A MAXIMAL FIGURE-OF-MERIT LEARNING APPROACH TO MAXIMIZING MEAN AVERAGE PRECISION WITH DEEP NEURAL NETWORK BASED CLASSIFIERS

Kehuang Li † *Zhen Huang* *You-Chi Cheng* *Chin-Hui Lee*

School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332-0250, USA
† kehle@gatech.edu

ABSTRACT

We propose a maximal figure-of-merit (MFoM) learning framework to directly maximize mean average precision (MAP) which is a key performance metric in many multi-class classification tasks. Conventional classifiers based on support vector machines cannot be easily adopted to optimize the MAP metric. On the other hand, classifiers based on deep neural networks (DNNs) have recently been shown to deliver a great discrimination capability in automatic speech recognition and image classification as well. However, DNNs are usually optimized with the minimum cross entropy criterion. In contrast to most conventional classification methods, our proposed approach can be formulated to embed DNNs and MAP into the objective function to be optimized during training. The combination of the proposed maximum MAP (MMAP) technique and DNNs introduces nonlinearity to the linear discriminant function (LDF) in order to increase the flexibility and discriminant power of the original MFoM-trained LDF based classifiers. Tested on both automatic image annotation and audio event classification, the experimental results show consistent improvements of MAP on both datasets when compared with other state-of-the-art classifiers without using MMAP.

Index Terms— Nonlinearity, average precision, deep neural networks, maximal figure-of-merit

1. INTRODUCTION

During the process of designing a classifier, density based methods use joint probability density functions to minimize the Bayes risk, e.g., Gaussian mixture models (GMMs) [1] and hidden Markov models (HMMs) [2]. On the other hand, discriminant based methods define discriminant function to evaluate class scores and learn their parameters using certain optimization criteria [3]. Hybrid methods by combining these two approaches have also been explored [4]. Methods based on discriminative training (e.g., support vector machines [5]) are more flexible since they link data directly to the corresponding labels without modelling the relationship between the feature space and parameter space. However, it is always an interesting issue on how to train such models to optimize the desired performance of a task. Support vector machines (SVMs) maximize the minimum margin between samples and the separation hyperplane described by a linear discriminant function (LDF) [5]. Maximum mutual information (MMI) [6] maximizes the mutual information among competing classes, while minimum classification error (MCE) [3] can directly minimize the empirical classification error of the training data.

A performance gap between the training and testing data has been observed in practice making it hard to design a proper discriminative training (DT) criterion. SVMs have shown some generalization abilities [5], but another group of methods would suggest to directly optimize the desired performance metric used in testing. Among them, an interesting example is maximal figure-of-merit (MFoM) [7], which can be used to optimize precision, recall, F1-measure and receiver operating characteristic (ROC) as well [7, 8]. The MFoM learning approach has been successfully used in text categorization, automatic image annotation and audio event classification [8, 9]. The key idea of MFoM is to approximate the performance metric with a differentiable function, so that gradient based optimization methods can be properly applied. Similar to the ROC metric, we propose an approach to maximizing AP, which is widely used in multi-class classification. The difference between the ROC metric and the AP metric is that in AP every positive sample is an operating point with weight related to its normalized rank which turns out to be a difficult metric to optimise [10]. Compared with the AP approximation method used in [10], our proposed approach uses differentiable functions to estimate ranks so that the gradient of the objective function can be calculated directly.

Another problem of the conventional MFoM approaches is that it is hard to introduce nonlinearity to the discriminant function. There are mainly two branches of methods to deal with the problem. One is to use nonlinear discriminant function like quadratic functions [11], and the other is to apply kernel functions to map features to high dimension Hilbert space [12]. Using a nonlinear discriminant function will make the objective function hard to solve, while kernelization approaches need to deal with the increased dimensions when there are hundreds of thousands of training samples. On the other hand, deep neural networks with discriminative training are widely used in automatic speech recognition (ASR) [4, 13, 14]. DNNs introduce nonlinearity naturally and are able to handle massive data sets [13]. Therefore in this study we propose to introduce AP based MFoM to discriminative training of DNN based classifiers. DNNs in massive data problems use random part of data on each training step. MFoM needs an approximation of the performance metric on training data, and in most cases it is hard to have an accurate approximation based on a small subset of the training data. However, in our experiment, ranking based performance metric can properly deal with the problem, so that stochastic gradient descent learning methods on DNNs can work properly.

2. GENERAL FORMULATION OF MFOM TRAINING

Let us considering a multi-class classification problem with M classes $\{C_k|k = 1, \cdots, M\}$. Suppose the training dataset is $\boldsymbol{X} = \{\boldsymbol{x}_i \in \mathbb{R}^n | i = 1, \cdots, N\}$, and $\boldsymbol{L} = \{\boldsymbol{y}_i | i = 1, \cdots, N\}$ is their label set, where $\boldsymbol{y}_i \subset \{1, \cdots, M\}$. Every data must be

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

assigned to at least one class, $\boldsymbol{X} = \bigcup_{k=1}^{M} C_k$. For multi-label problems, total size of all C_k's must be greater than that of $\boldsymbol{X}$. Assume for each class C_k, a discriminant function is defined as $g_k(\boldsymbol{x}; \boldsymbol{\omega}_k)$, where $\boldsymbol{\omega}_k$ are parameters for the class, and they belong to parameter set $\boldsymbol{\Omega} = \{\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_M\}$. The predicted class of data $\boldsymbol{x}$ is $\tilde{C}_y = \arg\max_y g_y(\boldsymbol{x}; \boldsymbol{\omega}_y)$. If there are multi-labels, first several candidates should be considered or a threshold can help. Normally, LDF is adopted, which means $g_k(\boldsymbol{x}; \boldsymbol{\omega}_k) = \boldsymbol{\omega}_k^T \hat{\boldsymbol{x}}$, where $\hat{\boldsymbol{x}} = (\boldsymbol{x}; 1)$ makes the notation of weights and bias in LDF consistent.

To qualify the classification performance, a misclassification measure [3] is defined for each class C_k as

$$d_k(\boldsymbol{x}) = -g_k(\boldsymbol{x}; \boldsymbol{\omega}_k) + \frac{1}{\eta} \ln \left(\frac{1}{M} \sum_{j \neq k, j=1}^{M} e^{\eta g_j(\boldsymbol{x}; \boldsymbol{\omega}_j)} \right), \quad (1)$$

where η is a smoothing constant. A greater η will make competing classes, which are summed up in the right side of Eq. (1), with high scores have greater influence on the misclassification measure.

The sign of the misclassification measure indicates whether the predicted class is correct or not. A positive sign means the prediction is incorrect, and vice versa. Furthermore, a sigmoid function is embedded to estimate the count on misclassified samples. That function is always called loss function [3], $l_k(\boldsymbol{x}) = \frac{1}{1+e^{-\alpha_k d_k(\boldsymbol{x}) + \beta}}$, where α_k and β are two parameters controlling the active region of the 0-1 approximation. α_k can be estimated based on the distribution of $d_k(\boldsymbol{x})$ [15], and in this paper we take $\beta = 0$.

3. FORMULATION OF MMAP

The loss function defined in last section can only deal with misclassification count, while in the calculation of AP, we also need an estimation of the ranks of data samples. In [16], gradient $\nabla_{\boldsymbol{\omega}} AP$ is got by estimating $\frac{\partial AP_k}{\partial d_k(\boldsymbol{x}_i)} \nabla_{\boldsymbol{\omega}} d_k(\boldsymbol{x}_i)$ and summing over all $\boldsymbol{x}_i$. It was claimed to be efficient, but a lot of smoothing would be necessary. Suppose $s = g(\boldsymbol{x})$ is the score of a data sample, where $g(\cdot)$ is $g_k(\cdot; \omega_k)$ when we are calculating scores for class C_k. The discrete form of AP for class C_k is

$$AP_k = \frac{1}{|C_k|} \sum_{\boldsymbol{x}_i \in C_k} \frac{\mathrm{rank}(s_i; \{s_j | \boldsymbol{x}_j \in C_k\})}{\mathrm{rank}(s_i; \{s_j | \boldsymbol{x}_j \in \boldsymbol{X}\})}, \quad (2)$$

where $|C_k|$ denotes the cardinality of set C_k, and $\mathrm{rank}(s; \boldsymbol{S})$ is the rank of s in set $\boldsymbol{S}$.

A smoothed pair-wise rank function [8] can be used to estimate ranking relation between two samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and it has the form

$$r_k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{1 + e^{-\alpha_k [d_k(\boldsymbol{x}_i) - d_k(\boldsymbol{x}_j)]}}. \quad (3)$$

If s_i is ranked higher than s_j, $r_k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ will be less than 0.5 and approaches 0 as either α_k or $d_k(\boldsymbol{x}_i) - d_k(\boldsymbol{x}_j)$ becomes pretty large. And summing Eq. (3) over all $\boldsymbol{x}_j$ in set C_k will give an estimation of $\mathrm{rank}(s_i; \{s_j | \boldsymbol{x}_j \in C_k\}) \approx \sum_{\boldsymbol{x}_j \in C_k} r_k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, while $\mathrm{rank}(s_i; \{s_j | \boldsymbol{x}_j \in \boldsymbol{X}\}) \approx \sum_{\boldsymbol{x}_j \in \boldsymbol{X}} r_k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

For M-class classification, we will generally use the mean of all APs of different classes to evaluate the overall performance. A regularize term could be added to extend generalization ability. In this point, to maximize mean-AP is to solve the following constrained optimizing problem,

$$\begin{aligned} \min \quad & -\frac{1}{M} \sum_{k=1}^{M} AP_k \\ \text{s.t.} \quad & \frac{1}{2} \|\boldsymbol{\omega}_k\|_2^2 = 1, k = 1, \cdots, M. \end{aligned} \quad (4)$$

Applying a penalty term ρ, we can transform (4) to a non-constrained programming problem, $f(\boldsymbol{\Omega}) = -\frac{1}{M} \sum_{k=1}^{M} AP_k + \frac{\rho}{2} \sum_{k=1}^{M} \|\boldsymbol{\omega}_k\|_2^2$.

We adopt L-BFGS to solve the optimization problem. L-BFGS is a gradient based convex programming algorithm being widely used in many areas [17, 18, 19]. It has been shown that L-BFGS can work on non-convex problems as well [20]. Replace rank terms in $f(\boldsymbol{\Omega})$ with their approximation, we can get the gradient of $f(\boldsymbol{\Omega})$ by taking partial derivative of it as in Table 1.

4. MMAP ON NEURAL NETWORKS

To apply our proposed MMAP method on deep neural networks (will be denoted as MMAP-DNN), we shall change the form of our equations in some degree. Neural networks are composed of input, output and hidden layers. Normally, in deep belief networks (DBNs) the output layer of the neural network will adopt softmax functions as the final output units to have an estimation of posterior probabilities,

$$P(C_k | \boldsymbol{x}) \approx \frac{e^{z_k}}{\sum_{k'=1}^{M} e^{z_{k'}}}, \quad (5)$$

where $z_{k'}$'s are values fed to softmax function of class C_k. If we think of the output of the last hidden layer as new features mapped from the original data space, and let us denote them as $\boldsymbol{u}$ and $\boldsymbol{U} = \{\boldsymbol{u}_i \in \mathbb{R}_m^+ | i = 1, \cdots, N\}$, then z_k is the inner product of $\boldsymbol{u}$ and the weight $\boldsymbol{w}$ times a constant, $z_k = \eta \boldsymbol{w}_k^T \boldsymbol{u}$, which has a similar form to LDF. Furthermore, the final output of the softmax functions can be linked to an exponential form of the misclassification measure by

$$d_k'(\boldsymbol{z}) = \frac{1}{\eta} \ln \left(\frac{1}{M-1} \left(\frac{1}{\mathrm{softmax}(\boldsymbol{z}, k)} - 1 \right) \right), \quad (6)$$

where $\boldsymbol{z} = (z_1, \cdots, z_M)^T$, and $\mathrm{softmax}(\boldsymbol{z}, k)$ is defined as the right side of Eq. (5). Since we only care about the sign of misclassification measure, we shall see that η can be combined to parameter vector $\boldsymbol{w}$'s. Note that the mapping from $\boldsymbol{x}$ to $\boldsymbol{u}$, $\mathbb{R}^n \mapsto \mathbb{R}_+^m$, is a nonlinear mapping. Thus by adding our MMAP on neural network, we improve the discriminant power of neural network, and introduce a nonlinear space transform to original MFoM approach. The smoothed pair-wise rank function, in neural network, would be

$$r_k'(\boldsymbol{z}_i, \boldsymbol{z}_j) = \left[1 + \left(\frac{\frac{1}{\mathrm{softmax}(\boldsymbol{z}_j, k)} - 1}{\frac{1}{\mathrm{softmax}(\boldsymbol{z}_i, k)} - 1} \right)^{\frac{\alpha_k}{\eta}} \right]^{-1}. \quad (7)$$

In general, DNNs will be trained using a random subset of the training data at every iteration. We will not be able to estimate α_k on each iteration. If we set α_k to be η, α_k and η will no longer exist in Eq. (7). Actually, by doing so, we force neural network to learn $\eta \boldsymbol{w}$ instead of $\boldsymbol{w}$ where η can be different from class to class.

During the back-propagation stage of DNN training, the gradient on $\boldsymbol{z}$ should be estimated. In case of the cross-entropy (CE) criterion, in which DNNs are trained to minimize CE between output units, the gradient per sample is the absolute error of the output. In our MMAP training, the gradient for each data $\boldsymbol{x}$ is not just related to the output error but also correlated with other data samples. To improve the

Table 1. Discriminant training target functions and their gradients

	Objective	Gradient
MCE	$\sum_{k=1}^{M}\sum_{i=1}^{N} v_k(\boldsymbol{x}_i) l_k(\boldsymbol{x}_i)$	$\sum_{k=1}^{M}\sum_{i=1}^{N} v_k(\boldsymbol{x}_i)\nabla_{\boldsymbol{\omega}} l_k(\boldsymbol{x}_i), \quad v_k(\boldsymbol{x}_i)=\begin{cases}1 & ,\boldsymbol{x}_i\in C_k\\ -1, & \boldsymbol{x}_i\notin C_k\end{cases}$
MFoM-μF1	$-\frac{2\sum_{k=1}^{M}\sum_{\boldsymbol{x}_i\in C_k}(1-l_k(\boldsymbol{x}_i))}{\sum_{k=1}^{M}\left(\lvert C_k\rvert+\sum_{i=1}^{N}(1-l_k(\boldsymbol{x}_i))\right)}$	$\sum_{k=1}^{M}\sum_{i=1}^{N}\left(\frac{B}{A^2}-\frac{v_k(\boldsymbol{x}_i)}{A}\right)\nabla_{\boldsymbol{\omega}} l_k(\boldsymbol{x}_i), \quad A=\sum_{k=1}^{M}\left(\lvert C_k\rvert+\sum_{i=1}^{N}(1-l_k(\boldsymbol{x}_i))\right)$ $B=\sum_{k=1}^{M}\left(\lvert C_k\rvert-\sum_{i=1}^{N} v_k(\boldsymbol{x}_i)(1-l_k(\boldsymbol{x}_i))\right)$
MMAP	$-\frac{1}{M}\sum_{k=1}^{M}\sum_{\boldsymbol{x}_i\in C_k}\frac{1}{\lvert C_k\rvert}\frac{\sum_{\boldsymbol{x}_j\in C_k} r_k(\boldsymbol{x}_i,\boldsymbol{x}_j)}{\sum_{\boldsymbol{x}_j\in\boldsymbol{X}} r_k(\boldsymbol{x}_i,\boldsymbol{x}_j)}$	$\frac{1}{2M}\sum_{k=1}^{M}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{Q_{k,i}}{P_{k,i}^2}-\frac{v_k(\boldsymbol{x}_i)}{P_{k,i}}\right)\nabla_{\boldsymbol{\omega}} r_k(\boldsymbol{x}_i,\boldsymbol{x}_j), \quad P_{k,i}=\lvert C_k\rvert\sum_{j=1}^{N} r_k(\boldsymbol{x}_i,\boldsymbol{x}_j)$ $Q_{k,i}=\lvert C_k\rvert\sum_{j=1}^{N} v_k(\boldsymbol{x}_j) r_k(\boldsymbol{x}_i,\boldsymbol{x}_j)$
MMAP-DNN	$-\frac{1}{M}\sum_{k=1}^{M}\sum_{\boldsymbol{x}_i\in C_k^{'}}\frac{1}{\lvert C_k^{'}\rvert}\frac{\sum_{\boldsymbol{x}_j\in C_k} r_k^{'}(\boldsymbol{z}_i,\boldsymbol{z}_j)}{\sum_{\boldsymbol{x}_j\in\boldsymbol{X}} r_k^{'}(\boldsymbol{z}_i,\boldsymbol{z}_j)}$	$\frac{1}{2M}\sum_{k=1}^{M}\sum_{i=1}^{N^{'}}\sum_{j=1}^{N^{'}}\left(\frac{Q_{k,i}^{'}}{P_{k,i}^{'2}}-\frac{v_k(\boldsymbol{x}_i)}{P_{k,i}^{'}}\right)\nabla_{\boldsymbol{z}} r_k^{'}(\boldsymbol{z}_i,\boldsymbol{z}_j), \quad P_{k,i}^{'}=\lvert C_k^{'}\rvert\sum_{j=1}^{N^{'}} r_k^{'}(\boldsymbol{z}_i,\boldsymbol{z}_j)$ $Q_{k,i}^{'}=\lvert C_k^{'}\rvert\sum_{j=1}^{N^{'}} v_k(\boldsymbol{x}_j) r_k^{'}(\boldsymbol{z}_i,\boldsymbol{z}_j)$

generalization and decrease the complexity, DNNs are trained by applying stochastic gradient descent with momentum [21] on some mini-batches of the training set. In each iteration, only subsets of M classes' sets might be chosen. Let us assume the size of the mini-batch is $N^{'}$ and denote the subset of C_k as $C_k^{'}$, which can even be $\varnothing$. Though mini-batch methods show great power on DNN training, AP estimated on mini-batch will be different from the exact AP of whole data set. It could be a problem if we demands accurate estimations of AP. But if we consider APs as a contributing part of the weight assigned to the gradient of each data sample, it comes out with non-accurate AP yet we can still give greater weights to positive samples with low scores, and give smaller weights to positive samples with high scores. Thus low ranked positive samples are more likely to increase their ranks at the next iteration, while high ranked ones will not grow too fast. Similarly, negative samples will be forced to low ranks by different weights based on their observed ranks.

5. EXPERIMENTS AND RESULTS

We test our proposed algorithm on data sets of auto image annotation (AIA) and audio event classification problems. We use Corel 5k for the first part which has about 5000 images labelled with 374 key concepts that each image might be labelled with more than one key word. It is divided into training and testing sets with 4500 and 500 samples, respectively. 44 out of 374 key words that occur more than 100 times in the training set were selected in our experiment, and they all appear in the testing set. Thus the real size of the training set is 4350 and that of the testing set is 485. For the second part, we use Buffalo 3k, which is an expansion of the data set used in [22] used for video event classification and collected from the internet. It has 3056 video clips, among them 1327 are labelled with 30 event notations. Each clip can have just one label. In our experiment, only the audio track of the data set is used, thus we dropped out 17 clips without audio tracks. We used support vector machines

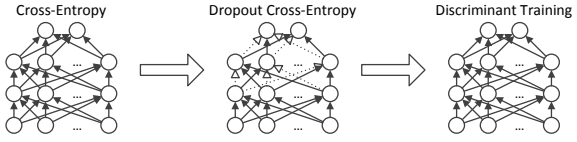

Fig. 1. Training neural network of audio event classifier

trained by LIBSVM [23] and deep neural networks trained using the Kaldi tool kit [24] as our baselines. In our experiments, we train all DNNs without pre-training since pre-training has been shown to be not always necessary [21].

5.1. Experimental Setup and Task Description

5.1.1. AIA Data

Features of Corel 5k were extracted using the methods given in [25]. A dimension of 600 was adopted to create both texture and colour histogram features. Base MMAP models for each of the two different features were first trained using method introduced in section 3, and their output for the 44 classes were then combined to 88-dimension vectors to be used in late fusion. We use a L2 penalty of 0.001 in training of the texture and colour models, and that of 0.005 in fusion model training. On the other hand, to build the MMAP-DNN models, one DNN for each of the two different features was first trained using the CE criterion and followed by one fusion neural network fed with their outputs. Then the MMAP criterion was further used to train the fusion model with the initial parameters learnt from CE optimization. Two high level feature models both have two hidden layers with 1024 hidden nodes, while the fusion model has one layer of 256 hidden nodes. To compare the performance on different features, we also trained models for each feature type with the MMAP criterion, but they were not used in the fusion step. A batch size of 32 was used, and we set the learning rate to be 0.04 with a momentum factor 0.9 and L2-penalty 5×10^{-6} in DNN training. For the SVM baseline, we used RBF kernels. The fusion step was the same as that for our MMAP model that the output scores for both features were concatenated as a new feature for the fusion model.

5.1.2. Audio Event Classification Data

On Buffalo 3k, we split the data set randomly into two subsets, with 678 training and 632 testing clips. We extract the 39-dimension mel-frequency cepstral coefficients (MFCCs) on 25 ms windows at a frame rate of 10 ms. K-means clustering was used to build a codebook of size 1024. Then we created feature using bag-of-word similar to [26], which will give us a 1024 dimension feature vector per clip. Those feature vectors were then normalized by dividing every element in a vector with their sum. We have the MMAP models and SVMs trained on the 678 training clips and tested on the remaining 632 clips. In linear MMAP training, L2-penalty was 0.002. In the MMAP-DNN model, we used 2 hidden layers each with 512 hidden

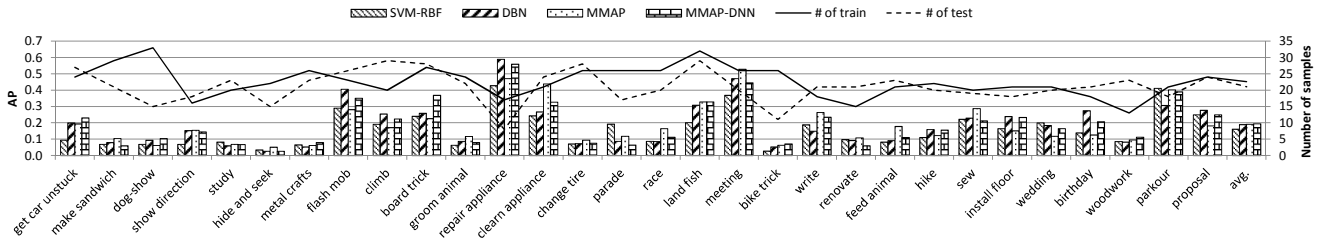

Fig. 2. Mean average precision of all classes in Buffalo 3k

nodes. All models were trained as multi-class classifiers. It could be a really difficult task to train a 30-classes classifier with only 678 samples, especially for DNNs.

To reduce the over-fitting problem, an L2 penalty was used and we also adopted the mini-batch and drop-out [27] mechanism. In the stage of training DNNs to minimize classification errors, no L2-penalty was used but the momentum factor was set at 0.91. In the stage of drop-out training, the drop-out rate was 0.5. And in the discriminative training stage, we used a 1×10^{-4} L2-penalty and a 0.9 momentum factor. The learning rate was set to 0.04 in the CE stage, and then set to 0.004 in the other two stages.

5.2. Results and Discussion

Results of our experiments are listed in Table 2 where it shows the mean average precision for the two data sets in the two rightmost columns. We compare SVM with the RBF kernel, DBN with the CE criterion, linear MMAP, MMAP embedded DNN, and a late fusion of the output of SVM and MMAP-DNN denoted as SVM+MMAP-DNN. The proposed MMAP method obtains a relative 6.5% and 19% gains in MAP when compared with the SVM systems. Neural network trained to minimize errors has the worst performance in Corel 5k, it could be caused by multiple labels and insufficient training data. However, our proposed method improves the result of neural network by 26% in Corel 5k. On Buffalo 3k, DBN with a relative large node size learnt pretty well and MMAP doesn't bring much gain to it. It seems DBN, MMAP and MMAP-DNN shows similar MAP, but their performance on each class are different, which can be reflected by the top 5 concepts shown in Table 3 that are different between SVM and DNN. The improved fusion results shown in the bottom row in Table 2 clearly indicate the complimentary nature.

Table 2. MAP comparison for different classifiers

	Corel 5k	Buffalo 3k
SVM	0.356	0.161
DBN	0.301	0.189
MMAP	0.346	0.191
MMAP-DNN	**0.379**	**0.195**
SVM+MMAP-DNN	**0.432**	**0.201**

In Figure 2, bars show MAPs of different classes and their corresponding classifiers, and lines indicate the number of training samples in each class showing on the right-side vertical axis. We can find that the performance of different classes is somewhat correlated with the training sample size in that class. Classes with more training samples are always able to get a better performance. However, MMAP is able to give an extra bonus gain to some classes that have only few training samples, e.g., the *write* and *sew* classes have less than 20 training samples and MMAP still attains a 0.25 average precision on these classes. DBN shows a great performance in the *dog-show*, *repair appliance* and *birthday* categories when compared with other methods. These events all contain specific sounds, and DNN is capable of reproducing some characterized samples even they are only a few. MMAP-DNN usually has a combinational performance of MMAP and DBN. The only case SVM outperforms is *parade* class, which contains a lot of different sounds.

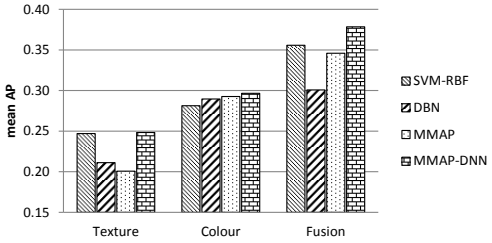

Fig. 3. Performance on Corel 5k

Table 3. Top 5 words with best performance

	SVM	DBN	MMAP	MMAP-DNN
Corel 5k	jet plane tracks birds ocean	ocean horses polar tiger sun	ocean horses cat bear birds	ocean horses polar bear tiger
Buffalo 3k	repair appl. parkour meeting flash mob proposal	repair appl. meeting flash mob land fish parkour	meeting repair appl. clean appl. parkour land fish	repair appl. meeting parkour board trick flash mob

6. SUMMARY

In this paper, we have presented a technique to maximize the mean average precision of multi-class multi-label classifiers. By further embedding DNNs into the MAP objective function we gain a flexibility to use nonlinear classifiers to improve the discriminative power of conventional linear classifiers. Experiments on two different data sets show an up to 25% relative MAP improvement. In the future, we will study problems with large size training sets and ways to further improve the MMAP-DNN framework.

7. ACKNOWLEDGMENT

The authors would like to thank Chao Weng for fruitful discussion during our experiments. The Buffalo 3k dataset is shared by Chengliang Xu who is a PhD student of SUNY Buffalo.

8. REFERENCES

[1] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using gaussian mixture speaker models," *IEEE Trans. on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[2] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[3] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 5, no. 3, pp. 257–265, 1997.

[4] A.-R. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *INTERSPEECH*, 2010, pp. 2846–2849.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[6] S. Kapadia, V. Valtchev, and S. J. Young, "Mmi training for continuous phoneme recognition on the timit database," in *Acoustics, Speech, and Signal Processing (ICASSP), 1993 IEEE International Conference on*. IEEE, 1993, vol. 2, pp. 491–494.

[7] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua, "A maximal figure-of-merit (mfom)-learning approach to robust classifier design for text categorization," *ACM Trans. on Information Systems (TOIS)*, vol. 24, no. 2, pp. 190–218, 2006.

[8] S. Gao, C.-H. Lee, and J. H. Lim, "An ensemble classifier learning approach to roc optimization," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*. IEEE, 2006, vol. 2, pp. 679–682.

[9] Z. Huang, Y.-C. Cheng, K. Li, V. Hautamäki, and C.-H. Lee, "A blind segmentation approach to acoustic event detection based on i-vector," in *INTERSPEECH*, 2013.

[10] I. Kim and C.-H. Lee, "An efficient gradient-based approach to optimizing average precision through maximal figure-of-merit learning," *Journal of Signal Processing Systems*, pp. 1–11, 2013.

[11] Y. Wang and Q. Huo, "Sample-separation-margin based minimum classification error training of pattern classifiers with quadratic discriminant functions," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1866–1869.

[12] B. Byun and C.-H. Lee, "A kernelized maximal-figure-of-merit learning approach based on subspace distance minimization," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2068–2071.

[13] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[14] O. Vinyals, S. V. Ravuri, and D. Povey, "Revisiting recurrent neural networks for robust asr," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4085–4088.

[15] H. Watanabe, J. Tokuno, T. Ohashi, S. Katagiri, and M. Ohsaki, "Minimum classification error training with automatic setting of loss smoothness," in *Machine Learning for Signal Processing (MLSP), 2011 IEEE International Workshop on*. IEEE, 2011, pp. 1–6.

[16] C. Ma and C.-H. Lee, "An efficient gradient computation approach to discriminative fusion optimization in semantic concept detection," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[17] J. Nocedal, "Updating quasi-newton matrices with limited storage," *Mathematics of computation*, vol. 35, no. 151, pp. 773–782, 1980.

[18] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[19] J. J. Moré and D. J. Thuente, "Line search algorithms with guaranteed sufficient decrease," *ACM Trans. on Mathematical Software (TOMS)*, vol. 20, no. 3, pp. 286–307, 1994.

[20] J. J. E. Dennis and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, vol. 16, Siam, 1983.

[21] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[22] P. Das, C. Xu, R. F. Doell, and J. J. Corso, "A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 2634–2641.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[24] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.

[25] S. Gao, D.-H. Wang, and C.-H. Lee, "Automatic image annotation through multi-topic text categorization," in *Acoustics, Speech and Signal Processing (ICASSP), 2006 IEEE International Conference on*. IEEE, 2006.

[26] Y.-G. Jiang, X. Zeng, G. Ye, D. Ellis, S.-F. Chang, S. Bhattacharya, and M. Shah, "Columbia-ucf trecvid2010 multimedia event detection: Combining multiple modalities, contextual concepts, and temporal matching," in *TRECVID*, 2010.

[27] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.