Class Consistent Multi-Modal Fusion with Binary Features

Ashish Shrivastava Mohammad Rastegari Sumit Shekhar Rama Chellappa
Larry S. Davis
University of Maryland, College Park
{ashish, mrastega, sshekha, rama, lsd}@umiacs.umd.edu

Abstract

Many existing recognition algorithms combine different modalities based on training accuracy but do not consider the possibility of noise at test time. We describe an algorithm that perturbs test features so that all modalities predict the same class. We enforce this perturbation to be as small as possible via a quadratic program (QP) for continuous features, and a mixed integer program (MIP) for binary features. To efficiently solve the MIP, we provide a greedy algorithm and empirically show that its solution is very close to that of a state-of-the-art MIP solver. We evaluate our algorithm on several datasets and show that the method outperforms existing approaches.

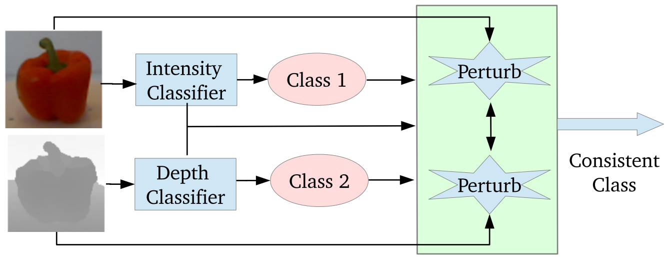

Figure 1. Overview of the proposed Class Consistent Multi-Modal (CCMM) fusion. The proposed algorithm perturbs the input features until all the modalities predict a consistent class.

1. Introduction

Combining information from multiple sources - multiple sensor modalities or multiple feature channels applied to a single sensor modality - is generally advantageous for recognition problems. For example, a self-driving car can better navigate its environment using multiple sensors including color cameras, depth sensors, inertial sensors, etc. Using both color and depth cameras, instead of either, can significantly improve performance of computer vision tasks such as object categorization, detection, tracking, segmentation and others [22, 9, 25]. In biometrics, fingerprints from multiple fingers can be used, or fingerprint and iris can be combined to determine identity. In this paper, we consider classification by fusing information from multiple modalities and present an algorithm for multi-modal fusion by enforcing the intuitive constraint that the predicted class label should be consistent across all modalities.

Fusing multiple modalities for classification has been explored in many computer vision applications. These approaches can broadly be divided into three categories: (1) feature level fusion, (2) score level fusion, and (3) decision level fusion. In feature level fusion, features from multiple modalities are combined before feeding them to the decision unit or a classifier, e.g., a support vector machine (SVM) [2]. A straight-forward way of combining the features is to concatenate them, which has been used in biometrics [33, 41, 36], object recognition [21], scene classification [34] etc. Feature concatenation preserves the raw information so that the classifier can utilize the correlation among modalities. However, these features are often very high dimensional and hence simple concatenation can be inefficient. Another approach to feature level fusion is multiple kernel learning (MKL), which learns a linear combination of multiple kernels. Finding appropriate feature combinations entails designing good kernel functions among a set of candidate kernels. MKL is a powerful way of determining the mixing weights of multiple kernels [15, 29, 38, 39]. For multi-modal fusion, each modality can be used to form a kernel matrix; an optimal linear combination of kernel matrices translates into optimal feature level fusion. When using only a linear kernel for each modality, the MKL methods are similar to feature concatenation, except that the features from each modality are weighted based on training accuracy. However, in order to make a good decision at test time, it is important that we also determine the quality of the test features from each modality. For example, from a training database of depth and intensity images, we might conclude that both modalities are equally useful for classification; however, at test time, the depth image may be noisy due to specularity on an object's surface. In such situations, it is useful to make the prediction based on score level or decision level fusion and not rely entirely on training accuracy. Score level fusion can be achieved by averaging the scores of decision functions and decision level fusion can be performed by taking a majority vote from all the modalities. The predicted scores from multiple modalities can also be combined using late fusion methods [23, 40]. Recently, [37] proposed a sparse representation-based multimodal biometric fusion method, which represents the test data by a sparse linear combination of training data, while constraining the observations from different modalities of the test subject to share their sparse representations. Effectively, they regularize the joint sparse coefficient matrix with the $\ell_{\{1,2\}}$ norm, which enforces the test feature to be reconstructed from the training features of the same class. This method is the closest to our approach in that it implicitly enforces that different modalities share a common class at test time. However, [37] does not learn a classification model, and training data for each class needs to be "paired" for each modality. That is, the number of samples in each modality must be the same to enforce the $\ell_{\{1,2\}}$ constraint on the joint sparse coefficient matrix. Furthermore, enforcing row sparsity on the joint sparse coefficient matrix of a test sample makes the method susceptible to the ordering of the training samples within each class. Also, sparse methods are generally slow for large training matrices.

Most algorithms for feature fusion have been developed for continuous features. Recently, with the the availability of large datasets, the need for efficient algorithms that can work with big data has increased. One way to efficiently process large numbers of features is to represent them as binary features. Binary codes are attractive representations of data for similarity based search and retrieval purposes, due to their storage efficiency and computational efficacy [20, 30, 16, 32]. For example, 250 million images can be represented by 64 bit binary codes requiring only 16 GB of memory. Hashing is a common method to convert high dimensional features to binary codes whose Hamming distances preserve the original feature space distances. Although shorter codes are more desirable due to direct implementation in hash tables, longer binary descriptors of data have been shown to be efficient for fast similarity search tasks. For example, [26] proposed a multi-index hashing method, and [31] introduced a branch and bound approach to perform exact k-nearest neighbors search in sub-linear time with long binary codes. To the best of our knowledge, ours is the first work proposing multi-modal fusion using binary codes. We propose to modify the test features so that all the modalities agree on a common class label. We call this approach *class consistent multi-modal* (CCMM) fusion. The key idea, summarized in Fig. 1, is to minimize the magnitude of perturbations to feature values for each modality to reach a point where all the modalities are predicting a common class label. We develop this intuition into an optimization problem that can be solved via quadratic programming for continuous features, and mixed integer programming for binary features. We evaluate this algorithm on several state-of-the-art datasets and results show that the method outperforms many previous algorithms. Although the idea of perturbation has been used previously (e.g. [35, 27]), we believe it has not been utilized for fusing multiple modalities. The contributions of this paper are as follows:

- We enforce class consistency across all available modalities in a perturbation model to determine the class of multi-modal data item.
- Based on this notion of class consistency, we develop an efficient binary feature fusion algorithm.

2. Class Consistent Multi-Modal Fusion (CCMM)

Our method relies on the intuition that when multiple modalities are available, each of them should predict the same class. When there is disagreement in the prediction of a test sample, we employ a scheme that enforces consistency of the predicted class across modalities. We achieve this consistency by perturbing the test sample in each modality so that their predictions are consistent. This is formally posed as an optimization problem which minimizes the perturbation to satisfy the constraint that all modalities predict the same class label. In what follows, we establish our notation and develop the algorithm, first for continuous features and then, for binary features.

Assume that there are M modalities each with N_m labeled samples where $m = 1, \ldots, M$. Let the data matrix of the m^{th} modality be denoted by $\mathbf{Y}^{(m)} \in \mathbb{R}^{d \times N_m}$, where each column of $\mathbf{Y}^{(m)}$ is a d-dimensional data sample denoted by $\mathbf{y}_i^{(m)} \in \mathbb{R}^d$, for $i = 1, \ldots, N_m$. Let the class label of the i^{th} sample in the m^{th} modality be denoted by $l_i^{(m)} \in \{1, \ldots, C\}$, where C is the number of classes. Note that, for now, we regard the features as continuous; subsequently we adapt our method for binary features.

Let $\mathbf{W}^{(m)} := \begin{bmatrix} \mathbf{w}_1^{(m)} \\ \vdots \\ \mathbf{w}_C^{(m)} \end{bmatrix}$, be the classifier matrix for all categories in modality m, where the c^{th} row vector $\mathbf{w}_c^{(m)} \in \mathbb{R}^{1 \times d}$ denotes the parameters of a linear classifier for the c^{th} class, which we refer to as a classification weight vector. These weight vectors are learned in a way that the class of a test sample $\mathbf{y}_p^{(m)}$ can be computed as,

$$\text{class of } \mathbf{y}_p^{(m)} = \arg\max_c \mathbf{w}_c^{(m)} \mathbf{y}_p^{(m)}. \tag{1}$$

In our implementation we use an SVM ([10], [13]) to learn these classification weight matrices $\mathbf{W}^{(m)}$ for all modalities.

First, we describe the method for two modalities and then extend it to multiple modalities. Denote a given test sample's two modalities by $\mathbf{y}_p^{(1)}$ and $\mathbf{y}_p^{(2)}$ which by construction belong to the same class. Our goal is to minimize the total perturbation needed to reach the condition that the predicted classes using SVM matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ are identical. This is captured in the following optimization problem,

$$\begin{aligned} &\min_{\mathbf{y}^{(1)},\mathbf{y}^{(2)}} \|\mathbf{y}^{(1)} - \mathbf{y}_p^{(1)}\|_2 + \|\mathbf{y}^{(2)} - \mathbf{y}_p^{(2)}\|_2 \\ \text{subject to,} \quad &\arg\max_c \mathbf{w}_c^{(1)}\mathbf{y}^{(1)} = \arg\max_c \mathbf{w}_c^{(2)}\mathbf{y}^{(2)} \end{aligned} \tag{2}$$

The optimization problem in (2) is non-smooth and non-convex due to the $\arg\max$ functions. In order to solve it efficiently, we approximate it with a tractable convex problem. To achieve this, we employ an alternating optimization approach. First, we assume that the class predicted by the second modality is correct and optimize for $\mathbf{y}^{(1)}$, and then, we fix the class to the one predicted by the first modality and optimize for $\mathbf{y}^{(2)}$. When optimizing for the m^{th} modality feature $\mathbf{y}^{(m)}$, the class that is assumed to be correct is called the target class and is denoted by t_m , i.e.,

$$t_1 := \arg\max_c \mathbf{w}_c^{(2)}\mathbf{y}_p^{(2)}, \tag{3}$$

and,

$$t_2 := \arg\max_c \mathbf{w}_c^{(1)}\mathbf{y}_p^{(1)}. \tag{4}$$

We seek to perturb the feature $\mathbf{y}_p^{(m)}$ so that its predicted class is t_m, which can be achieved by solving the following problem:

$$\begin{aligned} &\min_{\mathbf{y}^{(m)}} \|\mathbf{y}^{(m)} - \mathbf{y}_p^{(m)}\|_2 \\ \text{subject to,} \quad &\arg\max_c \mathbf{w}_c^{(m)}\mathbf{y}^{(m)} = t_m. \end{aligned} \tag{5}$$

As explained later, the optimization problem in (5) is a quadratic program (QP). Let the solution of (5) be denoted by $\tilde{\mathbf{y}}_t^{(m)}$. Finally, the consistent class, denoted by l_p, across both modalities is the target class of the modality that requires the smallest change with respect to the original feature norm, i.e.,

$$l_p = t_{m^*}, \tag{6}$$

where,

$$m^* = \arg\min_m \frac{\|\tilde{\mathbf{y}}_p^{(m)} - \mathbf{y}_p^{(m)}\|_2}{\|\mathbf{y}_p^{(m)}\|_2}. \tag{7}$$

Next, we describe how the optimization problem in (5) can be written as a quadratic convex program. The constraints in (5) can be re-written as,

$$\begin{aligned} \arg\max_c \quad & \mathbf{w}_c^{(m)}\mathbf{y}^{(m)} = t_m \\ & \Rightarrow \mathbf{w}_{t_m}^{(m)}\mathbf{y}^{(m)} \geq \mathbf{w}_i^{(m)}\mathbf{y}^{(m)}, \qquad \forall i \neq t_m \\ & \Rightarrow \mathbf{w}_i^{(m)}\mathbf{y}^{(m)} - \mathbf{w}_{t_m}^{(m)}\mathbf{y}^{(m)} \leq \mathbf{0}, \qquad \forall i \neq t_m \\ & \Rightarrow \mathbf{A}_{t_m}\mathbf{y}^{(m)} \leq \mathbf{0}, \end{aligned} \tag{8}$$

where, $\mathbf{A}_{t_m}\mathbb{R}^{C-1\times d}$ is a constraint matrix whose rows are computed as, $[\mathbf{A}_{t_m}]_{i,:} = \mathbf{w}_i^{(m)} - \mathbf{w}_{t_m}$. Hence, the problem in (5) can be optimized by solving the following QP program,

$$\begin{aligned} &\min_{\mathbf{y}^{(m)}} \|\mathbf{y}^{(m)} - \mathbf{y}_p^{(m)}\|_2 \\ \text{subject to,} \quad &\mathbf{A}_{t_m}\mathbf{y}^{(m)} \leq \mathbf{0}. \end{aligned} \tag{9}$$

2.1. CCMM for binary features

As stated earlier, binary features are very useful for large scale classification because they require smaller storage space and are efficient for classification. However, the optimization problem in (9) has been designed for continuous features. For binary features, if we predict a consistent class by solving this problem, we may not achieve good performance because the solution will not lie in a binary space. In face recognition, for example, a binary feature may represent an image attribute like sunglasses, which could be either present or not present in the image. Hence, we optimize for the binary features over a binary space.

In this sub-section, we assume the features $\mathbf{b}_i^{(m)} \in \{0,1\}^d$ are d-dimensional binary vectors. Furthermore, data matrices $\mathbf{B}^{(m)} = [\mathbf{b}_1^{(m)}, \ldots, \mathbf{b}_{N_m}^{(m)}] \in \{0,1\}^{d\times N_m}$ are of size $d \times N_m$ with binary elements. As with continuous features, we learn SVM weight matrices for each modality. The major difference in setting up our optimization problem is that, in the case of binary features, we want the solution of the optimization problem to lie in a binary space. We reformulate (9) for binary features as:

$$\begin{aligned} &\min_{\mathbf{b}^{(m)}} \|\mathbf{b}^{(m)} - \mathbf{b}_p^{(m)}\|_1 \\ &\text{subject to,} \\ &\qquad \mathbf{A}_{t_m}\mathbf{b}^{(m)} \leq \mathbf{0} \\ &\qquad \mathbf{b}^{(m)} \in \{0,1\}^d. \end{aligned} \tag{10}$$

Note that for binary features, we minimize the ℓ_1 instead of the ℓ_2 norm because the former counts the number of places in the binary vector where the solution differs from the input feature. In other words, we minimize the Hamming distance between the input feature and the solution. Minimizing the ℓ_1 norm is a non-smooth function; hence, we use an auxiliary variable $\mathbf{z}$ to make the cost differentiable,

$$\min_{\mathbf{b}^{(m)},\mathbf{z}} \mathbf{1}^T\mathbf{z}$$

subject to,

$$\begin{aligned} &\mathbf{A}_{t_m}\mathbf{b}^{(m)} \le \mathbf{0} \\ &\mathbf{z} = |\mathbf{b}^{(m)} - \mathbf{b}_p^{(m)}|, \\ &\mathbf{b}^{(m)} \in \{0,1\}^d, \qquad \mathbf{z} \in \{0,1\}^d, \end{aligned} \tag{11}$$

where $\mathbf{1} \in \mathbb{R}^d$ is a vector of 1's, $(.)^T$ denotes the matrix transpose, and $|.|$ denotes the element-wise absolute value. The ℓ_1 constraints involving $\mathbf{z}$ are difficult to optimize. In order to eliminate them, we replace them with a set of linear constraints as follows. Let the i^{th} element of vectors $\mathbf{z}$, $\mathbf{b}^{(m)}$, and $\mathbf{b}_p^{(m)}$ be denoted by z_i, $b_i^{(m)}$ and $b_{pi}^{(m)}$, respectively. Next, $\mathbf{z} = |\mathbf{b}^{(m)} - \mathbf{b}_p^{(m)}|$ can be replaced by the following linear constraints,

$$z_i \ge (b_{pi}^{(m)} - b_i) \tag{12}$$

$$z_i \ge -(b_{pi}^{(m)} - b_i). \tag{13}$$

Now the optimization problem in (11) can be modified as,

$$\min_{\mathbf{b}^{(m)},\mathbf{z}} \sum_{i=1}^{d} z_i$$

subject to,

$$\begin{aligned} &\mathbf{A}_{t_m}\mathbf{b}^{(m)} \le \mathbf{0} \\ &\begin{bmatrix} -\mathbf{I}_d & +\mathbf{I}_d \\ -\mathbf{I}_d & -\mathbf{I}_d \end{bmatrix} \begin{bmatrix} \mathbf{z} \\ \mathbf{b}^{(m)} \end{bmatrix} \le \begin{bmatrix} \mathbf{b}_p^{(m)} \\ -\mathbf{b}_p^{(m)} \end{bmatrix} \\ &\mathbf{b}^{(m)} \in \{0,1\}^d, \qquad \mathbf{z} \in \{0,1\}^d. \end{aligned} \tag{14}$$

The optimization in (14) is a linear programming problem in $\mathbf{b}^{(m)}, \mathbf{z}$ except for the fact that the solution space is binary. Although this problem can be solved with a mixed integer programming (MIP) solver, we propose an efficient greedy algorithm and, later, empirically demonstrate that the solution to the greedy algorithm is close to that of the MIP solver. In order to solve the problem in (14) or (10) greedily, we first find a feasible solution, which can simply be one of the training samples from the target class satisfying the constraints of problem (10). Now, starting from this feasible solution, we move towards the test sample as much as possible without leaving the feasible region. Let the initial feasible solution be denoted by $\mathbf{b}_0$ and the running solution, which we will keep updating, be denoted by $\mathbf{b}$. First we initialize $\mathbf{b}$ to $\mathbf{b}_0$. Next, we find all the elements of $\mathbf{b}$ that are different from the test sample $\mathbf{b}_p^{(m)}$. Let this set of bit locations be denoted by $\mathcal{S}$, i.e.,

$$\mathcal{S} := \{i \mid b_i \ne b_{pi}^{(m)}\},$$

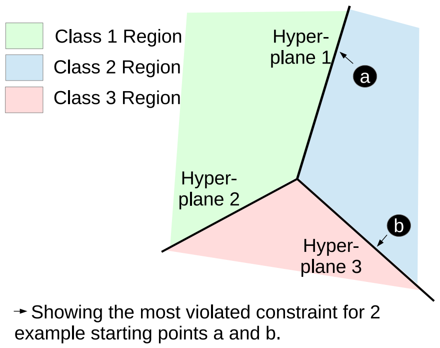

Figure 2. Example of most violated constraint. Both solutions a and b lie in target class 2. Since a is likely to move to class 1, hyperplane 1 corresponds to the most violated constraint. Similarly, if the current solution is b, hyperplane 3 corresponds to the most violated constraint.

where b_i is the i^{th} bit of vector $\mathbf{b}$ and $b_{pi}^{(m)}$ is the i^{th} bit of $\mathbf{b}_p^{(m)}$. Our goal is to change as many bits from this set $\mathcal{S}$ as possible because every change takes $\mathbf{b}$ one step closer to the test feature $\mathbf{b}_p^{(m)}$. Choosing the optimal subset of bits is an NP-hard problem and, hence, we resort to an approximate greedy method. Next, we present this greedy algorithm that changes one bit at a time from this set $\mathcal{S}$. The solution of the greedy algorithm can further be improved by various MIP solvers; however, empirically we observe that the greedy solution is quite good. In order to select a bit from $\mathcal{S}$ we flip all the bits in $\mathcal{S}$ and compute the following score,

$$s_i = \min_{c \ne t}(\mathbf{w}_t^{(m)}\mathbf{b} - \mathbf{w}_c^{(m)}\mathbf{b}_{\bar{i}}), \tag{15}$$

where $\mathbf{b}_{\bar{i}}$ is $\mathbf{b}$ with its i^{th} bit flipped. Note that, since we start from a feasible solution, s_i is bounded below by 0. Recall that $\mathbf{w}_c^{(m)}$ is the SVM weight vector for the c^{th} class of the m^{th} modality. The score s_i is the difference of scores between the target class and its closest one if the i^{th} bit is flipped. In other words, s_i corresponds to the constraint closest to the current solution $\mathbf{b}$ and is most likely to be violated if b's i^{th} bit is changed. This is illustrated in Fig. 2. Now, our goal is to change that bit which will keep the current solution in the feasible region as much as possible. Hence, we change that bit of $\mathbf{b}$ that belongs to set $\mathcal{S}$ and for which the solution remains as feasible as possible. Feasibility is measured by the maximum distance of $\mathbf{b}$ from all the constraints. We denote the index of the bit to be flipped by j, and it is computed as,

$$j = \arg\max_i s_i. \tag{16}$$

All the steps of the greedy algorithm are summarized in Algorithm 1. Having computed the solution to the optimization problem in (14), we can compute the consistent class l_p from both modalities based on the perturbation as follows:

$$l_p = t_{m^*}, \tag{17}$$

Algorithm 1: Greedy algorithm to optimize for binary feature

```
Input: Binary test data $\mathbf{b}_p^{(m)}$, Classification matrices
    $\mathbf{W}^{(m)}$, a feasible solution $\mathbf{b}_0$, target class $t$
Output: $\mathbf{b}$
Initialize $\mathbf{b} = \mathbf{b}_0$, $v = 1$.
$\mathcal{S} \leftarrow \{i \mid b_i \neq b_{pi}^{(m)}\}$
while ($v > 0$) AND ($\mathcal{S}$ is not empty) do
    1. $\mathbf{b}_{\bar{i}} \leftarrow \mathbf{b}$ with $i^{th}$ bit flipped.
    2. $v \leftarrow \max_{i \in \mathcal{S}} \min_{c \neq t} (\mathbf{w}_t^{(m)} \mathbf{b} - \mathbf{w}_c^{(m)} \mathbf{b}_{\bar{i}})$
    if $v > 0$ then
        $j \leftarrow \arg\max_{i \in \mathcal{S}} \min_{c \neq t} (\mathbf{w}_t^{(m)} \mathbf{b} - \mathbf{w}_c^{(m)} \mathbf{b}_{\bar{i}})$
        $b_j \leftarrow 1 - b_j$
    end
    3. $\mathcal{S} \leftarrow \{i \mid b_i \neq b_{pi}^{(m)}\}$
end
return $\mathbf{b}$
```

where,

$$m^* = \arg\min_m \frac{\|\tilde{\mathbf{b}}_p^{(m)} - \mathbf{b}_p^{(m)}\|_1}{\|\mathbf{b}_p^{(m)}\|_1}. \tag{18}$$

However, we find in our experiments that weighting the perturbations based on the quality of the modalities, improves the performance. This is explained in the next sub-section. Finally, we summarize all the steps to compute the consistent class from two modalities in Algorithm 2.

Algorithm 2: Summary of CCMM for binary features

```
Input: Binary test data $\mathbf{b}_p^{(m)}$, Classification matrices
    $\mathbf{W}^{(m)}$, for $m = 1, 2$
Output: Class consistent label $l_p$
1. Predict class labels $l_p^{(m)} = \arg\max_c w_c^{(m)} b_p^{(m)}$ for
individual modalities.
2. Compute target labels as $t_1 = l_p^{(2)}, t_2 = l_p^{(1)}$.
3. Compute $\tilde{b}_p^{(m)}$ by solving optimization problem in
(14) using greedy Algorithm 1.
4. Predict class consistent label $l_p$ using (17) or (23).
return $l_p$
```

2.2. Extension to Multiple Modalities

So far we have described how to predict a consistent class with two modalities. For multiple modalities we construct a set of target labels based on the classifiers for the individual modalities. This set, denoted by $\mathcal{Z}$, consists of the labels predicted by all the modalities. Let the number of modalities be denoted by M. We compute the score s_{mc} of the mth modality based on the perturbation needed to predict the cth target class as follows:

$$s_{mc} = \frac{\exp(-\epsilon_{mc}/\sigma)}{\sum_{c \in \mathcal{Z}} \exp(-\epsilon_{mc}/\sigma)}, \tag{19}$$

where,

$$\epsilon_{mc} = \frac{\|\tilde{\mathbf{b}}_p^{(m)} - \mathbf{b}_p^{(m)}\|_1}{\|\mathbf{b}_p^{(m_z)}\|_1},$$

and σ is a parameter that controls the sharpness of the distribution of the scores over classes. For each class, we compute the combined score as a weighted sum of each modality,

$$s_c = \sum_{m=1}^{M} \eta_m * s_{mc}, \tag{20}$$

where, η_m is the quality of the mth modality based on the training data. Although, we could have given the same weight to each modality, i.e. set $\eta_m = 1, \forall m$, setting these weights based on training data slightly improves performance. We compute η_m based on the kernel alignment criterion [12] which has been shown to generalize well to unseen test data and has been used for combining multiple kernels [15]. η_m is computed based on similarity between the linear kernel matrix of the mth modality, i.e. $\mathbf{B}^{(m)T}\mathbf{B}^{(m)}$, and the ideal kernel matrix $\boldsymbol{\mathcal{K}d}_m \in \mathbb{R}^{N_m \times N_m}$, defined as,

$$\boldsymbol{\mathcal{K}d}_m(i,j) = \begin{cases} 1, & \text{if} \quad l_i^{(m)} = l_j^{(m)}, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

The score η_m is computed as,

$$\eta_m = \frac{\langle \boldsymbol{\mathcal{K}}_m, \boldsymbol{\mathcal{K}d}_m \rangle}{\sqrt{\langle \boldsymbol{\mathcal{K}}_m, \boldsymbol{\mathcal{K}}_m \rangle \langle \boldsymbol{\mathcal{K}d}_m, \boldsymbol{\mathcal{K}d}_m \rangle}}, \tag{22}$$

where, $\boldsymbol{\mathcal{K}}_m := \mathbf{B}^{(m)T}\mathbf{B}^{(m)}$, and $\langle ., . \rangle$ denotes the dot product between the argument matrices. Finally, the class l_p of the test input is predicted as the one with the maximum score,

$$l_p = \arg\max_c s_c. \tag{23}$$

3. Experiments

We evaluate the method on publicly available computer vision datasets. First, we use the fusion algorithm to combine image and depth data for object categorization. Next, we apply the method to fuse multiple modalitites for biometrics applications and demonstrate significant improvement over previous methods. Finally, we combine intensity and semantic features on the Pascal-Sentence dataset. We compare the method to state-of-the-art multimodal fusion methods such as recently proposed sparse multimodal biometric recognition (SMBR) [37], sparse logistic regression (SLR) [19], support vector machine (SVM) [7] , and

	CCMM	SMBR	SLR-Sum	SLR-Major	SVM-Sum	SVM-Major	MKL Fusion
Intensity Features	70.1	64.7	64.3	64.3	70.1	70.1	68.4
Depth Features	64.9	61.1	63.6	63.6	64.9	64.9	61.8
Combined Features	**83.7**	73.5	73.4	71.9	79.1	74.1	77.1

Table 1. Classification Accuracy for RGB-D data

multiple kernel learning (MKL) [29] algorithms. As SLR and SVM methods cannot handle multiple modalities, [37] explored score-level and decision level fusion to combine modalities. Score level fusion was achieved by adding probability outputs of all the modalities to obtain a final score vector. Classification was performed by choosing the class corresponding to the maximum score. For decision level fusion, the class predicted by the maximum number of modalities was chosen. The score level fusion using SLR is called SLR-Sum and the decision level fusion is called as SLR-Major; for SVM, they are called SVM-Sum and SVM-Major, respectively. The parameter σ is set to 0.01 for all our experiments. We implemented our algorithm in MATLAB and used the Gurobi optimizer [17] for solving MIP and QP problems, for binary features and continuous features, respectively. We observed that the performance of the Gurobi optimizer was similar to the proposed greedy algorithm for biometrics application, and slightly better for object categorization. We plot the normalized difference between the solutions of the Gurobi and the greedy algorithm in Fig. 3, for fusing two iris features. As can be seen from this figure, the greedy algorithm's solution is close to that of the Gurobi for most of the test samples. Hence, we report results using only the greedy algorithm to solve the MIP for biometrics application, and report results using the Gurobi optimizer initialized with the greedy solution for object categorization. Furthermore, for classification or rank-one recognition, we include the top 5 classes into target class set. In the following subsections, we describe each of the datasets and present our results.

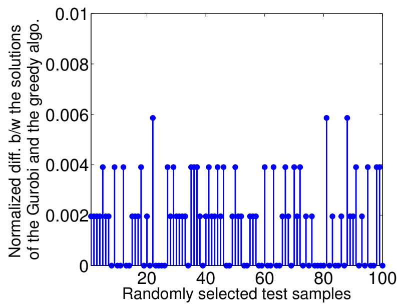

Figure 3. The performance of the greedy algorithm compared to the Gurobi MIP solver. For most of the test samples, the difference is less than 0.004 which corresponds to approximately 2 bits. That is, loosely speaking, the greedy algorithm's solution is, on an average, within 2 bits of the sophisticated MIP solver.

3.1. RGB-D data

Recently, there has been a growing interest in using both intensity and depth data for computer vision algorithms. For example, with Microsoft's Kinect camera one can capture videos of both color as well as corresponding depth data. The purpose of this experiment is to evaluate CCMM on binary features computed using color and depth data. We use the RGB-D dataset from the University of Washington [22] which consists of 51 object categories. A few examples of pairs of color and depth images from this dataset are shown in Fig. 4. Most of the depth images are noisy. Hence, we apply a recursive median filter to fill in missing values. Processed images are shown in the third row of Fig. 4.

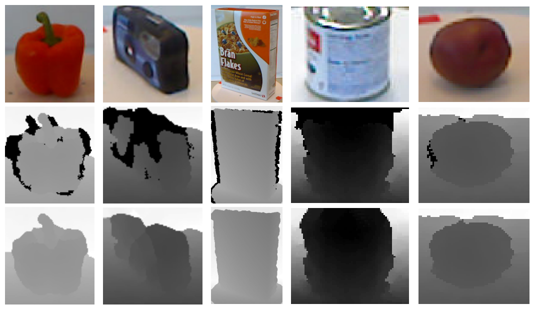

Figure 4. Example images of the RGBD dataset. First row shows the color images, second row displays the corresponding depth images, and the third row is the denoised version of the second row after applying the recursive median filter.

We test CCMM on the subset of the dataset by randomly selecting 15 images for training from each category. For the intensity images, we compute gradient based kernel descriptors [3] on 16×16 patches over a dense regular grid with spacing of 8 pixels. With these features, we compute a dictionary of 1000 words using k-means. Using this dictionary of visual words, we employed efficient match kernels and used 1×1, 2×2, and 4×4 pyramid sub-regions [5] to compute image level features. For the depth features, we compute the shape features over point clouds as described in [4] and gradient kernel descriptor features on the depth image. Similar to intensity features, image level depth features are computed using efficient match kernels over 1×1, 2×2, and 4×4 pyramid sub-regions using a dictionary of 1000 words. Finally, image level intensity and depth features are converted into binary features using the method proposed in [32]. We evaluate CCMM on individual modalities as well as their combination in Table 1. From the first two rows of the table, we note that SLR and SMBR methods have lower accuracy than SVM (CCMM is, of course, equivalent to SVM for the single modality case since we employ SVM as the per modality classifier). The reason for this is that these methods do not learn any classification model and rely on a sparse linear combination of training features to represent the test feature. Furthermore, none of the methods take into consideration that the input features are binary. By treating binary features in an appropriate way, CCMM is able to significantly improve the performance using a fusion of depth and intensity features, as seen in the last row of Table 1.

3.2. WVU dataset

The WVU biometrics dataset [11] consists of multiple biometrics such as fingerprints, iris, palmprint, hand geometry and voice samples from different subjects. Following the standard setting proposed in [37], we chose iris and fingerprint for testing the method. Furthermore, the evaluation was done on the subset of 219 subjects having samples in both modalities. Some challenging examples of fingerprints and iris images are shown in Fig. 6. We used the same Gabor features as in [37] for fingerprints and iris images. Before computing these features a robust pre-processing was applied to the images. Iris images were first segmented using the method proposed in [28], and then, a 25×240 iris template was created using the publicly available code of Masek et al [24]. Fingerprint images were first enhanced using filtering methods, then a core point was detected using algorithms from [18]. Finally, Gabor features were computed around the detected core point. Furthermore, to evaluate our algorithm on binary features, we computed the binary features of size 512 for each of the modalities using the method from [32]. Table 2 shows the accuracy of individual modalities. As can be seen, the performance of all the methods is comparable when only a single modality is considered. Next, we evaluate CCMM on various combinations of modalities. Following standard settings in [37], we compare the method on three combinations : (1) All the fingerprints (2) both iris images (3) all modalities. We present the comparison of different multi-modal fusion algorithms in Table 3, which shows that CCMM outperforms the competing algorithms despite the fact that, for individual modalities, the performance of CCMM is lower than SMBR. This demonstrates that forcing the class predictions of multiple modalities to be consistent is useful for multi-modal fusion.

We also compare cumulative match curves (CMC) of different methods with multiple modality combinations in Fig. 5. The CMC is a popular tool to analyze the performance of biometric systems [37], [8], [6]. To compute CMCs, the target label set $\mathcal{Z}$ is composed of all the class labels. As can be seen from this figure, the proposed method consistently outperforms the compared methods.

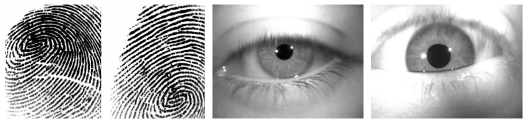
Figure 6. Example of challenging fingerprints and iris images from WVU dataset [11]. Many images in the dataset suffer from various artifacts such as blur, occlusion, noise etc.

3.3. CASIA Fingerprints dataset

The CASIA Fingerprint Image Database Version 5.0 (or CASIA-FingerprintV5) [1] contains a total of $20,000$ fingerprint images from 500 subjects. Each subject contributed a total of 40 images from 8 fingers, 4 from each hand. Each finger was scanned 5 times and the volunteers were asked to rotate their fingers with various levels of pressure to generate significant intra-class variations. In order to effectively compare all the algorithms, we took the subset of the first 50 subjects. Furthermore, we randomly selected 3 training images per modality for each subject, and kept the remaining 2 for testing. The results, presented in Table 4, are the average classification accuracies over 5 random trials. For each fingerprint, we compute the same features as for the WVU dataset. In this experiment, we evaluate the idea of class consistency with continuous features only. Furthermore, we compare CCMM on three combinations of modalities: (1) four fingerprints from the left hand, (2) four fingerprints from the right hand, and (3) all 8 fingerprints. As can be observed from Table 4, CCMM is comparable to SMBR when 4 fingers are fused, however it performs slightly better when fusing all 8 fingers.

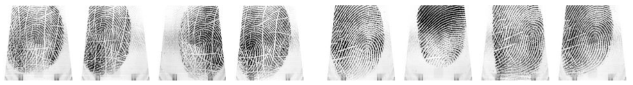
Figure 7. Example images of CASIA v5 dataset showing large intra-class variation. The first four images belong to one finger of subject 1 and the last four images are from subject 2.

3.4. Pascal-Sentence Dataset

This dataset has two modalities - images and sentences. The images in the dataset are collected from PASCAL VOC 2008, which is one of the most popular benchmark datasets for object recognition and detection. For each of the 20 categories of the PASCAL 2008 challenge, 50 images are randomly selected. Each image is annotated with 5 sentences using Amazon's Mechanical Turk. For our task we randomly picked just one sentence for each image. These sentences represent the semantics of the image.

Our image features, following [14], are collections of responses from a variety of detectors, image classifiers and scene classifiers. The details of the image features can be

	Finger 1	Finger 2	Finger 3	Finger 4	Iris 1	Iris 2
CCMM	67.8	86.9	69.4	89.3	60.5	61.2
SMBR	68.1	88.4	69.2	87.5	60.0	62.1
SLR	67.4	87.9	66.0	87.5	57.1	57.9

Table 2. Rank-one recognition of single modalities for WVU data

	CCMM	SMBR	SLR-Sum	SLR-Major	SVM-Sum	SVM-Major	MKL Fusion
4 Fingers	**98.8**	97.9	96.3	74.2	90.0	73.0	86.2
2 Irises	**82.9**	76.5	72.7	64.2	62.8	49.3	76.8
All Modalities	**99.6**	98.7	97.6	84.2	94.9	81.3	89.8

Table 3. Comparison of Rank-one recognition performance on WVU dataset for different combinations of modalities

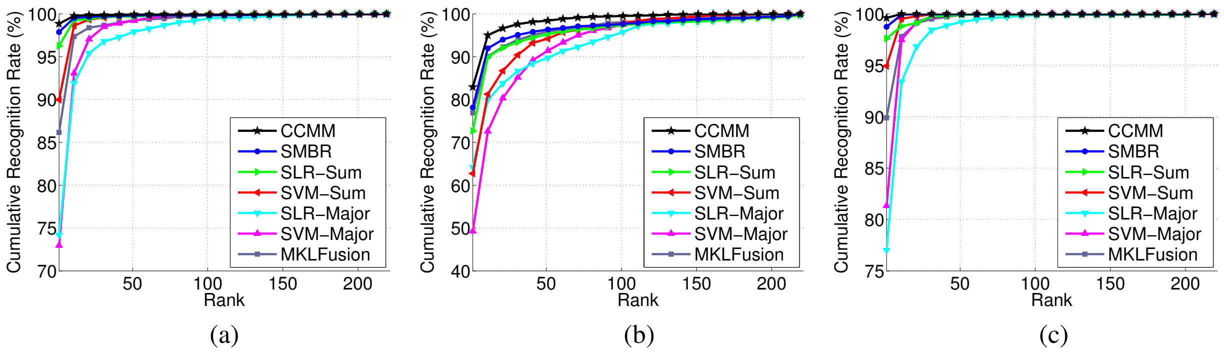

Figure 5. The comparison of CMCs for different modality combinations of WVU dataset. (a) Four fingers, (b) Two Irises, and (c) All the modalities (four fingers and two irises).

	CCMM	SMBR	SLR-Sum	SLR-Major	SVM-Sum	SVM-Major	MKL Fusion
Left Fingers	**92.4**	90.4	88.2	83.2	85.8	76.4	81.8
Right Fingers	91.8	**92.2**	90.0	84.6	82.2	74.6	78.8
All Fingers	**97.0**	96.2	95.4	87.8	92.6	83.6	91.2

Table 4. Comparison of rank-one recognition performance on multi-modal CASIA fingerprint data

	CCMM	SMBR	SLR-Sum	SLR-Major	SVM-Sum	SVM-Major	MKL Fusion
Intensity Features	66.2	66.2	65.4	65.4	66.2	66.2	67.2
Semantic Features	63.2	69.6	47.0	47.0	63.2	63.2	64.4
Combined Features	**77.2**	75.4	64.2	63.4	76.2	71.2	76.0

Table 5. Classification Accuracy for Pascal-Sentence dataset

found in [14]. The semantic features are constructed by using word-net semantic similarity with a dictionary of 1,200 words. These are followed by a quantization step that reduces the dimension to 20. The details of the text features are presented in [14]. Finally, the features are converted to binary codes using [32]. We present our result in Table 5, which shows that CCMM works better than the competing methods.

4. Conclusions

We described a multi-modal fusion algorithm- CCMM-based on class consistency and demonstrated that it performs significantly better than previous methods for binary features. The main idea is to perturb the input modalities until they predict the same class. Although CCMM used linear classification models, the class consistent prediction can be explored with other classification models.

Acknowledgement

This work was partially supported by an ONR grant N00014-12-1-0124.

References

[1] CASIA-FingerprintV5, http://biometrics.idealtest.org/. 7

[2] P. K. Atrey, M. A. Hossain, A. El-Saddik, and M. S. Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16(6):345–379, 2010. 1

[3] L. Bo, X. Ren, and D. Fox. Kernel Descriptors for Visual Recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2010. 6

[4] L. Bo, X. Ren, and D. Fox. Depth kernel descriptors for object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011. 6

[5] L. Bo and C. Sminchisescu. Efficient Match Kernel between Sets of Features for Visual Recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2009. 6

[6] R. Bolle, J. Connell, S. Pankanti, N. Ratha, and A. Senior. The relation between the ROC curve and the CMC. In *IEEE Workshop on Automatic Identification Advanced Technologies*, 2005. 7

[7] C. J. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, June 1998. 5

[8] K. Chang, K. Bowyer, and P. Flynn. An evaluation of multimodal 2d+3d face biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):619–624, April 2005. 7

[9] W. Choi, C. Pantofaru, and S. Savarese. Detecting and tracking people using an RGB-D camera via multiple detector fusion. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2011. 1

[10] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, Mar 2002. 2

[11] S. Crihalmeanu, A. Ross, S. Schuckers, and L. Hornak. A protocol for multibiometric data acquisition, storage and dissemination. In *Technical Report, WVU, Lane Department of Computer Science and Electrical Engineering*, 2007. 7

[12] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On kernel-target alignment. In *Advances in Neural Information Processing Systems (NIPS)*, 2002. 5

[13] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. 2

[14] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *European Conference on Computer Vision (ECCV)*, 2010. 7, 8

[15] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, July 2011. 1, 5

[16] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 2

[17] Gurobi Optimization Inc. Gurobi Optimizer Reference Manual, 2014. http://www.gurobi.com. 6

[18] A. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing*, 9(5):846–859, May 2000. 7

[19] B. Krishnapuram, L. Carin, M. Figueiredo, and A. Hartemink. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):957–968, June 2005. 5

[20] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1092–1104, June 2012. 2

[21] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 1

[22] K. Lai, L. Bo, X. Ren, and D. Fox. Sparse Distance Learning for Object Recognition Combining RGB and Depth Information. In *IEEE International Conference on on Robotics and Automation (ICRA)*, 2011. 1, 6

[23] D. Liu, K.-T. Lai, G. Ye, M.-S. Chen, and S.-F. Chang. Sample-specific late fusion for visual category recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[24] L. Masek and P. Kovesi. Matlab source code for biometric identification system based on iris patterns. *The University of Western Australia, Tech. Rep.*, 2003. 7

[25] A. Mishra, A. Shrivastava, and Y. Aloimonos. Segmenting "simple" objects using RGB-D. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. 1

[26] M. Norouzi, A. Punjani, and D. Fleet. Fast search in hamming space with multi-index hashing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[27] M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid. Transformation pursuit for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[28] S. Pundlik, D. Woodard, and S. Birchfield. Non-ideal iris segmentation using graph cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2008. 7

[29] A. Rakotomamonjy, U. Rouen, F. Bach, S. Canu, and Y. Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, 2008. 1, 6

[30] M. Rastegari, J. Choi, S. Fakhraei, H. Daume III, and L. S. Davis. Predictable dual-view hashing. In *IEEE International Conference on Machine Learning (ICML)*, 2013. 2

[31] M. Rastegari, C. Fang, and L. Torresani. Scalable object-class retrieval with approximate and top-k ranking. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 2

[32] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In *European Conference on Computer Vision (ECCV)*, 2012. 2, 6, 7, 8

[33] A. Rattani, D. Kisku, M. Bicego, and M. Tistarelli. Feature level fusion of face and fingerprint biometrics. In *IEEE International Conference on Biometrics: Theory, Applications, and Systems*, 2007. 1

[34] X. Ren, L. Bo, and D. Fox. RGB-(D) scene labeling: Features and algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1

[35] T. Riopka and T. Boult. Classification enhancement via biometric pattern perturbation. In *IAPR Conference on Audio- and Video-based Biometric Person Authentication*, Lecture Notes in Computer Science, 2005. 2

[36] A. Ross and R. Govindarajan. Feature level fusion using hand and face biometrics. *Proceedings of the SPIE*, 5779:196–204, Mar. 2005. 1

[37] S. Shekhar, V. Patel, N. Nasrabadi, and R. Chellappa. Joint sparse representation for robust multimodal biometrics recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):113–126, Jan 2014. 2, 5, 6, 7

[38] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *International Conference on Computer Vision (ICCV)*, 2007. 1

[39] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *International Conference on Computer Vision (ICCV)*, 2009. 1

[40] G. Ye, D. Liu, I.-H. Jhuo, and S.-F. Chang. Robust late fusion with rank minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[41] X. Zhou and B. Bhanu. Feature fusion of face and gait for human recognition at a distance in video. In *International Conference on Pattern Recognition (ICPR)*, 2006. 1