Fast Action Detection via Discriminative Random Forest Voting and Top-K Subvolume Search

IEEE Transactions on Multimedia (JCR: 1.790)

Gang Yu (Nanyang TU, Singapore), N. A. Goussies (U. B. Aires), Junsong Yuan (Nanyang TU, Singapore), Zicheng Liu (Microsoft, USA)

Ion Marqués, Grupo de Inteligencia Computacional, UPV/EHU

27-01-2012

Introduction

- **Problem**: Multiclass action detection in complex video scenes.
  - Action detection: Identify which type of action occurs and specify when (spatial location in each frame) and when (temporal location).
- **Proposal**: A new Random Forest-based template matching method. It's accurate and much faster than many other algorithms.

Introduction

- The proposed method summarized:
  - Each video sequence is characterized by a collection of spatio-temporal interest points (STIPs).
  - Training: A Random Forest (RF) is built to model the distribution of STIPs.
  - Testing:
    - Each point is run through the RF, providing an individual voting score toward each action type.
    - Video is downsampled. We find the spatio-temporal video subvolume with the maximum total mututal information score (via a fast top-K subvolume search algorithm).

Introduction

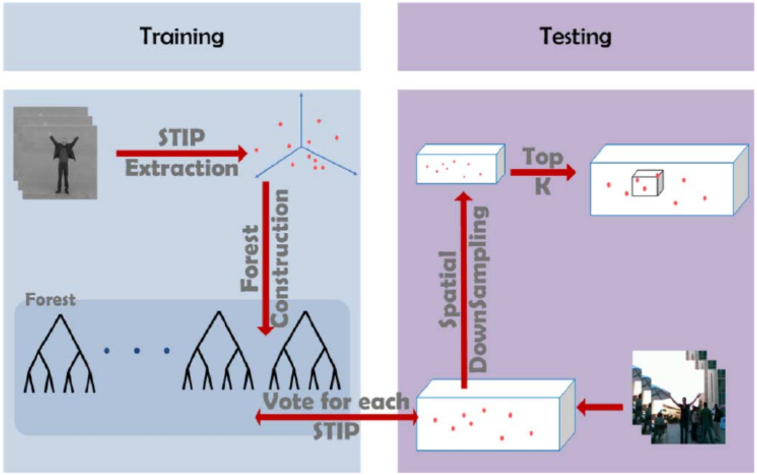
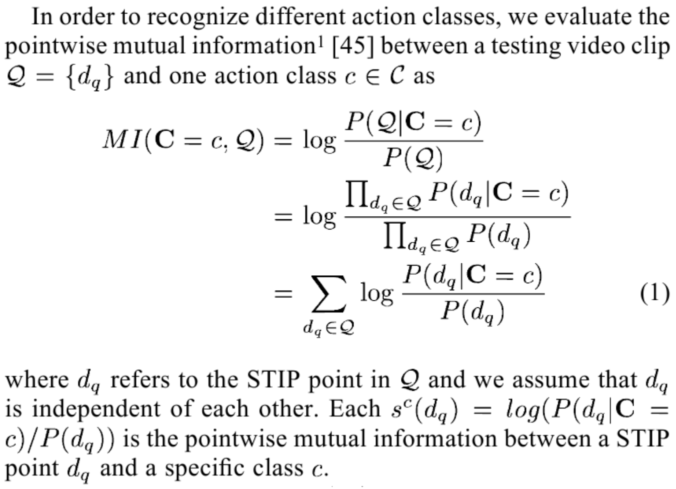


Fig. 1. Overview of our random forest-based video subvolume search.

Multiclass Action Recognition

- An action is represented as a collection of STIPs.
- A STIP is $d \in \mathbb{R}^N$, a N-dimensional feature vector. Denote class label set as $\mathcal{C} = \{1, 2, \ldots, C\}$



In order to recognize different action classes, we evaluate the pointwise mutual information[1] [45] between a testing video clip $\mathcal{Q} = \{d_q\}$ and one action class $c \in \mathcal{C}$ as

$$MI(\mathbf{C} = c, \mathcal{Q}) = \log \frac{P(\mathcal{Q}|\mathbf{C} = c)}{P(\mathcal{Q})} = \log \frac{\prod_{d_q \in \mathcal{Q}} P(d_q|\mathbf{C} = c)}{\prod_{d_q \in \mathcal{Q}} P(d_q)} = \sum_{d_q \in \mathcal{Q}} \log \frac{P(d_q|\mathbf{C} = c)}{P(d_q)} \qquad (1)$$

where d_q refers to the STIP point in $\mathcal{Q}$ and we assume that d_q is independent of each other. Each $s^c(d_q) = log(P(d_q|\mathbf{C} = c)/P(d_q))$ is the pointwise mutual information between a STIP point d_q and a specific class c.

Multiclass Action Recognition

- The voting score $s^c(d_q)$ is formulated as:

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

- $P(\mathbf{C} = c)$ is a prior, so the problem is to compute the posterior $P(\mathbf{C} = c \mid d_q)$.
- They approximate this probability with a Random Forest.

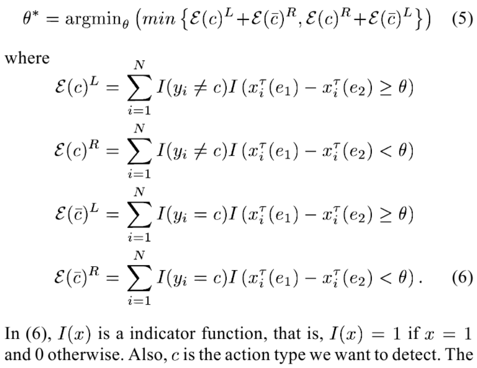
Multiclass Action Recognition: Random Forest

- **How they generate trees:**

1. We have N STIPs in the training set, defined as $\{(x_i, y_i), i = 1, 2, \ldots, N\}$, where $x_i = (x_i^1, x_i^2); x_i^1 \in R^{72}$ and $x_i^2 \in R^{90}$ refer to the HoG (Histogram of Gradient) and HoF (Histogram of Flow). $y_i \in \mathcal{C}$ is the label of the STIP.
2. We generate $\tau \in \{1, 2\}$ to indicate the feature type to use.
3. Generate the dimension index e_1 and e_2 and compute feature difference $D_i = x_i^\tau(e_1) - x_i^\tau(e_2), i = 1, 2, \cdots, N$.

Multiclass Action Recognition: Random Forest

4. For each x_i we assign it to the left child if $x_i^\tau(e_1) - x_i^\tau(e_2) \geq \theta$ or to the right if $x_i^\tau(e_1) - x_i^\tau(e_2) < \theta.$. The threshold is selected by minimizing binary classification error



$$\theta^* = \operatorname{argmin}_\theta \left(min \left\{ \mathcal{E}(c)^L + \mathcal{E}(\bar{c})^R, \mathcal{E}(c)^R + \mathcal{E}(\bar{c})^L \right\} \right) \quad (5)$$

where

$$\mathcal{E}(c)^L = \sum_{i=1}^{N} I(y_i \neq c) I \left(x_i^\tau(e_1) - x_i^\tau(e_2) \geq \theta \right)$$

$$\mathcal{E}(c)^R = \sum_{i=1}^{N} I(y_i \neq c) I \left(x_i^\tau(e_1) - x_i^\tau(e_2) < \theta \right)$$

$$\mathcal{E}(\bar{c})^L = \sum_{i=1}^{N} I(y_i = c) I \left(x_i^\tau(e_1) - x_i^\tau(e_2) \geq \theta \right)$$

$$\mathcal{E}(\bar{c})^R = \sum_{i=1}^{N} I(y_i = c) I \left(x_i^\tau(e_1) - x_i^\tau(e_2) < \theta \right). \quad (6)$$

In (6), $I(x)$ is a indicator function, that is, $I(x) = 1$ if $x = 1$ and 0 otherwise. Also, c is the action type we want to detect. The

Multiclass Action Recognition: Random Forest

- They perform this tree construction process 200 times and select the one with smallest classification error.
- Now, **how to compute $P(\mathbf{C} = c \mid d_q)$ with a random forest?**
- Suppose we have M trees. For a tree T_i, a STIP d_q falls in a leaf with N_i^+ positive samples and N_i^- negative samples. The psoterior distribution of d_q can be approximated by the average density of the M nodes in M different trees as

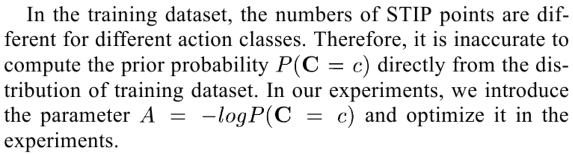
$$P(\mathbf{C} = c|d_q) \approx \frac{1}{M} \sum_{i=1}^{M} \frac{N_i^+}{N_i^+ + N_i^-}.$$

Multiclass Action Recognition: Random Forest

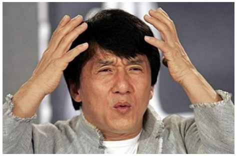
- Then we reformulate the voting score:

$$\begin{aligned} S^c(d_q) &= \log P(\mathbf{C} = c | d_q) - \log P(\mathbf{C} = c) \\ &= \log \frac{1}{M} \sum_{i=1}^{M} \frac{N_i^+}{N_i^+ + N_i^-} - \log P(\mathbf{C} = c). \end{aligned}$$

Multiclass Action Recognition: Random Forest



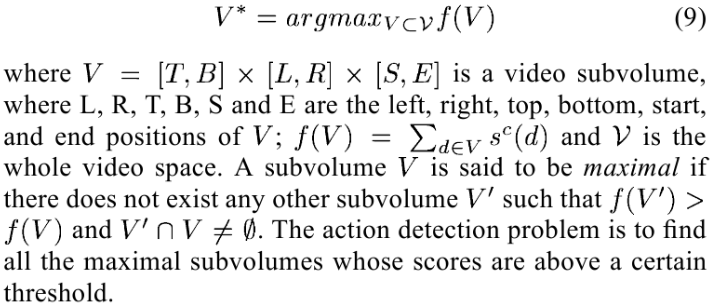
In the training dataset, the numbers of STIP points are different for different action classes. Therefore, it is inaccurate to compute the prior probability $P(\mathbf{C} = c)$ directly from the distribution of training dataset. In our experiments, we introduce the parameter $A = -logP(\mathbf{C} = c)$ and optimize it in the experiments.



- Advantages of Random forest:
  - Is way faster than locality sensitive hash nearest network approach,
  - the label information is integrated in the trees,
  - and the construction of random forest is flexible, as it is easy to combine other types of feature descriptors and spatial information of STIPs.

Action Detection and Localization

- The purpose of action detection is to find a subvolume V with the maximum similarity to the predefined action type. With each STIP being associated with an individual score $s^c(d_q)$, our goal is to find the video subvolume with the maximum score



$$V^* = argmax_{V \subset \mathcal{V}} f(V) \tag{9}$$

where $V = [T, B] \times [L, R] \times [S, E]$ is a video subvolume, where L, R, T, B, S and E are the left, right, top, bottom, start, and end positions of V; $f(V) = \sum_{d \in V} s^c(d)$ and $\mathcal{V}$ is the whole video space. A subvolume V is said to be *maximal* if there does not exist any other subvolume V' such that $f(V') > f(V)$ and $V' \cap V \neq \emptyset$. The action detection problem is to find all the maximal subvolumes whose scores are above a certain threshold.

Action Detection and Localization

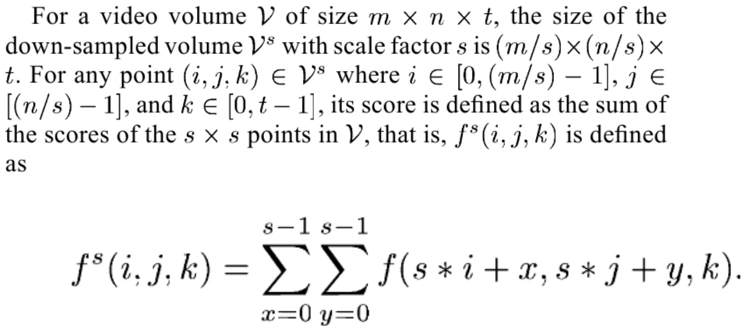
- our goal is to fins spatial windows W^* that maximizes

$$F(W) = \max_{T \subseteq \mathbb{T}} f(W \times T)$$

where W is a spatial window and T a temporal segment. This way we separate the temporal parameter. the complexity is linear in time, which is usually the largest of dimensions. This, however, is not very effective with high resolution videos. To quicken the process the authors propose tow things: Donsampling and Top-k search algorithm.

Action Detection and Localization: Downsampling.

- To handle high-resolution videos, the technique is to spatially down-sample the video space by a factor before the branch-and-bound search. Note that the interest point detection, descriptor extraction, and the scores are all done in the original video sequence.

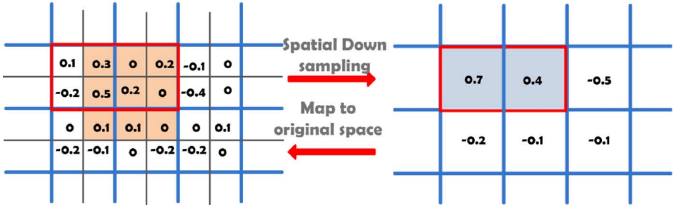


For a video volume $\mathcal{V}$ of size $m \times n \times t$, the size of the down-sampled volume $\mathcal{V}^s$ with scale factor s is $(m/s) \times (n/s) \times t$. For any point $(i, j, k) \in \mathcal{V}^s$ where $i \in [0, (m/s) - 1]$, $j \in [(n/s) - 1]$, and $k \in [0, t - 1]$, its score is defined as the sum of the scores of the $s \times s$ points in $\mathcal{V}$, that is, $f^s(i, j, k)$ is defined as

$$f^s(i, j, k) = \sum_{x=0}^{s-1} \sum_{y=0}^{s-1} f(s * i + x, s * j + y, k).$$

Action Detection and Localization: Downsampling.

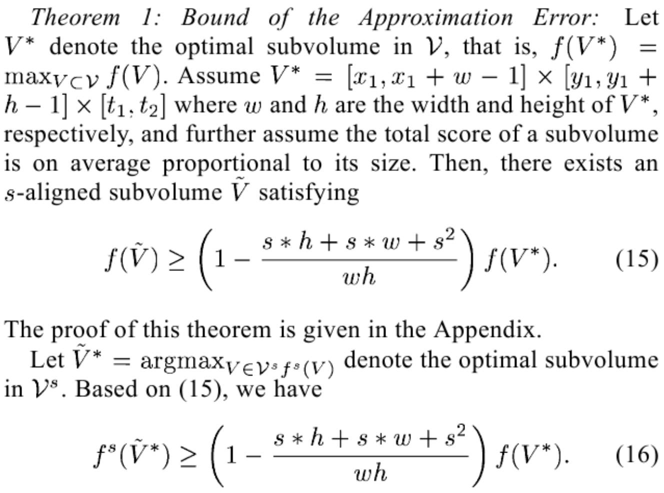
, graphically, these are two score images:



the red rectangle is the solution of the downsampled version. Is worse than the one of the original space (orange-ish rectangle): $(f^s(\tilde{V}^*) = 1.1 < f(V^*) = 1.4)$.

Action Detection and Localization: Downsampling.

- To further analyze that approximation error, the authors introduce a theorem:



Theorem 1: Bound of the Approximation Error: Let V^* denote the optimal subvolume in $\mathcal{V}$, that is, $f(V^*) = \max_{V \subset \mathcal{V}} f(V)$. Assume $V^* = [x_1, x_1 + w - 1] \times [y_1, y_1 + h - 1] \times [t_1, t_2]$ where w and h are the width and height of V^*, respectively, and further assume the total score of a subvolume is on average proportional to its size. Then, there exists an s-aligned subvolume $\tilde{V}$ satisfying

$$f(\tilde{V}) \geq \left(1 - \frac{s * h + s * w + s^2}{wh}\right) f(V^*). \qquad (15)$$

The proof of this theorem is given in the Appendix.

Let $\tilde{V}^* = \operatorname{argmax}_{V \in \mathcal{V}^s f^s(V)}$ denote the optimal subvolume in $\mathcal{V}^s$. Based on (15), we have

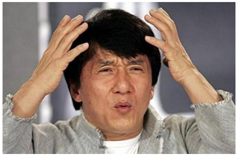
$$f^s(\tilde{V}^*) \geq \left(1 - \frac{s * h + s * w + s^2}{wh}\right) f(V^*). \qquad (16)$$

Action Detection and Localization: Downsampling.

- That theoretical analysis is consistent with the numerical experiments they performed.

Action Detection and Localization: Top-K Search Algorithm.

- The next slide contains the Top-K seach algorithm. I am very



  with it.

```
1: Initialize $P$ as empty priority queue
2: set $\mathbb{W} = [T, B, L, R] = [0, m] \times [0, m] \times [0, n] \times [0, n]$
3: push($\mathbb{W}, \hat{F}(\mathbb{W})$) into $P$
4: c = 1
5: **repeat**
6:   Initialize $(\{W_i^*, F_i^*\})_{i=c\ldots k}$ where $F_k^* \le \ldots \le F_c^*$
7:   **repeat**
8:     retrieve top state $\mathbb{W}$ from P based on $\hat{F}(\mathbb{W})$
9:     **if** $\hat{F}(\mathbb{W}) > F_k^*$ **then**
10:       split $\mathbb{W}$ into $\mathbb{W}^1 \cup \mathbb{W}^2$
11:       **if** $\hat{F}(\mathbb{W}^1) > F_k^*$ **then**
12:         push $(\mathbb{W}^1, \hat{F}(\mathbb{W}^1))$ into $P$
13:         update $(\{W_i^*, F_i^*\})_{i=c\ldots k}$
14:       **end if**
15:       **if** $\hat{F}(\mathbb{W}^2) > F_k^*$ **then**
16:         push $(\mathbb{W}^2, \hat{F}(\mathbb{W}^2))$ into $P$
17:         update $(\{W_i^*, F_i^*\})_{i=c\ldots k}$
18:       **end if**
19:     **end if**
20:   **until** $\hat{F}(\mathbb{W}) \le F_c^*$
21:   $T^* = argmax_{T \in [0,t]} f(W^*, T)$;
22:   output $V_c^* = [W^*, T^*]$ as the $c$-th detected subvolume
23:   for each point $(i, j, k) \in V_c^*$, set $f(i, j, k) = 0$.
24:   $c = c + 1$
25:**until** $c > k$
```

Experiments: Action Classification

- Database: KTH dataset (publicly available). Clips from 16 subjects for training, the other 9 for testing.

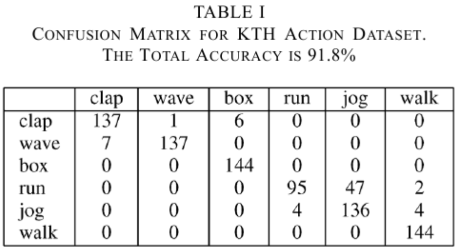


TABLE I
CONFUSION MATRIX FOR KTH ACTION DATASET. THE TOTAL ACCURACY IS 91.8%

	clap	wave	box	run	jog	walk
clap	137	1	6	0	0	0
wave	7	137	0	0	0	0
box	0	0	144	0	0	0
run	0	0	0	95	47	2
jog	0	0	0	4	136	4
walk	0	0	0	0	0	144

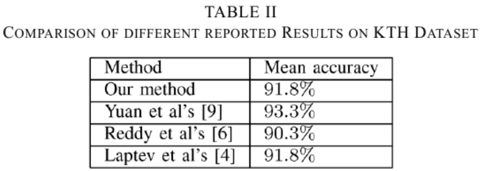


TABLE II
COMPARISON OF DIFFERENT REPORTED RESULTS ON KTH DATASET

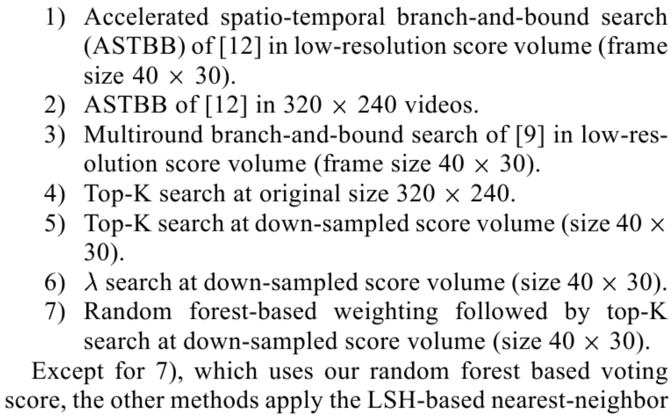
Method	Mean accuracy
Our method	91.8%
Yuan et al's [9]	93.3%
Reddy et al's [6]	90.3%
Laptev et al's [4]	91.8%

Experiments: Action Detection

- Database: KTH dataset (publicly available). Clips from 16 subjects for training. for testing, they use the challenging dataset MSRII of 54 video sequences where each video consist of several actions performed by different people in crowded environment.
  - Each MSRII video is approximately one minute long.
  - The videos contain three different types of actions: handwaving, handclapping, and boxing.
  - Some videos contain different people performing different actions simultaneously. There are also instances where a person performs two different actions consecutively.
- KTH walking data is the negative dataset when constructing forests.
- Parameters like A or K are fixed by hand.

Experiments: Action Detection

- Tested methods:



1) Accelerated spatio-temporal branch-and-bound search (ASTBB) of [12] in low-resolution score volume (frame size 40 $\times$ 30).
2) ASTBB of [12] in 320 $\times$ 240 videos.
3) Multiround branch-and-bound search of [9] in low-resolution score volume (frame size 40 $\times$ 30).
4) Top-K search at original size 320 $\times$ 240.
5) Top-K search at down-sampled score volume (size 40 $\times$ 30).
6) λ search at down-sampled score volume (size 40 $\times$ 30).
7) Random forest-based weighting followed by top-K search at down-sampled score volume (size 40 $\times$ 30).

Except for 7), which uses our random forest based voting score, the other methods apply the LSH-based nearest-neighbor

Experiments: Action Detection

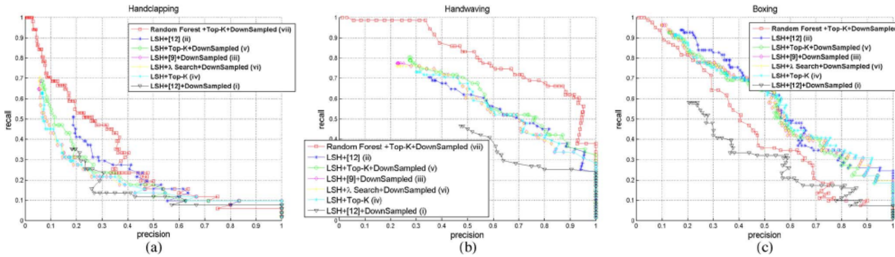


Fig. 3. Precision-recall curves for action detections with different methods. (a) Handclapping. (b) Handwaving. (c) Boxing.

Experiments: Action Detection

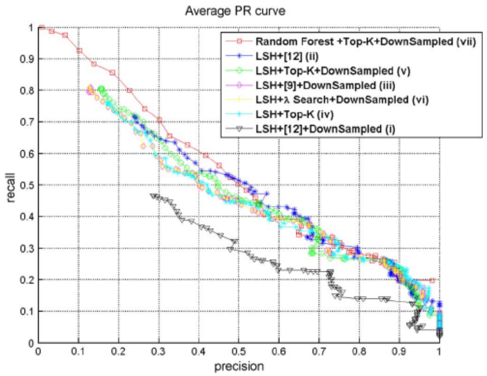


Fig. 4. Comparisons of average precision-recall curves.

TABLE IV
TIME CONSUMED FOR EACH METHOD TO SEARCH ACTIONS IN THE 54 VIDEOS

Method	Running Time
Low resolution (40×30) [12]	40 mins
High resolution (320×240) [12]	20 hours
Down-sampled B&B (40×30)	10 hours
λ search + Down-sampled B&B (40×30)	1 hour 20 mins
Top-K + Down-sampled B&B (80×60)	6 hours
Top-K + Down-sampled B&B (40×30)	26 mins

Experiments: Action Detection

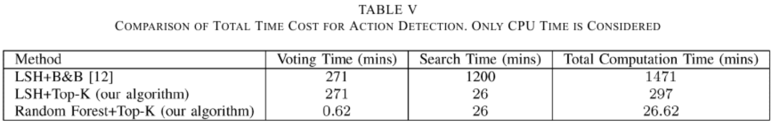


TABLE V
COMPARISON OF TOTAL TIME COST FOR ACTION DETECTION. ONLY CPU TIME IS CONSIDERED

Method	Voting Time (mins)	Search Time (mins)	Total Computation Time (mins)
LSH+B&B [12]	271	1200	1471
LSH+Top-K (our algorithm)	271	26	297
Random Forest+Top-K (our algorithm)	0.62	26	26.62

Conclusion

Main contributions:

- The proposed a random forest-based voting technique to compute the scores of the interest points, which achieves a **multiple orders-of-magnitude speed-up** compared with the nearest-neighbor-based scoring scheme.
- Top-k search technique which detects multiple action instances simultaneously with a single round of branch-and-bound search.
- To reduce the computational complexity of searching higher resolution videos, they performed a subvolume search on the down-sampled score volumes.

Conclusion

Results:

- The experiments have been made with challenging videos.
- The results show that the proposed system is robust to dynamic and cluttered background and is able to perform faster-than real-time action detection on high-resolution videos.