Heterogeneous Feature Machines for Visual Recognition

Liangliang Cao[♮] Jiebo Luo[†] Feng Liang[‡] Thomas S. Huang[♮]
[♮]Beckman Institute and Coordinated Science Lab, Department of ECE, UIUC
[†]Kodak Research Laboratories, Eastman Kodak Company, Rochester, USA
[‡]Department of Statistics, University of Illinois at Urbana-Champaign

Abstract

With the recent efforts made by computer vision researchers, more and more types of features have been designed to describe various aspects of visual characteristics. Modeling such heterogeneous features has become an increasingly critical issue. In this paper, we propose a machinery called the Heterogeneous Feature Machine (HFM) to effectively solve visual recognition tasks in need of multiple types of features. Our HFM builds a kernel logistic regression model based on similarities that combine different features and distance metrics. Different from existing approaches that use a linear weighting scheme to combine different features, HFM does not require the weights to remain the same across different samples, and therefore can effectively handle features of different types with different metrics. To prevent the model from overfitting, we employ the so-called group LASSO constraints to reduce model complexity. In addition, we propose a fast algorithm based on co-ordinate gradient descent to efficiently train a HFM. The power of the proposed scheme is demonstrated across a wide variety of visual recognition tasks including scene, event and action recognition.

1. Introduction

As one of the most popular machine learning tools, Kernel Machines (KMs), such as support vector machines, kernel discriminant analysis, and kernel partial least squares regression, are extensively used for visual recognition tasks. In KMs, the input data is implicitly embedded into a high dimensional (or infinite dimensional) space by a nonlinear mapping. Linear functions in the transformed kernel space correspond to a rich class of nonlinear functions in the original data space, which form the so-called reproducing kernel Hilbert space (RKHS). Most of the KMs, including the popular support vector machines, solves the classification function $f(\mathbf{x})$ which minimizes

$$\min_{f\in\mathcal{F}} L(f) + \lambda R(f), \tag{1}$$

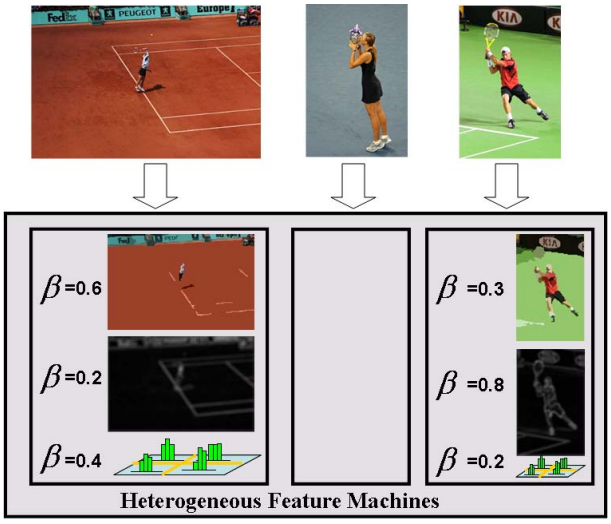

Figure 1. HFM for recognizing a "tennis" event: HFM provides a flexible weighting scheme to combine multiple heterogeneous features (e.g., color, MSER, HOG, and SIFT) while also removing irrelevant samples (e.g., the middle image). Here $\boldsymbol{\beta}$ denotes the weights of features assigning to different samples. Note in our scheme the summation of $\boldsymbol{\beta}$ is not necessarily 1.

where $L(f)$ denotes the empirical loss over the training samples, λ is a tuning parameter, and the regularization term $R(f)$ is usually a monotone function of the RKHS norm of f. Though the function space $\mathcal{F}$ may be infinite dimensional, by the representer theorem [17], the solution of (1) takes the following finite dimensional form

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i), \tag{2}$$

where α_i's are the unknown kernel regression coefficients and $K(\cdot,\cdot)$ is the kernel function that represents the inner product between $\mathbf{x}$ and $\mathbf{x_i}$ in the kernel space.

In practice, choices of kernel functions are critical. Classic kernels functions, such as the linear, polynomial and Gaussian kernel functions, are based on one global similarity metric in the data space. Such a limitation was not

recognized in problems concerned with only a single type of features, such as in text analysis and computational biology. For complex visual recognition tasks, however, a subject is often associated with multiple visual measurements, which can be either global features (e.g., shape, texture, color moment [35], HOG [10], and GIST[27]) or local features (e.g., SIFT [22], MSER [24], shape context [3], and Local Binary Pattern (LBP) [26]). Different features describe different aspects of the visual characteristics and demand different metrics, for example, Euclidean distances are often used as a similarity measure for global color histograms, while spatial matching kernels [20] have been widely employed for local SIFT descriptors. How to handle heterogeneous features becomes an important problem in computer vision.

In this paper, we propose a novel approach to handling multiple features. Suppose we have a collection of samples $\{\mathbf{x}_i, y_i\}$ with $1 \leq i \leq N$. Each y_i denotes the label of a sample, and $\mathbf{x}_i = [\mathbf{x}_i^1, \mathbf{x}_i^2, ...\mathbf{x}_i^M]$ denotes the corresponding measurements of M features, where each feature $\mathbf{x}_i^m$ may be a multivariate vector or a collection of vectors when representing an image in the bag-of-feature framework. For each feature m, the similarity metric between two samples is represented by $s^m(\mathbf{x}^m, \mathbf{z}^m)$. Then we model the classification function by

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{N} \sum_{m=1}^{M} \beta_i^m s^m(\mathbf{x}^m, \mathbf{x}_i^m), \tag{3}$$

where β_i^m is the unknown kernel regression coefficient associated with the ith sample and the mth feature. The new model provides a flexible way to fuse multiple features, where the fusion weights are formulated as part of the kernel regression coefficients and will be adaptively estimated from the data. Instead of having a global set of fusion weights to balance the contribution of each feature, our model has data dependent weights and therefore leads to a nonlinear and locally adaptive fusion of multiple features. Since our new model (Eq. 3) is designed to integrate heterogeneous features, we call it a Heterogeneous Feature Machine (HFM).

We illustrate the motivation of employing a flexible, sample-dependent weighting scheme in Figure 1. A visual event such as a tennis game concerns multiple visual characteristics (e.g, color, MSER, HOG, and SIFT). Note that the importance of a feature varies from one sample image to another. From a distant court view (left photo), the background of tennis court provides most of the cues for classification and thus the color feature is the most important. On the other hand, from a close-up (right photo), the player's body figure contains most of the information and thus shape features become relevant. Therefore to model their varying influence, features should be combined in a nonlinear fashion. In addition to a flexible weighting scheme, we also intend to reduce the effects of outlier samples and exclude features related to the irrelevant samples. In Figure 1, the middle photo in which the tennis star Sharapova kisses the US Open trophy should not contribute features to the event model, as indicated by the blank box, because it does not exhibit any typical characteristics of a general tennis event.

The challenge brought by the flexibility of our HFM is the large number of parameters we need to estimate. To reduce the model complexity and computation burden, we employ a novel regularization called group LASSO [36] in the final cost function. Different from the traditional l_2 norm regularization used in SVM or l_1 norm used in sparse coding approach [33], the group LASSO is designed specially for grouped coefficients and induces shrinkage and variable selection at the group level. In our model, it regularizes multiple features simultaneously to capture different characteristics of the same object.

The remaining of this paper is arranged as follows. In Section 2, we present the heterogeneous feature machine as a flexible model for multiple features. In Section 3, we propose an efficient learning algorithm for parameter estimation. In Section 4, we discuss the difference and connection with respect to related work, and particularly draw contrast between HFM and multiple kernel learning. Section 5 and Section 6 show the experimental results in recognizing events from images and actions from videos, respectively. Conclusions are given in Section 7.

2. A Model for Heterogeneous Features

Rewrite the HFM defined in Eq. (3) as the following

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{N} \boldsymbol{\beta}_i^T \mathbf{s}_i(x), \tag{4}$$

where we group the similarity measures and regression coefficients associated with the ith sample as

$$\mathbf{s}_i(\mathbf{x}) = [s^1(\mathbf{x}_i, \mathbf{x}), s^2(\mathbf{x}_i, \mathbf{x}), ..., s^M(\mathbf{x}_i, \mathbf{x})]^T \tag{5}$$

and

$$\boldsymbol{\beta}_i = [\beta_i^1, \beta_i^2, ..., \beta_i^M]^T, \tag{6}$$

which both are M-dimensional vectors.

To solve $f(\mathbf{x})$, we minimize the following loss function

$$L(\boldsymbol{\beta}) = \sum_i \log \frac{\exp(y_i f(\mathbf{x}_i))}{1 + \exp(f(\mathbf{x}_i))}, \tag{7}$$

which is the negative log-likelihood from a logistic regression. The logistic loss is appealing because: (a) besides producing a hard classification result, (7) also offers a natural estimate of the posterior probability $p(\mathbf{x}) = e^f/(1+e^f)$, while the traditional SVM scores are not reliable for measuring probability ; and (b) the logistic loss function is differentiable, which is preferred for optimization. We will

show later that such a loss function gives rise to an efficient learning algorithm.

Though Eq. (7) holds only for two-class problems where $y_i = 0$ or 1, it can be easily extended to model multiple classes following the extension of multivariate logistic regression. For simplicity, we focus on binary problems in this paper, and use a one-vs.-all approach for multivariate classification problems.

Due to the large number of parameters, directly minimizing the logistic loss will cause an overfitting problem. To learn a sparse classifier, we employ the group LASSO regularization, with which the final cost function becomes

$$C_{group} = -L(\boldsymbol{\beta}) + \lambda \sum_{i=1}^{N} ||\boldsymbol{\beta}_i||_2 \tag{8}$$

where λ is a tuning parameter and $||\cdot||_2$ denotes the l_2 norm. Note that the regularization term $\lambda \sum_{i=1}^{N} ||\boldsymbol{\beta}_i||_2$ uses the l_2 norm within a group and the l_1 norm between groups. Recall that the classic l_2 norm produces soft shrinkage on the regression coefficients, while l_1 norm yields sparse solutions by shrinking some of the coefficients to zero. The combination of these two norms in group LASSO leads to a sparse constraint at the group level. In other words, for a noisy sample, all the corresponding M coefficients $\boldsymbol{\beta}_i = [\beta_i^1, \beta_i^2, ..., \beta_i^M]$ will be set to zero. Only the discriminant samples with nonzero $\boldsymbol{\beta}$ contribute to the final classifier. In the remaining of this paper, we refer to those samples as *support samples* analogous to *support vectors* in SVMs.

3. An Efficient Learning Algorithm

Considering the computational burden, the model in (8) calls for a new, more efficient algorithm. In this work, we proposed an efficient approach to solve our optimization problem (8), using the block Co-ordinate Gradient Descent (CGD) method from Tseng and Yun [31]. Their method has also been employed by Meier *et al*. [25] for solving group LASSO constraints in a different context.

Since the group LASSO regularization term is not differentiable everywhere, it is hard to minimize (8) directly. However, since the regularization term can be separated into smaller components due to the group structure, Tseng and Yun [31]suggested to carry out the optimization procedure in a group-wise way. For simplicity, we outline the group-wise optimization framework in Algorithm 1 and explain the details in the remaining of this section.

Following the suggestion in [31], we first approximate the original cost function C_{group} by its quadratic expansion:

$$C_{group} \approx -\left(L(\boldsymbol{\beta}) + \nabla L^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \mathbf{H} \mathbf{d}\right) + \lambda \sum_{i=1}^{N} ||\boldsymbol{\beta}_i + \mathbf{d}_i||_2$$

Algorithm 1 : Outline of group-wise optimization.

1: do iteration $t = 1, 2, ...$
2: **for each** group i **do**
3: find the optimal $\boldsymbol{\beta}_i^{t+1}$ to update $\boldsymbol{\beta}_i^t$
4: **end for**
5: stop if the optimization is converged.

Output: Obtain the classification function f with parameters $(\beta_0, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ...)$.

where $L(\boldsymbol{\beta})$ represents the first item in eq. (8), and $\mathbf{d}_i$ denotes the direction in which $\boldsymbol{\beta}_i$ should be updated. $\mathbf{H}$ is a diagonal matrix which approximates the Hessian of $L(\boldsymbol{\beta})$ with the form of $\mathbf{H} = \text{diag}(\mathbf{H}_1, \mathbf{H}_2, ..., \mathbf{H}_N)$, where $\mathbf{H}_i$ is a diagonal matrix $\mathbf{H}_i = h_i \mathbf{I}$, with $h_i = \max[diag(-\nabla_{ii}^2 L), 10^{-3}]$.

It is easier to minimize the quadratic approximation than (8). For each group, we minimize the following equation iteratively:

$$C_i = -\left(L(\boldsymbol{\beta}) + \nabla_i L^T \mathbf{d}_i + \frac{1}{2}\mathbf{d}_i^T \mathbf{H}_i \mathbf{d}_i\right) + \lambda ||\boldsymbol{\beta}_i + \mathbf{d}_i||_2. \tag{9}$$

When $||\nabla_i L(\boldsymbol{\beta}) - h_i \boldsymbol{\beta}_i||_2 \leq \lambda$, it is straightforward to update $\boldsymbol{\beta}_i$ by

$$\boldsymbol{\beta}_i^* = \boldsymbol{\beta}_i + \mathbf{d}_i = 0. \tag{10}$$

Otherwise

$$\mathbf{d}_i = -\mathbf{H}_i^{-1}\left[\nabla_i L(\boldsymbol{\beta}) - \lambda \frac{\nabla_i L(\boldsymbol{\beta}) - h_i \boldsymbol{\beta}_i}{||\nabla_i L(\boldsymbol{\beta}) - h_i \boldsymbol{\beta}_i||_2}\right]. \tag{11}$$

Then the parameter is updated by $\boldsymbol{\beta}_i^* = \boldsymbol{\beta}_i + \alpha \mathbf{d}_i$, where α is the step size. The optimal α is obtained by Armijo rule [16], which satisfies

$$C_i(\boldsymbol{\beta}_i + \alpha \mathbf{d}_i) - C_i(\boldsymbol{\beta}_i) \leq \sigma \alpha ||\nabla C_i||, \tag{12}$$

where $||\nabla C_i|| = -\mathbf{d}_i \nabla L(\boldsymbol{\beta}) + \lambda(||\boldsymbol{\beta}_i + \mathbf{d}_i||_2 - ||\boldsymbol{\beta}_i||_2)$.

Note that we need to compute $L(\boldsymbol{\beta})$ and $\nabla_i L(\boldsymbol{\beta})$ for each i in minimizing C_i via (11) and (12). This can be very costly when the number of groups (i.e., the sample size here) is large. To improve the efficiency, we employ the following technique in our algorithm.

By defining $\mathbf{s}_0(\mathbf{x}_j) = 1$, we can write the likelihood as

$$L = \sum_{j=1}^{N} y_j \sum_{i=0}^{N} \boldsymbol{\beta}_i^T \mathbf{s}_i(\mathbf{x}_j) - \sum_{j=1}^{N} \log(1 + \exp(\sum_{i=0}^{N} \boldsymbol{\beta}_i^T \mathbf{s}_i(\mathbf{x}_j))).$$

Observe that the computation of $\sum_i \beta_i^T \mathbf{s}_i(\mathbf{x}_j)$ involves N^2 multiplication and $N + 1$ addition. Our trick is to introduce a matrix $\mathbf{Z} = [Z(i,j)]_{(N+1)\times N}$, with $Z(i,j) = \boldsymbol{\beta}_i^T \mathbf{s}_i(\mathbf{x}_j)$. Since the CGD approach minimizes the cost function in a group-wise way, only the i^*th column of $\mathbf{Z}$ will be updated

for group i^*. Then the likelihood can be computed using

$$L = \sum_j y_j \left(\sum_{i \neq i^*} Z(i,j) + \boldsymbol{\beta}_{i^*}^T \mathbf{s}_{i^*}(\mathbf{x}_j) \right) - \sum_j \log \left(1 + \exp(\sum_{i \neq i^*} Z(i,j) + \boldsymbol{\beta}_{i^*}^T \mathbf{s}_{i^*}(\mathbf{x}_j)) \right). \quad (13)$$

Now the computation of $\sum_i \beta_i^T \mathbf{s}_i(\mathbf{x}_j)$ is reduced to N multiplication with $N+1$ addition. Similarly, the computation of ∇L can also be reduced by the use of $\mathbf{Z}$. Algorithm 2 summarizes the detailed algorithm.

Algorithm 2 : Efficient CGD solver.

Initialize $\beta^0 \in \mathbb{R}^M$.
compute $\mathbf{Z}$
do iteration $t = 1, 2, ...$
    Compute $\mathbf{H}^t$.
    for each group i
        Compute L and $\nabla_i L$ based on $\mathbf{Z}$.
        Get optimal $\mathbf{d}^t$ from (10) and (11).
        Get optimal α^t using line search.
        Set $\boldsymbol{\beta}_i^{t+1} = \boldsymbol{\beta}_i^t + \alpha^t \mathbf{d}^t$.
        update the ith column of $\mathbf{Z}$
    end for
stop if the optimization is converged.

Output: Obtain the classification function f with parameters $(\beta_0, \boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ...)$.

After the parameters are learned, we can estimate how likely a testing sample belongs to a class by computing the probability $p = (1 + e^{-f})^{-1}$ and hence find the most possible class label.

4. Related Work

In this section, we address the differences and connections between our model and some recently developed KM algorithms, especially those dealing with feature fusion.

A common approach to fusing multiple features is to use different kernels for different features and then combine them by a weighted summation

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_m w_m K_m(\mathbf{x}_i, \mathbf{x}_j), \quad (14)$$

which is called Multiple Kernel Learning (MKL) [18][2] [30] [28] [32]. In MKL, the weight w_m for the mth feature does not depend on $\mathbf{x}$ and remains the same across all the samples. Consequently, such a linear weighting approach has a restricted fusion capability and would fail to describe possible nonlinear relationships among different types of features.

Recently, Gnen and Alpaydin [13] proposed a localized weighting approach to MKL by introducing a weighting function $w(\mathbf{x})$ that is assumed to either a linear or quadratic function of the input vector $\mathbf{x}$. The localized MKL in [13] works well for low dimensional data, however, for high dimensional data from visual applications, its performance is deteriorated by the extra model complexity from estimating a large amount of unknown parameters for the weighting function. Furthermore, some data from visual applications are non-vectors embedded with spatial information. It is difficulty to choose appropriate weighting function for such data, since there does not exist a default choice of metric or coordinate system for the weighting function to be defined on.

Our HFM differs from the MKL work by Bach [1] which also employed the group LASSO penalty. In [1] the kernel coefficients related to the same kernel form a block, while in our work, the coefficients related to the same sample form a block. The approach in [1] works well for problems with more features than the number of samples, where the common premise held there is that many features are irrelevant. In computer vision, however, features characterize different aspects of samples and should not be given up, and on the other hand, we often have too many samples. So HFM employ the group lasso to eliminate samples with less discriminant power and to find the support samples as an analogue of support vectors in SVM.

To illustrate how well our HFM works, we compare the classification accuracy of HFM and MKL algorithms including SILP [30] SimpleMKL [28], and localized MKL [13] on UCI liver repository [4]. We use 70% of the samples for training and 30% for testing and repeat the experiment 20 times. Following [28], we construct 91 kernel matrix with different parameters and types, which is based either on a single variable or on all the variables using the same parameters as in [28]. Table 1 lists the average accuracy of SILP, SimpleMKL, Localized MKL, and our HFM with different tuning parameter λ. Our model is consistently better than all MKL approaches, and our computational time is

Table 1. Comparing the recognition results between HFM and MKL in UCI liver data set

Algorithm	Accuracy	Time (secs)
SILP [30]	$65.9 \pm 2.6\%$	47.6 ± 9.8
SimpleMKL [28]	$65.9 \pm 2.3\%$	18.9 ± 12.6
Localized MKL [13]	$61.5 \pm 2.1\%$	140.1 ± 10.4
HFM ($\lambda = 0.2$)	$69.6 \pm 4.8\%$	16.2 ± 1.7
HFM ($\lambda = 0.1$)	$70.8 \pm 4.4\%$	19.9 ± 2.1
HFM ($\lambda = 0.06$)	$72.0 \pm 4.3\%$	26.2 ± 2.5
HFM ($\lambda = 0.05$)	$72.6 \pm 3.9\%$	27.0 ± 4.0
HFM ($\lambda = 0.04$)	$72.8 \pm 4.0\%$	29.9 ± 4.9
HFM ($\lambda = 0.02$)	$72.7 \pm 4.3\%$	37.0 ± 6.9
HFM ($\lambda = 0.01$)	$72.7 \pm 4.0\%$	45.2 ± 9.7

comparable to the best one (SimpleMKL).

The advantage of HFM over MKL lies in the flexibility in combining features. In our model, the weights of kernels vary from sample to sample, which in fact amounts to a nonlinear fusion of the multiple kernel functions. The fusion scheme of HFM is more general than that of MKL, and therefore can lead to a better performance in practice. Note that SimpleMKL selects the sparse representation over 91 kernel matrices, while our HFM looks for a sparse selection over 241×91 training vectors. Performance of our model on some real applications will be represented in Section 5 and 6.

The MKL approach has been adopted by vision researchers. Varma and Ray [32] and Bosch *et al*. [7] reported excellent recognition results on the Caltech 101 dataset. Same as MKL, their methods keep the feature weights fixed across all the samples. As future work, we plan to generalize our logistic loss to multiple classes and it would be interesting to compare the generalized HFM model with their methods on problems with a large number of classes.

Recently, there has been a growing interest on l_1 norm regularization in the field of machine learning. Roth [29] extended the l_1 (i.e., LASSO) regression to the kernel setting:

$$C = \sum_{i=1}^{N} \left[y_i - \big(\alpha_0 + \sum_j \alpha_j K(x_i, x_j)\big) \right]^2 + \lambda ||\alpha||_1, \tag{15}$$

which is termed as the generalized LASSO. Another group of approaches is to combine SVM with l_1 regularization [37], which employs the following cost function

$$C = \sum_{i=1}^{N} \left[1 - y_i \big(\alpha_0 + \sum_j \alpha_j K(x_i, x_j)\big) \right]_+ + \lambda ||\alpha||_1, \tag{16}$$

Instead of the least-square loss or hinge loss, our model uses a likelihood-based loss function. Moreover, we employ a group LASSO regularization instead of l_1 to handle both sparsity and multiple-feature fusion.

The l_1 norm regression has also been studied in the field of remote sensing and sparse coding (see [9] for a survey). For example, Wright *et al*. [33], Mairal *et al*. [23] and Yang *et al*. [34] all adopt the idea of sparse coding for feature selection on computer vision problems. Their methods select a proportion of all the features and provide a robust prediction when noise occurs in some of the feature dimensions, or when part of the image is corrupted. In contrast, our model considers the noise at the sample level, and selects a small set of discriminant samples among all the training data. In addition, the work in [33] [23] considers only a single type of features, while our model aims to efficiently fuse heterogeneous features.

5. Event Recognition from Images

We tested our proposed model on two event databases: the Princeton sports event dataset [21] and Jain's Flickr sports event dataset [14]. The Princeton dataset was collected from the Internet, and was later annotated thoroughly by Lotus Hill Research Institute. There are 8 sports categories in the dataset: bocce, croquet, polo, rowing, snowboarding, badminton, sailing, and rock climbing. The number of images in each category varies from 137 to 250. In each image, there are always one or more athletes, and the poses and sizes of these athletes differ significantly. In this work, we take the same classification protocol setup as in [21]: For each event class, 70 randomly selected images are used for training and 60 of the remaining images are used for testing.

The second dataset is collected by Jain et al. [14]. This dataset contains 2449 Flickr images mostly taken by amateur photographers, and often from a distance. Unlike the Princeton dataset, this dataset focuses more on popular American sports: baseball, basketball, football, soccer, and tennis. Following the protocol of [14], we randomly select 50% images for training, and the remaining 50% for testing.

In our experiments, we employ the popular image features such as GIST [27], Histogram of Oriented Gradients (HOG) [10], color moment [35], and Local Binary Pattern (LBP) [26]. These features characterized different aspects of the images. More specifically, GIST is built on the responses of Gabor filters to represent the spatial structure of a scene, and HOG aims to model the distribution of intensity gradients and edge directions, while color moments and LBP are used as measures of colors and textures, respectively. In our work, each image is divided into 4×4 tiled regions, and each region is represented by a specified feature vector. These vectors are concatenated into a long vector and then normalized by the standard deviation multiplied by the dimension of the feature. We use the Gaussian radial basis function to measure the similarity between two images.

Since our model can effectively handle features with different metrics, we also adopt Lazebnik's spatial pyramid matching (SPM) [20] due to its proven effectiveness. Spatial pyramid matching is a simple yet effective approach to compare similarity between images. The idea is to partition the image into sub-regions, and then compute the intersections of SIFT histograms [22] at different resolutions. Suppose we have a sequence of resolutions $0, 1, .., L$, we have 2^l sub-regions for the lth resolution. Let H_i^l and H_j^l denote the histogram of image i and image j in resolution l, we can compute the histogram intersection as $I(H_1^l, H_2^l) = \sum_{r=1}^{2^l} \min\big(H_i^l(r), H_j^l(r)\big)$. Then the simi-

Table 2. Recognition results on the Princeton sports event dataset.

Method	Accuracy
Scene model in [21]	60%
Object model in [21]	49%
Object + geometry model in [21]	46%
Object + scene model in [21]	73.4%
Our HFM	**79.2%**

Table 3. Recognition results on Jain's sports event dataset.

Method	Accuracy
SVM [14]	61.4%
SHRF [14]	65.3%
Our HFM	**73.1%**

larity between two images is then defined as

$$s(H_1, H_2) = \sum_{l=0}^{L} \omega_l I(H_1^l, H_2^l), \tag{17}$$

where the weight is $\omega_l = \max\left(\frac{1}{2^L}, \frac{1}{2^{L-l+1}}\right)$. The resulting similarity is normalized between $[0, 1]$.

We use the proposed HFM to combine multiple features for image classification. Since our approach is not sensitive to λ, we fix it as $1e^{-5}$ instead of searching for the best performance among a set of values. To show the effectiveness of our HFM approach, we compared with the classification results using random forests [8] and Bayesian Net with the help of public software Weka [12]. Note that MKL algorithm is not fit for our setting since there are only a small number of features ($M = 5$ for image and $m = 4$ for video) and it is difficult to normalize kernels of different metrics.

On the Princeton event dataset, the original authors built generative models considering the events themselves, as well as the scenes and objects associated with the events. Several variations of their basic model were evaluated in [21], among which the model with objects and scenes was reported to be the best. Note that their approach requires thorough annotation of the photos in terms of scenes and objects. In contrast, our model builds a discriminant model and only needs the event labels. As shown in Table 2, our HFM outperforms all the models in [21] by significant margins. It shows that by combining heterogeneous features, HFM not only improves the recognition accuracy, but also eliminates the burden of acquiring detailed annotation for training the models.

Table 3 compares the performance of HFM with the previous work by [14]. Jain *et al*. present a model named Selective Hidden Random Fields (SHRF) by generalizing the popular conditional random field model. They showed that SHRF outperformed the performance of classic SVM and was the state-of-the-art algorithm on their dataset [14]. Our HFM approach improves the recognition accuracy from 65.3% to 72.3%.

Table 4. Comparison of the effect of individual features. Dataset A is the Princeton event dataset, and Dataset B is Jain's event dataset. Each number is the accuracy in percentage.

Features	Acc. on dataset A	Acc. on dataset B
Gist	*63.1%*	40.0%
HOG	49.8%	42.7%
LBP	51.2%	62.3%
Color Moment	59.8%	56.7%
SPM	59.4%	*63.3%*
Bayesian Net	61.3%	58.5%
Random forest	64.2%	68.4%
HFM	**79.2%**	**73.1%**

We also compared the accuracy of using single type features versus multiple type features. When single type features are used, the dimension of each feature group is 1 and the group LASSO regularization degenerates to l_1 norm regularization. CGD algorithm can still be employed to find the optimal parameters. Table 4 shows the performance of each feature type. From our experiments, GIST feature is the best for event recognition on the Princeton dataset, while SPM is the best for Jain's event dataset. When using multiple features, we compare the performances of Bayesian net, random forests, and our HFM model. Table 4 shows that both Bayesian net and random forests obtain good performance in the two datasets. However, our proposed HFM achieved the highest recognition accuracy and decisively outperforms the all single type features.

6. Action Recognition from Videos

Figure 2. CMU event dataset. From (a) to (e) are examples of five action categories: jumping, one-hand waving, pickup, push-up, and two-hand waving. These videos were shot by hand-held cameras, with crowded and cluttered backgrounds.

Next, we will show that the proposed HFM is also suitable for video action recognition tasks. We are interested in recognizing human actions in the real world, where the background might be cluttered and the camera is not always steady. We are not interested in action recognition with clean background and in well-controlled environments. In the real world, human actions often occur in crowded, dy-

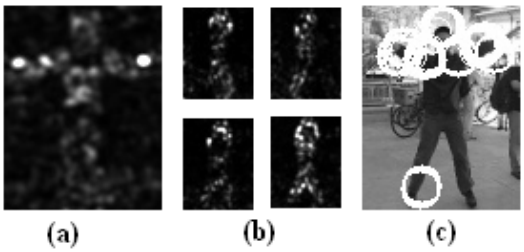

Figure 3. Heterogeneous features used for video action recognition. (a) motion history image. (b) Efros et al.'s four channel motion descriptor. (c) spatio-temporal feature detector.

namic environments. It is unlikely to find one feature that can reliably characterize such complex scenarios. We hope to use HFM to find the optimal feature combination.

We tested our algorithm on the CMU event dataset collected by Ke et al. [15]. Unlike the "clean" videos such as KTH and Weizman, the CMU event videos were taken with a hand-held camera in crowded environments with moving people or cars in the background. Figure 2 shows a few shots from this dataset . There are more than 100 action clips, which represent real world actions such as jumping, picking up an object from the ground, waving for a bus, or pushing an elevator button.

In Ke et al.'s original work [15], a user needs to interactively segment a spatio-temporal template for each action, which is then manually broken into parts. The recognition performance is heavily dependent on the generality of the user-specified model. In this paper, we try to recognize the events in a fully automatic manner. By randomly selecting 50% of the video clips, we train a HFM without any human interaction and then use the trained model to recognize the actions in the remaining video clips.

We consider three types of features for the video recognition task: Efros et al.'s four-channel motion descriptor [11], Bobick and Davis's motion history image [5], and Laptev's spatio-temporal features [19]. These descriptors are certainly of different nature. By using the first two descriptors, each video i is represented by a sequence of features $\mathbf{x}_i^m(t)$, with $1 \leq t \leq T_i$. The similarity between two videos can be measured by averaging the similarity over the the corresponding frames from the two sequences.

Unlike the first two descriptors, Laptev et al.'s feature is based on local detectors. They proposed to find the salient regions in the spatio-temporal domain, and then use different descriptors to describe the detected regions. Note that there is no perfect solution for the salient region descriptors, and we tried both Histogram of Oriented Gradients (HOG) and Histograms of Oriented Flow (HOF) as in [19]. These features are converted into histograms and the Chi-square distance is used to measure the similarity between two histograms. Suppose there are two histograms $H_i = (H_i(1), H_i(2), ..., H_i(n))^T$ and $H_j = (H_j(1), H_j(2), ..., H_j(n))^T$, where n denotes the number of bins of the histogram. Then the Chi-square similarity becomes

$$s(H_i, H_j) = \sum_{k=1}^{n} \frac{(H_i(k) - H_j(k))^2}{H_i(k) + H_j(k)}, \tag{18}$$

which is also normalized between $[0, 1]$.

Table 5. Comparison of the effect of different features for action recognition.

Features	Acc. on CMU event dataset
Efrons's feature	*65.8%*
Motion History Image [5]	49.5%
Laptev's feature 1 (HOG)	55.6%
Laptev's feature 2 (HOF)	78.7%
Bayesian Net	79.1%
Random forest	59.5%
HFM	**88.1**%

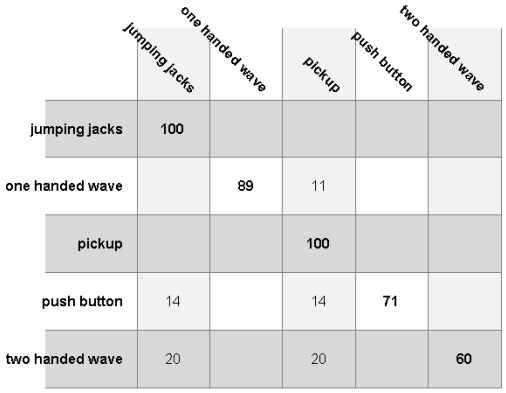

Figure 4. Confusion matrix of the action recognition results by HFM on the CMU event dataset.

Table 5 shows the recognition results using different types of features. When we use single type features, the accuracy is relatively modest and the highest is 78.7%. When using multiple features, the Bayesian net produces a better accuracy of 79.1%. It is interesting to note that that although random forest performs better than Bayesian net for image recognition, it does worse for action recognition. However, our HFM is able to achieve a recognition accuracy as high as 88.1%. Close examination of the confusion matrix shown in Fig. 4 reveals that our model achieved perfect recognition for jumping and pickup actions. The most challenging event for our model is the two-hand wave, which were confused with jumping and pickup, especially when the human body is partially occluded.

7. Conclusions

In this paper, we propose a novel model called Heterogeneous Feature Machine, which generalizes beyond the classic Kernel Machines such that multiple types of visual

features can be effectively integrated for visual recognition tasks. Our model is built on a logistic regression loss function and a group LASSO regularization term. This novel formulation makes it possible to integrate diverse yet complementary features with different metrics while removing the noise of irrelevant samples. We employ the framework of block Co-ordinate Gradient Descent (CGD) to find the optimal coefficients for HFM, and improve the optimization speed by efficiently computing the likelihood and its gradients. As a result, the proposed model can achieve better performance than state-of-the-art MKL algorithms [30] [28] within comparable computation time.

Acknowledgement

We would like to thank Lukas Meier, Sangwoon Yun and anonymous reviewers for their constructive suggestions and comments. Lianglaing Cao would like to thank Kodak for supporting his PhD study. This work was also supported in part by U.S. Government VACE Program and the National Science Foundation DMS-0732276.

References

[1] F. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

[2] F. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. *International Conference on Machine Learning*, 2004.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. *Advances in Neural Information Processing Systems*, 2000.

[4] C. Blake, and C. Merz. UCI repository of machine learning databases. University of California Irvine, 1998.

[5] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2001.

[6] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. *ACM International Conference on Image and Video Retrieval*, 2007.

[7] A. Bosch, A. Zisserman, and X. Munoz. Image Classification Using ROIs and Multiple Kernel Learning. *submitted to International Journal of Computer Vison*, 2008.

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] E. Candes and M. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, 2008.

[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2:886–893, 2005.

[11] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. *IEEE International Conference on Computer Vision*, pages 726–733, 2003.

[12] E. Frank, M. Hall, L. Trigg, G. Holmes, and I. Witten. Data mining in bioinformatics using Weka, 2004.

[13] M Gnen, and E Alpaydin. Localized multiple kernel learning. *International Conference on Machine Learning*, 2008

[14] V. Jain, A. Singhal, and J. Luo. Selective hidden random fields: Exploiting domain specific saliency for event classification. *IEEE Conference on Computer Vision and Pattern Recognition*, 1, 2008.

[15] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. *IEEE International Conference on Computer Vision*, 2007.

[16] C. Kelley. *Iterative Methods for Optimization*. Society for Industrial Mathematics, 1999.

[17] G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.

[18] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, 5:27–72, 2004.

[19] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[20] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[21] L.-J. Li and L. Fei-Fei. What, where and who? classifying event by scene and object recognition. *IEEE International Conference on Computer Vision*, 2007.

[22] D. Lowe. Object recognition from local scale-invariant features. *Proc. Int'l Conf. Computer Vision*, 1999.

[23] J. Mairal, *et al.* Discriminative sparse image models for class-specific edge detection and image interpretation. *European Conference on Computer Vision*, 2008.

[24] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[25] L. Meier, S. van de Geer, and P. Buhlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, 70(1):53–71, 2008.

[26] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 971–987, 2002.

[27] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[28] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[29] V. Roth. The generalized LASSO. *IEEE Transactions on Neural Networks*, 15(1), 2004.

[30] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *The Journal of Machine Learning Research*, 7:1531–1565, 2006.

[31] P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming B*, 117(1-2), 2009.

[32] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. *IEEE International Conference on Computer Vision*, 2007.

[33] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.

[34] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear Spatial Pyramid Matching using Sparse Coding for Image Classification. *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[35] H. Yu, M. Li, H. Zhang, and J. Feng. Color texture moments for content-based image retrieval. *International Conference on Image Processing*, 3:929–932, 2002.

[36] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B*, 68(1):49–67, 2006.

[37] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm Support Vector Machines. *Advances in Neural Information Processing Systems*, 2004.