Discriminative Video Pattern Search for Efficient Action Detection

Junsong Yuan, *Member*, *IEEE*, Zicheng Liu, *Senior Member*, *IEEE*, and Ying Wu, *Senior Member*, *IEEE*

Abstract—Actions are spatiotemporal patterns. Similar to the sliding window-based object detection, action detection finds the reoccurrences of such spatiotemporal patterns through pattern matching, by handling cluttered and dynamic backgrounds and other types of action variations. We address two critical issues in pattern matching-based action detection: 1) the intrapattern variations in actions, and 2) the computational efficiency in performing action pattern search in cluttered scenes. First, we propose a discriminative pattern matching criterion for action classification, called naive Bayes mutual information maximization (NBMIM). Each action is characterized by a collection of spatiotemporal invariant features and we match it with an action class by measuring the mutual information between them. Based on this matching criterion, action detection is to localize a subvolume in the volumetric video space that has the maximum mutual information toward a specific action class. A novel spatiotemporal branch-and-bound (STBB) search algorithm is designed to efficiently find the optimal solution. Our proposed action detection method does not rely on the results of human detection, tracking, or background subtraction. It can handle action variations such as performing speed and style variations as well as scale changes well. It is also insensitive to dynamic and cluttered backgrounds and even to partial occlusions. The cross-data set experiments on action detection, including KTH, CMU action data sets, and another new MSR action data set, demonstrate the effectiveness and efficiency of the proposed multiclass multiple-instance action detection method.

Index Terms—Video pattern search, action detection, spatiotemporal branch-and-bound search.

1 Introduction

Detecting human actions in video sequences is an interesting yet challenging problem. It has a wide range of applications, including video surveillance, telemonitoring of patients and senior citizens, medical diagnosis and training, video indexing, and intelligent human-computer interaction, etc. Actions can be treated as spatiotemporal objects which are characterized as spatiotemporal volumetric data. Like the use of sliding windows in object detection, action detection can be formulated as locating spatiotemporal subvolumes in videos (i.e., video patterns) that contain the target actions. Despite previous successes of sliding window-based object detection [1], [2], this approach cannot be easily extended to action detection. It is still a challenging problem to detect and locate actions in video sequences, mainly due to the following two difficulties.

First, the computational complexity of pattern searching in the video space is much higher than that of object search in the image space. Without any prior knowledge about the location, temporal duration, and the spatial scale of the action, the search space for video patterns is prohibitive for exhaustive search. For example, a one-minute video sequence of size $160 \times 120 \times 1{,}800$ contains billions of valid spatiotemporal subvolumes of various sizes and locations. Therefore, although the state-of-the-art approaches of object detection can efficiently search the spatial image space [3], [1], they are, in general, not scalable to search videos due to such an enormous search space. To reduce this huge search space, some other methods try to avoid exhaustive search by sampling the search space, e.g., only considering a fixed number of spatial and temporal scales [4]. However, this treatment is likely to cause missing detections. Moreover, the solution space is still quite large, even after subsampling.

Second, human actions often exhibit tremendous amount of intrapattern variations. The same type of actions may look very different in their visual appearances. There are many factors that contribute to such variations, including the performing speed, clothing, scale, viewpoints, not to mention partial occlusions and cluttered backgrounds. When using a single and rigid action template for pattern matching as in [4], [5], the actions that vary from the template cannot be detected. One potential remedy is to use multiple templates to cover more variations, but the required number of templates will increase rapidly, resulting in formidable computational costs.

We propose an efficient action detection approach that addresses these two challenges mentioned above. Each action is characterized by a set of spatiotemporal interest points (STIPs) [6]. Provided a test video sequence, each STIP casts a positive or negative-valued vote for the action class, based on its mutual information with respect to that action class. As illustrated in Fig. 1, detection of an action is to search for a spatiotemporal subvolume that has the

- *J. Yuan is with the School of Electrical and Electronics Engineering, 50 Nanyang Avenue, Nanyang Technological University, Singapore. E-mail: jsyuan@ntu.edu.sg.*
- *Z. Liu is with Microsoft Research, 1 Microsoft Way, Redmond, WA 98052. E-mail: zliu@microsoft.com.*
- *Y. Wu with the Department of Electrical Engineering and Computer Science, 2145 Sheridan Road, Northwestern University, Evanston, IL 60208. E-mail: yingwu@eecs.northwestern.edu.*

Manuscript received 10 Oct. 2009; revised 30 Mar. 2010; accepted 25 Nov. 2010; published online 17 Feb. 2011.
Recommended for acceptance by P. Perez.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2009-10-0676.
Digital Object Identifier no. 10.1109/TPAMI.2011.38.

0162-8828/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

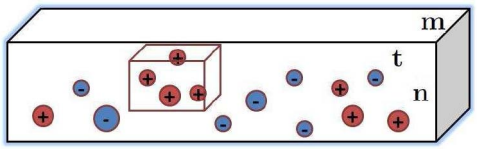


Fig. 1. Action detection through spatiotemporal pattern search. The highlighted subvolume has the maximum mutual information toward the specific event class. Each circle represents a spatiotemporal feature point which contributes a vote based on its own mutual information. The whole video volume is of size $m \times n \times t$, where $m \times n$ is the image size and t is the temporal length.

maximum total vote. Such a subvolume with maximum voting score also maximizes the pointwise mutual information toward a specific action class. Thus, it is treated as a valid detection instance of that action class. This is a new formulation of action detection.

To handle the intraclass action variations, various action templates that belong to the same class (or pattern) are collected and the collection of all the STIPs forms a pool of positive STIPs. As each action pattern is represented by more than one template, it is able to tolerate the variations in actions. In terms of pattern matching, to make an analogy to the template-based pattern matching where only positive templates are utilized, our pattern matching is called *discriminative matching* because of the use of both positive and negative templates. Given both the positive and negative training STIPs, a classification scheme called naive Bayes mutual information maximization (NBMIM) is proposed to classify a query video clip, i.e., a cloud of STIPs. By exploring discriminative learning, such a discriminative matching method can better distinguish the target action patterns from the cluttered and the moving backgrounds. Thus, a more robust pattern matching can be achieved.

To handle the large search space in video, we propose a method that decouples the temporal and spatial spaces and applies different search strategies, respectively. By combining dynamic programming in the temporal space and the branch-and-bound in the spatial space, the proposed spatiotemporal branch-and-bound (STBB) method significantly speeds up the search of the spatiotemporal action patterns. Moreover, we also investigate how to detect multiple action instances simultaneously, i.e., search for multiple subvolumes whose scores are higher than the detection threshold. Based on the new scheme, it can terminate many unnecessary candidates earlier during the process of branch-and-bound to save computation. It leads to a much faster search, without significantly degrading the quality of the detection results.

The benefits of our new method are threefold. First, the proposed discriminative pattern matching can well handle action variations by leveraging all of the training data instead of a single template. By incorporating the negative training information, our pattern matching has stronger discriminative power across different action classes. Second, our method does not rely on object tracking, detection, and background subtraction. It can handle background clutter and other moving objects in the background. Last but not least, the proposed spatiotemporal branch-and-bound search algorithm is computationally efficient and can find the global optimal solution. To validate the proposed action detection method, it has been tested on various data sets, including cross-data set experiments where the positive training data, negative training data, and test data are from different sources. The action categorization results on the KTH data set are comparable to the state-of-the-art results. The multiclass multiple-instance action detection results demonstrate the effectiveness and efficiency of our method.

2 Related Work

2.1 Action Recognition

Action recognition has been an active research topic. Given a video sequence, it requires identifying which type of action is performed in this video. Some previous works perform human action recognition based on the tracking of human body parts. For example, motion trajectories are used to represent actions in [7], [8], [9], [10], [11]. Unfortunately, robust object tracking is itself a nontrivial task. The problem becomes particularly difficult when there are occlusions or when the background is cluttered. Instead of relying on the body part information, some approaches use the silhouette information, such as using key poses in representing actions [12], [13]. Some other approaches treat actions as spatiotemporal shapes and characterize them by using manifold invariants in the spatiotemporal space. For example, in [14], a spatiotemporal volume (STV) is generated by the 2D contours of the object along the temporal axis. By considering the STV as a 3D manifold, this method extracts algebraic invariants of the manifold which correspond to the changes in direction, speed, and shape of the parts. Space-time shapes are also applied in [15]. It utilizes the properties of the solution to the Poisson equation to extract space-time features. Because both silhouettes and spatiotemporal shapes can be obtained through foreground-background separation, such action recognition approaches perform well when the background is reasonably clean or static. If the background is cluttered and dynamic, extracting the foreground becomes very difficult. The noisy and erroneous silhouettes or spatiotemporal shapes largely limit the performance of these methods.

To avoid the foreground-background separation, many recent methods applied local spatiotemporal features to characterize actions and perform action classification over the set of local features [16], [17], [18], [19], [20], [21], [22], [23], [24]. In [19], STIPs are proposed and applied to characterize human actions. In [18], local spatiotemporal features are quantized into "visual words" and the support vector machine is applied for classification. In [25], a video sequence is characterized by a "bag-of-words," where each frame corresponds to a "word." A semilatent topic model is then trained for action recognition. These previous good recognition results validate the advantages of using the spatiotemporal local features.

Besides using local features, there are many previous works in designing and fusing various types of features for classification. In [26], a set of kinematic features are proposed for action recognition. In [27], midlevel motion features are developed from low-level optical flow information for action recognition. In [28], a motion descriptor based on optical flow measurements in a spatiotemporal volume is introduced. All of these methods require optical flow estimation. In [29], a motion-based representation

called motion context is proposed for action recognition. To improve the recognition performance, both shape and motion information are used for action detection [30], [31], [32]. In [33], multiple features are fused using Fiedler Embedding. To select informative features, the PageRank algorithm is applied in [17]. Based on the Gaussian processes with multiple kernel covariance functions, [34] proposes a Bayesian classification method to automatically select and weight multiple features. Spatiotemporal context information is utilized in [11] to improve the performance of action recognition. In [35], a generative model is learned by using both semantic and structure information for action recognition and detection.

2.2 Action Detection

Differently from action recognition or categorization [36], [19], [37], where each action video is classified into one of the pre-defined action classes, the task of action detection [38], [39], [4], [40], [41] needs to identify not only which type of action occurs, but also where (spatial location in the image) and when (temporal location) it occurs in the video. As discussed in [42], it is in general a more challenging problem as it needs to not only recognize the action, but also to locate it in the video space [39], [4], [43], [40], [41], [44]. Some previous methods apply template-based action matching [5], [39], [45]. For example, two types of temporal templates are proposed in [5] for characterizing actions: 1) the motion energy image (MEI) and 2) the motion history image (MHI). To provide a viewpoint-free representation for human actions, [46] introduces motion history volume (MHV). Besides motion templates, some other approaches also characterize an action as a sequence of postures so that sequence matching methods can be applied to action recognition and detection [47], [48], [32]. In general, the template-based approach is sensitive to the cluttered and dynamic backgrounds. To address this problem, [4] proposes to oversegment the video into many spatiotemporal video volumes. An action template is then matched by searching among these oversegmented video volumes. However, because only one template is utilized in matching, previous template-based methods also have difficulties in handling intraclass action variations.

In addition to template matching, discriminative learning methods have also been applied to action detection. Motivated by the successful face detector [2], [38] extends the Haar features to the three-dimensional space, followed by the boosting algorithm to integrate these features for classification. In [41], multiple instance learning is presented for learning a human action detector. However, head detection and tracking are required to help locate the person. To learn the action representations, [49] proposes collecting images from the Web and using this knowledge to automatically annotate actions in videos. In [50], boosted space-time window classifiers are introduced to detect human actions on real movies with substantial variation of actions in terms of subject. Both human motion and shape features are applied. As an earlier version of this paper, [40] proposes an efficient three-dimensional branch-and-bound search for efficient action detection. This method is further developed in [51] for transductive action detection, where the requirement of training labels is reduced.

2.3 Object Recognition and Detection

Besides action recognition and detection, some recent works in object recognition and detection were also related to our work. A recent work in [52] proposed the naive Bayes nearest neighbor (NBNN) classifier for image classification. In [1], object detection is formulated as finding the optimal bounding box that gives the highest detection score in the image. An efficient branch-and-bound method is proposed to search for the optimal bounding box in the image. Despite the successful applications in object detection [1] and image retrieval [53], it still is a nontrivial problem to extend the efficient search method from the spatial image space to the spatiotemporal video space. Thus, a further study is required.

3 Classification Model of Actions

3.1 Interest Point Representation for Actions

We represent an action as a space-time object and characterize it by a collection of the STIPs [6]. Two types of features are used to describe the STIPs [19]: histogram of gradient (HOG) and histogram of flow (HOF), where HOG is the appearance feature and HOF is the motion feature. These features have shown promising results in action categorization [19]. We denote a video sequence by $\mathcal{V} = \{\mathbf{I}_t\}$, where each frame $\mathbf{I_t}$ consists of a collection of STIPs. We do not select key-frames, but collect all STIPs to represent a video clip by $\mathcal{Q} = \{d_i\}$.

3.2 Naive Bayes Mutual Information Maximization

We denote by $d \in \mathbb{R}^N$ a feature vector describing a STIP and by $\mathbf{C} \in \{1, 2, \ldots, C\}$ a class label. Based on the naive Bayes assumption and by assuming the independence among the STIPs, we can evaluate the *pointwise mutual information* between a video clip $\mathcal{Q}$ and a specific class $c \in \{1, 2, \ldots, C\}$ as:

$$\begin{aligned} &MI(\mathbf{C} = c, \mathcal{Q}) \\ &\quad = \log \frac{P(\mathcal{Q}|\mathbf{C} = c)}{P(\mathcal{Q})} = \log \frac{\prod_{d_q \in \mathcal{Q}} P(d_q|\mathbf{C} = c)}{\prod_{d_q \in \mathcal{Q}} P(d_q)} \\ &\quad = \sum_{d_q \in \mathcal{Q}} \log \frac{P(d_q|\mathbf{C} = c)}{P(d_q)} = \sum_{d_q \in \mathcal{Q}} s^c(d_q), \end{aligned} \tag{1}$$

where $s^c(d_q) = MI(\mathbf{C} = c, d_q)$ is the pointwise mutual information score to measure the association between d_q and class c. Assuming the independence among d_q, the final decision of $\mathcal{Q}$ is based on the summation of the mutual information from all primitive features $d_q \in \mathcal{Q}$ w.r.t. class c.

To evaluate the contribution $s^c(d_q)$ of each $d_q \in \mathcal{Q}$, we develop the pointwise mutual information through discriminative learning [54]:

$$\begin{aligned} s^c(d_q) &= MI(\mathbf{C} = c, d_q) = \log \frac{P(d_q|\mathbf{C} = c)}{P(d_q)} \\ &= \log \frac{P(d_q|\mathbf{C} = c)}{P(d_q|\mathbf{C} = c)P(\mathbf{C} = c) + P(d_q|\mathbf{C} \neq c)P(\mathbf{C} \neq c)} \\ &= \log \frac{1}{P(\mathbf{C} = c) + \frac{P(d_q|\mathbf{C} \neq c)}{P(d_q|\mathbf{C} = c)} P(\mathbf{C} \neq c)}. \end{aligned}$$

If the prior probabilities are equal, i.e., $P(\mathbf{C} = c) = \frac{1}{C}$, we further have

$$s^c(d_q) = \log \frac{C}{1 + \frac{P(d_q|\mathbf{C}\neq c)}{P(d_q|\mathbf{C}=c)}(C-1)}. \quad (3)$$

From (3), we can see that the likelihood ratio test $\frac{P(d_q|\mathbf{C}\neq c)}{P(d_q|\mathbf{C}=c)}$ determines whether d_q votes positively or negatively for the class c. When $MI(\mathbf{C}=c, d_q) > 0$, i.e., likelihood ratio $\frac{P(d_q|\mathbf{C}\neq c)}{P(d_q|\mathbf{C}=c)} < 1$, d_q votes a positive score $s^c(d_q)$ for the class c. Otherwise, if $MI(\mathbf{C}=c, d_q) \leq 0$, i.e., $\frac{P(d_q|\mathbf{C}\neq c)}{P(d_q|\mathbf{C}=c)} \geq 1$, d_q votes a negative score for the class c. After receiving the votes from every $d_q \in \mathcal{Q}$, we can make the final classification decision for $\mathcal{Q}$ based on its mutual information toward C classes.

For the C-class action categorization, we build C one-against-all classifiers. The test action $\mathcal{Q}$ is classified as the class that gives the maximum detection score:

$$c^* = \arg \max_{c\in\{1,2,..,C\}} MI(c, \mathcal{Q}) = \arg \max_{c\in\{1,2,..,C\}} \sum_{d\in\mathcal{Q}} s^c(d).$$

We call this naive Bayes mutual information maximization. Compared with the NBNN [52], each score $s^c(d)$ corresponds to the pointwise mutual information and can either be positive or negative. As will be explained in Section 4, such a property brings extra benefits in formulating action detection as a subvolume search problem where a computationally efficient detection solution can be found.

3.3 Likelihood Ratio Measurement

Denote by $\mathbb{T}^{c+} = \{\mathcal{V}_i\}$ the positive training data set of class c, where $\mathcal{V}_i \in \mathbb{T}^{c+}$ is a video of class c. As each $\mathcal{V}$ is characterized by a collection of STIPs, we represent the positive training data by the collection of all positive STIPs: $\mathbb{T}^{c+} = \{d_j\}$. Symmetrically, the negative data are denoted by $\mathbb{T}^{c-}$, which is the collection of all negative STIPs.

To evaluate the likelihood ratio for each $d \in \mathcal{Q}$, we apply the Gaussian kernel density estimation based on the training data $\mathbb{T}^{c+}$ and $\mathbb{T}^{c-}$. With a Gaussian kernel

$$K(d - d_j) = \frac{1}{\sqrt{2}\sigma} \exp^{-\frac{1}{2\sigma^2}\|d-d_j\|^2},$$

we adopt the nearest neighbor approximation as in [52]. The likelihood ratio becomes:

$$\frac{P(d|\mathbf{C}\neq c)}{P(d|\mathbf{C}=c)} = \frac{\frac{1}{|\mathbb{T}^{c-}|}\sum_{d_j\in\mathbb{T}^{c-}} K(d-d_j)}{\frac{1}{|\mathbb{T}^{c+}|}\sum_{d_j\in\mathbb{T}^{c+}} K(d-d_j)} \approx \exp^{-\frac{1}{2\sigma^2}(\|d-d^{c-}_{NN}\|^2 - \|d-d^{c+}_{NN}\|^2)}. \quad (4)$$

Here, d^{c-}_{NN} and d^{c+}_{NN} are the nearest neighbors of d in $\mathbb{T}^{c+}$ and $\mathbb{T}^{c-}$, respectively. We approximate the numerator $\frac{1}{|\mathbb{T}^{c-}|}\sum_{d_j\in\mathbb{T}^{c-}} K(d-d_j)$ by $\exp^{-\frac{1}{2\sigma^2}\|d-d^{c-}_{NN}\|^2}$, and the denominator

$$\frac{1}{|\mathbb{T}^{c+}|}\sum_{d_j\in\mathbb{T}^{c+}} K(d-d_j)$$

by $\exp^{-\frac{1}{2\sigma^2}\|d-d^{c+}_{NN}\|^2}$.

In kernel-based density estimation, it is difficult to select an appropriate kernel bandwidth σ. A large kernel bandwidth may oversmooth the density estimation, while a too small kernel bandwidth only counts on the nearest neighbors in the Parzen estimator. Let

$$\gamma(d) = \|d - d^{c-}_{NN}\|^2 - \|d - d^{c+}_{NN}\|^2. \quad (5)$$

According to (3) and (4), a positive $\gamma(d)$ will generate a positive score $s^c(d_q)$, while a negative $\gamma(d)$ will generate a negative $s^c(d)$. To avoid the selection of the best bandwidth σ, we adaptively adjust σ based on the *purity* in the neighborhood of a STIP d:

$$\frac{1}{2\sigma^2} = \begin{cases} \frac{|NN^{c+}_\epsilon(d)|}{|NN_\epsilon(d)|}, & \text{if } \gamma(d) \geq 0, \\ \frac{|NN^{c-}_\epsilon(d)|}{|NN_\epsilon(d)|}, & \text{if } \gamma(d) < 0, \end{cases} \quad (6)$$

where $NN^{c+}_\epsilon(d) = \{d_j \in \mathbb{T}^{c+} : \|d_j - d\| \leq \epsilon\}$ is the ϵ-nearest neighbors of point d in the positive class c; $NN^{c-}_\epsilon(d) = \{d_j \in \mathbb{T}^{c-} : \|d_j - d\| \leq \epsilon\}$ is the ϵ-nearest neighbors of point d in the negative class; $NN_\epsilon(d) = \{d_j \in \mathbb{T}^{c+} \cup \mathbb{T}^{c-} : \|d_j - d\| \leq \epsilon\}$ is the entire set of ϵ-nearest neighbors of d. With $\gamma(d)$ determining the sign of the vote $s^c(d)$, if d is located in a high purity region of the corresponding class, its vote $s^c(d)$ is stronger.

3.3.1 Efficient Nearest Neighbor Search

To obtain the voting score $s^c(d)$, for each $d \in \mathcal{Q}$ we need to search for its nearest neighbors (NNs). To improve the efficiency of searching in the high-dimensional feature space and to obtain $NN^c_\epsilon(d)$ quickly, we employ locality sensitive hashing (LSH) [55] to perform the approximate ϵ-nearest neighbors (ϵ-NN) search.

Based on $NN^{c+}_\epsilon(d)$ and $NN^{c-}_\epsilon(d)$, instead of searching for the global nearest neighbor for each class, we approximate it by the closest point to the query d in the ϵ-NN set. Taking the negative class as an example, we have:

$$\|d - d^{c-}_{NN}\| = \min_{d_i\in NN^{c-}_\epsilon(d)} \|d - d_i\|.$$

It is worth noting that d^{c-}_{NN} depends on the selection of ϵ. If we happen to have $|NN^{c-}_\epsilon(d)| = 0$, we assume the negative nearest neighbor is at distance ϵ, namely, $\|d - d^{c-}_{NN}\|^2 = \epsilon^2$ in (5). Applying the same strategy to the positive class, we have:

$$\|d - d^{c+}_{NN}\| = \min_{d_i\in NN^{c+}_\epsilon(d)} \|d - d_i\|.$$

When $|NN^{c+}_\epsilon(d)| = 0$, we assume $\|d - d^{c+}_{NN}\|^2 = \epsilon^2$ in (5).

4 Discriminative Video Pattern Search

Based on the proposed NBMIM criterion, action detection is to find a subvolume of maximum mutual information. As illustrated in Fig. 1, given a video sequence $\mathcal{V}$, we want to find a spatiotemporal subvolume $V^* \subset \mathcal{V}$ with the highest mutual information score. Since STIPs are sparse features and involve only a very small number of pixels $d \in \mathcal{V}$, the optimal subvolume V^* may not be a unique one. For example, if a frame does not contain any STIPs, it becomes arbitrary for V^* to include this empty frame as it does not affect the total voting score. To avoid this problem, we introduce a very small negative vote $s(d_\emptyset) < 0$ to the empty pixels that are not associated with any STIP. Such a negative prior discourages the inclusion of empty pixels into V^*.

Given a specific class c, our target is to search for the optimal subvolume:

$$V^* = \arg\max_{V\subseteq\mathcal{V}} MI(V, \mathbf{C} = c) = \arg\max_{V\subseteq\mathcal{V}} \sum_{d\in V} s^c(d) = \arg\max_{V\in\mathbf{\Lambda}} f(V), \quad (7)$$

where $f(V) = \sum_{d\in V} s^c(d)$ is the objective function and $\mathbf{\Lambda}$ denotes the candidate set of all valid subvolumes in $\mathcal{V}$. Suppose the target video $\mathcal{V}$ is of size $m \times n \times t$. The optimal solution $V^* = t^* \times b^* \times l^* \times r^* \times s^* \times e^*$ has six parameters to be determined, where $t^*, b^* \in [0, m]$ denote the top and bottom positions, $l^*, r^* \in [0, n]$ denote the left and right positions, and $s^*, e^* \in [0, t]$ denote the start and end positions. As a counterpart of the bounding-box-based object detection, the solution V^* is the 3D bounding volume that has the highest score for the target action.

The total number of the subvolumes is in the order of $O(n^2m^2t^2)$. Therefore, it is computationally prohibitive to perform an exhaustive search to find the optimal subvolume V^* from such an enormous candidate pool. In the following, we first present the conventional branch-and-bound solution extended directly from 2D bounding box search in [1], and then present our new method to find V^* more efficiently.

4.1 Spatiotemporal Branch-and-Bound Search

4.1.1 Conventional Branch-and-Bound Search

A branch-and-bound solution is proposed in [1] for searching the optimal bounding box in an image for object detection. This idea can be directly extended to find the optimal subvolume in videos, by replacing the spatial bounding box by a spatiotemporal subvolume.

Denote by $\mathbb{V}$ a collection of subvolumes. Assume there exist two subvolumes V_{min} and V_{max} such that for any $V \in \mathbb{V}$, $V_{min} \subseteq V \subseteq V_{max}$. Then, we have

$$f(V) \leq f^+(V_{max}) + f^-(V_{min}), \quad (8)$$

where $f^+(V) = \sum_{d\in V} \max(s^c(d), 0)$ contains the positive votes, while $f^-(V) = \sum_{d\in V} \min(s^c(d), 0)$ contains the negative ones. We denote the upper bound of $f(V)$ for all $V \in \mathbb{V}$ by:

$$\hat{f}(\mathbb{V}) = f^+(V_{max}) + f^-(V_{min}) \geq \max_{V\in\mathbb{V}} f(V). \quad (9)$$

Moreover, it is easy to see that if V is the only element in $\mathbb{V}$, we have the equality:

$$\hat{f}(\mathbb{V}) = f(V). \quad (10)$$

Equations (9) and (10) thus meet the two requirements discussed in [1] for the effective upper bound in the branch-and-bound search. With the first condition in (9), $\hat{f}(\mathbb{V})$ is an upper bound of $f(V)$. Therefore, it does not incur miss detection by using $\hat{f}(\mathbb{V})$ for pruning unsatisfactory candidates. It guarantees the optimality of the solution. The second condition in (10) provides the termination condition of the branch-and-bound.

In order to distinguish this method from our new method, we call it conventional branch-and-bound method (Algorithm 1). Compared to the spatial bounding box searching, the search of spatiotemporal subvolume is much more difficult. In videos, the search space has two additional parameters (the start and end on the time dimension) and expands from four dimensions to six dimensions. As the complexity of the branch-and-bound grows exponentially with respect to the number of dimensions, the conventional branch-and-bound solution is too slow for videos.

Algorithm 1: Conventional branch-and-bound (BB) search (extension of [1])

input : video $\mathcal{V} \in \mathbb{R}^{n\times m\times t}$;
quality bounding function $\hat{f}$ (see text)
output : $V^* = \arg\max_{V\subseteq\mathcal{V}} f(V)$

```
initialize P as an empty priority queue
set 𝕍 = [0,n] × [0,n] × [0,m] × [0,m] × [0,t] × [0,t]
while 𝕍 contains more than one element do
    split 𝕍 → 𝕍¹ ∪ 𝕍²
    get upper bound f̂(𝕍¹)
    push (𝕍¹, f̂(𝕍¹)) into P
    get upper bound f̂(𝕍²)
    push (𝕍², f̂(𝕍²)) into P
    retrieve top state 𝕍 from P based on f̂(𝕍)
return V* = 𝕍
```

4.1.2 Spatiotemporal Branch-and-Bound Search

We present a new method called STBB search to search the video space. Instead of directly applying branch-and-bound in the 6D parameter space, our new method decomposes it into two subspaces: 1) 4D spatial parameter space and 2) 2D temporal parameter space. We denote by $W \in \mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}$ a spatial window and $T \in \mathbb{R} \times \mathbb{R}$ a temporal segment. A subvolume V is uniquely determined by W and T. The detection score of a subvolume $f(V_{W\times T})$ is

$$f(V_{W\times T}) = f(W, T) = \sum_{d\in W\times T} s(d). \quad (11)$$

Let $\mathbb{WW} = [0, m] \times [0, m] \times [0, n] \times [0, n]$ be the parameter space of the spatial windows, and $\mathbb{TT} = [0, t] \times [0, t]$ be the parameter space of temporal segments. Our objective here is to find the spatiotemporal subvolume which has the maximum detection score:

$$[W^*, T^*] = \arg\max_{W\in\mathbb{WW}, T\in\mathbb{TT}} f(W, T). \quad (12)$$

The optimal detection score is

$$F(W^*) = \max_{W\in\mathbb{WW}} F(W) = \max_{W\in\mathbb{WW}} \max_{T\in\mathbb{TT}} f(W, T). \quad (13)$$

We take different search strategies in the two subspaces $\mathbb{WW}$ and $\mathbb{TT}$ and search alternately between $\mathbb{WW}$ and $\mathbb{TT}$. First, if the spatial window W is determined, we can easily search for the optimal temporal segment in space $\mathbb{TT}$:

$$F(W) = \max_{T\in\mathbb{TT}} f(W, T), \quad (14)$$

This relates to the max subvector problem, where given a real vector, the output is the contiguous subvector of the input that has the maximum sum (see Fig. 3). We will discuss its efficient solution later.

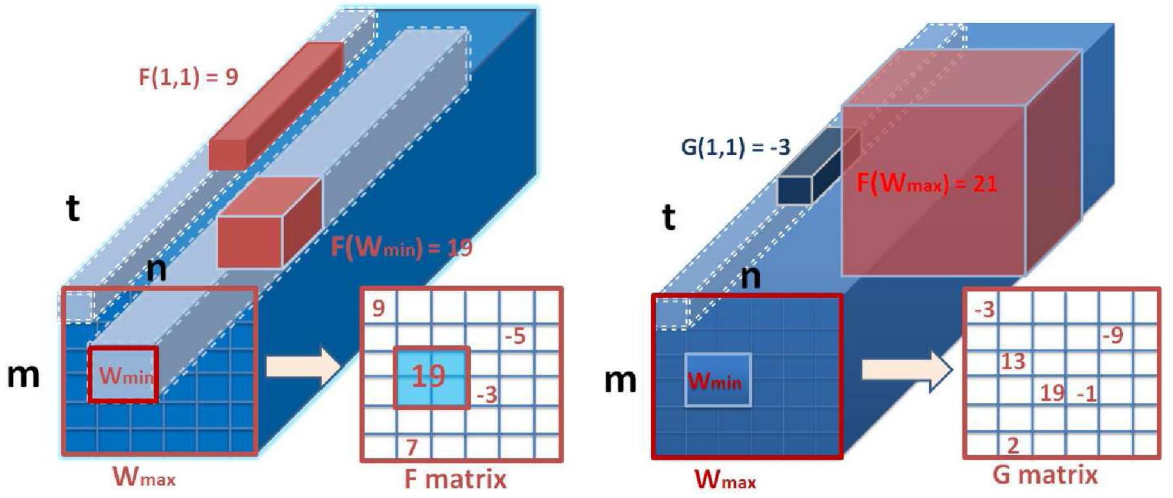


Fig. 2. Illustration of the upper bound estimation. $\mathbb{W}$ denotes a set of spatial windows. W_{max} and W_{min} are the maximum and minimum windows in $\mathbb{W}$, respectively. F and G are two matrices obtained through (16) and (17), respectively. Empty cells in F and G matrices correspond to null entries. Left figure: The first upper bound in Lemma 1. The upper bound is $\hat{F}_1(\mathbb{W}) = 19 + 9 + 7 = 35$. Right figure: The second upper bound in Lemma 2. The upper bound is $\hat{F}_2(\mathrm{W\ W}) = 21 - (-3 - 9 - 1) = 34$.

To search the spatial parameter space $\mathbb{W}$, we employ a branch-and-bound strategy. Since the efficiency of a branch-and-bound-based algorithm critically depends on the tightness of the upper bound, we first derive a tighter upper bound.

Given an arbitrary parameter space $\mathbb{W} = [m_1, m_2] \times [m_1, m_2] \times [n_1, n_2] \times [n_1, n_2]$, the optimal solution is

$$W^* = \arg\max_{W \in \mathbb{W}} F(W). \tag{15}$$

We define $F(\mathbb{W}) = F(W^*)$. Assume there exist two sub-rectangles W_{min} and W_{max} such that $W_{min} \subseteq W \subseteq W_{max}$ for any $W \in \mathbb{W}$. For each pixel $i \in W_{max}$, we denote the maximum sum of the 1D subvector along the temporal direction at pixel i's location by:

$$F(i) = \max_{T \subseteq \mathbb{T}} f(i, T). \tag{16}$$

Letting $F^+(i) = \max(F(i), 0)$, we have the first upper bound for $F(\mathbb{W})$, as presented in Lemma 1.

Lemma 1 (upper bound $\hat{F}_1(\mathbb{W})$). *Given a spatial parameter space $\mathbb{W} = \{W : W_{min} \subseteq W \subseteq W_{max}\}$, we have*

$$F(\mathbb{W}) \leq \hat{F}_1(\mathbb{W}) = F(W_{min}) + \sum_{i \in W_{max}, i \notin W_{min}} F^+(i).$$

When $W_{max} = W_{min}$, we have the tight bound $\hat{F}_1(\mathbb{W}) = F(W_{min}) = F(W^*)$.

Symmetrically, for each pixel $i \in W_{max}$, we denote the minimum sum of the 1D subvector at pixel i's location by

$$G(i) = \min_{T \subseteq \mathbb{T}} f(i, T). \tag{17}$$

Let $G^-(i) = \min(G(i), 0)$, and Lemma 2 presents the other upper bound of $F(\mathbb{W})$.

Lemma 2 (upper bound $\hat{F}_2(\mathbb{W})$). *Given a spatial parameter space $\mathbb{W} = \{W : W_{min} \subseteq W \subseteq W_{max}\}$, we have*

$$F(\mathbb{W}) \leq \hat{F}_2(\mathbb{W}) = F(W_{max}) - \sum_{i \in W_{max}, i \notin W_{min}} G^-(i).$$

When $W_{max} = W_{min}$, we have the tight bound $\hat{F}_2(\mathbb{W}) = F(W_{max}) = F(W^*)$.

The proofs of Lemmas 1 and 2 are given in the Appendix, which can be found in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.38. The two lemmas are illustrated in Fig. 2, where $F(W_{min}) = 19$, and $F(W_{max}) = 21$. The values of $F(i)$ are shown in the F matrix where blank cells indicate zeros. The values of $G(i)$ are shown in the G matrix. Lemma 1 gives the upper bound $\hat{F}_1(\mathbb{W}) = 19 + 9 + 7 = 35$ and Lemma 2 gives the upper bound $\hat{F}_2(\mathbb{W}) = 21 - (-3 - 9 - 1) = 34$. Based on Lemmas 1 and 2, we can obtain a final tighter upper bound, which is the minimum of the two available upper bounds.

Theorem 1 (Tighter upper bound $\hat{F}(\mathbb{W})$). *Given a spatial parameter space $\mathbb{W} = \{W : W_{min} \subseteq W \subseteq W_{max}\}$, the upper bound of the optimal solution $F(\mathbb{W})$ is*

$$F(\mathbb{W}) \leq \hat{F}(\mathbb{W}) = \min\{\hat{F}_1(\mathbb{W}), \hat{F}_2(\mathbb{W})\}. \tag{18}$$

Based on the tighter upper bound derived in Theorem 1, we propose our new branch-and-bound solution in the spatial parameter space $\mathbb{W}$ in Algorithm 2. Since the convergence speed of branch-and-bound method highly depends on the tightness of the bound, the new algorithm can converge much faster with a better upper bound estimation. Moreover, compared to the conventional branch-and-bound solution in Algorithm 1, the new STBB algorithm keeps track of the current best solution which is denoted by W^*. Only when a parameter space $\mathbb{W}$ contains potentially better solution (i.e., $\hat{F}(\mathbb{W}) \geq F^*$), we push it into the queue. Otherwise, we discard the whole space $\mathbb{W}$. It thus saves the memory in maintaining the priority queue of $\mathbb{W}$.

Fig. 3. Max subvector search: The highlighted element in red is the subvector of max sum, which is $8 - 1 + 5 = 12$.

Algorithm 2: Spatio-temporal branch-and-bound (STBB) search

input : video $\mathcal{V} \in \mathbb{R}^{m \times n \times t}$
quality bounding function $\hat{F}$ (see text)
output : $V^* = \arg\max_{V \subseteq \mathcal{V}} f(V)$

initialize P as an empty priority queue
set $\mathbb{W} = [T, B, L, R] = [0, n] \times [0, n] \times [0, m] \times [0, m]$
set $\hat{F}(\mathbb{W}) = \min\{\hat{F}_1(\mathbb{W}), \hat{F}_2(\mathbb{W})\}$
push $(\mathbb{W}, \hat{F}(\mathbb{W}))$ into P
set current best solution $\{W^*, F^*\} = \{W_{max}, F(W_{max})\}$;
repeat
  retrieve top state $\mathbb{W}$ from P based on $\hat{F}(\mathbb{W})$
  if $(\hat{F}(\mathbb{W}) > F^*)$ **then**
    split $\mathbb{W} \rightarrow \mathbb{W}^1 \cup \mathbb{W}^2$
    CheckToUpdate($\mathbb{W}_1$, W^*, F^*, P);
    CheckToUpdate($\mathbb{W}_2$, W^*, F^*, P);
  else
    $T^* = \arg\max_{T \subset [0,t]} f(W^*, T)$;
    return $V^* = [W^*, T^*]$.
until *stop* ;

Function CheckToUpdate($\mathbb{W}$, W^*, F^*, P)
Get W_{min} and W_{max} of $\mathbb{W}$
if $(F(W_{min}) > F^*)$ **then**
  update $\{W^*, F^*\} = \{W_{min}, F(W_{min})\}$;
if $(F(W_{max}) > F^*)$ **then**
  update $\{W^*, F^*\} = \{W_{max}, F(W_{max})\}$;
if $(W_{max} \neq W_{min})$ **then**
  get $\hat{F}(\mathbb{W}) = \min\{\hat{F}_1(\mathbb{W}), \hat{F}_2(\mathbb{W})\}$
  if $\hat{F}(\mathbb{W}) \geq F^*$ **then**
    push $(\mathbb{W}, \hat{F}(\mathbb{W}))$ into P

4.1.3 Efficient Upper Bound Estimation for Branch-and-Bound Search

To estimate the upper bound in Theorem 1, as well as to search for the optimal temporal segment T^* given a spatial window W, we design an efficient way to evaluate $F(W_{max})$, $F(W_{min})$, and, in general, $F(W)$.

According to (14), given a spatial window W of a fixed size, we need to search for a temporal segment with maximum summation. To present our efficient solution, we first review the classic max subvector problem in one-dimensional pattern recognition. It is the degeneration of the maximum subvolume problem in spatiotemporal space. There exists an elegant solution called Kadane's algorithm which is of a linear complexity using dynamic programming [56]. We present Kadane's algorithm in Algorithm 3. The max sum problem is illustrated in Fig. 3.

Kadane's algorithm can accelerate the temporal search and provide an efficient estimation of the upper bounds. Given any spatial window W, the summation within W at each frame j is $f(W, j) = \sum_{d \in W \times j} s(d)$. By applying the trick of integral-image, $f(W, j)$ can be obtained in a constant time. Let $v(j) = f(W, j)$, the evaluation of $F(W)$ in (14) is to find the max subvector in v. By using Kadane's algorithm, it can be done in a linear time. As a result, both upperbounds in Lemmas 1 and 2 can be obtained in a linear time. Therefore, the estimation of the upper bound $\hat{F}(\mathbb{W})$ in Theorem 1 is of a linear complexity $O(t)$.

TABLE 1
Complexity Comparison between Spatiotemporal Branch-and-Bound and Conventional Branch-and-Bound

	BB (Alg. 1)	STBB (Alg. 2)
Dimensions for B&B	6 (spatio-temporal)	4 (spatial)
Upper bound est.	$O(1)$	$O(t)$
Worst case	$O(m^2n^2t^2)$	$O(m^2n^2t)$

Algorithm 3: The linear algorithm of max subvector [56]

input : real vector v of length $t + 1$
output : $T^* = \arg\max_{T \subseteq [0,t]} \sum_{i \in T} v(i)$

set $MaxSofar = MaxEndingHere = 0$;
set $Start = End = 0$;
for $i = 0 : t$ **do**
  $MaxEndingHere = \max(0, MaxEndingHere + v(i))$;
  if $MaxEndingHere = 0$ **then**
    $CurStart = \min(i + 1, t)$;
  if $MaxSoFar \leq MaxEndingHere$ **then**
    $Start = CurStart$;
    $End = i$;
  $MaxSofar = \max(MaxSofar, MaxEndingHere)$;
return $T^* = [Start, End]$;

The complexity comparison between our proposed method (Algorithm 2) and the conventional branch-and-bound (Algorithm 1) is presented in Table 1. As our branch-and-bound is only performed in the spatial space, the worst-case complexity of our Algorithm 2 ($O(m^2n^2t)$) is better than that of Algorithm 1 ($O(m^2n^2t^2)$), which needs to perform branch-and-bound in the spatiotemporal space.

Algorithm 4: Multiple-instance action detection

input : video $\mathcal{V} \in \mathbb{R}^{m \times n \times t}$;
detection threshold D_t
output : a collection of detections: $V^* \subseteq \mathcal{V}$, s.t. $f(V^*) \geq D_t$

repeat
  V^* = STBBSearch($\mathcal{V}$);
  clear V^* to zero values and update $\mathcal{V}$.
until *the current detection is invalid:* $f(V^*) < D_t$;

5 Multiclass Multiple-Instance Action Detection

5.1 Multiple-Instance Detection Algorithm

The branch-and-bound approaches in Algorithms 1 and 2 are designed to search for a unique subvolume of maximum score. For multiple instance action detection, the same algorithm needs to be performed multiple rounds. At each round, the score of the detected subvolume V^* is compared against a predefined detection threshold D_t in order to determine whether it is a valid detection. If it is a valid

detection, we clear it by setting the score of $i \in V^*$ to $s(d_\emptyset)$ and continue to find the next subvolume of the maximum detection score. This process continues until the current best subvolume is not a valid detection.

5.2 Accelerated STBB (A-STBB) for Multiple-Instance Detection

To improve the efficiency of multiple-instance detection, we modify the original STBB search in Algorithm 2 and propose an accelerated STBB. We briefly explain the main idea below. For multiple-instance detection, the detection threshold $D_t > 0$ can be used to speed up the search process by terminating many unnecessary branches earlier during the branch-and-bound process. First of all, if there is no valid detection in a video sequence, then instead of finding the optimal subvolume V^* with the maximum detection score, we can safely terminate the search at an earlier stage. For example, if a parameter space $\mathbb{W}$ satisfies $\hat{f}(\mathbb{W}) \leq D_t$, it indicates that $\mathbb{W}$ is an invalid parameter space because the score of the best candidate is still below the detection threshold. Therefore, $\mathbb{W}$ does not require a further inspection. If none of the remaining candidates satisfies $\hat{f}(\mathbb{W}) \geq D_t$, then the search can be safely terminated because no valid detection will be found.

Furthermore, if a subvolume V with valid detection score $f(V) \geq D_t$ is already found, we can quickly finalize the detection based on the current solution, instead of keeping looking for the maximum V^*. In such a case, although the final detection may not be the optimal subvolume V^*, it still provides a valid detection where $f(V) \geq D_t$. Therefore, it leads to a much faster search without significantly degrading the quality of the detection results.

Incorporating the above two heuristics, we present the A-STBB search in Algorithm 5. Compared with the STBB search in Algorithm 2, during each search iteration, we retrieve an upper bounded estimation $\hat{F}(\mathbb{W})$ from the heap. If $\hat{F}(\mathbb{W}) < D_t$, we directly reject the whole video sequence $\mathcal{V}$, since no V^* can achieve the detection threshold. This strategy largely speeds up the scanning of negative video sequences which do not contain the target action. Moreover, at each search iteration, we also keep track of the current best score F^*. When $F^* \geq D_t$, it indicates that there exists a valid detection in the corresponding parameter space $\mathbb{W}$. In such a case, we speed up the search by limiting the rest of the search space within $\mathbb{W}$ only. In other words, instead of searching for the optimal $f(V^*)$ globally, we are satisfied with the local optimal solution $f(\tilde{V}) > D_t$. Since only one subvolume with qualified score will be selected while other subvolumes are discarded, our A-STBB performs the nonmaxima suppression implicitly during the search process.

Algorithm 5: Accelerated STBB (A-STBB) search

input : video $\mathcal{V} \in \mathbb{R}^{m \times n \times t}$;
detection threshold D_t
output : a subvolume $\tilde{V} \subseteq \mathcal{V}$, s.t. $f(\tilde{V}) \geq D_t$ (if no valid detection, return $\tilde{V} = \emptyset$)

set $\mathbb{W} = [T, B, L, R] = [0, n] \times [0, n] \times [0, m] \times [0, m]$
get $\hat{F}(\mathbb{W}) = \min\{\hat{F}_1(\mathbb{W}), \hat{F}_2(\mathbb{W})\}$
push ($\mathbb{W}$, $\hat{F}(\mathbb{W})$) into empty priority queue P
set current best solution $\{W^*, F^*\} = \{W_{max}, F(W_{max})\}$;
repeat
  retrieve top state $\mathbb{W}$ from P based on $\hat{F}(\mathbb{W})$
  if $\hat{F}(\mathbb{W}) < D_t$ **then**
    return $\tilde{V} = \emptyset$
  if $(\hat{F}(\mathbb{W}) > F^*)$ **then**
    split $\mathbb{W} \rightarrow \mathbb{W}^1 \cup \mathbb{W}^2$
    CheckToUpdate($\mathbb{W}_1$, W^*, F^*, P);
    CheckToUpdate($\mathbb{W}_2$, W^*, F^*, P);
  else
    $T^* = \arg\max_{T \subset [0,t]} f(W^*, T)$;
    return $\tilde{V} = [W^*, T^*]$.
until *stop* ;

Function CheckToUpdate($\mathbb{W}$, W^*, F^*, P)
if $(F^* \geq D_t)$ **then**
  clear priority queue P push ($\mathbb{W}$, $\hat{F}(\mathbb{W})$) into empty priority queue P
else
  Get W_{min} and W_{max} of $\mathbb{W}$ **if** $(F(W_{min}) > F^*)$ **then**
    update $\{W^*, F^*\} = \{W_{min}, F(W_{min})\}$;
  if $(F(W_{max}) > F^*)$ **then**
    update $\{W^*, F^*\} = \{W_{max}, F(W_{max})\}$;
  if $(W_{max} \neq W_{min})$ **then**
    get $\hat{F}(\mathbb{W}) = \min\{\hat{F}_1(\mathbb{W}), \hat{F}_2(\mathbb{W})\}$
  if $\hat{F}(\mathbb{W}) \geq F^*$ **then**
    push ($\mathbb{W}$, $\hat{F}(\mathbb{W})$) into P

6 Experiments

6.1 Action Categorization

We use the KTH data set to evaluate the proposed NBMIM classifier on action categorization. The KTH data set contains six types of human actions: walking, jogging, running, boxing, hand waving, and hand clapping, each of which is performed several times by 25 subjects. There are four different environments where the video sequences are captured: outdoors, outdoors with scale variation, outdoors with different clothes, and indoors. The video is captured at 25 frames per second and at a low image resolution of 160×120.

We follow the standard experiment setting of KTH data set as in [36], [19]. The whole data set contains 598 video sequences, taken over homogeneous backgrounds with a static camera. Each sequence is further segmented into four subsequences according to [36], thus it gives, in total, 2,391 action videos. Each action video has an average length of four seconds. Among the 25 people, 16 of them are used for training and the remaining nine are used for testing. The training data set contains 1,528 individual actions and the testing data set contains 863 individual actions. We apply both motion (histogram of motion) and appearance (histogram of gradient) features as in [19]. By concatenating the HOG and HOF features, a 162-dimensional feature vector is used to characterize each STIP. The average euclidean length of the STIP descriptor is 4.46. The training data set generates a pool of 308,110 STIPs. Given a query STIP, we search its ϵ-nearest neighbors using locality sensitive hashing. The E2LSH package [55] is employed and the probability for correct retrieval is set to $p = 0.9$.

The threshold of the nearest neighbor search, ϵ, is the only parameter of the proposed NBMIM classifier. Its influence is

TABLE 2
The Comparison between NBMIM (Adaptive and Fixed Kernel Bandwidth) and NBNN, with Different Selections of ϵ

	$\epsilon = 1.8$	$\epsilon = 2.0$	$\epsilon = 2.2$	$\epsilon = 2.4$	$\epsilon = 2.6$
adaptive	91.8%	**93.0%**	**93.7 %**	**93.4 %**	**93.3 %**
$\frac{1}{2\sigma^2} = 1$	**91.9%**	92.2%	92.7 %	92.7 %	92.6 %
NBNN	91.7%	91.8%	92.5%	92.6%	92.7 %

TABLE 3
Confusion Matrix for the KTH Action Data Set, Where the Total Accuracy Is 93.7 Percent

	clap	wave	walk	box	run	jog
clapping	**144**	0	0	0	0	0
waving	5	**139**	0	0	0	0
walking	0	0	**144**	0	0	0
boxing	0	0	0	**143**	0	0
running	1	0	0	0	**105**	38
jogging	2	0	4	0	4	**134**

twofold. First of all, it affects the search speed and the quantity of the nearest neighbors. The larger the ϵ, the slower the approximate ϵ-NN search using LSH, but the more nearest neighbors it will find. Second, ϵ also controls the bandwidth σ in the kernel density estimation according to (6). To evaluate the influence of ϵ, we test different choices of ϵ and compare three different classification models: NBMIM (adaptive kernel bandwidth), NBMIM (fixed kernel bandwidth), and NBNN in [52]. To make a fair comparison to NBNN, we use the same parameter for the approximate nearest neighbor search as described in Section 3.3.1. All three classifiers share the same d_{NN}^{c+} and d_{NN}^{c-}. The only difference is the voting score $s_c(d)$. In this experiment, since each action class has approximately the same number of video sequences, we assume the prior probabilities are equal and apply (3) to calculate $s_c(d)$. The result in Table 2 shows that the classification performance is not very sensitive to the selection of ϵ. Our proposed NBMIM with the adaptive kernel bandwidth performs slightly better than NBMIM with a fixed bandwidth, as well as NBNN. It is worth noting that the NBNN classifier cannot be directly applied to the detection formulation of (7) because its voting score is always positive.

The best action categorization results are presented in Table 3, with $\epsilon = 2.2$ and using the adaptive kernel bandwidth. Among the 863 testing actions, we obtained 54 errors, and the total accuracy is 93.7 percent. Among the six types of actions, hand clapping, walking, and boxing receive 100 percent accuracy. Most of the errors are due to the misclassification of running to jogging.

In Table 4, we further compare our results with that of [19] by applying exactly the same training[1] and testing data set, as well as the same STIP features. However, we do not quantize STIPs into "words." Instead of using the SVM, we match the raw STIPs in the original high-dimensional feature space and apply the NBMIM classifier. Our results show that, without quantizing primitive features into "words," the classification performance can be further improved. This is consistent with the discussion in [52] which pointed out that the nearest neighbor approach has the potential to provide better classification performance than the SVM based on the "bag-of-words" representation, where the quantization step can introduce a loss of discriminative information.

6.2 Detecting Two-Hand Waving Action

We select the two-hand waving action as a concrete example for action detection. To validate the generalization ability of our method, we apply completely different data sets for training (KTH data set) and testing (CMU action data set [4]). As summarized in Table 5, for the positive training data, we apply the KTH hand waving data set that contains 16 people. The negative training data are constituted of two parts: 1) the KTH walking data set, which contains 16 people, and 2) one office indoor sequence, which contains actions of sitting down and standing up.

The testing data set has 48 sequences, which include two types of actions in the CMU data set: 1) two-hand waving and 2) jumping jacks. Both of them contain the two-hand waving actions. The duration of each test sequence ranges from 10 to 40 seconds, and the video resolution is 160×120. Among the 48 sequences, 19 of them contain a total number of 52 positive instances. The other 29 sequences do not contain positive examples. For the density estimation, we set $\epsilon = 2.6$ for the nearest neighbor search. To avoid noisy detection, we regard a detection as invalid if its temporal duration is shorter than 0.4 seconds.

To evaluate the results, we apply a similar measurement proposed in [4], but with a relatively loose criterion. For the precision score, a detection is regarded as correct if at least $1/8$ of the volume size overlaps with the ground truth. For the recall score, the ground truth is regarded as retrieved if at least $1/8$ of its volume size is covered by at least one detection. We use such a measurement because the ground truth labels cover the entire human body while the region that exhibits hand waving action is much smaller. Even though the localized bounding box of the proposed method is not very accurate, one can always refine the bounding box by using other cues after roughly determining the location. We apply the precision and recall scores to evaluate the detection performance, where precision = # correct detect/# total detect, and recall = # correct detect/# total action.

We apply the efficient A-STBB search to detect two-hand waving actions. To evaluate the influence of the parameter $s(d_\emptyset)$, we test a number of different values of $s(d_\emptyset)$, including $s(d_\emptyset) = -10 \times 10^{-5}$, -7.5×10^{-5}, -6×10^{-5}, -5×10^{-5}, -4×10^{-5}, -2.5×10^{-5}, -1×10^{-5}. Fig. 4 presents the precision-recall curves by increasing the detection threshold D_t from 5 to 40. It shows that $s(d_\emptyset)$ is an important parameter that can influence the detection results significantly. When a small $s(d_\emptyset)$ is selected, the detected maximum subvolume is of a large size, thus having a sufficient overlap with the ground truth. Therefore, we obtain a higher recall score while the precision score gets worse. On the other hand, when a large $s(d_\emptyset)$ is selected, the detected subvolume is of a small size thus the overlap with the ground truth volume becomes smaller. This results in a worse recall score but a

1. In [19], eight people are used for training and another eight people are used as cross validation for parameter tuning of the SVM. We use the whole 16 people as the training data.

TABLE 4
Comparison between NBMIM and SVM

	training	testing	features	classifier	accuracy
[19]	8 persons + 8 person CV	9 persons	STIP + "bag of words"	non-linear SVM	91.8 %
ours	16 persons	9 persons	STIP	NBMIM	**93.7 %**

better precision score. When selecting $s(d_\emptyset) = -4 \times 10^{-5}$, both precision and recall scores achieve above 70 percent at a specific detection threshold.

Some detection examples are presented in Figs. 10, 11, 12, 13, and 14. The yellow bounding box is the ground truth label of the whole human body action and the red bounding box is our detection of the two-hand waving action. Since both motion and appearance features are used, our method can tolerate action pattern variations caused by the change of subjects. Our detection method can also handle scale changes of the actions, performing speed variations, background clutter, and even partial occlusion. Fig. 10 shows the same person performing two-hand waving with two different styles and different speeds. In Fig. 11, two actions with large-scale variations are detected successfully. Fig. 12 shows action detection results on cluttered backgrounds and with severe partial occlusions, where target tracking is very difficult.

Most of the missing and false detections are caused by the bad lighting conditions, crowded scenes, large viewpoint changes, or moving cameras. Fig. 13 presents a false detection example where two single-hand waving actions occur together and brings a false detection of the two-hand waving action. As the naive Bayes assumption does not consider the geometric relations among STIPs, our approach cannot distinguish whether the two waving hands are from the same person or not. To better handle this problem, a geometric model would be required for a further verification. Finally, Fig. 14 shows an example of the missed detection. Although it is indeed a partial detection, the overlap region with the ground truth is less than $1/8$, thus it is treated as a missed detection.

6.3 Multiclass Multiple-Instance Action Detection

Based on the multiclass action recognition model, we can perform multiclass multiple-instance action detection. To validate the generalization ability of our method, we still apply a cross-data set training and testing. We select three classes of actions for positive training from the KTH data set: boxing, hand waving, and hand clapping, including 16 subjects for each class of action. Because the action instances are captured in different environments and viewpoints, and exhibit spatial scale and style variations, the intraclass variations of actions are well captured in the training data. To better distinguish the three types of target actions from other types of movements, we also use the walking class in the KTH data set as the common negative class. As a result, for each of the three action classes, the negative training data set includes the STIPs from the walking class, as well as the STIPs from other two action classes.

The testing videos are captured by ourselves (see Fig. 5). Each testing sequence is of a higher resolution 320×240, compared with that of 160×120 in the training videos in the KTH data set. The frame rate is 15 frames per second. The testing data set contains 16 video sequences. Each video sequence is between 32 and 76 seconds. It has, in total, 63 action instances: 14 hand clapping, 24 hand waving, and 25 boxing, performed by 10 different subjects who do not appear in the training data. Each sequence contains multiple types of actions, performed by one or multiple subjects. As a challenging data set, all of the video sequences are captured in cluttered and moving backgrounds, including both indoor and outdoor scenes. The style and scale of actions can vary significantly depending on the subject. To evaluate the performance, we manually label a spatiotemporal bounding box for each action instance. A detected action is regarded as correct if at least $1/8$ of the volume size overlaps with a ground truth label. On the other hand, an action is regarded as retrieved if at least $1/8$ of its volume size overlaps with that of a valid detection. To filter out noisy detections, we require a valid detection lasts between 20 and 200 frames. For the kernel density estimation, we set the nearest neighbor search parameter to be $\epsilon = 2.6$.

We apply the A-STBB search for multiclass multiple-instance action detection. In Fig. 6, we show the precision and recall curves for three action classes, by increasing the detection threshold D_t from 3 to 30. We also compare a few different values of $s(d_\emptyset)$, including $s(d_\emptyset) = -1 \times 10^{-5}$, $-2 \times 10^{-5} - 3 \times 10^{-5}$, -4×10^{-5}, and -6×10^{-5}. For different action classes, the optimal parameters of D_t and $s(d_\emptyset)$ may be different. Among the three classes of actions, hand waving and boxing provide better performance, where both precision and recall rates are higher than or close to

TABLE 5
Cross-Data Set Training and Testing of Two-Hand Waving Detection

positive training	hand-waving 16 persons (KTH)
negative training	walking 16 persons (KTH) + 1 indoor seq.
testing	two-hand waving + jumping jacks (CMU)

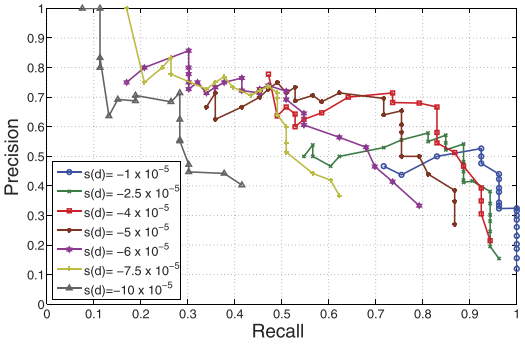


Fig. 4. Performance of two-hand wave detection in the CMU data set, with different choices of $s(d_\emptyset)$. See text for the measurement of precision and recall scores.

Fig. 5. Examples of multiclass multiple-instance action detection. Each row shows an action class: hand waving (first row), hand clapping (second row), and boxing (third row). Each image is a sample frame from the action video. The first column shows the training videos from the KTH data set. The second to the fifth columns show some detection results. The highlighted bounding boxes correspond to the detected spatial window W^* and we apply different colors to distinguish different action classes: clapping (turquoise), waving (magenta), and boxing (yellow). The final column shows miss detection examples, where the bounding boxes are ground truth labeling: clapping (red), waving (green), and boxing (blue). The whole data set is accessible at http://research.microsoft.com/en-us/downloads/fbf24c35-a93e-4d22-a5fe-bc08f1c3315e/.

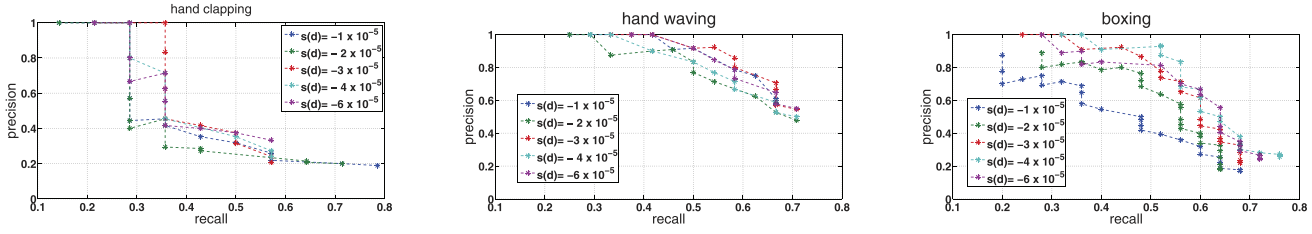


Fig. 6. Performance of three-class action detection. Left: hand clapping; middle: hand waving; right: boxing.

Fig. 7. Detection of multiple actions in the same scene. Top: Detection of two boxings (yellow). Bottom: Detection of one boxing (yellow) and one hand waving (purple).

65 percent. However, hand clapping is more challenging, especially if the clapping movement is subtle. Hand clapping is also easily confused with the hand waving action. For all of the three classes, most missing detections are due to the small spatial scales, bad lighting conditions, or crowded scenes. In Fig. 7, we show the detection results of multiple actions in the same scene.

The computational cost of multiclass multiple-instance action detection contains three parts: 1) extraction of STIPs, 2) kernel density estimation and calculation of voting scores for each class, and 3) search for qualified subvolumes for each class. First, for videos at resolution 320×240, the speed of STIP detection is 2-4 frames per second using the binary code provided by [19]. Second, the major cost of obtaining the voting score $s_c(d)$ comes from the ϵ-NN search in density estimation. By using the E2LSH code for efficient NN search, the query time of each STIP is 40-50 milliseconds with $\epsilon = 2.6$ and retrieve probability $p = 0.9$. However, if performing exhaustive search of ϵ-NN, the query time of each STIP increases to 130 milliseconds. If parallel search can be performed using a four-core CPU, the estimated query time can achieve around 12 milliseconds per STIP. As each frame contains 20-40 STIPs on average, the processing time can achieve 2-4 frames per second. Finally, to evaluate the CPU cost of subvolume search through A-STBB, we record the computational cost for each of the 16 testing sequences in Table 6. The test is performed on a four-core CPU desktop.

In Table 6, we notice that the computational cost of A-STBB depends on the video sequence, including the number of STIPs and the number of action instances. On average, the A-STBB search can achieve 4-5 frames per second using a four-core CPU. The search tends to be slower for video sequences with a larger number of moving objects in the background since a lot of STIPs will be extracted. On the other hand, if a video sequence does not contain any target actions, the search will finish quickly thanks to the early termination strategy.

TABLE 6
The CPU Cost of the A-STBB Search on the MSR Action Data Set for Multiple-Instance Detection of Three Classes of Actions

video #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
length #min	0.75	0.73	0.95	0.57	1	0.82	0.65	0.85	0.85	1.05	0.90	0.63	0.53	1.27	0.72	0.88
cost #min	0.25	2	3	1.5	1.5	6	0.75	0.25	2.75	0.5	1.25	0.25	0.25	10	3.75	10.25

The algorithm is implemented in C++ and runs on a four-core CPU desktop. Only the A-STBB search cost is listed, while the STIP extractions and score calculation are not included. The measurement of CPU cost is rounded into $\frac{1}{4}$ minutes.

6.4 Search Complexity and Efficiency Comparison

6.4.1 Comparison between STBB Search and Conventional BB Search

To validate the efficiency gain of our STBB search, we compare our STBB (Algorithm 2) to the conventional branch-and-bound (Algorithm 1) by searching the MVI-142a sequence in the CMU action data set [4]. The max subvolume is of size $43 \times 32 \times 112$. The input video $\mathcal{V}$ is of size $120 \times 160 \times 141$, a temporal segment from MVI-142a. We intentionally choose such a target video of a short length, such that the sizes of its three dimensions are balanced. This gives a fair comparison to the conventional branch-and-bound because the longer the video length t, the less efficient the conventional branch-and-bound will be.

The left figure in Fig. 8 shows that our proposed method converges much faster than the conventional branch-and-bound. In terms of the number of branches, our method converges after 10,302 branches, an order of magnitude faster than the conventional branch-and-bound which needs 103,202 branches before convergence. This validates that the upper bound proposed in Theorem 1 is tighter than that of the conventional method. In Algorithm 1, the upper bounded estimation $\hat{f}(\mathbb{W})$ decreases slowly when the current state converges to the optimal solution. In comparison, the convergence of the upper bound in our proposed method (Algorithm 2) is much faster. For example, after 2,000 branches, our method reaches a very good solution $f(V) = 15.78$, which is close to the optimal one $f(V^*) = 16.21$. On the other hand, after 2,000 branches, the largest upper bound given by the conventional branch-and-bound is still as large as $\hat{f}(\mathbb{W}) = 24.06$.

As mentioned earlier, another advantage of our method is that it keeps track of the current best solution. A new subvolume is pushed into the queue only when its upper bound is better than the current best solution. In comparison, the method proposed in [1] needs to push every middle state into the priority queue, as there is no record of the current best solution. In Fig. 8, we also compare the required size of the priority queue between our method and the conventional branch-and-bound. The size of the priority queue in our method is well-controlled and is much smaller. In our method, during the branch-and-bound process, the size of the priority queue decreases after a peak value. However, for the conventional branch-and-bound, the size of priority queue always increases, almost linearly to the number of branches. Since each insertion or extraction operation of priority queue is $O(logn)$ for a queue of size n, the size of the priority queue affects both the computational and memory costs. It is especially important to limit a queue to a moderate size for the video space search because it can generate a much larger number of candidates than the spatial image case.

6.4.2 Evaluation of the Accelerated STBB Search

To evaluate the efficiency of the proposed A-STBB algorithm in Algorithm 5 for branch-and-bound, we select the first five video sequences of the two-hand waving action in the CMU action data set [4]. Each sequence contains one two-hand waving action. The algorithm searches for the subvolume of high detection score such that it covers the action. Compared with original STBB which targets the optimal subvolume with maximum score, A-STBB finds approximate optimal subvolume but at a faster search speed. For the original STBB search, we do not need to specify the detection threshold as it returns the subvolume with maximum detection score. For the A-STBB search, the detection threshold is selected as $D_t = 10$. Under this detection score, the first subvolume returned by A-STBB is compared with the optimal subvolume returned from the original STBB algorithm. As the detection score of all of the five target subvolumes is higher than $D_t = 10$, such a

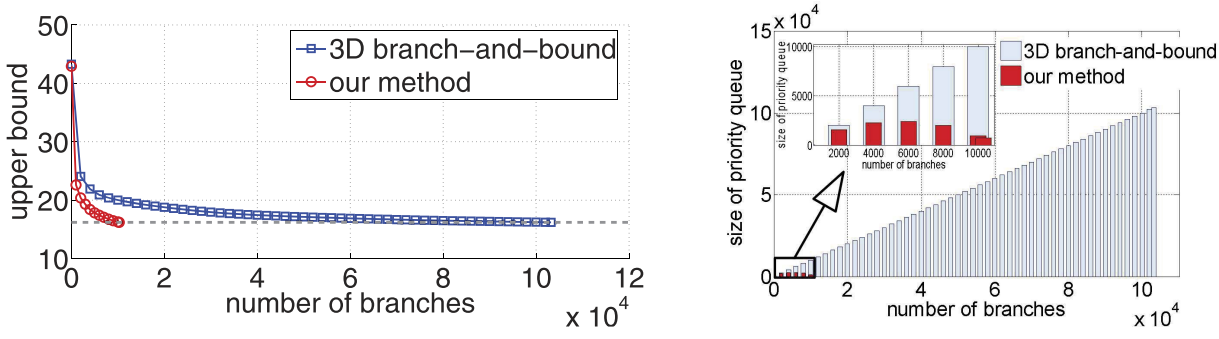


Fig. 8. Comparison between our STBB search (Algorithm 2) and conventional BB search in three-dimensional spatiotemporal space (Algorithm 1). Left: Comparison of the convergence speed. Both methods can find the optimal subvolume with detection score $f(V^*) = 16.21$. The red-circle curve and the blue-square curve show the convergence of the upper bound estimation of STBB method and conventional BB. Right: comparison between the length of the priority queue in branch-and-bound search.

TABLE 7
Comparison between STBB in Algorithm 2
and Accelerated STBB in Algorithm 5

video	W^*	T^*	score $f(V^*)$	# of branches
V1: STBB (Alg. 2)	55 97 23 54	442 553	16.21	206933
V1: A-STBB (Alg. 5)	55 97 23 52	442 546	15.97	4549
V2: STBB (Alg. 2)	61 122 20 38	673 858	37.39	67281
V2: A-STBB (Alg. 5)	60 122 20 39	673 858	37.36	4668
V3: STBB (Alg. 2)	72 118 22 71	11 700	89.01	71594
V3: A-STBB (Alg. 5)	82 114 23 73	10 705	85.21	2275
V4: STBB (Alg. 2)	73 112 23 77	420 1083	73.42	63076
V4: A-STBB (Alg. 5)	77 108 23 78	420 1083	70.93	2363
V5: STBB (Alg. 2)	18 144 7 114	418 451	46.50	315770
V5: A-STBB (Alg. 5)	41 151 7 114	419 451	45.36	133027

$V^* = [W^*, T^*]$ is the detected subvolume through branch-and-bound search. The four parameters of W^* determine the spatial location and the two parameters of T^* determine the start and end frames.

Fig. 9. Detection of person running in surveillance video using A-STBB search. Each row shows a detection instance. Top: A man runs in the corridor. Bottom: A child runs in the corridor.

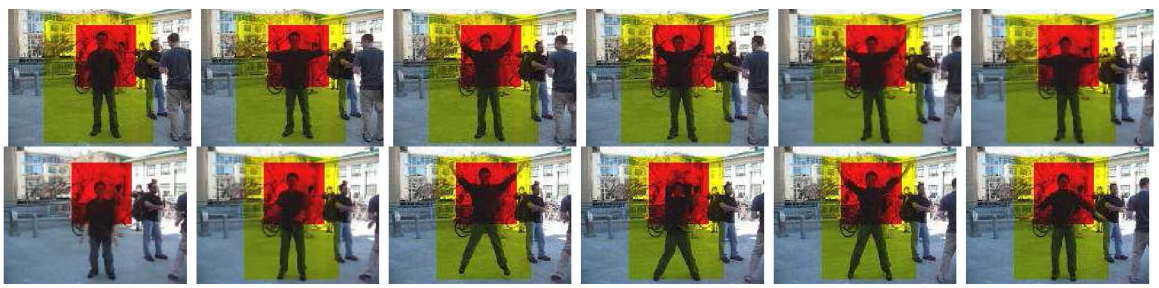


Fig. 10. A detection example with performing speed and style variations. The yellow bounding box is the ground truth label of the whole human body action and the red bounding box is our detection of two-hand waving.

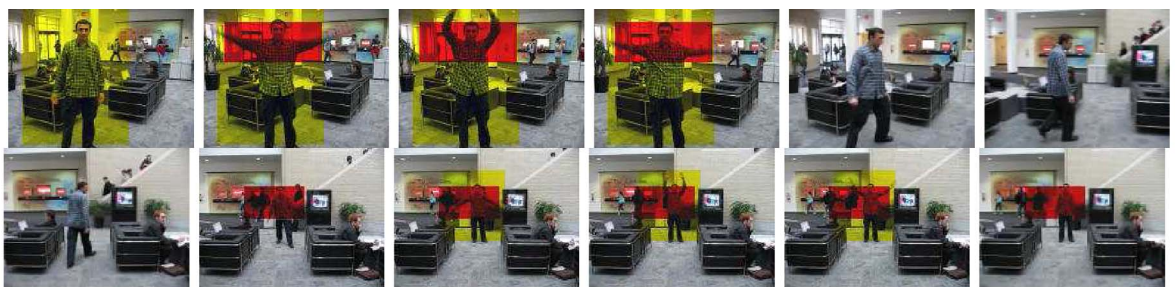


Fig. 11. A detection example with large spatial scale changes.

detection threshold will not affect the efficiency comparison between the original STBB and A-STBB search algorithms.

The comparison between the A-STBB in Algorithm 5 with the original STBB in Algorithm 2 is presented in Table 7. W^* is the spatial window containing left, right, top, and bottom parameters. T^* includes the start and end frames. Table 7 shows that detection results of A-STBB in Algorithm 5 are close to those of STBB in Algorithm 2. Both algorithms provide similar detections results, in terms of detection scores, locations, and sizes of the subvolumes.

Fig. 12. A detection example with cluttered and moving background, as well as severe partial occlusions.

Fig. 13. A false detection example caused by two individual hand wavings from two different persons.

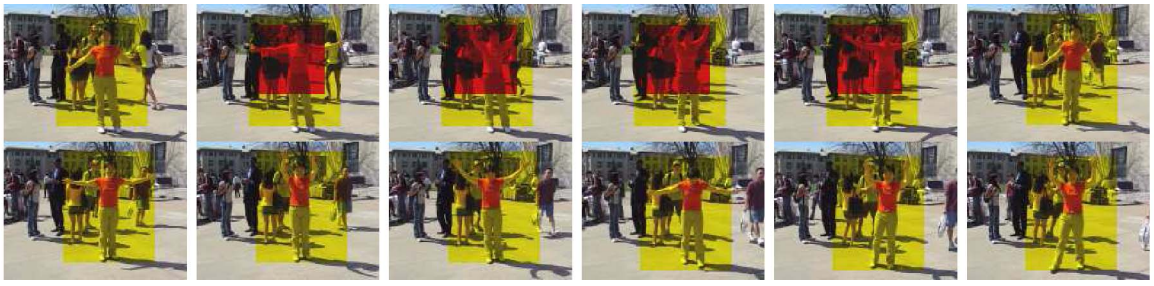


Fig. 14. An example of missed detection. Although the action is detected in the first few frames, the whole action instance is still treated as a missed detection due to limited overlap with the ground truth labeling.

However, the number of branches in STBB can be up to 20 times more than that of A-STBB. It validates the efficiency of the proposed A-STBB. Moreover, if a video sequence does not contain any target actions, A-STBB can be even more efficient by terminating the search process at a very early stage, and returns with nonvalid detection found.

To show the performance of our A-STBB search in real video surveillance scenario, we also test on two sequences from the TRECVID 2008 event detection data set, which is a very challenging one for video surveillance [57]. The videos are captured by the real surveillance cameras in an airport. Although there are a lot of actions defined in TRECVID 2008, we only use the running action since it is similar to those in the KTH data set. We use 16 running people from the KTH data set for positive training and 16 walking people for negative training. The two selected testing sequences are taken by the second camera (five cameras in total). The video resolution is 180×144, with 25 frames per second. Fig. 9 shows the detection results where each row corresponds to a video sequence.

7 Conclusion

Similarly to the sliding-window-based search for object detection, detection of actions is to search for qualified subvolumes in the volumetric video space. To address the search complexity of this new formulation of action detection, a novel STBB search solution is proposed. We extend the previous branch-and-bound solution from searching spatial image patterns to searching spatiotemporal video patterns. By tightening the upper bound and reducing the parameter space from six dimensions to four dimensions, the STBB search is significantly more efficient in searching video patterns. For multiclass multiple-instance action detection, the A-STBB search validates its efficiency and effectiveness on the CMU and MSR data sets.

In order to tolerate the intraclass action variations, we propose a discriminative pattern matching method, called NBMIM, for action classification. Compared with conventional template-based pattern matching, instead of using a single template for pattern matching, we apply both positive and negative templates for discriminative matching. Despite its simplicity, the proposed NBMIM approach can well distinguish one action class from other classes, as well as the background class. Although such a naive Bayes assumption ignores the spatiotemporal dependency among interest point features, it leads to better tolerance of intrapattern variations. Our action detection method does not rely on the detection and tracking of a person. It can handle scale changes well, performing speed and style variations of actions, cluttered and dynamic backgrounds,

even partial occlusions. The future work includes extending the STBB search to find subvolumes of more flexible shapes, i.e., nonrectangle shapes, and relaxing the naive Bayes assumption in discriminative matching to consider the spatiotemporal dependency among the interest points.

ACKNOWLEDGMENTS

This work was supported in part by the Nanyang Assistant Professorship to Dr. Junsong Yuan, US National Science Foundation grant IIS-0347877, IIS-0916607, and the US Army Research Laboratory and the US Army Research Office under grant ARO W911NF-08-1-0504. The authors thank Dr. Yan Ke, Dr. Cha Zhang, and Dr. Zhengyou Zhang for helpful discussions, and Liangliang Cao for the help on the experiments of the TRECVID data set.

REFERENCES

[1] C.H. Lampert, M.B. Blaschko, and T. Hofmann, "Beyond Sliding Windows: Object Localization by Efficient Subwindow Search," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[2] P. Viola and M.J. Jones, "Robust Real-Time Face Detection," *Int'l J. Computer Vision,* vol. 57, no. 2, pp. 137-154, 2004.

[3] M.B. Blaschko and C.H. Lampert, "Learning to Localize Objects with Structured Output Regression," *Proc. European Conf. Computer Vision,* pp. 2-15, 2008.

[4] Y. Ke, R. Sukthankar, and M. Hebert, "Event Detection in Crowded Videos," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1-8, 2007.

[5] A.F. Bobick and J.W. Davis, "The Recognition of Human Movement Using Temporal Templates," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 3, pp. 257-267, Mar. 2001.

[6] I. Laptev, "On Space-Time Interest Points," *Int'l J. Computer Vision,* vol. 64, nos. 2-3, pp. 107-123, 2005.

[7] C. Rao, A. Yilmaz, and M. Shah, "View-Invariant Representation and Recognition of Actions," *Int'l J. Computer Vision,* vol. 50, no. 2, pp. 203-226, 2002.

[8] N. Nguyen, D. Phung, S. Venkatesh, and H. Bui, "Learning and Detecting Activities from Movement Trajectories Using the Hierarchical Hidden Markov Models," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, 2005.

[9] S. Ali, A. Basharat, and M. Shah, "Chaotic Invariants for Human Action Recognition," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1-8, 2007.

[10] V. Parameswaran and R. Chellappa, "View Invariance for Human Action Recognition," *Int'l J. Computer Vision,* vol. 66, no. 1, pp. 83-101, 2006.

[11] J. Sun, X. Wu, S. Yan, L. Cheong, T. Chua, and J. Li, "Hierarchical Spatio-Temporal Context Modeling for Action Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2004-2011, 2009.

[12] F. Lv and R. Nevatia, "Single View Human Action Recognition Using Key Pose Matching and Viterbi Path Searching," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2007.

[13] D. Weinland and E. Boyer, "Action Recognition Using Exemplar-Based Embedding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-7, 2008.

[14] A. Yilmaz and M. Shah, "Actions as Objects: A Novel Action Representation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2005.

[15] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *Proc. IEEE Int'l Conf. Computer Vision,* vol. 2, pp. 1395-1402, 2005.

[16] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior Recognition via Sparse Spatio-Temporal Features," *Proc. IEEE Int'l Workshop Visual Surveillance Performance Evaluation Tracking Surveillance,* pp. 65-72, 2005.

[17] J. Liu, J. Luo, and M. Shah, "Recognizing Realistic Actions from Videos 'in the Wild'," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1996-2003, 2009.

[18] J. Liu and M. Shah, "Learning Human Actions via Information Maximization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[19] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning Realistic Human Actions from Movies," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[20] K. Jia and D.-Y. Yeung, "Human Action Recognition Using Local Spatio-Temporal Discriminant Embedding," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[21] J. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words," *Int'l J. Computer Vision,* vol. 79, no. 3, pp. 299-318, 2008.

[22] I. Laptev, B. Caputo, C. Schu, and T. Lindeberg, "Local Velocity-Adapted Motion Events for Spatio-Temporal Recognition," *Computer Vision and Image Understanding,* vol. 109, no. 1, pp. 207-229, 2007.

[23] P.S. Dhillon, S. Nowozin, and C.H. Lampert, "Combining Appearance and Motion for Human Action Classification in Videos," technical report, Max-Planck-Inst. for Biological Cybernetics, 2008.

[24] P. Scovanner, S. Ali, and M. Shah, "A 3-Dimensional Sift Descriptor and Its Application to Action Recognition," *Proc. ACM Multimedia,* 2007.

[25] Y. Wang and G. Mori, "Human Action Recognition by Semi-Latent Topic Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 10, pp. 1762-1774, Oct. 2009.

[26] S. Ali and M. Shah, "Human Action Recognition in Videos Using Kinematic Features and Multiple Instance Learning," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 32, no. 2, pp. 288-303, Feb. 2010.

[27] A. Fathi and G. Mori, "Action Recognition by Learning Mid-Level Motion Features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[28] A.A. Efros, A.C. Berg, G. Mori, and J. Malik, "Recognizing Action at a Distance," *Proc. IEEE Int'l Conf. Computer Vision,* vol. 2, 2003.

[29] Z. Zhang, Y. Hu, S. Chan, and L.-T. Chia, "Motion Context: A New Representation for Human Action Recognition," *Proc. European Conf. Computer Vision,* pp. 817-829, 2008.

[30] P. Natarajan and R. Nevatia, "View and Scale Invariant Action Recognition Using Multiview Shape-Flow Models," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[31] S.N. Vitaladevuni, V. Kellokumpu, and L.S. Davis, "Action Recognition Using Ballistic Dynamics," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[32] Z. Lin, Z. Jiang, and L.S. Davis, "Recognizing Actions by Shape-Motion Prototype Trees," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 444-451, 2009.

[33] J. Liu, S. Ali, and M. Shah, "Recognizing Human Actions Using Multiple Features," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.

[34] D. Han, L. Bo, and C. Sminchisescu, "Selection and Context for Action Recognition," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1933-1940, 2009.

[35] S.-F. Wong, T.-K. Kim, and R. Cipolla, "Learning Motion Categories Using Both Semantic and Structural Information," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-6, 2007.

[36] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local SVM Approach," *Proc. 17th Int'l Conf. Pattern Recognition,* vol. 3, pp. 32-36, Aug. 2004.

[37] K.K. Reddy, J. Liu, and M. Shah, "Incremental Action Recognition Using Feature-Tree," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1010-1017, 2009.

[38] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient Visual Event Detection Using Volumetric Features," *Proc. IEEE Int'l Conf. Computer Vision,* vol. 1, pp. 166-173, 2005.

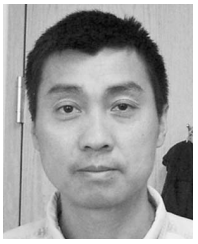
[39] E. Shechtman and M. Irani, "Space-Time Behavior Based Correlation," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 405-412, 2005.

[40] J. Yuan, Z. Liu, and Y. Wu, "Discriminative Subvolume Search for Efficient Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 2442-2449, 2009.

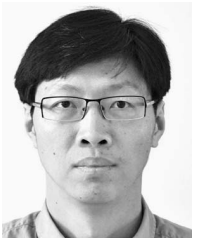
[41] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, and T.S. Huang, "Action Detection in Complex Scenes with Spatial and Temporal Ambiguities," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 128-135, 2009.

[42] J. Yuan and Z. Liu, "TechWare: Video-Based Human Action Detection Sources," *IEEE Signal Processing Magazine,* vol. 27, no. 5, pp. 136-139, Sept. 2010.
[43] C. Yeo, P. Ahammad, K. Ramchandran, and S.S. Sastry, "High-Speed Action Recognition and Localization in Compressed Domain Videos," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 8, pp. 1006-1015, Aug. 2008.
[44] J. Yuan, Z. Liu, Y. Wu, and Z. Zhang, "Speeding Up Spatio-Temporal Sliding-Window Search for Efficient Event Detection in Crowded Videos," *Proc. ACM Multimedia Workshop Events in Multimedia,* 2009.
[45] M.D. Rodriguez, J. Ahmed, and M. Shah, "Action MACH: A Spatio-Temporal Maximum Average Correlation Height Filter for Action Recognition," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.
[46] D. Weinland, R. Ronfard, and E. Boyer, "Free Viewpoint Action Recognition Using Motion History Volumes," *Computer Vision and Image Understanding,* vol. 104, nos. 2-3, pp. 207-229, 2006.
[47] H. Jiang, M.S. Drew, and Z.-N. Li, "Successive Convex Matching for Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1646-1653, 2006.
[48] W. Li, Z. Zhang, and Z. Liu, "Expandable Data-Driven Graphical Modeling of Human Actions Based on Salient Postures," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 11, pp. 1499-1510, Nov. 2008.
[49] O. Duchenne, I. Laptev, J. Sivic, F. Bach, and J. Ponce, "Automatic Annotation of Human Actions in Videos," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 1491-1498, Sept.-Oct. 2009.
[50] I. Laptev and P. Pérez, "Retrieving Actions in Movies," *Proc. IEEE Int'l Conf. Computer Vision,* 2007.
[51] L. Cao, Z. Liu, and T.S. Huang, "Cross-Data Set Action Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* 2010.
[52] O. Boiman, E. Shechtman, and M. Irani, "In Defense of Nearest-Neighbor Based Image Classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1-8, 2008.
[53] C.H. Lampert, "Detecting Objects in Large Image Collections and Videos by Efficient Subimage Retrieval," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 987-994, 2009.
[54] P.C. Woodland and D. Povey, "Large Scale Discriminative Training of Hidden Markov Models for Speech Recognition," *Computer Speech and Language,* vol. 16, no. 1, pp. 25-47, 2002.
[55] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-Sensitive Hashing Scheme Based on P-Stable Distribution," *Proc. 20th Ann. Symp. Computational Geometry,* pp. 253-262, 2004.
[56] J. Bentley, "Programming Pearls," *Algorithm Design Techniques,* vol. 27, no. 9, pp. 865-871, 1984.
[57] M. Dikmen et al., "Surveillance Event Detection," *Proc. Video Evaluation Workshop,* 2008.

Junsong Yuan received the BEng degree in communication engineering from Huazhong University of Science and Technology, Wuhan, P.R. China, and the MEng and PhD degrees from the National University of Singapore, and Northwestern University, Illinois, respectively. Before that, he graduated from the special program for the gifted young at Huazhong University of Science and Technology. During the summer 2008, 2007, and 2006, he was a research intern with the Communication and Collaboration Systems group, Microsoft Research, Redmond, Washington, Kodak Research Laboratories, Rochester, New York, and Motorola Laboratories, Schaumburg, Illinois, respectively. From 2003 to 2004, he was a research scholar at the Institute for Infocomm Research, Singapore. In September 2009, he joined Nanyang Technological University as a Nanyang assistant professor. His current research interests include computer vision, image and video data mining and content analysis, multimedia search, etc. He was a recipient of the Doctoral Spotlight award from the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '09), a recipient of the Nanyang Assistant Professorship from Nanyang Technological University, and received the Outstanding PhD Thesis award from the Electrical Engineering and Computer Science Department at Northwestern University. He is a member of the IEEE and the ACM.



Zicheng Liu received the BS degree in mathematics from Huazhong Normal University, China, the MS degree in operational research from the Institute of Applied Mathematics, and the PhD degree in computer science from Princeton University. He is a senior researcher at Microsoft Research, Redmond, Washington. He has worked on a variety of topics, including combinatorial optimization, linked figure animation, and microphone array signal processing. His current research interests include activity recognition, face modeling and animation, and multimedia collaboration. Before joining Microsoft Research, he worked at Silicon Graphics as a member of the technical staff for two years, where he developed a trimmed NURBS tessellator which was shipped in both OpenGL and OpenGL-Optimizer products. He has published more than 70 papers in peer-reviewed international journals and conferences, and holds more than 40 granted patents. He has served on the technical committees for many international conferences. He was the cochair of the 2003 ICCV Workshop on Multimedia Technologies in E-Learning and Collaboration, the technical cochair of the 2006 IEEE International Workshop on Multimedia Signal Processing, and the technical cochair of the 2010 International Conference on Multimedia and Expo. He is an associate editor of *Machine Vision and Applications*, and a senior member of the IEEE.



Ying Wu received the BS degree from Huazhong University of Science and Technology, Wuhan, China, in 1994, the MS degree from Tsinghua University, Beijing, China, in 1997, and the PhD degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC) in 2001. From 1997 to 2001, he was a research assistant at the Beckman Institute for Advanced Science and Technology at UIUC. During summer 1999 and 2000, he was a research intern with Microsoft Research, Redmond, Washington. In 2001, he joined the Department of Electrical and Computer Engineering at Northwestern University, Evanston, Illinois, as an assistant professor. He is currently an associate professor of electrical engineering and computer science at Northwestern University. His current research interests include computer vision, image and video analysis, pattern recognition, machine learning, multimedia data mining, and human-computer interaction. He serves as an associate editor for the *IEEE Transactions on Image Processing*, *SPIE Journal of Electronic Imaging*, and *IAPR Journal of Machine Vision and Applications*. He received the Robert T. Chien award at UIUC in 2001, and the US National Science Foundation (NSF) CAREER award in 2003. He is a senior member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.