Sample-Specific Late Fusion for Visual Category Recognition

Dong Liu[†], Kuan-Ting Lai[‡§], Guangnan Ye[†], Ming-Syan Chen[‡§], Shih-Fu Chang[†]
[†]Dept. of Electrical Engineering, Columbia University, USA
[‡]Graduate Institute of Electrical Engineering, National Taiwan University, Taiwan
[§]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
{dongliu,gnye,sfchang}@ee.columbia.edu, {ktlai,mschen}@arbor.ee.ntu.edu.tw

Abstract

Late fusion addresses the problem of combining the prediction scores of multiple classifiers, in which each score is predicted by a classifier trained with a specific feature. However, the existing methods generally use a fixed fusion weight for all the scores of a classifier, and thus fail to optimally determine the fusion weight for the individual samples. In this paper, we propose a sample-specific late fusion method to address this issue. Specifically, we cast the problem into an information propagation process which propagates the fusion weights learned on the labeled samples to individual unlabeled samples, while enforcing that positive samples have higher fusion scores than negative samples. In this process, we identify the optimal fusion weights for each sample and push positive samples to top positions in the fusion score rank list. We formulate our problem as a L_∞ norm constrained optimization problem and apply the Alternating Direction Method of Multipliers for the optimization. Extensive experiment results on various visual categorization tasks show that the proposed method consistently and significantly beats the state-of-the-art late fusion methods. To the best knowledge, this is the first method supporting sample-specific fusion weight learning.

1. Introduction

Recently, "multi-feature late fusion" has been advocated in the computer vision community, and its effectiveness has been demonstrated in various applications such as object recognition [22, 24], biometric analysis [15], video event detection [14, 24]. Given multiple classifiers trained with different low-level features, late fusion tries to combine the prediction scores of all classifiers (the prediction score of each sample generated by a classifier indicates the confidence of classifying the sample as positive). Such a fusion method is expected to assign positive samples higher fusion scores than the negative ones so that the overall performance can be improved. Although very simple, this method has proved to be effective in improving performance of each individual classifier and also produces highly comparative results to multi-feature early fusion methods [21, 24].

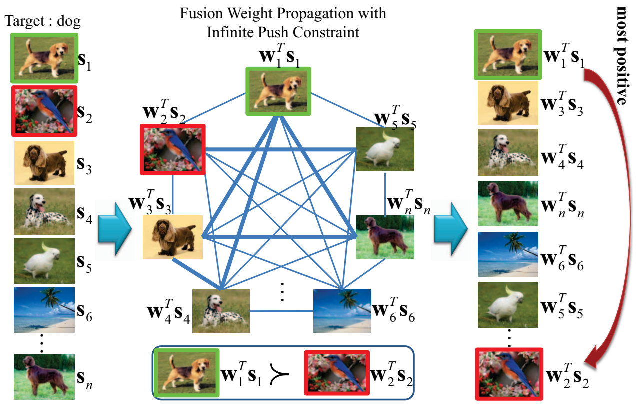


Figure 1. An illustration of the proposed sample-specific late fusion method. Given n images and their prediction score vectors $\mathbf{s}_i$ $(i = 1, \ldots, n)$, where the images with green and red borders are respectively labeled as positive and negative while the others are unlabeled, we want to learn a fusion weight vector $\mathbf{w}_i$ for each sample. The problem is cast into an information propagation procedure which propagates the fusion weights of the labeled images to the individual unlabeled ones along a graph built on low-level features. During the propagation, we use an infinite push constraint to ensure the positive samples have higher fusion scores than the negative samples. The fusion scores $\mathbf{w}_i^\top \mathbf{s}_i$ can be used to rank the images where the positive images will appear at the top positions of the rank list. This figure is best viewed in color.

The simplest approach to late fusion is to estimate a fixed weight for each classifier and then use a weighted summation of the prediction scores as the fusion result. Obviously, this assumes all the prediction scores of a classifier share the same weight and fails to consider the differences of the classifier's prediction capability on individual samples. A classifier, in fact, does have different prediction capabilities on different samples, where some samples are correctly predicted while others are not. Therefore, instead of using a fixed weight for each classifier, a promising alternative is to estimate the specific fusion weights for each sample to alleviate the individual prediction errors from the imperfect classifiers and achieve robust fusion.

Discovering the sample specific fusion weights is a nontrivial task due to the following issues. First, given the prediction scores of a test sample, since its label information is unavailable, it is unknown how to determine the sample specific fusion weights for such an unlabeled sample. Second, to get a robust late fusion result, we need to maximally ensure positive samples have the highest fusion scores in the fusion result. Indeed, the visual recognition task can be seen as a ranking process that aims at assigning positive samples higher scores than the negative samples.

In this paper, we address the above issues by proposing the Sample Specific Late Fusion (SSLF) method, which learns the optimal sample-specific fusion weights from supervision information while directly enforcing that positive samples have the highest fusion scores in the fusion result. Figure 1 illustrates the framework of our proposed method. Specifically, we define the fusion process as an information propagation procedure which propagates the fusion weights learned on the individual labeled samples to the individual unlabeled ones. The propagation is guided by a graph built on low-level features of all samples, which enforces visually similar samples have similar fusion scores and offers the capability to infer fusion weights for unlabeled samples. To ensure most positive samples have the highest fusion scores as possible, we use the L_∞ norm infinite push constraint to minimize the number of positive samples scored lower than the highest-scored negative sample. By this propagation process, we identify the optimal sample-specific fusion weights and push positive samples to have the highest fusion scores. We will demonstrate experimentally the proposed method can achieve significant performance gains when evaluated over various visual recognition tasks.

2. Related Work

Nandakumar *et al.* [15] employed the Gaussian mixture model to approximate the score distributions of the classifier, and then performed score fusion using likelihood ratio test. Terrades *et al.* [22] developed a supervised late fusion method which tried to minimize the classification error rates under L_1 constraints on the fusion weights. However, these works focus on classifier-level fusion which determines a fixed weight for all prediction scores of a specific classifier. Such fusion methods blindly treat the prediction scores of a classifier as equally important and cannot optimally determine the fusion weights for each sample.

Recently, Liu *et al.* [14] proposed a local expert forest model for late fusion, which partitioned the score space into local regions and learned the local fusion weights in each region. However, the learning can only be performed on the training samples whose label information is provided, and hence cannot be applied to learn the fusion weights on the test samples. In addition, the partition requires a threshold, which is difficult, if not impossible, to predefine. One promising work that tries to obtain sample specific fusion scores is the low rank late fusion method proposed by Ye *et al* [24]. Specifically, they converted the prediction score vectors of multiple classifiers into various pairwise relation matrices and then extracted a shared rank-2 matrix by decomposing each original matrix into a common rank-2 matrix and a sparse residual matrix. Finally, a score vector is extracted from the rank-2 matrix as the late fusion result. Despite its benefits, in practice, when seeking shared score patterns across the classifiers, supervision information is not present. As a result, it totally depends on the agreement of different classifiers, which may blindly bring the common prediction errors shared across different classifiers into the final fusion. Instead, we focus on learning the optimal fusion weights for the individual samples by exploiting the supervision information, which accounts for the differences in the classifiers' prediction abilities on the individual samples, and hence achieve robust fusion.

We are motivated by the recent infinite push ranking method in machine learning. One representative work is the support vector infinite push method [1], which introduces the L_∞ push loss function into the learning-to-rank problem with the purpose of maximizing the number of positive samples on the absolute top of the list [20]. Rakotomamonjy *et al.* [19] further developed a sparse support vector infinite push method, which incorporated feature selection into the support vector infinite push method. However, these methods can only learn a uniform ranking function for all the test samples, and cannot be applied to the sample specific fusion weight learning. Related work can also be found in graph-based semi-supervised learning [3, 25], but they are restricted to estimating the classification or ranking score of each node, and thus cannot be used to learn the weights for fusion. Our method is related to instance-specific metric learning [26], which aims at deriving a proper distance metric for each instance rather than optimally determining the fusion weights of each instance for ranking.

3. Learning Sample-Specific Fusion Weight

In this section, we will introduce our Sample-Specific Late Fusion (SSLF) method. We first present the notation and definition, and then describe the problem formulation.

3.1. Notation and Definition

Suppose we have a set of m classifiers C_i's ($i = 1, \ldots, m$), each of which is learned based on one type of feature. The proposed method works in a transductive setting in which l labeled samples $\{x_i, y_i\}_{i=1}^{l}$ and u unlabeled samples $\{x_i\}_{i=l+1}^{l+u}$ are available, where $y_i \in \{0, 1\}$ is the label of sample x_i. Specifically, the labeled samples are responsible for providing supervision information while the unlabeled samples correspond to test samples whose prediction confidences are expected from the fusion. Since our method works on the prediction scores of the classifiers, it is important to note that the labeled samples employed in our method should be disjoint from the training samples used for classifier training. This is due to the fact that the ground-truth labels of the training samples have been exploited by the classifiers, making the prediction scores on these training samples bias toward the ground-truth labels. Such prediction scores cannot reflect the classifier's prediction capabilities on unseen samples, defeating the value of a fusion method. In real-world visual classification tasks, such labeled sample set can be easily obtained. For example, besides training and test set, many visual classification tasks also provide a validation set for parameter selection. In this case, we can directly apply it as our labeled sample set. Even when the validation set is not available, we can also obtain such samples by splitting from the training samples before classifier training or crawling online resources.

By applying the classifiers on the labeled samples and unlabeled samples, we obtain a labeled score vector set $\{\mathbf{s}_i, y_i\}_{i=1}^{l}$ and unlabeled score vector set $\{\mathbf{s}_i\}_{i=l+1}^{l+u}$, where $\mathbf{s}_i = [s_i^1, \ldots, s_i^m]^\top$ denotes the prediction score vector of sample x_i ($i = 1, \ldots, l+u$) with s_i^j being the prediction score of the j-th classifier C_j. Furthermore, for ease of discussion, we divide the labeled sample set into a positive subset $\mathcal{P} = \{\mathbf{s}_i^+\}_{i=1}^{p}$ and a negative subset $\mathcal{N} = \{\mathbf{s}_i^-\}_{i=1}^{n}$, where $\mathbf{s}_i^+$ and $\mathbf{s}_i^-$ respectively denote the score vector of a positive sample and a negative sample, p and n are respectively the total number of positive and negative samples. Finally, we stack all score vectors into a matrix $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_{l+u}]$, where the positive samples are placed before the negative samples and all unlabeled samples are placed in the last columns of the matrix.

3.2. Problem Formulation

We want to learn a sample-specific fusion function $f_i(\mathbf{s}_i) = \mathbf{w}_i^\top \mathbf{s}_i$ for each sample ($i = 1, \ldots, l+u$), where $\mathbf{w}_i = [w_i^1, \ldots, w_i^m]^\top$ is a non-negative fusion weight vector with w_i^j being the fusion weight of s_i^j. Obviously, we can directly derive the fusion weights of the labeled samples based on their label information. However, it is non-trivial to learn fusion weights for the unlabeled samples since there is no supervision information that can be directly applied.

To solve this problem, we resort to the local smoothness property in graph-based semi-supervised learning which assumes similar samples have similar labels within a local region of the sample space. To realize this property, we build a nearest neighbor graph based on the low level features. For each sample x_i, we find its K nearest neighbors and establish an edge between x_i and each of its neighbors. The entry G_{ij} in the weight matrix $\mathbf{G}$ associated with the graph is calculated by

$$G_{ij} = \begin{cases} \exp(-\frac{\bar{d}(x_i,x_j)}{\sigma}), & \text{if } i \in \mathcal{N}_K(j) \text{ or } j \in \mathcal{N}_K(i), \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where $\mathcal{N}_K(i)$ denotes the index set of the K nearest neighbors of samples x_i (we set $K = 6$ here), $\bar{d}(x_i, x_j) = \frac{1}{m}\sum_{k=1}^{m} d^k(x_i, x_j)$ is the average distance between two samples, in which $d^k(x_i, x_j)$ denotes the distance calculated based on the k-th feature type (In our experiments, we use different distances for different features). σ is the radius parameter of the Gaussian function, which is set as the mean value of all pairwise average distances among the samples.

Our late fusion method is formulated as follows:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \Omega(\mathbf{W}) + \lambda \ell(\{f_i\}_{i=1}^{l}; \mathcal{P}, \mathcal{N}), \\ \text{s.t.} \quad & \mathbf{w}_i \geq 0,\ i = 1, \ldots, l+u, \end{aligned} \tag{2}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_{l+u}]$ consists of $l+u$ fusion weight vectors to be derived for both labeled and unlabeled samples, and λ is a trade-off parameter among the two competing terms. The first term is a regularization term designed for the purpose of fusion weight propagation:

$$\begin{aligned} \Omega(\mathbf{W}) &= \sum_{i,j=1}^{l+u} E_{ij}(\mathbf{w}_i^\top \mathbf{s}_i - \mathbf{w}_j^\top \mathbf{s}_j)^2 \\ &= \big(\pi(\mathbf{W})\big)^\top \mathbf{L}\big(\pi(\mathbf{W})\big), \end{aligned} \tag{3}$$

where $\mathbf{E} = \mathbf{U}^{-\frac{1}{2}}\mathbf{G}\mathbf{U}^{-\frac{1}{2}}$ is a normalized weight matrix of $\mathbf{G}$. The matrix $\mathbf{U}$ is a diagonal matrix where the (i,i)-entry is the i-th row or column sum of $\mathbf{G}$. $\mathbf{L} = (\mathbf{I} - \mathbf{E})$ is the graph laplacian [2]. $\pi(\mathbf{W})$ is a vector defined as $\pi(\mathbf{W}) = \big((\mathbf{W}^\top\mathbf{S}) \circ \mathbf{I}\big)\mathbf{1}$, where $\circ$ is the Hadamard matrix product. Intuitively, the minimization of Eq. (3) enforces a smooth fusion score propagation over the graph structure, making similar samples have similar fusion scores. An alternative to the above regularization term is to replace the score difference $(\mathbf{w}_i^\top \mathbf{s}_i - \mathbf{w}_j^\top \mathbf{s}_j)^2$ with the squared L_2 distance of weight vectors, i.e., $\|\mathbf{w}_i - \mathbf{w}_j\|_2^2$. However, this ignores the prediction scores of the individual test samples and does not fully take advantage of the prediction capability of the trained classifiers.

The second term is an infinite push loss function [1], which tries to minimize the number of positive samples scored below the highest-scored negative. In fact, the number of positive samples scored below the highest-scored negative is exactly the maximum number of positives scored below any negative, which is defined as:

$$\ell(\{f_i\}_{i=1}^{l}; \mathcal{P}, \mathcal{N}) = \max_{1 \leq j \leq n} \Big(\frac{1}{p}\sum_{i=1}^{p} I_{f_i(\mathbf{s}_i^+) < f_j(\mathbf{s}_j^-)}\Big), \tag{4}$$

where $I.$ is the indicator function whose value is 1 if the argument is true and 0 otherwise. The maximum operator over j equals to calculating the L_∞ norm of a vector consisting of n entries, each of which corresponds to one value based on j in the parentheses of Eq. (4). By minimizing this penalty, positive samples tend to score higher than any negative sample. This essentially ensures positive samples have higher fusion scores than the negative, leading to more accurate fusion results.

The minimization of Eq. (4) is intractable due to its discrete nature, so we minimize a convex upper bound [1]:

$$\ell(\{f_i\}_{i=1}^{l};\mathcal{P},\mathcal{N}) = \max_{1\le j\le n}\left(\frac{1}{p}\sum_{i=1}^{p}\left(1-(\mathbf{w}_i^\top\mathbf{s}_i^+ - \mathbf{w}_j^\top\mathbf{s}_j^-)\right)_+\right), \tag{5}$$

where $(u)_+ = u$ if $u > 0$ and 0 otherwise.

Finally, the objective function can be written as:

$$\begin{aligned} \min_{\mathbf{W}} \quad & (\pi(\mathbf{W}))^\top \mathbf{L}(\pi(\mathbf{W})) \\ & +\lambda \max_{1\le j\le n}\left(\frac{1}{p}\sum_{i=1}^{p}\left(1-(\mathbf{w}_i^\top\mathbf{s}_i^+ - \mathbf{w}_j^\top\mathbf{s}_j^-)\right)_+\right), \\ \text{s.t.} \quad & \mathbf{w}_i \ge 0,\ i=1,\dots,l+u. \end{aligned} \tag{6}$$

The above objective function is convex, and thus can achieve the global optimum. In the next section, we will introduce an efficient procedure to solve it.

4. Optimization Procedure

In this section, we follow the optimization procedure in [19] and derive an Alternating Direction Method of Multipliers (ADMM) method [4] for the optimization. To this end, we first drop the $\mathbf{w}_i \ge 0$ constraint, so that the ADMM method can be applied to solve Eq. (6), and then we project the solution back to the feasible region.

4.1. ADMM Formulation

We convert the optimization problem in Eq. (6) into the following problem with linear constraints:

$$\begin{aligned} \min_{\mathbf{W},a_{ij}} \quad & \Omega(\mathbf{W}) + \lambda \max_{1\le j\le n}\left(\frac{1}{p}\sum_{i=1}^{p}(a_{ij})_+\right), \\ \text{s.t.} \quad & a_{ij} = 1-(\mathbf{w}_i^\top\mathbf{s}_i^+ - \mathbf{w}_j^\top\mathbf{s}_j^-). \end{aligned} \tag{7}$$

Then, by defining the matrix $\mathbf{J} = [(\mathbf{A}\bigotimes\mathbf{B})^\top,(\mathbf{D}\bigotimes\mathbf{C})^\top,(\mathbf{F}\bigotimes\mathbf{B})^\top]^\top$, where $\mathbf{A}=\mathbf{I}_p$, $\mathbf{B}=\mathbf{1}_{n\times 1}^\top$, $\mathbf{C}=\mathbf{I}_n$, $\mathbf{D}=-\mathbf{1}_{p\times 1}^\top$, $\mathbf{F}=\mathbf{0}_{u\times p}$, $\bigotimes$ denotes the Kronecker product, $\mathbf{X} = (\mathbf{W}^\top\mathbf{S})\circ\mathbf{I}$, the vector $\mathbf{a}$ composing of all a_{ij}'s and the function $g(\mathbf{a}) = \lambda\max_{1\le j\le n}\left(\frac{1}{p}\sum_{i=1}^{p}\max(a_{ij},0)\right)$, we arrive at the following formulation:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{a}} \quad & \Omega(\mathbf{W}) + g(\mathbf{a}), \\ \text{s.t.} \quad & \mathbf{J}^\top\mathbf{X}\mathbf{1}+\mathbf{a}-\mathbf{1} = 0. \end{aligned} \tag{8}$$

The augmented Lagrangian of the above problem is

$$\begin{aligned} \mathcal{L}(\mathbf{W},\mathbf{a},\gamma,\mu) &= \Omega(\mathbf{W}) + g(\mathbf{a}) + \gamma^\top(\mathbf{J}^\top\mathbf{X}\mathbf{1}+\mathbf{a}-\mathbf{1}) \\ &+ \frac{\mu}{2}\|\mathbf{J}^\top\mathbf{X}\mathbf{1}+\mathbf{a}-\mathbf{1}\|_2^2, \end{aligned} \tag{9}$$

where γ is the vector of Lagrangian multipliers of the linear constraints and μ is a weighting parameter of the quadratic penalty. Following the experimental practices of ADMM [4], we set μ to be 10^{-4}. The above formulation can be equally rewritten as

$$\mathcal{L}(\mathbf{W},\mathbf{a},\beta) = \Omega(\mathbf{W}) + g(\mathbf{a}) + \frac{\mu}{2}\|\mathbf{J}^\top\mathbf{X}\mathbf{1}+\mathbf{a}-\mathbf{1}+\beta\|_2^2, \tag{10}$$

where $\beta = \gamma/\mu$. Finally, the optimization becomes iteratively solving the saddle point of the augmented Lagrangian. At iteration k, we need to solve the following three sub-problems:

$$\mathbf{W}^{k+1} = \arg\min_{\mathbf{W}}\mathcal{L}(\mathbf{W},\mathbf{a}^k,\beta^k), \tag{11}$$

$$\mathbf{a}^{k+1} = \arg\min_{\mathbf{a}}\mathcal{L}(\mathbf{W}^{k+1},\mathbf{a},\beta^k), \tag{12}$$

$$\beta^{k+1} = \beta^k + \mathbf{J}^\top\mathbf{X}^{k+1}\mathbf{1}+\mathbf{a}^{k+1}-\mathbf{1}, \tag{13}$$

where $\mathbf{X}^{k+1} = \left((\mathbf{W}^{k+1})^\top\mathbf{S}\right)\circ\mathbf{I}$. Next, we will show how to solve these sub-problems.

4.2. Alternating Optimization

The optimization of Eq. (11) can be written as

$$\min_{\mathbf{W}}\Xi(\mathbf{W}) \equiv \frac{\mu}{2}\|\mathbf{J}^\top\mathbf{X}\mathbf{1}-\mathbf{t}\|_2^2 + \Omega(\mathbf{W}), \tag{14}$$

where $\mathbf{t} = 1-\mathbf{a}^k-\beta^k$ and its gradient can be calculated as

$$\frac{\nabla\Xi(\mathbf{W})}{W_{ij}} = \mathrm{tr}\left[\left(\mu\mathbf{J}(\mathbf{J}^\top\mathbf{X}\mathbf{1}-\mathbf{t})\mathbf{1}^\top + 2\mathbf{L}\mathbf{X}\mathbf{1}\mathbf{1}^\top\right)^\top\frac{\partial\mathbf{X}}{\partial W_{ij}}\right], \tag{15}$$

through which the optimization problem can be solved by a conjugate gradient descent method.

The optimization problem in Eq. (12) becomes

$$\min_{\mathbf{a}} g(\mathbf{a}) + \frac{\mu}{2}\|\mathbf{a}-\mathbf{t}\|_2^2, \tag{16}$$

where $\mathbf{t} = \mathbf{1}-\beta^k-\mathbf{J}^\top\mathbf{X}^{k+1}\mathbf{1}$. To solve the two nested max operators in $g(\mathbf{a})$, we use the double trick and convert the problem as in [19]:

$$\begin{aligned} \min_{\mathbf{a}^+,\mathbf{a}^-} \quad & \frac{1}{2}\|\mathbf{a}^+-\mathbf{a}^--\mathbf{t}\|_2^2 + \max_{1\le j\le n}\left(\frac{\lambda}{\mu p}\sum_{i\in\mathcal{G}_j}a_i^+\right) \\ \text{s.t.} \quad & \mathbf{a}^+\ge 0, \mathbf{a}^-\ge 0, \end{aligned} \tag{17}$$

where $\mathbf{a} = \mathbf{a}^+-\mathbf{a}^-$ and $\mathcal{G}_j$ denotes the indices of the positive samples in vector $\mathbf{a}$ that are coupled with the negative sample $\mathbf{s}_j$. This problem can be solved by iterative optimization by employing the optimization method in [19], where $\mathbf{a}^-$ has a closed-form solution while $\mathbf{a}^+$ can be solved by Douglas-Rachford method [7], which alternately performs two proximal operators on the positive quadrant and the $L_{1,\infty}$ mixed norm until convergence [18]. The optimization procedure is shown in Algorithm 1.

Algorithm 1 ADMM for Sample Specific Late Fusion

```
1:  Input: $\mathbf{S} \in \mathbb{R}^{m\times(l+u)}$, $\mathbf{L} \in \mathbb{R}^{(l+u)\times(l+u)}$, $\{y_i\}_{i=1}^{l}$, $\lambda$.
2:  Initialization: $k = 0, \mathbf{a}^0, \beta^0 = 0, \mathbf{W}^0 > 0, \mu = 10^{-4}$.
3:  Calculate $\mathbf{J} \in \{-1, 0, 1\}^{(l+u)\times pn}$ based on the label information.
4:  repeat
5:      $\mathbf{t} = \mathbf{1} - \mathbf{a}^k - \beta^k$.
6:      $\mathbf{W}^{k+1} = \arg\min_{\mathbf{W}} \frac{\mu}{2}\|\mathbf{J}^\top \mathbf{X}\mathbf{1} - \mathbf{t}\|_2^2 + \Omega(\mathbf{W})$.
7:      Force the negative values in $\mathbf{W}^{k+1}$ to 0.
8:      $\mathbf{X}^{k+1} = \big((\mathbf{W}^{k+1})^\top \mathbf{S}\big) \circ \mathbf{I}$.
9:      $\mathbf{t} = \mathbf{1} - \beta^k - \mathbf{J}^\top \mathbf{X}^{k+1}\mathbf{1}$.
10:     Obtain $\mathbf{a}^-$ and $\mathbf{a}^+$ by solving Eq. (17) based on [19].
11:     $\mathbf{a}^{k+1} = \mathbf{a}^+ - \mathbf{a}^-$.
12:     $\beta^{k+1} = \beta^k + \mathbf{J}^\top \mathbf{X}^{k+1}\mathbf{1} + \mathbf{a}^{k+1} - \mathbf{1}$.
13:     $k = k + 1$.
14: until convergence.
15: Output: $f_i(\mathbf{s}_i) = (\mathbf{w}_i^*)^\top \mathbf{s}_i, i = l+1, \ldots, l+u$.
```

4.3. Algorithmic Analysis

Algorithm 1 is built upon ADMM and Douglas-Rachford procedure, each of which has shown very good convergence property. Since the objective function is convex, the algorithm will approach the global optimum. In our later experiment, we observe that the objective function value converges to the minimum after few iterations. For example, in the experiment on Oxford Flower 17 dataset (see Section 5.1) implemented on the MATLAB platform on an Intel XeonX5660 workstation with 3.2 GHz CPU and 18 GB memory, Algorithm 1 finishes efficiently within 3.56 seconds on average for each category. Note that the scalability of our algorithm is dominated by the total number of samples involved in the optimization.

5. Experiments

In this section, we will evaluate the proposed late fusion method by applying it to various visual recognition tasks including object categorization and video event detection. We compare the performances of the following five methods: (1) Kernel Averaging (KA), we average the kernel matrices of different features to obtain a fused kernel matrix. This is actually the most common method for early fusion of multiple features and is proved to achieve highly comparative results as multiple kernel learning [9]. Therefore, we do not include MKL as a comparing baseline since no significant performance difference is expected. (2) Average Late Fusion (ALF), we directly average the prediction scores from all the classifiers as the fusion results. (3) Low Rank Late Fusion (LRLF), in this method, the prediction scores of each classifier are first converted into a binary comparative relationship matrix and a shared rank-2 matrix is then discovered across all matrices. The final fusion score vector can be extracted from the rank-2 matrix by matrix decomposition. (4) Fixed Weight Late Fusion (FWLF), instead of learning sample-specific fusion functions, we learn a fixed fusion function $f(\mathbf{s}) = \mathbf{w}^\top \mathbf{s}$ for all the samples. This essentially applies the same weight w_i to all the scores of the i-th classifier. To achieve this, we replace the fusion function $f_i(\mathbf{s}_i) = \mathbf{w}_i^\top \mathbf{s}_i$ in our objective function with $f(\mathbf{s}_i) = \mathbf{w}^\top \mathbf{s}_i$ ($i = 1, \ldots, l+u$). (5) Our proposed Sample-Specific Late Fusion (SSLF).

Following previous work on late fusion [24], we employ the probabilistic outputs of the one-vs-all SVM classifier as the prediction scores, in which each value measures the possibility of classifying a sample as positive. On each dataset, we do not manually specify the number of classifiers (i.e., the number of features), but directly utilize the available features on the dataset to train classifiers (one for each classifier). To evaluate the performance of each method, the Average Precision (AP) is employed as the evaluation metric. We calculate AP for each visual category and then calculate the mean Average Precision (mAP) across all categories of the entire dataset as the final evaluation metric.

To determine the appropriate parameter for our method and FWLF, we vary the value of λ in the grid of $\{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$ and then run 2-fold cross validation on the labeled sample set to select the best parameter value based on validation performance. Regarding the parameter setting of LRLF, we follow the suggested parameter setting strategy as in [24] and choose the best parameter values based on 2-fold cross-validation. The tradeoff parameter for SVM is selected from $\{10^{-1}, \ldots, 10^{3}\}$ through 5-fold cross-validation on the training set.

5.1. Results for object categorization

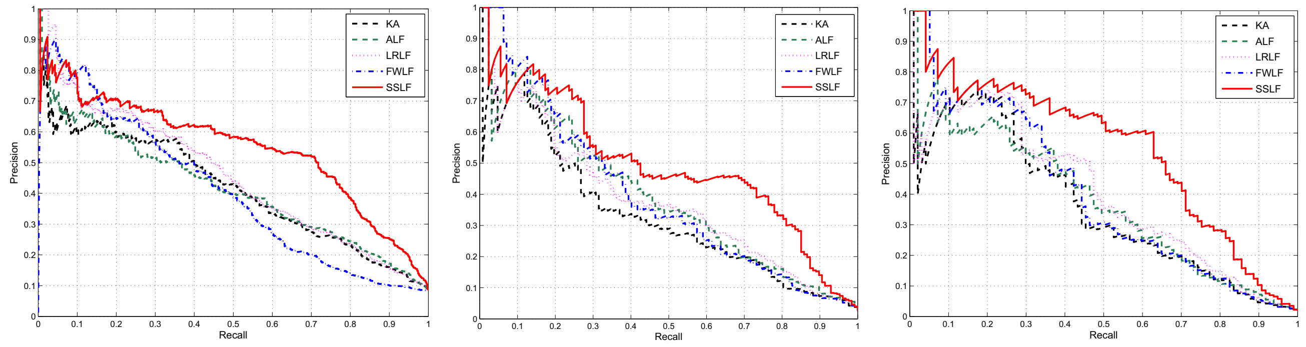
In this subsection, we evaluate our proposed method on the object categorization task. Two benchmark datasets are utilized: PASCAL VOC'07 and Oxford Flower 17.

PASCAL VOC'07. This dataset consists of $9,963$ images which are crawled by querying for images of 20 different object categories from Flickr website. For feature representations, we directly download the 15 features provided by [8], including 4 kinds of SIFT Bag-Of-Words (BoW) histograms [12], 4 kinds of Hue BoW histograms [23], 2 kinds of RGB color histograms, 2 kinds of HSV histograms, 2 kinds of LAB color histograms and 1 GIST feature [17]. The details on the features can be found in [8]. Following [8], we use L_1 distance for the color histograms, L_2 for GIST, and χ^2 for the visual word histograms. For a given distance matrix, the kernel matrix of SVM classifier is calculated as $\exp(-d(x, y)/\sigma)$ where $d(x, y)$ is the distance between x and y and σ is the mean value of all the pairwise distances on the training set.

We follow the standard training ($5,011$ images) and test ($4,952$ images) data split provided by this dataset in our experiment. To generate the labeled sample set for late fusion,

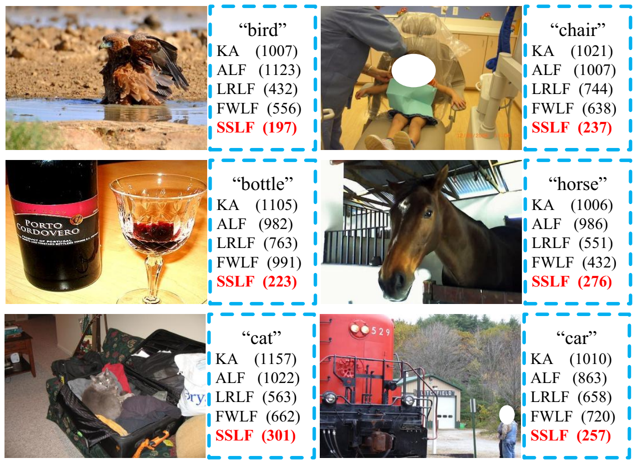
	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	moto	person	plant	sheep	sofa	train	tv	mAP
KA	73.9	54.0	48.5	68.3	19.8	55.2	71.5	50.2	44.1	33.5	44.4	41.6	77.4	60.5	84.9	37.5	36.1	39.1	77.4	42.0	53.0
ALF	70.0	55.4	47.2	62.8	20.4	56.0	69.6	51.9	43.2	39.1	43.5	41.7	76.7	58.2	82.3	33.1	37.7	37.0	76.4	41.1	52.2
LRLF	76.0	57.7	52.8	**73.2**	24.5	63.5	73.0	53.4	49.3	39.5	48.1	45.7	**80.9**	62.2	**88.1**	40.6	40.1	45.5	72.3	42.7	56.5
FWLF	74.6	55.2	50.7	68.9	22.8	57.1	72.5	51.9	46.7	37.1	47.0	43.5	77.8	61.4	85.5	39.4	39.0	42.0	78.1	44.2	54.8
SVM [8]	72.7	53.0	49.1	66.8	25.6	52.4	69.9	50.0	46.0	36.4	43.3	43.9	74.7	59.5	83.4	39.0	39.5	39.9	74.3	42.8	53.1
SSLF	**78.0**	**64.9**	**58.0**	73.1	**32.2**	**64.0**	**76.4**	**62.4**	**57.3**	**44.6**	**56.7**	**51.0**	80.4	**65.9**	87.5	**46.5**	**46.3**	**49.7**	**82.9**	**55.3**	**61.7**

Table 1. Per-category performance comparison (AP %) of different methods on PASCAL VOC'07 dataset. The standard deviations for FWLF and SSLF are respectively 0.7% and 0.5%.



(a) dog (b) cow (c) sheep

Figure 2. Precision/recall curves of different methods on three categories of PASCAL VOC'07 dataset. This figure is best viewed in color.


"bird"
KA (1007)
ALF (1123)
LRLF (432)
FWLF (556)
SSLF (197)

"chair"
KA (1021)
ALF (1007)
LRLF (744)
FWLF (638)
SSLF (237)

"bottle"
KA (1105)
ALF (982)
LRLF (763)
FWLF (991)
SSLF (223)

"horse"
KA (1006)
ALF (986)
LRLF (551)
FWLF (432)
SSLF (276)

"cat"
KA (1157)
ALF (1022)
LRLF (563)
FWLF (662)
SSLF (301)

"car"
KA (1010)
ALF (863)
LRLF (658)
FWLF (720)
SSLF (257)

Figure 3. Example images and their rank positions in the fusion score rank list obtained from different fusion methods. For each method, the rank list is obtained by ranking all $4,952$ test images in descending order based on the fusion scores.

we uniformly divide the training samples of each category into 5 folds, and then use 4 folds as the training data for SVM training while using the remaining 1 fold as the labeled sample set for late fusion. The experiments are repeated 5 times so that each fold can be used as the labeled sample set, and the average result is reported. Note that such splits are only applicable in the FWLF and SSLF methods which need supervision information. For other methods including KA, ALF, LRLF, we still use the original data split.

Table 1 shows the per-category performances of all the methods in comparison. From the results, we have the following observations: (1) The proposed SSLF method consistently beats all the other baseline methods by a large margin, which demonstrates its effectiveness in determining the optimal fusion weights for each sample. (2) The LRLF, FWLF and SSLF late fusion methods all outperform the ALF method. This is due to the fact that the former methods take advantage of additional knowledge (either consistent score patterns across the classifiers or supervision information) while the latter only blindly averages the scores from different classifiers without accounting their difference. (3) The sample level late fusion methods including LRLF and SSLF outperform the FWLF. The reason may be that FWLF only tries to learn uniform fusion weights for all the samples and hence cannot discover the optimal fusion weights for each sample. (4) Our SSLF method performs better than LRLF method, since the latter does not exploit the supervision information. In Figure 2, we show the precision-recall curves of different methods for some representative categories. As can be seen, the precisions of our method are higher than the other methods when the recall varies from 0 to 1. This clearly demonstrates that our method is able to assign higher fusion scores to the positive samples. In our experiments, we also observe that prediction scores from more reliable classifiers tend to have higher fusion weights than the scores from the less reliable classifiers. Figure 3 shows the rank positions of some example images after ranking the $4,952$ test images based on fusion scores of different methods. As seen, SSLF can successfully rank the images at higher positions in the rank list.

Oxford Flower 17. The Oxford Flower 17 dataset is a benchmark dataset for multi-feature object categorization [16]. This dataset consists of $1,360$ images falling into 17 different species of flowers, and each class contains 80 images. We use the predefined training (17×40), validation (17×20) and test (17×20) data splits along with the χ^2 distance matrices calculated from different features in our experiment. There are seven features provided by this dataset, including color, shape, texture, HOG [6], clustered HSV values, SIFT feature on the foreground boundary (SIFTbdy) and SIFT feature on the foreground internal region (SIFTint). For SVM classifier, we use the χ^2 kernel and the best parameter C is selected via validation performance on the validation set. We use the validation set as the labeled sample set for FWLF and SSLF.

The results of our proposed method and all other baseline methods are shown in Table 2. As can be seen, our proposed method outperforms all the baseline methods. Again, the experiment results demonstrate the superiority of the proposed method. Figure 4 shows the image ranking results of different fusion methods.

methods	mAP
KA	86.0 ± 1.7
ALF	86.9 ± 2.1
LRLF	91.7 ± 1.7
FWLF	91.2 ± 1.5
Our SSLF Method	$\mathbf{93.40 \pm 1.3}$

Table 2. Performance comparisons (AP %) on Oxford Flower 17.

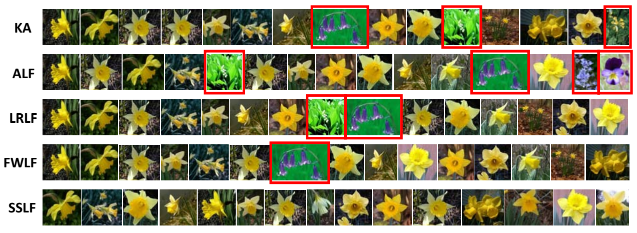


Figure 4. Top 15 images ranked with the fusion scores of different methods. Images with red borders are incorrect. This figure is best viewed in color.

5.2. Results for video event detection

We also test our method on the task of video event detection, in which the TRECVID 2011 Multimedia Event Detection (MED) development set and Columbia Consumer Video (CCV) are utilized as the testbed.

TRECVID 2011 MED development set. This dataset contains 10,804 video clips from 17,566 minutes of web video programs falling into five event classes and the background class. The five events are *attempting a board trick*, *feeding an animal*, *landing a fish*, *wedding ceremony*, and *working on a woodworking project* respectively. The dataset is partitioned into the training set (8,783 videos) and test set (2,021 videos). The training set contains 8,273 background videos that do not belong to any of the event classes, making the detection task challenging.

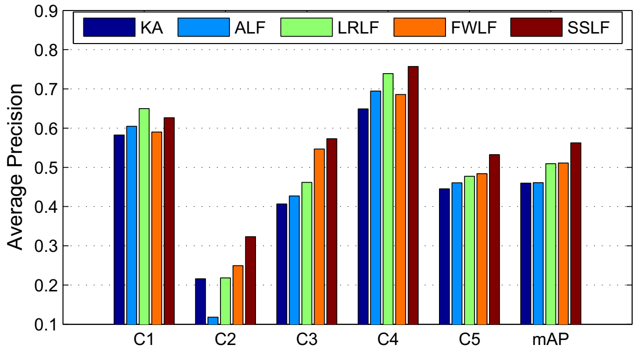


Figure 5. Per-category performance comparison on TRECVID MED 2011 development dataset. The five events from left to right in the horizontal axis are (1) "*attempting a board trick*", (2) "*feeding an animal*", (3) "*landing a fish*", (4) "*wedding ceremony*", and (5) "*working on a woodworking project*" respectively. The standard deviations of mAP for FWLF and SSLF are respectively 1.1% and 1.3%. This figure is best viewed in color.

Given a video clip, we sample 1 frame every 2 seconds. For each frame, we extract 5,000-dimension SIFT BoW and then average all frame features within a video as the static video representation. We also extract 5,000-dimension Spatial-Temporal Interest Points (STIP) BoW [13] and 4,000-dimension Mel-Frequency Cepstral Coefficients (MFCC) BoW. We use the L_2 distance to calculate the distance matrix of each feature and then train SVM classifiers with χ^2 kernel. Following the experiment setting on PASCAL VOC'07, we uniformly split the training samples into 5 folds and use 4 folds for SVM training and 1 fold for learning fusion weight. The experiments are repeated 5 times and the averaged result is reported.

Figure 5 shows the per-event performance of all the methods. As can be seen, our method achieves the best performance on four out of the five events. Specifically, our method outperforms the KA, ALF, LRLF and FWLF by 10.3%, 10.1%, 5.3% and 5.1% respectively in terms of mAP. Moreover, it achieves the best performances on most of the event categories. For instance, on event "feeding an animal", our method outperforms the best baseline FWLF by 7.4%. Once again the experiment results demonstrate the effectiveness of our method.

CCV. This dataset contains 9,317 YouTube videos annotated over 20 semantic categories, where 4,659 videos are used for training and 4,658 videos are used for testing [11]. We downloaded three kinds of low-level features provided by this dataset as the feature representations which include 5,000-dimension SIFT BoW, 5,000-dimension STIP BoW and 4,000-dimension MFCC BoW. We follow the same setting as in the TRECVID MED dataset and the per-category results are shown in Figure 6. From the results, we can see that the proposed SSLF achieves the best performance in terms of mAP, where it outperforms KA, ALF, LRLF and FWLF by 8.7%, 9.3%,

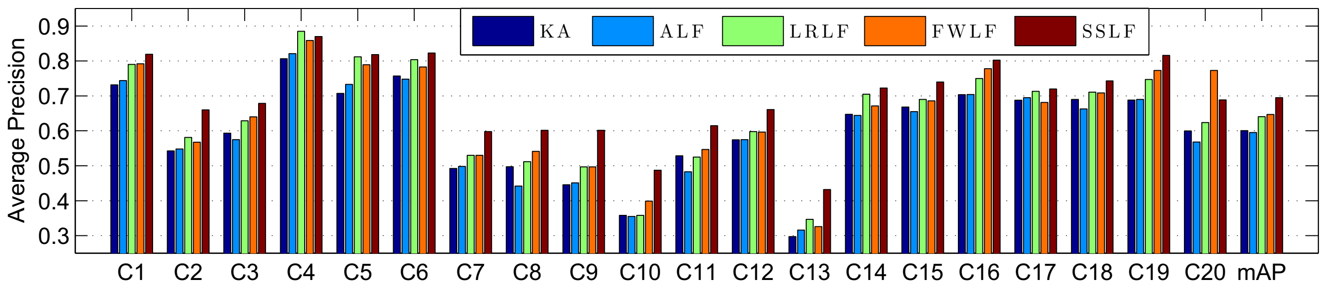


Figure 6. Per-category performance comparison on CCV dataset. The 20 categories from left to right in the horizontal axis are (1) "*basketball*", (2) "*baseball*", (3) "*soccer*", (4) "*ice skating*", (5) "*skiing*", (6) "*swimming*", (7) "*biking*", (8)"*cat*", (9)"*dog*", (10)"*bird*", (11)"*graduation*", (12)"*birthday*", (13)"*wedding reception*", (14)"*wedding ceremony*", (15)"*wedding dance*", (16)"*music performance*", (17)"*non-music performance*", (18)"*parade*", (19)"*beach*", (20)"*playground*" respectively. The standard deviations of mAP for FWLF and SSLF are respectively 0.2% and 0.4%. This figure is best viewed in color.

5.4% and 4.9% respectively. This demonstrates that our method is effective in the task of video event detection.

5.3. Discussion

We note that we can apply the classical out-of-sample extension method in transductive learning to estimate the fusion score of a new sample [5, 10, 24]. For a new test sample z, we can use the low-level feature to find a set of nearest neighbors $\{x_i\}_{i=1}^q$ from all samples in the original dataset, where x_i is a neighbor of z and q is the total number of neighbors. Based on the neighborhood set, the late fusion score can be determined as $f(z) = \sum_{i=1}^q \frac{G(z,x_i)}{\sum_{i=1}^q G(z,x_i)} (\mathbf{w}_i^*)^\top \mathbf{s}_i$, where $G(z, x_i)$ is the similarity between z and x_i, and $(\mathbf{w}_i^*)^\top \mathbf{s}_i$ is the fusion score of x_i obtained on the original dataset. In this way, we obtain the fusion score for the unseen sample.

6. Conclusions

We have introduced a sample-specific late fusion method to learn the optimal fusion weights for each sample. The proposed method works in a transductive setting which propagates the fusion weights of the labeled samples to the individual unlabeled samples, while leveraging the infinite push constraint to enforce positive samples to have higher fusion scores than negative samples. The process is formulated into a convex objective function, and the ADMM method is employed for the optimization. Extensive experiments have demonstrated the effectiveness of the proposed method on various visual category recognition tasks including object categorization and video event detection. For future work, we will pursue the sample-specific late fusion for multi-class and multi-label visual recognition tasks.

7. Acknowledgement

This work is supported in part by Office of Naval Research (ONR) grant (N00014-10-1-0242). We would like to thank Felix Yu, Brendan Jou, and anonymous reviewers for their insightful suggestions.

References

[1] S. Agarwal. The infinite push: A new support vector ranking algorithm that directly optimizes accuracy at the absolute top of the list. In *SDM*, 2011.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 2002.

[3] M. Belkin, etc. On manifold regularization. In *AISTATS*, 2005.

[4] S. Boyd, etc. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 2010.

[5] Y. Bengio, J. Paiement, and P. Vincent. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps and spectral clustering. In *NIPS*, 2003.

[6] N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. In *CVPR*, 2005.

[7] J. Eckstein and D. Bertsekas. On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Matehmatical Programming*, 1992.

[8] M. Guillaumin, J. Verbeek, and C. Schmid. Multimodal semi-supervised learning for image classification. In *CVPR*, 2010.

[9] P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *ICCV*, 2009.

[10] C. Hou, etc. Semisupervised learning using negative labels. In *TNN*, 2011.

[11] Y.-G. Jiang, etc. Consumer video understanding: A benchmark database and an evaluation of human and machine performance. In *ICMR*, 2011.

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[13] I. Laptev. On space-time interest points. *IJCV*, 2005.

[14] J. Liu, S. McCloskey, and Y. Liu. Local expert forest of score fusion for video event classification. In *ECCV*, 2012.

[15] K. Nandakumar, Y. Chen, S. Dass, and A. Jain. Likelihood ratio-based biometric score fusion. *TPAMI*, 2008.

[16] M. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICCVGIP*, 2008.

[17] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelop. *IJCV*, 2001.

[18] A. Quattoni, etc. An efficient projection for $\ell_{1,\infty}$ regularization. In *ICML*, 2009.

[19] A. Rakotomamonjy. Sparse support vector infinite push. In *ICML*, 2012.

[20] C. Rodin, etc. The p-norm push: A simple convex ranking algorithm that concetrates at the top of the list. *JMLR*, 2009.

[21] C. Snoek, M. Worring, and A. Smeulders. Early versus late fusion in semantic video analysis. In *ACM MM*, 2005.

[22] O. Terrades, E. Valveny, and S. Tabbone. Optimal classifier fusion in a non-bayesian probabilistic framework. *TPAMI*, 2009.

[23] J. Weijer and C. Schmid. Coloring local features for object image classification. In *ECCV*, 2006.

[24] G. Ye, D. Liu, I.-H. Jhuo, and S.-F. Chang. Robust late fusion with rank mininization. In *CVPR*, 2011.

[25] D. Zhou, etc. Ranking on data manifolds. In *NIPS*, 2003.

[26] D. Zhan, etc. Learning instance specific distances using metric propagation. In *ICML*, 2009.