A Novel Multiple Kernel Learning Framework for Heterogeneous Feature Fusion and Variable Selection

Yi-Ren Yeh, Ting-Chu Lin, Yung-Yu Chung, and Yu-Chiang Frank Wang, *Member, IEEE*

***Abstract*—We propose a novel multiple kernel learning (MKL) algorithm with a group lasso regularizer, called group lasso regularized MKL (GL-MKL), for heterogeneous feature fusion and variable selection. For problems of feature fusion, assigning a group of base kernels for each feature type in an MKL framework provides a robust way in fitting data extracted from different feature domains. Adding a mixed $\ell_{1,2}$ norm constraint (i.e., group lasso) as the regularizer, we can enforce the sparsity at the group/feature level and automatically learn a compact feature set for recognition purposes. More precisely, our GL-MKL determines the optimal base kernels, including the associated weights and kernel parameters, and results in improved recognition performance. Besides, our GL-MKL can also be extended to address heterogeneous variable selection problems. For such problems, we aim to select a compact set of variables (i.e., feature attributes) for comparable or improved performance. Our proposed method does not need to exhaustively search for the entire variable space like prior sequential-based variable selection methods did, and we do not require any prior knowledge on the optimal size of the variable subset either. To verify the effectiveness and robustness of our GL-MKL, we conduct experiments on video and image datasets for heterogeneous feature fusion, and perform variable selection on various UCI datasets.**

***Index Terms*—Feature fusion, multiple kernel learning, variable selection.**

I. Introduction

In order to produce satisfactory results in many pattern recognition and computer vision problems, one typically needs to consider the combination of heterogeneous features, i.e., features extracted from different domains for improved performance. For example, object recognition using real-world images deals with images with large intra and interclass variations plus background clutter presented. In such cases, using a single type of features like SIFT [1] or HOG [2] is not able to describe each object category well and thus the recognition performance is limited [3]. Similar remarks apply to video processing applications such as event recognition [4], [5], action recognition [6], affective classification [7], [8] and object detection [3], [9]. As illustrated in Fig. 1, it is common and necessary to integrate different types of features observed from the input video to address the corresponding recognition or annotation problems (e.g., [5]).

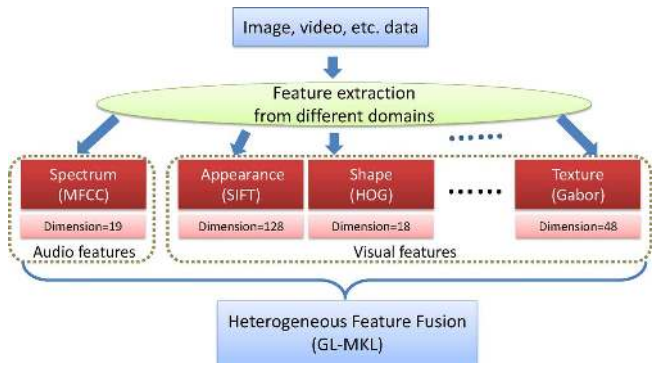

Fig. 1. Overview of feature fusion for video object classification. Note that the features of interest are collected from heterogeneous domains, and thus each feature type has a unique property and distribution.

While the use of heterogeneous features becomes more practical for real-world applications, how to properly integrate those features is still one of the main research topics in the areas of pattern recognition and machine learning. For *feature-level* fusion, one can simply concatenate different types of features and obtain a new feature representation for training and testing. On the other hand, one can train a classifier for each type of features, and the results predicted by different classifiers (and the associated features) will be combined via voting or averaging strategies to reach the final output. This can be considered as *classifier* or *decision-level* fusion. While these fusion techniques are easy to implement and promising results have been reported (e.g., [10]), a simple concatenation of heterogeneous features will result in the increase of feature dimensionality. Moreover, there is no guarantee that simple voting or averaging techniques would produce improved performance. As pointed out in [3], rather than adding features/classifers with similar performances, one should combine those with complementary information in order to achieve better recognition performance.

In this paper, we present a multiple kernel learning (MKL) framework for heterogeneous feature fusion and variable selection. Inspired by the recent success of MKL, our proposed framework aims to select a compact set of features/varaibles for improved recognition performance via the introduction of a

Manuscript received September 26, 2011; revised February 03, 2012; accepted February 12, 2012. Date of publication February 23, 2012; date of current version May 11, 2012. This work is supported in part by the National Science Council of Taiwan via NSC 99-2221-E-001-020 and NSC 100-2221-E-001-018-MY2. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Sethuraman Panchanathan.

Y.-R. Yeh and T.-C. Lin are with the Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan, 11529 (e-mail: yryeh@citi.sinica.edu.tw; tingchulin@citi.sinica.edu.tw).

Y.-Y. Chung was with the Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan, and is currently with the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA 50014, USA (e-mail: ychung@iastate.edu).

Y.-C. F. Wang is with the Research Center for Information Technology Innovation and Institute of Information Science, Academia Sinica, Taipei, Taiwan, 11529 (e-mail: ycwang@citi.sinica.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2188783

group lasso regularizer. Fig. 1 illustrates an example of video object classification via combining features extracted from different domains. Via the one-vs-rest learning strategy, our method is able to select the class-specific weights for different types of features, and thus is expected to outperform simple feature and decision-level fusion methods. As we will show later, this proposed framework can be easily extended to the use of heterogeneous variable selection, which is also practical in practical pattern recognition problems. Our experimental results will verify the feasibility of our GL-MKL formulation for both heterogeneous feature fusion and variable selection problems, and we will show that our method outperforms existing MKL based approaches in the above problems.

The remaining of this paper is organized as follows. Section II discusses related works on feature fusion and variable selection using MKL. We review the formulation for MKL and its use for feature fusion and variable selection in Section III. Section IV introduces our proposed MKL framework and details how we apply it to address the above problems. We present experimental results of heterogenous feature fusion for video and image datasets in Section V, and the performance of variable selection for several UCI datastes is reported in Section VI. Finally, Section VII concludes this paper.

II. Related Works

Kernel methods such as support vector machines (SVM) [11] have been shown to be very effective for data representation, dimension reduction, and classification. A more flexible learning model using multiple kernels instead of one, which is known as multiple kernel learning (MKL), has recently been proposed [12]. Since MKL is able to better represent or discriminate between data using multiple base kernels, it has been shown to improve the performance of many learning tasks, including feature fusion (e.g., [3], [6], [9]) and variable selection (e.g., [13], [14]).

For feature fusion, MKL has been used for visual classification tasks such as object classification [3], [6], [9], [15], [16]. The conventional MKL framework combines multiple features (e.g., appearance, texture, shape, etc.) by constructing base kernels for each type of feature. The optimal weights for each base kernel are determined by MKL, and they indicate the contribution/importance of the associated features. Several variants of MKL for feature fusion have also been proposed for improving the performance [3], [6], [9]. Yang *et al.* [9] introduced an intermediate representation "group", which collects images within each category. Images within a group shared the same weights to combine features, and these weights are determined by MKL. Since its performance is highly dependent on the grouping results, it is not easy for one to select an appropriate clustering algorithm without prior knowledge. Gehler *et al.* [3] integrated MKL and boosting techniques to learn the weights for feature combination. However, the learned weights were not jointly optimized with the data instances (in terms of support vectors), so there is no guarantee that their approach would always produce the best fusion results (as shown in our experiments later). Cao *et al.* [6] proposed a logistic regression model with multiple kernels to perform feature fusion and instance selection for video action recognition. While they chose to impose a group Lasso regularizer (as we do) on their framework, their work aims at removing irrelevant samples. Since our goal is to address general classification problems, we do not consider the case in which there exist irrelevant instances. Therefore, it is not easy to extend their work for these problems.

Different from feature fusion, variable selection (also known as feature selection) focuses on the use of a single type of feature. It aims at identifying a subset of relevant features for improved or comparable recognition performance. Prior works such as [13], [14] have applied MKL for variable selection problems. Dileep *et al.* [13] proposed to learn the optimal base kernels, while each is built from each feature attribute/dimension. As a result, this can be regarded as an extreme case of feature fusion. Although an improved MKL-based variable selection method was recently proposed by Xu *et al.* [14], the user needs to specify the preferable size of the feature subset in advance, which typically cannot be known in practice. Moreover, these prior MKL-based methods treat all features equally important, and none of them addresses the problem of heterogeneous variable selection (i.e., the feature attributes are collected from different domains, and thus each feature dimension has a unique property and distribution).

In this paper, we propose a novel MKL-based method for heterogeneous feature fusion and variable selection. We extend the standard MKL formulation and impose a mixed ℓ_1 and ℓ_2 norm constraint as the group lasso regularizer, which will determine the optimal weights for each base kernel and thus achieve the goal of feature selection fusion and variable selection. In our framework, each heterogeneous feature (or variable) is associated with multiple base kernels and is considered as a group. The imposed group lasso regularizer tends to maintain sparsity between different groups, while the associated weights of the selected kernels for each group need not be sparse. This allows our MKL algorithm to select more than one base kernel for each heterogeneous feature (or feature dimension), while a compact set of groups will be enforced due to the added sparsity at the group level. Since we associate each heterogeneous feature (or feature dimension) with multiple base kernels with different kernel parameter (e.g., width of the Gaussian kernel), our MKL has the capability to deal with heterogeneous data. Using our approach, the associated weights and kernel parameters can be learned automatically and simultaneously.

III. Multiple Kernel Learning

A. Review of MKL

The support vector machine (SVM) [17] has been known to be an effective binary classifier due to its generalization ability. It learns an optimal separating hyperplane to distinguish data between two different classes without any assumption on data distribution. However, a single kernel function might not be sufficient to model the data of interest and thus produce a satisfactory separating hyperplane. As a result, multiple kernels are recently applied for this purpose, and this is referred to as multiple kernel learning (MKL) [12]. More precisely, one can replace the single kernel by a linear combination of base kernels,

while each kernel describes a different property of the data of interest (i.e., different feature spaces or distributions). Thus, MKL is expected to provide improved generalization ability for the learning model.

Similar to SVM, one can approach MKL by formulating and solving its primal form. This process can be considered as describing the data in multiple feature spaces using different norm vectors $\mathbf{w}_\ell$. Let $\phi : \mathcal{X} \rightarrow \mathcal{H}$ be a feature mapping function, in which $\mathcal{X}$ is the input space and $\mathcal{H}$ is a dot product space associated with the inner product $\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle_{\mathcal{H}}$. Note that we have $\langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle_{\mathcal{H}} = k(\mathbf{x}, \mathbf{z})$, which computes the inner product between the transformed feature vectors $\phi(\mathbf{x})$ and $\phi(\mathbf{z})$, and $k(\cdot, \cdot)$ is a positive semidefinite kernel function. According to [18], the primal form of MKL is thus formulated as the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}, \phi(\mathbf{w}), b, \boldsymbol{\xi}} \quad & \frac{1}{2} \sum_{\ell=1}^{p} \frac{1}{\beta_\ell} \|\phi_\ell(\mathbf{w}_\ell)\|_{\mathcal{H}_\ell}^2 + C \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i \left(\sum_{\ell=1}^{p} \langle \phi_\ell(\mathbf{w}_\ell), \phi_\ell(\mathbf{x}_i) \rangle_{\mathcal{H}_\ell} + b \right) + \xi_i \geq 1 \\ & \xi_i \geq 0, \quad \text{for } i = 1, 2, \ldots, n, \\ & \|\boldsymbol{\beta}\|_1 = 1, \quad \boldsymbol{\beta} \geq \mathbf{0} \end{aligned} \tag{1}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_p]^\top$, and p is the number of feature spaces (i.e., number of base kernels). Similar to SVM, C in (1) is the trade-off between the generalization of MKL and its training errors ξ_i. From the above formulation, we see that the primal form of MKL restricts the weight of the norm vector $\phi_\ell(w_\ell)$ with the constraint of $\|\boldsymbol{\beta}\|_1 = 1$ and $\beta_\ell \geq 0$, which tends to produce a sparse solution for $\boldsymbol{\beta}$. In (1), it can be observed that if β_ℓ vanishes, then the corresponding $\|\phi_\ell(\mathbf{w}_\ell)\|_{\mathcal{H}_\ell}$ should be zero; otherwise, the value of the objective function will be unbounded. In [18], Rakotomamonjy *et al.* have shown that $\|\phi_\ell(\mathbf{w}_\ell)\|_{\mathcal{H}_\ell} \rightarrow 0$ as $\beta_\ell \rightarrow 0$, which prevents the objective function value from approaching infinity. Therefore, the use of the sparsity constraint $\|\boldsymbol{\beta}\|_1 = 1$ would still produce a valid and sparse solution for $\boldsymbol{\beta}$.

Similar to the SVM, one can also convert the above formulation and derive the dual form for MKL. With the constraint on β_ℓ, the minimization problem (1) can thus be transformed into the following min-max problem:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \sum_{\ell=1}^{p} \beta_\ell k_\ell(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \sum_{i=1}^{n} y_i \alpha_i = 0 \\ & \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{1}C, \quad \|\boldsymbol{\beta}\|_1 = 1, \quad \boldsymbol{\beta} \geq \mathbf{0} \end{aligned} \tag{2}$$

where α_i are the Lagrange coefficients. Comparing (2) to SVM, multiple base kernel functions (i.e., $k_\ell(\mathbf{x}_i, \mathbf{x}_j)$) are applied in (2) while only a single kernel is used in SVM. Since the constraint $\|\boldsymbol{\beta}\|_1 = 1$ tends to result in a sparse solution of β_ℓ, this learning process can be viewed as the removal of redundant kernels among the base ones. Simply speaking, the MKL formulation in (2) aims to determine an optimal and compact linear combination of base kernels for improved recognition performance, and this is achieved by learning the best weights β_ℓ for the base kernels and the predictors α_i for the associated data (for classification). For a test input $\mathbf{x}$, the decision function of MKL can be computed as

$$F_{MKL}(\mathbf{x}) = \text{sign} \left(\sum_{i=1}^{n} \sum_{\ell=1}^{p} \beta_\ell (k_\ell(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_i + b) \right). \tag{3}$$

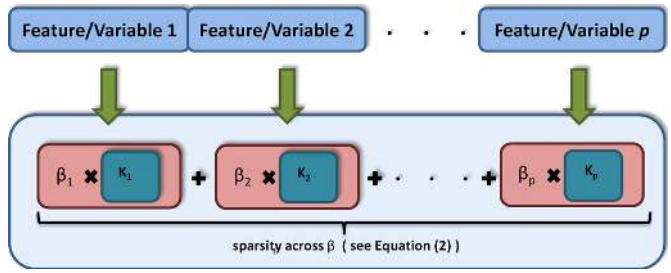

Fig. 2. Illustration of MKL for feature selection. Each feature constructs a base kernel, and the MKL determines weight coefficients β_ℓ for each base kernel with a sparsity constraint.

B. MKL for Feature Fusion and Variable Selection

As mentioned in Section II, it has been shown that feature fusion via MKL can improve the performance for many learning and vision tasks [3], [6], [9]. As shown in Fig. 2, the standard MKL framework constructs base kernels for each type of feature, and their optimal weight coefficients (i.e., β_ℓ) are determined by solving (2). However, the standard MKL needs to predetermine the parameters for each kernel, such as the bandwidth σ of Gaussian kernels, and this parameter selection procedure either requires prior knowledge or results in increased computational complexity due to the need to perform cross-validation.

MKL also has recently been applied for variable selection [3], [13], [14]. Existing methods typically approach this type of problem as solving a task of learning the optimal weights for each feature variable/attribute. More specifically, MKL uses each feature variable to construct a corresponding kernel. As shown in Fig. 2, it determines the weight coefficients for each for improved performance while those weights indicate the relevance of the associated features for the learning task. Although the weighted sum of these kernels calculated from individual variable is expected to improve the classification performance, results reported in previous works such as [13] did not achieve significant improvements on several benchmark datasets. Moreover, existing variable selection methods usually regard all variables from the same domain, and the distributions of each variable are assumed to be the same with some data normalization techniques applied. In other words, they did not address the problem of heterogeneous variable selection as we do. In the next section, we will detail our proposed MKL framework, which van be applied to both heterogeneous feature fusion and variable selection problems.

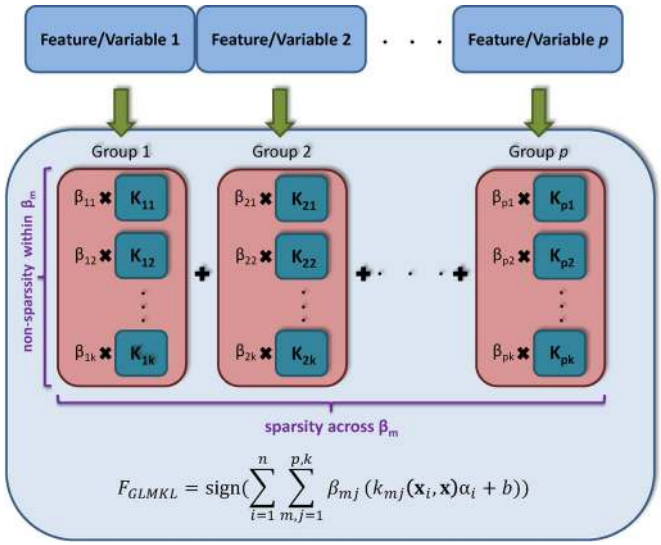

Fig. 3. Illustration of our GL-MKL for heterogeneous feature fusion. Different from Fig. 2, multiple kernels constructed for each heterogeneous feature form a group, and we enforce the group-lasso constraint on the weights of each base kernel for feature fusion purposes. Since different types of heterogeneous features should be associated with their preferable kernels, our GL-MKL allows the existence of nonsparsity for the kernel weights within each group.

IV. Group Lasso Regularized MKL

A. Algorithm of GL-MKL

To address the problems mentioned in the previous section, we propose a novel MKL with a group lasso regularizer, called *group lasso regularized MKL* (GL-MKL), which constrains the coefficient $\boldsymbol{\beta}$ with a $\ell_{1,2}$ norm. Suppose that we have p types of feature paired with k different kernel choices (e.g., different σ choices if using Gaussian kernels). There is a total of $p \times k$ base kernels in our group lasso regularized MKL. That is, we have $\boldsymbol{\beta} = [\boldsymbol{\beta}_1; \boldsymbol{\beta}_2; \cdots; \boldsymbol{\beta}_p] = [\beta_{11}, \beta_{12}, \ldots, \beta_{pk}]^\top \in \mathbb{R}^{(p\times k)\times 1}$, which are associated with base kernels as shown in Fig. 3. Our mixed $\ell_{1,2}$ constraint imposed on $\boldsymbol{\beta}$ will maintain sparsity between different groups (i.e., different feature types), while the associated β_{mj} values in each group need not be sparse (see Fig. 3). More precisely, we enforce the sparsity constraint at the feature space level (for feature fusion), and we allow our MKL to select more than one kernels for each feature to improve overall performance (to handle heterogeneous features).

With these $p \times k$ kernels and the corresponding coefficient $\boldsymbol{\beta} \in \mathbb{R}^{(p\times k)\times 1}$, the primal form of our GL-MKL is formulated as follows:

$$\begin{aligned}
&\min_{\boldsymbol{\beta},\phi(\mathbf{w}),b,\boldsymbol{\xi}} \ \frac{1}{2}\sum_{m,j=1}^{p,k}\frac{1}{\beta_{mj}}\|\phi_{mj}(\mathbf{w}_{mj})\|^2_{\mathcal{H}_{mj}} + C\sum_{i=1}^{n}\xi_i \\
&\text{s.t. } y_i\left(\sum_{m,j=1}^{p,k}\langle\phi_{mj}(\mathbf{w}_{mj}),\phi_{mj}(\mathbf{x}_i)\rangle_{\mathcal{H}_{mj}} + b\right) + \xi_i \geq 1 \\
&\qquad\qquad \xi_i \geq 0, \ \text{for } i = 1, 2, \ldots, n \\
&\qquad\qquad \sum_{m=1}^{p}\|\boldsymbol{\beta}_m\|_2 \leq 1, \ \boldsymbol{\beta}_m \in \mathbb{R}^k \geq \mathbf{0}, \ \forall\, m \quad (4)
\end{aligned}$$

where $\boldsymbol{\beta}_m = [\beta_{m1}, \beta_{m2}, \ldots, \beta_{mk}]$. We also apply the same setting in [19] and relax the equality constraint $\sum_{m=1}^{p}\|\boldsymbol{\beta}_m\|_2 = 1$ to $\sum_{m=1}^{p}\|\boldsymbol{\beta}_m\|_2 \leq 1$ due to the convexity of the optimization problem. More specifically, assigning $(k = 1, p = 0)$ or $(k = 0, p = 1)$ will convert our algorithm back to ℓ_1 or ℓ_2 regularized MKL problem, which can be considered as two special cases of our proposed MKL. In practice, one can choose different numbers of kernels for each feature using our MKL, while we fix this number k in this paper. From (4), we have $\|\phi_{mj}(\mathbf{w}_{mj})\|_{\mathcal{H}_{mj}} \to 0$ as $\beta_{mj} \to 0$ in our GLMKL formulation. As discussed in Section III-A, this property prevents the value of the objective function in (4) from approaching infinity, and thus a valid solution $\boldsymbol{\beta}$ with group-wise sparsity will be obtained.

We see that, if $\boldsymbol{\beta}$ is fixed in (4), our GL-MKL formulation becomes a Lagrangian function of variables $\phi(\mathbf{w})$, b, and $\boldsymbol{\xi}$

$$\begin{aligned}
&\mathcal{L}(\phi(\mathbf{w}), b, \boldsymbol{\xi}) \\
&= \frac{1}{2}\sum_{m,j=1}^{p,k}\frac{1}{\beta_{mj}}\|\phi_{mj}(\mathbf{w}_{mj})\|^2_{\mathcal{H}_{mj}} + C\sum_{i=1}^{n}\xi_i \\
&\quad + \sum_{i=1}^{n}\alpha_i\left(1 - \xi_i - y_i\left(\sum_{m,j=1}^{p,k}\langle\phi_{mj}(\mathbf{w}_{mj}),\phi(\mathbf{x}_i)\rangle_{\mathcal{H}_{mj}} + b\right)\right) \\
&\quad - \sum_{i=1}^{n}\nu_i\xi_i \qquad (5)
\end{aligned}$$

where α_i and ν_i are the Lagrangian multipliers. Setting the derivatives of (5) to zeroes with respect to the primal variables, we have the following conditions:

$$\begin{aligned}
&\frac{\partial\mathcal{L}(\phi(\mathbf{w}), b, \boldsymbol{\xi})}{\partial\phi_{mj}(\mathbf{w}_{mj})} = 0 \\
&\quad \Rightarrow \phi_{mj}(\mathbf{w}_{mj}) = \beta_{mj}\sum_{i=1}^{n}\alpha_i y_i\phi_{mj}(\mathbf{x}_i), \ \forall\, m, j \\
&\frac{\partial\mathcal{L}(\phi(\mathbf{w}), b, \boldsymbol{\xi})}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n}\alpha_i y_i = 0 \\
&\frac{\partial\mathcal{L}(\phi(\mathbf{w}), b, \boldsymbol{\xi})}{\partial\boldsymbol{\xi}} = 0 \Rightarrow C - \alpha_i - \nu_i = 0, \ \forall\, i. \qquad (6)
\end{aligned}$$

Substitute the above conditions to (5), we then transform (4) into the following min-max optimization problem:

$$\begin{aligned}
&\min_{\boldsymbol{\beta}}\max_{\boldsymbol{\alpha}} \ S(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{i=1}^{n}\alpha_i \\
&\qquad\qquad - \frac{1}{2}\sum_{i,r=1}^{n}y_i y_r\alpha_i\alpha_r\sum_{m,j=1}^{p,k}\beta_{mj}k_{mj}(\mathbf{x}_i,\mathbf{x}_j) \\
&\text{s.t. } \sum_{i=1}^{n}y_i\alpha_i = 0 \\
&\qquad\qquad \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{1}C, \ \sum_{m=1}^{p}\|\boldsymbol{\beta}_m\|_2 \leq 1, \ \boldsymbol{\beta}_m \in \mathbb{R}^k \geq \mathbf{0} \ \forall m. \\
&\qquad\qquad (7)
\end{aligned}$$

The above min-max problem can be solved by gradient based methods (e.g., [12], [18]). Alternatively, we can formulate (7)

as a semiinfinite programming (SIP) problem [20] and search for the best $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ iteratively. To be more specific, we fix $\boldsymbol{\beta}$ and solve the maximization problem of (7) with respect to $\boldsymbol{\alpha}$; we note that this procedure can be addressed using any regular SVM solver such as libSVM [21], which solves (7) with fixed $\boldsymbol{\beta}$. Once the variables $\boldsymbol{\alpha}$ are determined in an iteration, we fix $\boldsymbol{\alpha}$ and solve the minimization problem of (7) with respect to $\boldsymbol{\beta}$. Suppose that $\boldsymbol{\alpha}^*$ is the optimal solution in (7), we have the objective value $S(\boldsymbol{\alpha}^*, \boldsymbol{\beta}) = \theta$ and $\theta \geq S(\boldsymbol{\alpha}, \boldsymbol{\beta})$ for all $\boldsymbol{\alpha}$. Thus, by fixing $\boldsymbol{\alpha}$, we convert (7) into the following SIP problem (as suggested by [19]) which minimizes θ to its lower bound

$$\begin{aligned} \min_{\theta, \boldsymbol{\beta}} \quad & \theta \\ \text{s.t.} \quad & \theta \geq S(\boldsymbol{\alpha}, \boldsymbol{\beta}) \\ & \sum_{m=1}^{p} \|\boldsymbol{\beta}_m\|_2 \leq 1,\ \boldsymbol{\beta}_m \in \mathbb{R}^k \geq \mathbf{0}\ \forall m \\ & \mathbf{0} \leq \boldsymbol{\alpha} \leq \mathbf{1}C,\ \sum_{i=1}^{m} y_i \alpha_i = 0,\ \forall \boldsymbol{\alpha} \in \mathbb{R}^n. \end{aligned} \tag{8}$$

In our implementation, we use the function fmincon in MATLAB to solve (8). By iteratively solving the above two types of optimization problems (with respect to $\boldsymbol{\beta}$ or $\boldsymbol{\alpha}$), the optimal solution of (7) is thus determined. The pseudo code of our GL-MKL is described in Algorithm 1, and the decision function of GLMKL is calculated as

$$F_{GLMKL}(\mathbf{x}) = \operatorname{sign}\left(\sum_{i=1}^{n} \sum_{m,j=1}^{p,k} \beta_{mj} (k_{mj}(\mathbf{x}_i, \mathbf{x}) \alpha_i + b) \right). \tag{9}$$

Algorithm 1: Our Group Lasso Regularized MKL

Input: Data matrix $\mathbf{A}$, label $\mathbf{y}$, kernel function k_{mj}

Output: $\boldsymbol{\alpha}$ and $\boldsymbol{\beta} \in \mathbb{R}^{p \times k}$

begin

$t \leftarrow 1$; $S^0 \leftarrow 1$; $\theta^0 \leftarrow 0$; $\beta_{mj}^0 \leftarrow (1/p\sqrt{k})\ \forall\ m, j$;

while $\mid 1 - \theta^{t-1}/S^{t-1} \mid \leq \epsilon$ **do**

$\boldsymbol{\alpha}^t \leftarrow$ solve **(7)** with fixed $\boldsymbol{\beta}^{t-1}$;

$S^t \leftarrow S(\boldsymbol{\alpha}^t, \boldsymbol{\beta}^{t-1})$;

$\boldsymbol{\beta}^t,\ \theta^t \leftarrow$ solve **(8)** with all fixed $\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2, \ldots, \boldsymbol{\alpha}^t$;

$t \leftarrow t + 1$

B. GL-MKL for Heterogeneous Feature Fusion

When using our proposed GL-MKL for heterogenous feature fusion, we associate multiple types of kernels (with different kernel parameters) with each type of feature and consider them as a group, as illustrated in Fig. 3. The weight coefficients of each kernel, denoted by β_{mj}, are constrained by our group lasso regularization (see (7)), and thus our feature fusion framework maintains sparsity across $\boldsymbol{\beta_m}$ but allows nonsparsity within $\boldsymbol{\beta_m}$. We note that the above sparsity among different groups (feature types) is preferable, since it meets the goal of feature fusion (i.e., an emphasis of effective features). On the other hand, our group lasso constraint allows the nonsparsity within each group of base kernels; this is to accommodate the presence of heterogeneous data, which will require different (and possibly multiple) kernels with distinct kernel parameters to describe the data in different feature spaces at the same time. Therefore, the use of our GL-MKL provides additional flexibility in fitting heterogeneous data, and this cannot be easily achieved by standard MKL methods.

C. GL-MKL for Heterogeneous Variable Selection

For heterogeneous variable selection problems, it has been observed that each variable prefers a different set of base kernels which best represent its property/distribution for recognition purposes [22]. As a result, the use of our proposed GL-MKL can be extended for variable selection purposes. We associate multiple types of kernels (with different kernel parameters) with each variable and consider them as a group; that is to say, we consider the use of each variable to construct base kernels, and the learning process is to determine the associated weight for each kernel. If the weight is zero, the corresponding feature is redundant or trivial and thus it is discarded. Similar to the above fusion framework, The sparsity among different groups (features) meets the goal of variable selection (i.e., a compact set of variables is desirable), and the nonsparsity within each group of base kernels provides additional flexibility in fitting heterogeneous data. Another advantage is that we do not require the prior knowledge on the preferable/optimal size of the variable subset to be selected. This cannot be easily achieved by sequential-based feature selection methods. In the next section, we will evaluate our GL-MKL feature fusion and variable selection on a variety of datasets and show the effectiveness of our proposed method.

V. Experiments: Feature Fusion

A. Video Object Classification

1) Web Video Dataset: For our experiments on video object recognition with feature fusion, we collect a Web video dataset from YouTube, in which the videos are captured by uncontrolled and free-moving cameras, and the moving objects of interest are present in cluttered background. Significant scale and viewpoint variations of the objects can be observed, and the resolution of a large portion of videos in this dataset is low. We consider six different moving object categories: *Airplane*, *Ambulance*, *Race Car*, *Fire Engine*, *Helicopter*, and *Motorbike*. Each object category has 25 to 30 video sequences, and each sequence has one moving foreground object presenting in it. We randomly select 10 from each class for training, and the remaining for testing. Fig. 4 shows some video frames of each object category as an example of our dataset.

We subsample 20 frames from each of the video sequence. In order to preprocess our video data, we apply our recently proposed Consensus Foreground Object Template (CFOT) [10] to identify the region of interest (i.e., the foreground object with dominant motion information). We multiply the CFOT masks on the training and test video data, and we extract the associated visual features within the CFOT regions for training and

Fig. 4. Example videos in our Web video dataset.

testing. Note that only the visual features extracted from the training data are used to design the classifiers. For audio feature, in order to produce the same number (20) of each type of features, we uniformly divide each video clip into 20 segments and average the Mel-frequency cepstral coefficients (MFCC) to obtain 20 audio features. To classify a test video input, we first predict the label of each of the 20 subsampled frames from that input sequence, and we use a majority vote to determine the final object label for this input video. In our experiments, we consider the one-against-all strategy for classifier designs.

2) Features and Parameters: We consider five types of features collected from audio and visual domains for feature fusion. We now describe the setting for each below.

- MFCC
  We convert audio signals of each video sequence nito a stream of 19-dimensional Mel-frequency cepstral coefficients (MFCCs) using a 32-ms Hamming-windowed frame with 10-ms shifts.
- SIFT
  For visual appearance information, we use SIFT (scale-invariant feature transform) descriptors [1]. The dense SIFT visual features are extracted from 16×16 pixel patches from a video frame, and the horizontally and vertical spacing between adjacent patches is 6 pixels.
- HOG
  We capture shape information by HOG (histogram of oriented gradients) descriptors [2]. We consider a dense 8×8 pixel grid of uniformly spaced cells and extract gradient histograms, and only one scale in an octave of the pyramid is used.
- Gabor
  As for texture information, we extract the image texture by the Gabor filter [23], [24] at four different scales and with six orientations. We calculate the mean and standard deviations of these 24 output values, which result in a total dimension of this feature as 48.
- EDH
  The edge information is analyzed by an EDH (edge direction histogram) descriptor [24], [25]. A Canny filter is applied to detect edges within the region of interest. We then use a Sobel filter to calculate the gradient of each edge point, and quantize this result into a 72-bin descriptor.

TABLE I
PERFORMANCE COMPARISONS WITH SVM-BASED FUSION METHODS ON OUR WEB VIDEO DATASET. WE CALCULATE THE MEAN AVERAGE PRECISION (MAP) FOR EACH APPROACH AND FEATURE NORMALIZATION TECHNIQUE

Normalization	GL-MKL	Gaussian	Linear	Linear Sum	Linear Vote	Adaboost
N/A	72.72	70.57	62.62	68.41	62.69	68.31
Zero-mean	72.17	70.47	65.71	59.79	60.70	69.77
Min-Max	65.62	68.67	64.73	65.45	57.63	68.07

For the descriptors extracted from the visual domain, we apply sparse coding [26] techniques to convert them into a bag-of-words (BOW) model. In our implementation, we use the software package developed by Mairal *et al.* [27] to learn the dictionaries (one for each type of features), and to encode the associated sparse feature descriptor. The size of each dictionary K is set to 225, and we have $\lambda = 0.2$ controlling the sparsity of the encoded coefficient vector in our experiments. After obtaining the encoded sparse coefficient vectors for all features, we use the max pooling strategy to covert the encoded coefficients into a K-dimensional feature vector for each video frame.

3) Discussion: In our experiments, we compare our proposed GL-MKL with existing SVM and MKL-based feature fusion methods. Gaussian kernels are used for nonlinear mapping in SVM and all MKL-based methods. In all methods considered, the regularization parameter C is selected by 5-fold cross validation. For the standard SVM, the bandwidth σ in Gaussian kernel is also fine tuned by 5-fold cross validation. To deal with the heterogeneous audio and visual features using our GL-MKL, we allow each feature type to build 6 different base Gaussian kernels as a group, and the σ value for each Gaussian is determined by the standard deviation γ of the Euclidean distance between each pair of training instances. As a result, the six σ values for our base kernels are $\{\gamma, \gamma \cdot 10^1, \gamma \cdot 10^2, \gamma \cdot 10^3, \gamma \cdot 10^4, \gamma \cdot 10^5\}$.

We first compare the performance of our GL-MKL with those produced by SVM-based feature fusion methods. The results in terms of mean average precision (MAP) are shown in Table I. The first two SVM classifiers are linear and nonlinear SVMs, which are trained using concatenated audio and visual features. The other three methods considered can be considered as decision-level fusion using SVMs trained on individual features. They are sum rule, major vote, and Adaboost. To show that our GL-MKL does not require to explicitly normalize the features due to the use of multiple kernels, we also evaluate all methods with three normalization strategies: without normalization, zero-mean,and max-min. For zero-mean, we normalize features into a normal distribution $N(0, 1)$. On the other hand, we linearly normalize all features into the same $[0, 1]$ range for max-min. From Table I, it is clear that our GL-MKL significantly outperforms other SVM-based fusion methods in most cases. It shows that the direct concatenation of heterogenous

TABLE II
PERFORMANCE COMPARISONS WITH MKL BASED FUSION METHODS ON OUR WEB VIDEO DATASET. WE CALCULATE THE MEAN AVERAGE PRECISION (MAP) FOR EACH APPROACH AND FEATURE NORMALIZATION TECHNIQUE. NOTE THAT THE DECISION FUNCTIONS AND THE REQUIRED PARAMETERS TO LEARN ARE LISTED FOR EACH METHOD

Method	**Normalization**			**Decision function**	**Learning**
	N/A	Zero-Mean	Min-Max		
GL-MKL	72.72	72.17	65.62	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m,j=1}^{p,k} \beta_{mj}^c (\sum_{i=1}^{n} k_{mj}^c(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_i^c + b^c)]$	α_i^c, β_{mj}^c and b^c
MKL [13]	70.89	69.90	66.55	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m=1}^{p} \beta_{m}^c (\sum_{i=1}^{n} k_{m}^c(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_i^c + b^c)]$	α_i^c, β_{m}^c and b^c
NS-MKL [19]	69.44	69.49	68.47	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m=1}^{p} \beta_{m}^c (\sum_{i=1}^{n} k_{m}^c(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_i^c + b^c)]$	α_i^c, β_{m}^c and b^c
LP_B [3]	70.16	71.15	69.47	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m=1}^{p} B_{m}^c (\sum_{i=1}^{n} k_{m}(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_{i,m}^c + b_m^c)]$	$\alpha_{i,m}^c$, B_m^c and b_m^c
LP_β [3]	67.10	68.72	68.81	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m=1}^{p} \beta_{m} (\sum_{i=1}^{n} k_{m}(\mathbf{x}_i, \mathbf{x}) \cdot \alpha_{i,m}^c + b_m^c)]$	$\alpha_{i,m}^c$, β_m and b_m^c
Averaging	63.41	68.92	51.52	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\sum_{m=1}^{p} \frac{1}{p} (\sum_{i=1}^{n} k_{m}^c(\mathbf{x}_i, \mathbf{x})) \cdot \alpha_i^c + b^c]$	α_i^c and b^c
Product	55.25	67.57	52.45	$y(x) = \underset{c=1\ldots C}{\operatorname{argmin}} \ [\prod_{m=1}^{p} (\sum_{i=1}^{n} (k_{m}^c(\mathbf{x}_i, \mathbf{x})))^{\frac{1}{p}} \cdot \alpha_i^c + b^c]$	α_i^c and b^c

features directly might suffer from the dominant features and thus affect the performance.

Next, we compare our GL-MKL with existing MKL-based fusion methods, including the standard MKL [13], nonsparse MKL (or NS-MKL in short) [19], and LP_B and LP_β for MKL [3]. Besides, we consider two baseline MKL approaches which combine the base kernels by average or product. By a five-fold cross validation, we select the best σ value for all base Gaussian kernels jointly with the regularization parameter C for these two baseline methods. For the standard MKL and NS-MKL. we construct 6 base Gaussian kernels for each type of features and thus also have a total of 30 base kernels (5 audio/visual features available) to learn the associated MKL classifier. We note that we apply the same parameter setting for LP_B and LP_β in [3][1] for a fair comparison. The decision functions for each MKL-based method and the required parameters to learn are shown also in Table II, where b is the offset, $\boldsymbol{\alpha}$ is the support vector coefficients (for data instances), and $\boldsymbol{\beta}$ or B are the kernel weights for each feature type.

Table II lists the recognition performance in terms of MAP of our GL-MKL and other MKL-based fusion methods on the Web video dataset. It can be seen that all results of learning-based MKL approaches (i.e., those with the learning of kernel weights) are superior to those of simple fusion baseline methods (i.e., average and product). It confirms that an appropriate weighting scheme can achieve better performance for feature fusion, especially for the case of heterogenous features. Among the approaches with kernel weight learning, our proposed GL-MKL yields the best performance with either nonnormalization or zero-mean normalization. This verifies that assigning each heterogeneous feature with multiple base kernels with a group structure provides a more flexible way in fitting/representing data. In other words, our proposed is more robust and adaptive to features collected from heterogeneous domains even without a carefully normalization procedure, and thus an improved recognition performance can be achieved.

[1]The code used to produce the results of LP_B and LP_β [3] is available at http://www.vision.ee.ethz.ch/ pgehler/

TABLE III
COMPUTATION TIME OF KERNEL BASED FUSION METHODS ON WEB VIDEO DATASET

	Our Method	MKL [13]	NS MKL [19]
Running Time (seconds)	2148.68	2184.19	2251.24

Table III shows the computation time for training and test phases for MKL-based methods. It can be observed that our GL-MKL required comparable computation time as the standard MKL and NS-MKL did. Note that all runtime estimates are performed on a personal computer with Intel Core 2 Duo CPU 2.66 GHz and 4 G RAM. We did not list the computation time for LP_B and LP_β, since we simply applied their software package (programmed in C++) for our experiments. However, as noted in [3], their computation time is comparable to that of MKL. Therefore, it can be confirmed that our computational cost is comparable to other MKL-based methods, while we achieve an improved recognition performance in feature fusion.

In order to visualize the contribution of each feature type in such a MKL-based fusion scheme, we plot the kernel weights of the base kernels for MKL and our GL-MKL in Fig. 5(a) and the first row of Fig. 5(b), respectively. Note that we only list three binary classifiers (airplane, ambulance, and race car) for simplicity. From Fig. 5(b), we see that our GL-MKL selected more than one base kernel for each feature type, while the standard MKL tends to select sparse base kernels for feature fusion. Take the categories of ambulance and race car for examples, the audio feature is important for both categories, but the weights for the six audio kernels were very different. However, for the standard MKL, it only selected very few base kernels for recognition purposes, while the kernel weights for the audio feature for both categories were very low (see Fig. 5(a)). This is an example showing that our GL-MKL provides more flexibility in selecting kernels for improved recognition.

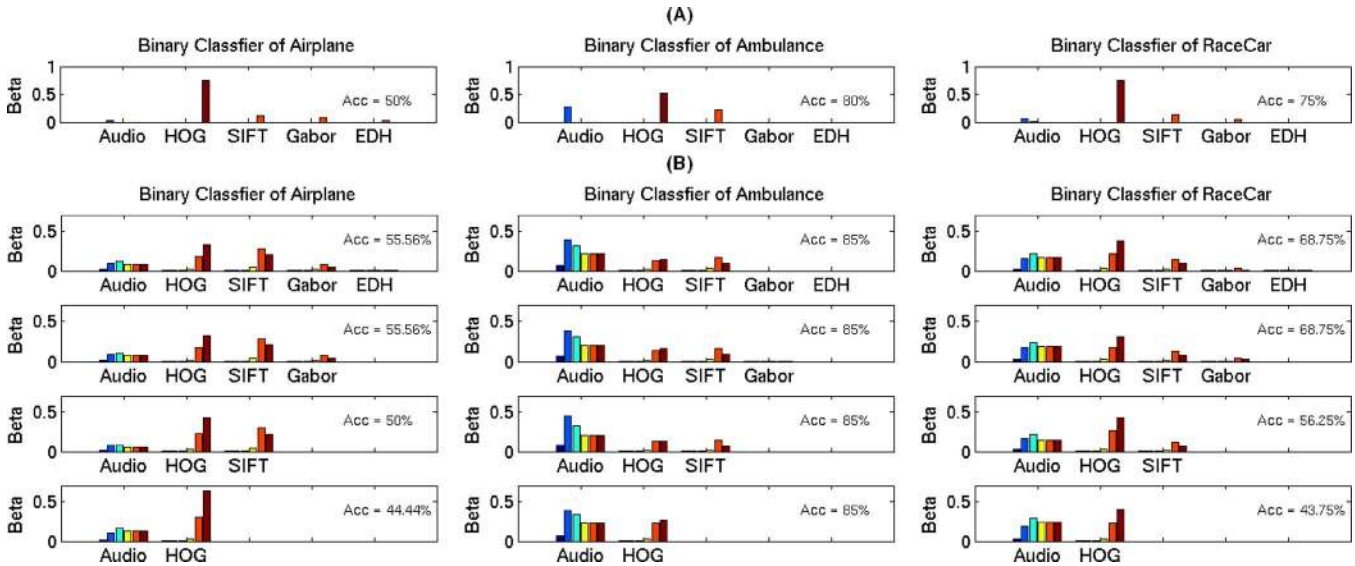

Fig. 5. Weights $\boldsymbol{\beta}$ determined for each base kernel and the corresponding feature type by MKL. The x-axis is the index of β_{mj} (6 Gaussian kernels for each feature type and shown in different colors), and the y-axis indicates the associated weights. The class-wise recognition rates with feature fusion are also noted. (B) The weights $\boldsymbol{\beta}$ determined for each base kernel and the corresponding feature type by GL-MKL. Note that we sequentially remove the features with least importance and observe comparable performance (and similar kernel weights) from the first row to the fourth row. (A). MKL. (B). GL-MKL.

Fig. 6. Example images in the UIUC Sport Event dataset.

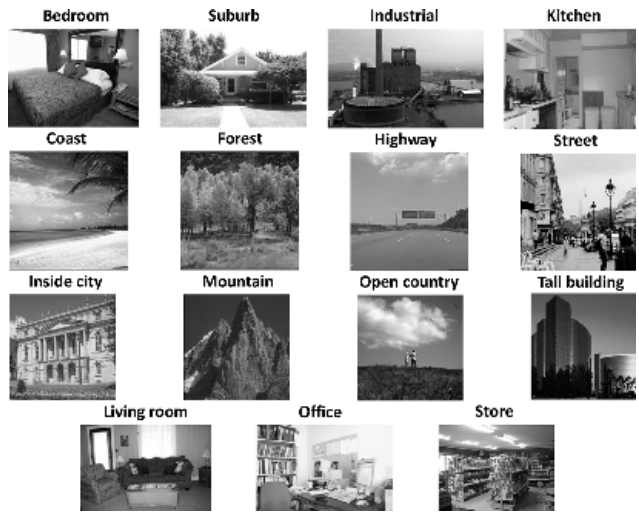

Fig. 7. Example images in the Scene-15 dataset.

To further evaluate the robustness of our GL-MKL, we observe the change of the kernel weights when removing the least significant feature types sequentially. The kernels weights for the above cases are shown in the second to the forth rows of Fig. 5(b). We found that, when we removed the least significant features according to our original observation/selection result, the kernel weights for the remaining features (with more importance) remained the same, and the recognition rates did not vary remarkably (see the recognition rates noted in Fig. 5(b)). These results support that our proposed GL-MKL is very effective for heterogeneous feature fusion, while it is able to identify the features with greater importance with proper feature/kernel weights.

B. Image Classification

1) Datasets and Features: We now conduct feature fusion experiments for image classification on two benchmark datasets: Scene-15 [28] and UIUC Sport Event datasets [29]. The Scene-15 dataset is collected from COREL collection, Google Image Search, and personal photographs (see example images shown in Fig. 7). Each class has 200 to 400 images, and the average image size is about 250 × 300 pixels. We randomly choose 100 images per categories for training and the rest for testing. We repeat this process 5 times and report the average results. For the UIUC Sport Event dataset, there is a total of eight sport event categories available (see Fig. 6 for examples), and each class has 137 to 250 high-resolution images with sizes varying 800 × 600 to thousands of pixels. We randomly choose 70 images per category for training and the rest for testing. We also repeat this process 5 times and report the average results.

For image classification, we consider five different types of visual features/descriptors. We first choose to extract PACT descriptors [30], which can be considered as a variant of the CENTRIST [30] descriptors and are known to be among the

TABLE IV
PERFORMANCE COMPARISONS ON THE SPORT EVENT AND SCENE IMAGE DATASET. WE CALCULATE THE MEAN AVERAGE PRECISION (MAP) AND THE ASSOCIATED STANDARD DEVIATION FOR EACH METHOD

Dataset	UIUC Sport Event	Scene-15
Gaussian	78.85 ± 1.30	82.75 ± 0.55
MKL [13]	77.38 ± 1.31	81.87 ± 0.37
NS-MKL [19]	80.17 ± 1.19	82.97 ± 0.55
LP_β [3]	80.57 ± 1.32	83.87 ± 0.34
LP_B [3]	80.19 ± 0.90	83.06 ± 0.55
GL-MKL	81.11 ± 0.78	83.54 ± 0.16

stat-of-the-art features especially for scene image recognition. We follow the setting in [30] and extract a 2-level PACT descriptor for an input image, and thus a 1302 dimensional feature vector will be obtained for each. The self-similarity descriptor [31] which measures the similarity between visual entities describing the structural information is the second type of visual feature considered. We use 5×5 pixel patches and set the correlation radius equal to 10 pixels in our experiments. The dimension of the resulting feature vector is encoded by a codebook of 600 visual words constructed by k-means clustering. Grey-scale dense SIFT descriptors [1] are extracted from each image to describe the image appearance information. The feature vectors are encoded by a codebook with 100 visual words (also learned from k-means clustering). For SIFT descriptors with the spatial pyramid matching (SPM) strategy [28], we consider the descriptors pooled from three different scales and with the same weights. To extract textural information, we extract local binary pattern (LBP) [32] as another visual feature, and the histograms of uniformly rotation-invariant $LBP_{8,1}$ are calculated. Finally, HOG descriptors [2] (with 40 bins) are extracted to describe the shape information presented in images.

2) Discussions: We compare our GL-MKL with SVM/MKL-based fusion methods, and the MAP performance is listed in Table IV. The settings for LP_β, LP_B and the standard SVM with Gaussian kernel are identical to those in our web video experiment. For MKL, NS-MKL and GL-MKL, we construct 3 different base Gaussian kernels for each feature type. The sigma values are determined by the mean μ of the Euclidean distance between all pairs of training instances, and we have $\{\mu \cdot 10^{-1}, \mu, \mu \cdot 10\}$ as the sigma values for the Gaussian kernels.

From this table, we can see that our GL-MKL is among the best of the MKL-based fusion methods. It is worth noting that, as pointed out in [30], the PACT descriptor exhibited promising discriminating ability especially for scene image data. Using the nonlinear SVMs with Gaussian kernels, it was reported in [30] that the use of PACT achieved about 83% and 78% MAP accuracies for the Scene-15 and UIUC Sport Event datasets, respectively. From our experiments, we observed that our GL-MKL automatically chose PACT as the most dominant feature after the feature fusion and learning stage, and thus a comparable performance at 83.54% was achieved for the Scene-15 dataset. As for the UIUC Sport Event dataset, in which both scene (i.e., places, courts, environments, etc.) and object (i.e., athletes, sport equipments, etc.) information are presented in images, our GLMKL considered features besides PACT also describing representative information, and thus a better performance at 81.11% was obtained. From the above experiments on both video and image datasets, we successfully verify the effectiveness of our GL-MKL for heterogeneous feature fusion.

TABLE V
SELECTED UCI DATASETS FOR OUR EXPERIMENTS ON HETEROGENEOUS VARIABLE SELECTION

Dataset	Wpbc	Wdbc	Ionosphere	Wine
Number of instances	198	569	351	178
Number of features	31	30	34	13
Number of classes	2	2	2	3
Heterogeneous features	Yes	Yes	No	Yes

VI. EXPERIMENTS: VARIABLE SELECTION

In this section, we evaluate the performance of variable selection on four UCI datasets[2] (see Table V for detailed descriptions). Among the datasets we consider, all contain heterogeneous variables except for the Ionosphere dataset. Besides, the Wine dataset contains multiple classes to be recognized.

A. Comparisons With MKL-Based Variable Selection Methods

For the first part of the experiments on variable selection, we compare our proposed GL-MKL with SVM (using all features), standard MKL [13], and NS-MKL [19]. Gaussian kernels are used for nonlinear mapping in SVM and all MKL-based methods. To deal with heterogeneous variables, we allow our proposed GL-MKL to choose among four different Gaussian kernels (with different σ) for each variable in the p-dimensional data space, so that our GL-MKL has a total of $4 \times p$ base kernels. We then group these kernels at feature level to enforce the group lasso constraint. Recall that, since our approach learns the optimal kernels for variable selection, we do not require any validation data to select σ. For all our tests, we randomly select 80% of the data for training, and the remaining as the test set data. Each experiment is repeated with 5 random trials, and we present the average recognition rate and the average size of the selected variable subset for each case, as shown in Tables VI and VII.

We note that none of the MKL-based methods assume that the optimal number of variables are known in advance, which is practical for variable selection. For different datasets and variable selection methods, the averaged recognition performance and the size of the selected variable subset are presented in Table VI. While the nonlinear SVM does not have the capability of selecting discriminating variables, it is used as the baseline classifier for comparisons. From Table VI, it can be observed that the recognition rates reported by nonsparse MKL, MKL, and our heterogenous variable selection method are statistically comparable to each other. However, it is worth noting that *our GL-MKL resulted in the most compact variable subset for each dataset*, as shown in the last column of Table VI. Therefore, these results verify the use of our method for variable selection with comparable recognition performance achieved.

[2]The UCI datasets are available at http://archive.ics.uci.edu/ml/

TABLE VI
PERFORMANCE COMPARISONS. FOR EACH DATASET AND FEATURE SELECTION APPROACH, THE AVERAGE RECOGNITION ACCURACY (%) AND ITS STANDARD DEVIATION ARE PRESENTED, FOLLOWED BY THE AVERAGE SIZE OF THE SELECTED FEATURE SUBSET NOTED IN (). WHILE COMPARABLE RECOGNITION RATES AMONG DIFFERENT APPROACHES ARE OBSERVED IN THIS TABLE, OUR METHOD SELECTS THE SMALLEST FEATURE SUBSET FOR EACH DATASET (HIGHLIGHTED IN BOLD), AND THUS PRODUCES PREFERABLE FEATURE SELECTION RESULTS

Dataset	SVM	Non-sparse MKL [19]	MKL [13]	Our method
Wdbc	93.81 ± 2.65 (30)	95.40 ± 1.81 (23)	95.22 ± 2.04 (7.2)	94.87 ± 4.26 (**4.2**)
Wpbc	76.12 ± 1.16 (31)	75.10 ± 2.46 (12)	75.71 ± 0.85 (3.2)	75.71 ± 1.33 (**2.4**)
Ionosphere	95.14 ± 2.17 (34)	87.43 ± 3.70 (33)	89.71 ± 1.86 (11.2)	93.71 ± 3.29 (**10.8**)
Wine	81.76 ± 2.46 (13)	92.94 ± 6.10 (11.8)	90.00 ± 4.92 (4.5)	95.29 ± 1.61 (**5.3**)

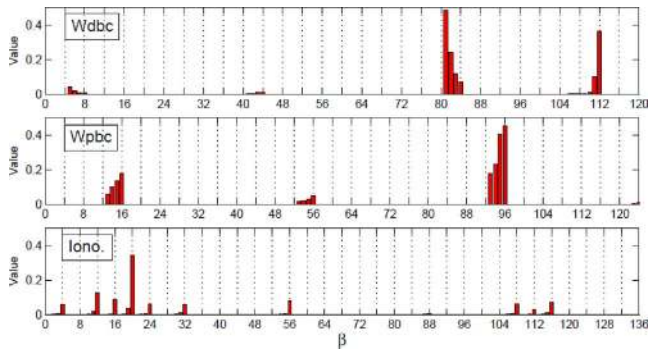

Fig. 8. Weights $\boldsymbol{\beta}$ determined for each base kernel and the corresponding variable. The x-axis is the index of β_{mj} (out of $k \times p = 4p$ for each dataset), and the y-axis is the associated weight. Each grid in the figure indicates a variable of interest, and the four components (red bars) in each grid represent the selected kernel weights.

Fig. 8 shows the weight β_{mj} for each base kernel and the associated variable using our GL-MKL. For each variable (i.e., each grid in Fig. 8), we assign a total of $k = 4$ base kernels, and the variable selection results are depicted by their weights (i.e., red bars in Fig. 8). It can be seen that our approach selected a compact variable subset (e.g., only 3 out of 31 variables were selected in Wpbc, as shown in Fig. 8), while the associated kernels need *not* be sparse. These results support that our approaches is able to provide a sparse yet discriminating variable subset, and achieves comparable performance as standard methods do.

B. Comparisons With Sequential-Based Selection Methods

We also compare our results with IFFS [33], and SFFS [34], which are state-of-the-art sequential-based variable selection methods and are popular due to its simplicity in implementation. The major concern of this type of approaches is that it needs to exhaustively search for the entire variable space for variable selection. Since these two methods need the prior knowledge of the variable subset size, we use the number of variables selected by our GL-MKL (determined in Table VI), and compare the recognition performance using the same size of the variable subset.

From Table VII, we see that our method outperforms SFFS and IFFS in terms of recognition on both heterogeneous or homogeneous variable data. Comparing with sequential-based variable selection methods, our method exhibits excellent ability in automatically determining the least number of variables when producing satisfactory recognition performance.

To make the comparisons more complete, we also search for the entire variable space on heterogeneous datasets using SFFS and IFFS, and we plot their corresponding averaged recognition rates in Fig. 9. It can be seen that, if the user does not specify the preferable number of variables to be selected, one will need to exhaustively search for the optimal size of the variable subset using sequential-based methods. When using our GL-MKL, the optimal number of variables can be selected automatically, while we achieve improved or comparable recognition performance (marked by black $*$ in Fig. 9) as the sequential-based methods do.

TABLE VII
PERFORMANCE COMPARISONS WITH SEQUENTIAL-BASED VARIABLE SELECTION METHODS. NOTE THAT BOTH SFFS AND IFFS USE ABOUT THE SAME NUMBER OF VARIABLES SELECTED BY OUR METHOD

Dataset	SFFS [34]	IFFS [33]	Our method
Wdbc	91.68 ± 2.28 (5)	94.51 ± 1.17 (5)	**94.87** ± 4.26 (4.2)
Wpbc	**76.92** ± 7.60 (3)	68.20 ± 8.82 (3)	75.71 ± 1.33 (2.4)
Ionosphere	89.42 ± 2.94 (11)	91.14 ± 1.89 (11)	**93.71** ± 3.29 (10.8)
Wine	90.85 ± 3.33 (5)	88.00 ± 5.54 (5)	**95.29** ± 1.61 (5.3)

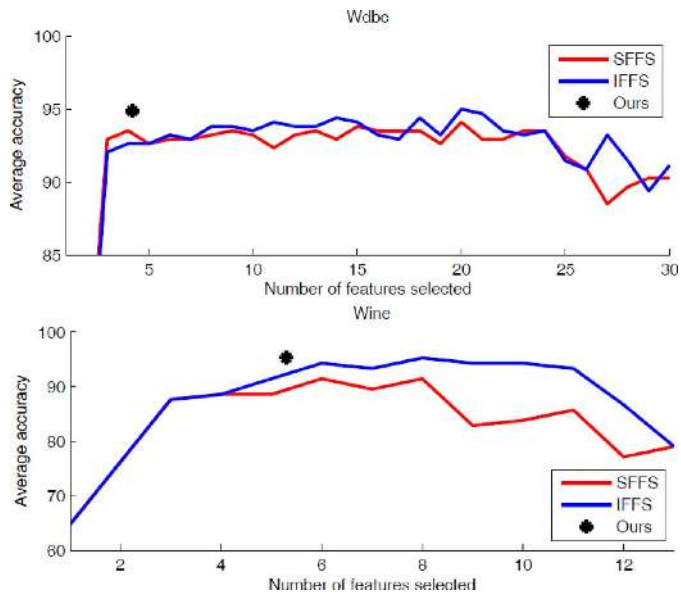

Fig. 9. Recognition performance of SFFS and IFFS on Wdbc and Wine datasets using different numbers of variables. The one reported by our GL-MKL is denoted by $*$ in the figure, which achieves the best or comparable performance with the smallest numbers of variables.

VII. CONCLUSION

A novel group lasso regularized MKL (GL-MKL) was proposed in this paper to address both heterogeneous feature fusion and variable selection problems. To deal with such heterogeneous data, we proposed to assign a group of base kernels

for each feature type or feature attribute. In order to automatically determine the kernel weights and parameters for above cases, we introduced a mixed $\ell_{1,2}$ norm constraint (i.e., group lasso) into the existing MKL algorithm. This regularizer would enforce the sparsity at the group level, while nonsparsity can be preserved for base kernels within the same group, which allows the MKL model to better describe the feature/variable extracted from different domains. It is worth noting that we do not need to explicitly choose any particular normalization techniques in the above problems, and we do not assume that the size of the variable subset to be known (or search for the entire variable space) as sequential-based selection methods do. Our experimental results on both video and image datasets (for heterogeneous feature fusion) and several UCI datasets (for heterogeneous variable selection) confirmed the effectiveness and robustness of our proposed framework.

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.

[3] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 221–228.

[4] T. TZhang, G. Xu, G. Zhu, S. Liu, and H. Lu, "A generic framework for event detection in various video domains," in *ACM Multimedia*, 2010, pp. 103–112.

[5] W. Jiang, C. Cotton, S.-F. Chang, D. Ellis, and A. C. Loui, "Audio-visual atoms for generic video concept classification," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 6, no. 3, pp. 14:1–14:19, 2010.

[6] L. Cao, J. Luo, F. Liang, and T. S. Huang, "Heterogeneous feature machines for visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 1095–1102.

[7] A. Hanjalic and L.-Q. Xu, "Affective video content representation and modeling," *IEEE Trans. Multimedia*, vol. 7, no. 1, pp. 143–154, Feb. 2005.

[8] G. Irie, T. Satou, A. Kojima, T. Yamasaki, and K. Aizawa, "Affective audio-visual words and latent topic driving model for realizing movie affective scene classification," *IEEE Trans. Multimedia*, vol. 12, no. 6, pp. 523–535, Oct. 2010.

[9] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao, "Group-sensitive multiple kernel learning for object categorization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 436–443.

[10] S.-W. Sun, Y.-C. F. Wang, Y.-L. Hung, C.-L. Chang, S.-S. Chen, H.-M. Wang, and H.-Y. M. Liao, "Automatic annotation of web videos," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2011, pp. 1–6.

[11] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: The MIT Press, 2002.

[12] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proc. Int. Conf. Mach. Learn.*, 2004, p. 6.

[13] A. D. Dileep and C. C. Sekhar, "Representation and feature selection using multiple kernel learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2009, pp. 717–722.

[14] Z. Xu, R. Jin, J. Ye, M. R. Lyu, and I. King, "Non-monotonic feature selection," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 1145–1152.

[15] A. Kumar and C. Sminchisescu, "Support kernel machines for object recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[16] M. Varma and D. Ray, "Learning the discriminative power-invariance trade-off," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

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

[18] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *J. Mach. Learn. Res.*, vol. 9, pp. 2491–2521, 2008.

[19] M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg, "Nonsparse multiple kernel learning," in *Proc. NIPS Workshop Kernel Learn.: Automatic Selection of Optimal Kernels*, 2008.

[20] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *J. Mach. Learn. Res.*, vol. 7, pp. 1531–1565, 2006.

[21] C.-C. Chang and C.-J. Lin, LIBSVM: A Library for Support Vector Machines 2001 [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[22] Y.-R. Yeh, Y.-Y. Chung, T.-C. Lin, and Y.-C. F. Wang, "Group lasso regularized multiple kernel learning for heterogeneous feature selection," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2011, pp. 103–112.

[23] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.

[24] A. Yanagawa, W. Hsu, and S. F. Chang, "Brief descriptions of visual features for baseline Trecvid concept detectors," Columbia Univ., Tech. Rep., 2006.

[25] A. Vailaya, A. Jain, and J. Z. Hong, "On image classification: City vs. landscape," in *Proc. IEEE Workshop Content-Based Access of Image and Video Libraries*, 1998, pp. 3–8.

[26] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.

[27] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 689–696.

[28] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 2169–2178.

[29] L.-J. Li and L. Fei-Fei, "What, where and who? classifying event by scene and object recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.

[30] J. Wu and J. M. Rehg, "Centrist: A visual descriptor for scene categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1489–1501, Aug. 2011.

[31] E. Shechtman and M. Irani, "Matching local self-similarities across images and videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[32] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[33] S. Nakariyakul and D. Casasent, "An improvement on floating search algorithms for feature subset selection," *Pattern Recognit.*, vol. 42, no. 9, pp. 1932–1940, 2009.

[34] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognit. Lett.*, vol. 15, no. 11, pp. 1119–1125, 1994.

Yi-Ren Yeh received the M.S. and Ph.D. degrees from the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taiwan, in 2006 and 2010, respectively.

From August 2008 to May 2009, he was a visiting scholar of CyLab, Carnegie Mellon University, Pittsburgh, PA. He is currently a postdoctoral research fellow of the Research Center for Information Technology Innovation (CITI) at Academia Sinica, Taipei, Taiwan. His research interests include machine learning, data mining, optimization, numerical methods, and pattern recognition

Ting-Chu Lin received the B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, in 2004 and 2010, respectively.

She is currently a research assistance of Multimedia and Machine Learning (MML) lab, the Research Center for Information Technology Innovation (CITI) at Academia Sinica, Taipei, Taiwan. Her research interests include machine learning, data mining, information retrieval, and multimedia.

Yung-Yu Chung received the B.S. degree from the Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, in 2007.

From November 2009 to July 2010, he was a research assistant of Multimedia and Machine Learning (MML) lab, the Research Center for Information Technology Innovation (CITI), Academia Sinica, Taipei, Taiwan. He is currently a graduate student in the Department of Computer Engineering, Iowa State University, Ames. In the summer of 2011, he took the intern in MML lab, CITI, Academia Sinica, Taipei, Taiwan. His research interests include machine learning, feature selection, pattern recognition, network optimization, and heterogeneous network.

Yu-Chiang Frank Wang (M'09) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2001, and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2004 and 2009, respectively.

From 2001 to 2002, he worked as a research assistant at the National Health Research Institutes, Taiwan. Since 2009, Dr. Wang has joined the Research Center for Information Technology Innovation (CITI) of Academia Sinica, Taiwan, where he holds the position as a tenure-track assistant research fellow. He leads the Multimedia and Machine Learning Lab at CITI, and works in the fields of signal and image processing, computer vision, and machine learning. From 2010 to 2011, he was a visiting scholar of the Department of Computer Science and Information Engineering at National Taiwan University Science and Technology, Taiwan.