number of metaclasses, and each column corresponds to a number of metaclasses in training. 4th and 5th columns correspond to merged metaclass training.

		Discriminative training			Partial training		Cross-training	
Classifier	#Class	52	38	26	38	26	38	26
SLNN	52	**66.15**			65.49	65.23	52.66	34.18
	38	71.93	70.40		**72.75**	72.47	70.38	52.66
	26	72.53	70.94	67.94	73.36	**73.44**	70.92	67.98
MLP	52	**78.97**			78.21	78.20	60.83	45.46
	38	84.64	84.98		**85.42**	85.06	84.37	68.08
	26	85.00	84.34	84.42	**85.79**	85.57	84.81	83.62
RBF	52	**78.06**			77.71	77.75	61.20	44.84
	38	83.76	**84.37**		84.25	84.31	84.00	66.68
	26	84.16	84.72	84.28	84.61	**84.81**	84.36	83.70
PNC	52	**81.09**			80.67	80.64	63.34	43.34
	38	86.87	87.11		**87.62**	87.61	86.80	65.41
	26	87.18	87.42	86.29	87.93	**88.03**	87.10	85.65

framework of multi-labeled classification does not match the problem of overlapping classification.

On three one-versus-all classifiers (CFPC, SVM-poly and SVM-rbf), I used two training schemes: ordinary discriminative training and PDT. The test accuracies are shown in Table 2. Again, all-class discriminative training (3rd column of Table 2) gives the highest accuracies for 52-class classification. When evaluated at reduced number of metaclasses, both merged metaclass training (4th and 5th columns) and PDT (6th and 7th columns) gives higher accuracies than all-class training. For the CFPC, PDT outperforms merged metaclass training. For the SVM classifiers, merged metaclass training gives the highest accuracies of metaclass classification, but the accuracies of PDT are closely competitive.

Overall, when evaluated at metaclass level, PDT gives higher accuracies than ordinary all-class discriminative training on all the seven classifiers, outperforms merged metaclass training on five neural classifiers, and performs comparably with merged metaclass training on two SVM classifiers. On the C-Cube database of handwritten letters, the remaining classification error rate of 26 metaclasses is still appreciable (over 10%) because of the inherent confusion of handwritten shapes between different letters. This can be alleviated by extracting more discriminant features which provide better between-class separation.

Compared to merged metaclass training, PDT has an advantage that it still outputs confidence scores for all classes. Thus, if a pattern of partially overlapping classes resides in the non-overlape region, it can still be classified unambiguously. By merged metaclass training, however, the boundary between partially overlapping classes is totally ignored.

Table 2. Test accuracies (%) of one-versus-all classifiers on the C-Cube Letter database. Each row gives the accuracies evaluated at a number of metaclasses, and each column corresponds to a number of metaclasses in training. 4th and 5th columns correspond to merged metaclass training.

		Discriminative training			Partial training	
Classifier	#Class	52	38	26	38	26
CFPC	52	**81.07**			80.73	80.71
	38	86.76	86.65		**87.22**	87.21
	26	87.09	86.99	85.86	87.56	**87.70**
SVM-poly	52	**82.19**			81.97	81.88
	38	88.13	**88.66**		88.61	88.51
	26	88.41	88.94	**89.03**	88.88	88.99
SVM-rbf	52	**82.65**			82.42	82.35
	38	88.73	**89.12**		89.10	89.02
	26	89.00	89.33	**89.43**	89.39	89.40

5 Conclusion

This paper proposed a partial discriminative training (PDT) scheme for classification of overlapping classes. It is applicable to all types of binary one-versus-all classifiers, including neural networks and SVM classifiers. The rationale of PDT is to ignore the difference between overlapping classes in training so as to improve the separation between metaclasses. Experiments in handwritten letter recognition show that when evaluated at metaclass level, the PDT scheme mostly outperforms ordinary all-class discriminative training. Compared to merged metaclass training, the PDT gives higher or comparable accuracies at metaclass level and provides more informative confidence scores. The PDT scheme is especially useful for such applications where overlapping classes are not necessarily discriminated before contextual information is exploited. This work will be extended by experimenting with different datasets.

Acknowledgements

This work is supported by the Hundred Talents Program of Chinese Academy of Sciences and the National Natural Science Foundation of China (NSFC) under grant no. 60775004 and grant no.60723005.

References

1. B.-L. Lu, M. Ito, Task decomposition and modular combination based on class relations: a modular neural network for pattern classification, *IEEE Trans. Neural Networks*, 10 (5): 1244-1256, 1999.

2. I.T. Podolak, Hierarchical classifier with overlapping class groups, *Expert Systems with Applications*, 34(1): 673-682, 2008.
3. M.R. Boutell, J. Luo, X. Shen, C.M. Browm, Learning multi-label scene classification, *Pattern Recognition*, 37 (9): 1757-1771, 2004.
4. G. Tsoumakas, I. Katakis, Multi-label classification: an overview, *Int. J. Data Warehousing and Mining*, 3 (3): 1-13, 2007.
5. F. Camastra, M. Spinetti, A. Vinciarelli, Offline cursive character challenge: a new benchmark for machine learning and pattern recognition algorithms, *Proc. 18th ICPR*, Hong Kong, 2006, pp.913-916.
6. F. Camastra, A SVM-based cursive character recognizer, *Pattern Recognition*, 40 (12): 3721-3727, 2007.
7. K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd edition, Academic Press, 1990.
8. C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.
9. C.J.C. Burges, A tutorial on support vector machines for pattern recognition, *Knowledge Discovery and Data Mining*, 2(2): 1-43, 1998.
10. A.L. Koerich, Unconstrained handwritten character recognition using different classification strategies, M. Gori and S. Marinai (Eds.), *Proc. 1st IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 2003, pp.52-56.
11. M. Blumenstein, X.Y. Liu, B. Verma, An investigation of the modified direction feature for cursive character recognition, *Pattern Recognition*, 40(2): 376-388, 2007.
12. J. Shürmann, *Pattern Classification: A Unified View of Statistical and Neural Approaches*, Wiley Interscience, 1996.
13. U. Kreßel, J. Schürmann, Pattern classification techniques based on function approximation, *Handbook of Character Recognition and Document Image Analysis*, H. Bunke and P.S.P. Wang (eds.), World Scientific, 1997, pp.49-78.
14. C.-L. Liu, H. Sako, Class-specific feature polynomial classifier for pattern classification and its application to handwritten numeral recognition, *Pattern Recognition*, 39 (4): 669-681, 2006.