This is a postprint version of the following published document:

Murtaza, F., Yousaf, M.H. y Velastín, S. A. (2019). TAB: Temporally aggregated bag-of-discriminant-words for temporal action proposals. *Computer Vision and Image Understanding*, 183, pp. 42-52.

DOI: <https://doi.org/10.1016/j.cviu.2019.04.008>

© 2019 Elsevier Ltd. All rights reserved.

This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/).

TAB: Temporally Aggregated Bag-of-Discriminant-Words for Temporal Action Proposals

Fiza Murtaza^a, Muhammad Haroon Yousaf^{a,**}, Sergio A. Velastin SMIEEE^{b,c}

^aDepartment of Computer Engineering, University of Engineering and Technology Taxila, Taxila 47080, Pakistan

^bDepartment of Computer Science, Applied Artificial Intelligence Research Group, University Carlos III de Madrid, Madrid 28270, Spain

^cSchool of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, UK

ABSTRACT

In this work, we propose a new method to generate temporal action proposals from long untrimmed videos named Temporally Aggregated Bag-of-Discriminant-Words (TAB). TAB is based on the observation that there are many overlapping frames in action and background temporal regions of untrimmed videos, which cause difficulties in segmenting actions from non-action regions. TAB solve this issue by extracting class-specific codewords from the action and background videos and extracting the discriminative weights of these codewords based on their ability to discriminate between these two classes. We integrate these discriminative weights with Bag of Word encoding, which we then call Bag-of-Discriminant-Words (BoDW). We sample the untrimmed videos into non-overlapping snippets and temporally aggregate the BoDW representation of multiple snippets into action proposals using a binary classifier trained on trimmed videos in a single pass. We present the effectiveness of our TAB proposal extraction method on two challenging temporal action detection datasets: MSR-II and Thumos14, where it improves upon state-of-the-art with recall rate of 82.0% and 80.65% respectively at a temporal intersection over union ratio of 0.8.

1. Introduction

An enormous amount of video data is being produced by the millions of cameras deployed every year to record store and transmit video data. A large portion of this data contains human activities and other events associated with human beings. From this continuous video data, effective recognition and interpretation of human related activities is required, which can aid many applications e.g. content based searching and indexing, surveillance etc. To achieve this, there is a great demand for computer vision based algorithms to temporally detect the actions of interest from the long untrimmed video data.

Most existing research work in action recognition is based on the classification of pre-segmented short clips, comprising a single action, into one of a set of predefined action classes (Herath et al., 2017). However, in real-time scenarios, video streams are streamed or recorded in a continuous way. This results in long streams of videos containing multiple actions (and

no actions) of different durations over time. Hence, algorithms must be able to simultaneously detect the time interval and category of different actions as they appear in continuous video streams. This task is known as temporal action detection in which we need to answer two questions 1) *when* and 2) *what* is happening in the long untrimmed videos.

In recent years, researchers have come up with the notion of using temporal action proposals (Shou et al., 2016; Caba Heilbron et al., 2016; Escorcia et al., 2016) for the detection of actions in the video streams. These action proposals find related temporal windows and an action classifier independently classifies these temporal windows in the subsequent stage. An approach based on a sliding window method is used for the generation of these action proposals, where the video is divided into small and overlapping temporal windows. To deal with the issue of varying action lengths in the video, multiple temporal scales are used at the time of applying temporal windows (Herath et al., 2017). However, the use of multiple temporal scales increases computational cost because of the exhaustive search in temporal location and scale. Instead, a better approach is to use a sliding window method that makes use of a fixed temporal scale to generate action proposals of varying durations for

^{**}Corresponding author: Tel.: 92-51-9047574; fax: +92-51-9047420;
e-mail: haroon.yousaf@uettaxila.edu.pk (Muhammad Haroon Yousaf)

Fig. 1: Illustration of an untrimmed video taken from the Thumos14 dataset containing ‘HighJump’ action (shown in green). Note that the ‘HighJump’ action starts at 7.30 seconds but there are background frames (running) which have similar poses to the starting frames of the ‘HighJump’ action, which cause difficulty in finding an exact start time.

action detection (Escorcia et al., 2016). However, this approach is still computationally infeasible as it gives rise to redundant processing and computations on each frame due to overlapping temporal windows. For real-time action detection systems and applications, it is necessary to avoid redundant computations so that action localization can be done efficiently. Another major issue with the untrimmed videos is that the videos mostly contain many frames having semantic similarity between both action and background (not containing any targeted action) regions as shown in Fig. 1. This leads to difficulties in finding the correct starting/ending time of an action because the features obtained on these common frames will be similar for two different classes. Therefore, in this research we exploit the discriminative power of the codewords which highlights the importance of the small regions of videos to distinguish between action and non-action regions of the untrimmed videos.

In long untrimmed videos, the duration of an action is not fixed but it can vary considerably over time, ranging from a few seconds to quite a few minutes. In addition, the length of different action intervals can be small as compared to the length of the untrimmed video. The considerable differences in the duration of an action and varying action intervals give rise to some major challenges: (1) the algorithms that use a sliding window approach for action detection will have to adapt to actions of varying length. For this purpose, a large number of sliding windows with different step size and length would be needed to obtain high localization accuracy. This will ultimately increase the computational cost. (2) Commonly used methods for action recognition work on densely sampled frames. This process is computationally expensive for long video streams. Moreover, end-to-end learning becomes very difficult in such cases.

This work aims to develop a novel framework that can detect human actions of varying durations and accurately locate the starting and ending time of each action from long uncut videos. To handle the challenges outlined earlier, we devise several contributions:

- First, we propose a codeword-weighting scheme for learning the discriminative power of the codewords, which will

aid in the encoding process. We embed discriminating weights of each codeword with the traditional Bag of Word approach. Therefore, we call it Bag of Discriminating Words (BoDW) encoding.

- Second, we apply BoDW on non-overlapping snippet level features of untrimmed videos. Then we aggregate multiple snippets into candidate proposals and we call it Temporal Aggregation of BoDWs (TAB). We classify these proposals into actions and background proposals by applying a classifier trained on trimmed action videos.
- Finally, we remove all background proposals and the remaining action proposals are used for the task of temporal action detection. We verify that our proposed TAB action proposals provide state-of-the-art results when integrated with an existing action recognition framework. We evaluated our proposed temporal proposal method on two challenging benchmarks; MSR-II and Thumos14. The results on the two datasets, having different challenges, show that our method is robust for detecting actions in different scenarios where the videos are long and untrimmed.

The rest of the paper is organized as follows: Section 2 presents related work; Section 3 introduces our proposed proposal generation method. The experimental results are discussed in Section 4. The conclusions are drawn in Section 5.

2. Related Work

2.1. Action Recognition

Extensive research has been carried out on addressing the issue of action recognition from video clips of small duration (Herath et al., 2017). In this case, the video streams to be processed are pre-segmented into short video clips that are well-trimmed. Each video clip only contains a single action of interest for the entire duration. Therefore, there is no need for action localization and the point of focus is action classification only. Numerous approaches use a global representation for action classification (Tran et al., 2015), while others use modeling of temporal motion structures to perform action classification (Gaidon et al., 2014). Unfortunately, these schemes are not feasible to operate on lengthy video sequences that contain actions with reasonably short temporal duration and where the major portion of the video sequence is composed of background i.e. non-action regions.

2.2. Temporal Action Detection and Proposals

We found many research works addressing the issue of temporal action localization (Shou et al., 2016; Caba Heilbron et al., 2016; Escorcia et al., 2016; Karaman et al., 2014; Buch et al., 2017; Oneata et al., 2014; Gao et al., 2017; Su and Grauman, 2016; Sun et al., 2015; Wang et al., 2014; Yuan et al., 2015; Tran et al., 2015; Singh et al., 2016; Duchenne et al., 2009). A traditional way of performing temporal action detection is to densely apply action classifiers in a sliding window fashion (Duchenne et al., 2009). Recent works have familiarized the use of temporal action proposals (Shou et al., 2016;

Caba Heilbron et al., 2016; Escorcia et al., 2016) for applying action classifiers on a lesser number of temporal windows in an efficient way. The basic notion is the generation of a fewer number of candidate temporal windows at first, which can be done using dictionary learning used in Sparse-prop (Caba Heilbron et al., 2016) or recurrent neural architecture used in Deep Action Proposals (DAPs) (Escorcia et al., 2016) and Single Stream Temporal action proposals (SST) (Buch et al., 2017). After that, an action classifier is applied to each action proposal in an independent fashion to categorize each proposal into one of the pre-defined actions. Multi-Stage CNN (SCNN) (Shou et al., 2016) proposed a temporal action localization, which is based on three segment-based 3D ConvNets. A first stage is temporal proposal generation; a second stage is designed for the classification of these proposals and a third stage is used for the action localization. Another approach for temporal action detection is the Temporal Unit Regression Network (TURN) (Gao et al., 2017), which refines the temporal boundaries of the proposals by temporal coordinate regression. Although these methods (Shou et al., 2016; Escorcia et al., 2016; Buch et al., 2017; Gao et al., 2017) incorporate deep networks for proposal generation, these methods do not handle the inter-class similarity problem. On the other hand, our method for proposal generation reduces inter-class similarity by exploiting the discriminative power of the codewords during encoding of each candidate proposal.

2.3. Spatiotemporal Action Detection

Another related problem is spatiotemporal action detection, which aims at detecting actions of interest not only in the temporal but also in the spatial domain (Chen et al., 2014; Gkioxari and Malik, 2015; Jain et al., 2014; van Gemert et al., 2015; Yao et al., 2010; Yu and Yuan, 2015). Spatiotemporal action detection also answers about *where* in the image an action is present along with *when* it occurs. In this case, the algorithms give rise to spatiotemporal action localization, providing added comprehensive localization information. As a result, computational complexity increases, which makes these algorithms infeasible to use for real-time systems and applications that require fast processing. Moreover, there is a possibility that temporal proposals may essentially help in reducing the search space for these methods. In this research work, we address temporal action detection instead of spatiotemporal detection.

3. Proposed Temporal Action Proposal Generation Framework

This work aims to produce temporal action proposals in long untrimmed videos. Given an untrimmed video, our method should output a small number of temporal locations, which are likely to contain a human action. A good temporal action proposal method should retrieve tight action locations with high recall while rejecting background regions, which do not contain targeted actions. The retrieved action locations, known as temporal action proposals, should have a high temporal overlap with the actual (ground truth) locations. Temporal proposals with high overlap will aid in classification of these proposals in one of the predefined action classes, i.e. temporal action detection. Our temporal action proposal method is motivated by the

observation that the proposal generation method must differentiate between the desired human actions and the background actions. In this way, we can retrieve only temporal locations that contain true action locations. However, generating uniform samples without utilizing any content-based information of the predefined action classes can generate proposals with less computational cost, but will retrieve many proposals, which do not contain human actions. In this work, we propose a Temporally Aggregated BoDW (TAB) that produces action proposals with high recall.

Fig. 2 shows an outline of our proposed method for producing temporal action proposals. Given a set of trimmed training videos, we first extract features (Section 3.1) to utilize spatial and temporal information of the videos. After feature extraction, we perform class-specific clustering (Section 3.2) to obtain the codewords for action and background classes. We then learn the weights for all codewords based on their discriminative power to differentiate between action and non-action class. After learning weights for codewords, we use this information in the encoding process (Section 3.3). From the untrimmed test video, we produce a set of candidate temporal regions having varying duration (Section 3.4 and 3.5). To produce the final temporal action proposals, these temporal regions are classified into actions or non-action regions using a binary (action/background) classifier.

3.1. Feature extraction

We use the popular Improved Dense Trajectories (IDT) (Wang and Schmid, 2013) based approach to extract the trajectories from the training and testing data. We use IDT as it has become the de facto feature extraction method in many activity recognition methods (Gorban et al., 2015). For the training data, we extract features from the trimmed training videos. As a standard practice (Wang and Schmid, 2013), we describe the extracted trajectories using Histograms-of-Oriented-Gradients (HOG) (Dalal and Triggs, 2005), Histograms-of-Optical-Flow (HOF) (Laptev et al., 2008) and Motion-Boundary Histograms (Dalal et al., 2006) in x and y directions (MBHx and MBHy). For IDT implementation, we extract these features using the publicly available source code provided by the authors (Wang and Schmid, 2013) with the default parameter settings. For each trajectory, the dimensionality of the HOF descriptor is 108 whereas the HOG, MBHx and MBHy descriptors have dimensionality of 96. We reduce the dimensionality D of these features in half, using Principal Component Analysis (PCA). By doing so, the dimensions D of these feature descriptors become 48 for HOG, MBHx and MBHy, and 54 for HOF. We emphasise that our proposed BoDW approach can also work with other features including deep features, which we will incorporate in future.

3.2. Learning Discriminative Power of Codewords

Our ultimate goal is to discriminate between action and background classes so that we can extract true action regions from the long untrimmed videos. For training, we take the trimmed videos containing targeted action classes as positive and the other background videos as negative data. We assume that the

Fig. 2: Block diagram of our proposed Temporally Aggregated BoDW (TAB) action proposal approach. Note that the Codebook Generation and Weight Learning is performed only once, which are used for the BoDW Encoding of snippets.

features within a specific action class would not be independent but instead they share commonalities with the features of the background class. Therefore, using global clustering features from two different classes (action and background) are most likely to be assigned to the single codeword during the encoding process, which may result in incorrect representation. To solve this issue, we perform class-specific clustering on each class and obtain the discriminative power of the codewords. We extract the discriminative power of the codewords from the training data using two successive steps. First, we find the codewords from the positive and negative class by performing the class-specific clustering separately on the feature sets of these two classes. From each trimmed training video, we extract a set of D -dimensional feature descriptors $\phi = \{x_l\}_{l=1}^m$ where m represents the total number of features extracted from the video. From the training data of action and background class $A = \{a_1, a_2\}$, we compile all feature descriptors from videos be-

longing to the class a_i into a feature matrix $X_i \in \mathbb{R}^{D \times n_i}$ where n_i represents the total number of features in the training set of class a_i . To perform class-specific clustering, we divide the feature matrix X_i into K codewords using K-means clustering with Euclidean distance. In this way for action and background class we obtain the set of $2K$ codewords $\{C_j\}_{j=1}^{2K}$.

Second, we compute the relative importance of the codewords by obtaining their weights w_j in accordance to their ability to differentiate between the action and background class. From the training feature descriptors of two classes, each feature descriptor x_l is assigned to the nearest codeword C_j , such that $\|x_l - C_j\|$ is minimum (line 4 of Algorithm 1). For each codeword C_j , we record the number of within-class q_j and out-of-class q'_j assignments of feature descriptors to C_j which show the correct and false assignments of feature descriptors respectively (lines 5-9 of Algorithm 1). This assignment will be used to find out the discriminative importance of each codeword C_j .

Algorithm 1 Finding the number of within-class q_j and out-of-class q'_j assignments for corresponding codewords C_j

Input: Set of extracted descriptors $\{x_1, \dots, x_n\}$ from training videos and codewords $C_r, \forall r \in [1 : 2K]$

Output: q_j and q'_j

```

1:  $q_j = 0, q'_j = 0$ 
2: for all actions  $a_i \in A$  do
3:   for all features  $x_l \in a_i$  do
4:      $j = \arg \min_r \|x_l - C_r\|$ 
5:     if  $C_j \in a_i$  then
6:        $q_j = q_j + 1$ 
7:     else
8:        $q'_j = q'_j + 1$ 
9:     end if
10:  end for
11: end for

```

in terms of weights w_j , as given below:

$$w_j = \frac{q_j}{q_j + q'_j} \quad \forall j \in [1 : 2K] \quad (1)$$

From Eq.1 it follows that if C_j is not assigned to any of the feature descriptors of its own class then $w_j = 0$ or if it is only assigned to the descriptors of its own class then $w_j = 1$ else $0 < w_j < 1$. From the viewpoint of codewords; if a codeword is common to both classes, action and background, it will have lower weight, which would decrease its importance. From the viewpoint of classes; those codewords which are assigned only to the feature descriptors of their corresponding class are discriminative and therefore they have high weights. We further integrate this discriminative power of codewords during the encoding process.

3.3. BoDW encoding

After calculating the codewords C_j and their corresponding weights w_j , feature descriptors $\phi = \{x_l\}_{l=1}^m$ (with m equals to the number of features extracted from the video clip) are then assigned to their nearest codeword. For simplicity, we perform a hard voting scheme (Sivic and Zisserman, 2003) to make the bags of these discriminating codewords i.e. each descriptor x_i votes for its nearest codeword C_j . For each codeword, we do the weighted voting as:

$$v(j) = w_j \times N_j, \quad \forall j \in [1 : 2K] \quad (2)$$

where N_j represents the number of votes for the codeword C_j . The multiplication with w_j results in discriminative representation therefore we call this Bag of Discriminative Words (BoDW) encoding. This multiplication with the corresponding weights will dominate or highlight the importance of the codewords according to their capability to discriminate between the action and the background class. Those frames which belong to two different classes (action and background) but have very high similarity, will have less weight. Therefore, those common frames will contribute less in the further encoding process. For the specific video clip, the resulted BoDW encoded vector

is of size $2K$ (2 is because we are discriminating between two classes i.e. action and background class). The dimensions of the final BoDW encoding approach for proposal generation depends upon the number of targeted classes therefore we prefer a simple hard voting scheme for feature aggregation than other high dimensional super vector based approaches (Schuldt et al., 2004; Perronnin et al., 2010).

3.4. Non-overlapping snippets

At testing time, our method takes a long untrimmed video $V = \{f_i\}_{i=1}^N$ with N frames. Unlike previous methods that sample the untrimmed video into highly overlapping temporal windows (Caba Heilbron et al., 2016), we discretize the untrimmed video into non-overlapping snippets of δ frames. In this way, we divide the video into $T = N / \delta$ non-overlapping temporal snippets $S = \{s_t\}_{t=1}^T$. We use non-overlapping snippet sampling because it can be computationally inexpensive for long videos. In the next step, we extract the IDT features at each time step t in a single pass. During the feature extraction process, each frame is processed only once for a long-untrimmed video. The extracted IDT features over the snippets S are individually passed to the BoDW module (as discussed in Section 3.3) to obtain the snippet level encoded vectors v (Eq. 2), which are further aggregated into proposals using the temporal aggregation module as described in the following section.

3.5. Temporally Aggregated BoDW (TAB)

After extraction of snippet level BoDW features from the untrimmed videos, our next step is to produce the proposals of variable durations for the untrimmed video so that they have high overlaps with the ground truth locations. For each untrimmed video, we aggregate the consecutive multiple snippets into the proposal as shown in Fig. 3. We adopt the proposal of multiple durations from (Buch et al., 2017), but instead of having right aligned proposals we obtain left aligned proposals because it is more beneficial to add future snippets than previous snippets at time step t . By doing so we can reuse the previously obtained proposal level features for obtaining the representation for proceeding proposals (as shown in Fig. 3) which makes it computationally fast. At each time step t , we produce a set of τ proposals, $P_t = \{p_h, o_h, o'_h\}_{h=1}^\tau$, where o_h and o'_h are respectively the starting and ending time of h -th candidate proposal and p_h represents its aggregated feature vector. At each time step t , each candidate proposal h has starting time $o_h = t$ and ending time $o'_h = h + t - 1$. We use a sum pooling for aggregation of the features of corresponding $1, 2, 3, \dots, \tau$ snippets at time step t into proposal based features p_h as given by:

$$p_h = p_{h-1} + v_{t+h-1}, \quad \forall h \in [1 : \tau], p_0 = 0 \quad (3)$$

Where v represents the snippet level BoDW feature vector of dimension $2K$. In this way at each time step t , our temporal aggregation module outputs the features of the candidate proposals having duration $1, 2, 3, \dots, \tau\delta$ frames. We perform the sum pooling to aggregate snippet level features of $1, 2, 3, \dots, \tau$ because this is respectively equivalent to extract the BoDW encoded vectors over $1\delta, 2\delta, 3\delta, \dots, \tau\delta$ frames as a whole. The advantage of this procedure is that it scans the whole untrimmed

Fig. 3: Generation of τ candidate proposals at time step t by temporal Aggregation of τ snippets.

video only once with less computational cost as we are simply summing the snippet level features which are already extracted during feature extraction process.

3.6. Classifying Candidate Proposals

For each training video clip, we apply our proposed BoDW approach on all four IDT descriptors (HOG, HOF, MBHx and MBHy) separately then we perform the concatenation of these four representations. We use representation-level fusion (also known as early fusion) for the concatenation because for IDT features along with hard voting scheme, the representation-level fusion performs best (Peng et al., 2016). The final concatenated BoDW vector is of size $8K$ i.e. $4 \times 2K$. Similarly, for each candidate proposal h extracted from untrimmed video we extract its corresponding proposal level feature representations p_h (Eq. 3) for four IDT descriptors separately. Then we perform early fusion of these four feature representations resulting in a $8K$ sized feature representation for each proposal. As mentioned earlier, our method does not depend on these specific features but, for example, deep-obtained features could be used instead. Once we have created and obtained the feature representations for all candidate proposals, next step is to classify them into action and background proposals using, for example, a Support Vector Machine (SVM) classifier. We use all trimmed videos containing predefined action classes as positive and other trimmed background videos as negative data to train a linear SVM classifier. Details of the positive and negative training data is provided in Section 4.1.1. Finally, we use the learned classifier to classify each candidate proposal into action and background regions. We remove the proposals classified as background and use the remaining action proposals for temporal action detection (Section 4.4).

4. Experimental Results

In this section, we evaluate the performance of our proposed method for the generation of temporal action proposals. We provide the details of the dataset used, experimental settings and results obtained. We also describe the application of the obtained temporal action proposals to the task of temporal action detection.

4.1. Experimental Setup

4.1.1. Datasets

We perform the evaluation of our proposal generation method on two publicly available datasets. First, we use the MSR-II Action dataset (Yuan et al., 2009) containing 54 videos where each video has a duration of 51 seconds on average. Each video has three actions performed different times by different actors. These actions are ‘Boxing’, ‘Clapping’ and ‘Waving’. The most challenging part of this dataset is that the actions are performed in a crowded environment and the background contains the movement of people and other objects. As a standard practice, we use the KTH (Soomro et al., 2012) dataset for training. We use ‘Boxing’, ‘Clapping’ and ‘Waving’ action from the KTH dataset as positive training examples whereas for negative training data we use the ‘Walking’ action. Second, we evaluate the quality of our proposed method on the challenging Thumos14 dataset (Dalal et al., 2006). This dataset contains untrimmed videos having 20 sports actions compiled from YouTube. It is considered as the most challenging dataset for action detection due to the large intra-class variability in the actions. It contains videos of about 3 minutes duration containing multiple occurrences of an action but it is confined to a small portion of the video. As a standard practice, we use trimmed videos of 20 sports actions from UCF101 dataset (Duta et al., 2017) as a positive training data. For negative training example, we use the videos of other 20 classes at random (to maintain the ratio of 1:1) from UCF101 for training the SVM classifier. For testing, we use the 213 untrimmed test videos provided in the Thumos14 challenge.

4.1.2. Parameter settings

For generating the codewords for action and the background data, we perform the class-specific k-means clustering with $K = 256$. We set $K = 256$ as a trade-off between computational complexity and accuracy as done in (Schuldt et al., 2004). We apply the BoDW encoding scheme to these four descriptors separately and perform the early fusion for the concatenation of these four BoDW encoded vectors (as discussed in Section 3.6). Finally, we apply Power Normalization (PN) followed by L2-normalization with $\alpha = 0.1$ as recommended in (Peronnin et al., 2010). Without normalization, the feature descriptors with a higher value will dominate the feature representation and hence influence the results of the classifier negatively. To have proposals of variable lengths, we extract multiple candidate proposals by setting $\tau = 64$ snippets for Thumos14 and $\tau = 20$ for MSRII dataset (Eq. 3). This depicts minimum and maximum proposals duration equal to 16 and 1024 (16×1 and 16×64 where 16 shows the snippet duration) frames respectively for Thumos14. Whereas for MSRII dataset the obtained proposals have minimum and maximum duration of 16 and 320 frames respectively. We adopt this minimum and the maximum length of the actions from the training data of both datasets and this will cover almost all types of actions having different durations. In this way, our proposed BoDW encoding provides the features for the proposals of all possible durations as contained in the training data. For classification, we choose a linear SVM classifier. As common practice (Shou et al., 2016; Escor-

cia et al., 2016; Buch et al., 2017), we finally perform non-maximal suppression to remove the highly overlapping action proposals having overlap of greater than 0.75.

4.1.3. Comparisons

For Thumos14 dataset, we compare our proposed TAB proposal generation approach with SCNN (Shou et al., 2016), Sparse-prop (Caba Heilbron et al., 2016), DAPs (Escorcia et al., 2016), SST (Buch et al., 2017) and TURN-FLOW (Gao et al., 2017). All of these methods use deep networks for proposal generation except Sparse-prop (Caba Heilbron et al., 2016) that uses Spatio Temporal Interest Points (STIPs) (Laptev, 2005) described by HOG and HOF descriptors. As mentioned earlier, (in Section 3.5), we adopt multiple length proposal length from SST (Buch et al., 2017), but SST requires sliding windows during the training stage for producing dense training data. In contrast, our method takes positive and negative videos for training the binary classifier, which makes it computationally inexpensive. Similarly, SCNN (Shou et al., 2016), Sparse-prop (Caba Heilbron et al., 2016), DAPS (Escorcia et al., 2016) and TURN-FLOW (Gao et al., 2017) produce proposals of varying durations by applying sliding windows at multiple temporal scales, which makes them computationally expensive.

For MSR-II dataset, we use the comparisons provided by (Caba Heilbron et al., 2016) in which the comparison is performed with two baselines; 1) Uniform sampling and 2): Binary Proposal Classifier (BPC). Uniform sampling provides a uniform ranking of each candidate action proposal in a random manner whereas the BPC discriminates between action and non-action from the raw video descriptors using a supervised learning approach. For MSR-II, we also compare our proposal generation method with Action localization Proposals from dense Trajectories (APT) (van Gemert et al., 2015), Fast Action Proposals (FAP) (Yu and Yuan, 2015), and Sparse-prop [3]. The IDT (Wang and Schmid, 2013) based features are used by APT (van Gemert et al., 2015) and FAP (Yu and Yuan, 2015) proposal generation methods.

4.1.4. Evaluation Metrics

For measuring the quality of our proposed temporal action proposals, we use Average Recall (AR) as used in other proposal generation methods (Shou et al., 2016; Caba Heilbron et al., 2016; Escorcia et al., 2016; Buch et al., 2017; Gao et al., 2017; van Gemert et al., 2015; Yu and Yuan, 2015). AR is computed at temporal Intersection over Union (tIoU) between 0.5 and 1 for varying number of retrieved proposals. We also compute the recall at different tIoU thresholds to show how tight our action proposals are with the ground truth for the fixed number of proposals. For this experiment, we use 1000 proposals retrieved per video. Finally, we examine the effect of using the proposed proposals in the context of temporal action detection in Section 4.4. We use mean Average Precision (mAP) as a standard evaluation measure to evaluate the temporal action detection results using evaluation toolkit provided by Thumos14 (Oneata et al., 2014).

4.2. TAB Proposal Generation

Given an input untrimmed video, the task of the temporal action proposal generation is to produce multiple temporal segments, which contain human actions. As discussed earlier, a good temporal proposal method should produce a small number of proposals with high recall rate at high tIoU. Moreover, it should be fast enough to produce proposals from long untrimmed videos. In this section, we provide the detailed experiments and the results to evaluate these three characteristics of our proposed temporal proposal method. For comparison, we also provide the performance of other state-of-the-art methods.

In the first experiment, we evaluate the ability of our method to obtain proposals with high recall. For this, we find the AR by extracting the fixed number of proposals over different tIoU thresholds. The recall is averaged over different tIoU thresholds in the range 0.5 - 1.0. In Fig. 4, we plot AR versus different number of retrieved proposals. From Fig. 4, we can observe that our proposed method outperforms the other state-of-the-art temporal proposal methods at different number of retrieved proposals for MSR-II and Thumos14 dataset. As compared to the TURN-FLOW (Gao et al., 2017), the performance of our proposed TAB proposals is better at lesser number of retrieved proposals for Thumos14 dataset. For Thumos14 our proposed TAB proposals achieve AR of 19% at only 20 proposals, whereas for moderate number of retrieved proposals we achieve the comparable performance, which again increases at 1000 proposals. Similarly, for MSR-II we achieve comparable performance with Sparse-prop (Caba Heilbron et al., 2016) for different number of retrieved proposals.

In the second experiment, we evaluate the ability of our proposed method to obtain high recall at high tIoU overlaps. In Fig. 5, we plot the proposal recall versus different tIoU overlap thresholds at 1000 proposals for Thumos14 and MSR-II dataset. Results in Fig. 5, show that our proposed proposal method has a comparable recall at low tIoUs but high recall at high tIoUs as compared to other methods. For Thumos14, we got smaller recall at lower tIoU threshold as compared to the other methods because our method gives less weight to the starting and ending frames of those actions that have high similarity with the background frames. Therefore, there is the chance that the proposals having much smaller durations (e.g. 16 frames) are misclassified due to less weights and the predicted start time for the action is somehow beyond the actual start time. In addition, the proposals generated by our method have duration multiple of size of the snippets i.e. 16 frames. Therefore, it can only produce candidate proposals having duration of 16, 32, 48, $\dots$, 1024 frames (for Thumos14). For this reason, there is a margin of about 16 frames in starting and ending times. However, our method got high recall at higher threshold than other methods. It can be observed that for Thumos14 dataset, our method achieves 0.604 recall as compared to 0.26 recall achieved by the competitive method i.e. TURN-FLOW (Gao et al., 2017) at 0.9 tIoU threshold. Similarly, for MSR-II dataset we achieve a comparable recall rate of 0.82 as compared to 0.80 achieved by Sparse-prop (Caba Heilbron et al., 2016) at 0.8 tIoU. This shows that our method can produce the proposals having high overlap with the ground truth temporal locations.

Fig. 4: Average recall against different number of retrieved proposals per video. Recall is averaged for different tIoU thresholds in the range 0.5 - 1.0. For this metrics, we use code provided by (Escorcia et al., 2016).

Fig. 5: Recall rate at different tIoU thresholds for 1000 proposals per video. For this metrics, we use code provided by (Escorcia et al., 2016).

Finally, in the third experiment, we compare the speed of our proposal generation method with the APT (van Gemert et al., 2015) on the Thumos14 dataset. For APT proposal generation, we use the publicly available code (van Gemert et al., 2015). For fair comparison, we only compare with APT because it also uses IDT features. All experiments are performed on a workstation having an Intel Xeon E5-2609 CPU and 16 GB memory. We use frames per second (FPS) to measure the speed of both methods as shown in Table 1. Results in Table 1, show that our proposed TAB proposal method is about 11 times faster than APT (van Gemert et al., 2015) because, for proposal generation, TAB scans each frame only once during feature extraction phase. As TAB has the ability to produce multiple sized pro-

posals by reusing the already computed BoDW features, this makes it computationally efficient. Note that IDT features take more time because the code (van Gemert et al., 2015) provided by authors only have CPU implementation and it is not possible to extract IDT features on GPU. Therefore for both feature extraction and proposal generation tasks, we only mention the time for CPU based implementations.

4.3. Qualitative Analysis of TAB proposals

For visualization of the outcome of our proposed proposal method, we generate proposals having high and low tIoU with the ground truth. From Fig. 6(a), it is clear that our proposal method correctly finds the starting and ending time of the

Fig. 6: Visualization of proposals obtained by our method on Thumos14 (a, b) and MSR-II (c) dataset. The candidate proposal is marked as true positive if $tIoU \geq 0.5$ otherwise false positive.

Table 1: Comparison of proposal generation performance in terms of average processing time (in sec) for a 3-minute video from the Thumos14 dataset.

Methods	Features	Proposals	Total	FPS
APT	457.8	5120.3	5578.1	0.97
Our	457.8	8.56	466.36	11.6

Table 2: Temporal action detection performance in terms of mAP (%) @ tIoU threshold equal to 0.5 at different number of proposals for Thumos14.

Methods	@50	@100	@200	@500	@1000
Sparse-prop	5.7	6.3	7.6	8.2	8
SCNN	5.6	7.7	10.5	13.57	13.45
DAPs	8.4	12.1	13.9	12.5	12
SST	10.9	13.2	13.94	13.1	12
TURN-FLOW	-	-	-	-	25.6
Our	11.8	17.7	23.6	30.45	32.3

‘LongJump’ action with high overlap with the ground truth. We observe that the true positive proposals with high confidence score have the highest overlap with the ground truth annotations. We provide the qualitative results for the false positive proposals in Figure 6(b). From Figure 6(b) we can see that a false positive proposal is generated because there is no clear transition between the background and the true action region of ‘GolfSwing’ action. In Figure 6(c), we provide the results for the untrimmed video taken from MSR-II dataset, which contains three actions in order of ‘Boxing’, ‘Waving’, ‘Clapping’. Note that the MSR-II dataset has cluttered background but still our method correctly finds the starting and ending time for the ‘Waving’ and ‘Clapping’ actions with tIoU of 0.93 and 0.97 re-

Table 3: Temporal action detection performance on Thumos14 in terms of mAP (%) @ different tIoU thresholds.

mAP @ tIoU threshold	0.3	0.5	0.7
Wang et al. (2014)	14.6	8.5	1.5
Oneata et al. (2014)	28.8	15	3.2
Yuan et al. (2015)	33.6	18.8	-
SCNN (Shou et al., 2016)	36.3	19	5.3
SST (Buch et al., 2017)	37.8	23	-
Our	39.6	32.3	20

spectively. Whereas for ‘Boxing’ action, our method finds the corresponding proposal with tIoU of 0.4 (which is less than the tIoU threshold of 0.5) therefore it is marked as a false positive.

4.4. Application to Temporal Action Detection

The key goal of generating temporal action proposals with high tIoU is to increase the accuracy of human action detection in long untrimmed video streams. For achieving this goal, we integrate our scheme into an action detection framework in the following manner. At first, we perform training of action classifiers by means of the trimmed action clips available for both datasets. As a standard practice (as mentioned in Section 4.1.1), we use the KTH and UCF-101 datasets for training MSR-II and Thumos14 dataset respectively. Subsequently, in the testing phase, we generate temporal proposals for the untrimmed input video using our proposed TAB action proposal method. In the end, we classify each temporal proposal by applying the trained action classifiers. We assess the performance of action detection scheme using mean Average Precision (mAP).

Fig. 7: Comparison of per class average precision (%) for Thumos14 and MSR-II dataset for tIoU threshold equal to 0.5 and 0.125 respectively.

Table 4: Temporal action detection performance in terms of mAP (%) at different number of proposals for MSR-II

	Uniform Sampling			BPC			APT			Sparse-prop			Our Method		
# Proposals	0.125	0.5	0.8	0.125	0.5	0.8	0.125	0.5	0.8	0.125	0.5	0.8	0.125	0.5	0.8
10	45.3	29.7	11.5	65.8	46.3	18.9	66.5	45.1	22.1	72.9	55.4	26.3	69.7	36.4	12.1
50	50.1	31.3	13.2	73.1	49.9	21.1	72.7	49.8	28.7	76.1	57.7	29.1	78.8	63.6	27.3
100	52.2	30.9	9.7	75.1	55.2	2.3	74.1	54.5	33.7	80.1	60.3	33.9	81.8	66.7	42.4

We then compare the final performance of action detection of our TAB proposals with the state-of-the-art approaches.

For MSR-II and Thumos14 dataset, we reuse the extracted IDT (Wang and Schmid, 2013) features and use the Fisher vectors (Sánchez et al., 2013) to perform encoding of the extracted HOG, HOF, MBHx and MBHy descriptors. We learn a Gaussian Mixture Model (GMM) with 256 Gaussian components as suggested in (Peng et al., 2016) for better performance. We then apply PCA for reducing the dimensions of the extracted descriptors into half. The resultant Fisher vector based representation of the four descriptors is concatenated using early fusion. For the training stage, we choose a one-vs-all linear SVM classifier.

In Table 2, we present the comparison results of action detection achieved using the action proposals extracted by our method with the other state-of-the-art methods on Thumos14 dataset at 0.5 tIoU overlap for different number of retrieved proposals. Our temporal proposals generation method achieves better performance as compared to other action proposals obtained by computationally challenging methods such as SCNN (Shou et al., 2016) and Sparse-prop (Caba Heilbron et al., 2016). Furthermore, the results also depict that our approach provides a satisfactory mAP using merely a few number of temporal proposals. In Table 3 we also provide the comparison results with other methods at different tIoU thresholds and using 1000 proposals for both methods. Results show that our method achieves a high mAP for all tIoU thresholds, which proves that the proposals obtained by our method are tighter with the ground truth regions.

Table 4 provides a comparison of the results obtained by applying temporal action detection scheme on our TAB proposals

with the state-of-the-art methods for MSR-II dataset. For a 0.8 tIoU threshold value, our proposed proposal generation method obtains an mAP score of 42.4% in comparison to the best value of 33.9% in Sparse-prop (Caba Heilbron et al., 2016). In addition, Sparse-prop method (Caba Heilbron et al., 2016) is computationally expensive as compared to our method. Although, our method scans the whole video in a single pass during feature extraction but it can aggregate multiple snippets into proposals of different durations. Our results are promising due to our proposed BoDW based encoding scheme which discriminates action and non-action regions effectively. We provide the average precision of each action in Fig. 7. It can be observed that the detection results on our proposed TAB proposals are higher than other methods for 13 out of 20 classes of Thumos14 dataset but we outperform in terms of mAP than other methods. Similarly, for MSR-II dataset, we got a high mAP as compared to proposals obtained using APT (van Gemert et al., 2015).

5. conclusion

In this work, we have described a TAB proposal technique for the generation of temporal action proposals. We show that the discriminating power of the proposed BoDW encoding scheme results in a good performance for distinguishing between action and non-action regions. We prove that our method can produce temporal proposals with high recall from long uncut video. Due to its ability to generate proposals of multiple lengths in a single pass, it is faster than other state-of-the-art methods with 11.6 FPS. This shows that TAB can aid in many large-scale video analysis applications e.g. content-based video retrieval, support video analysis, and video summarization etc. due to its lower

processing time. We demonstrate that the action proposals obtained by our method can aid in the task of temporal action task and produce a gain in mAP for two challenging untrimmed datasets i.e. Thumos14 and MSR-II. For future work, we will test CNN features with the BoDW encoding scheme. In addition, we will see the results by directly training BoDW for the multi-class problem for final temporal detection task.

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research. We also acknowledge the support from the Directorate of Advance Studies, Research and Technological development (ASR) & TD, University of Engineering and Technology Taxila, Pakistan. Sergio A Velastin acknowledges funding by the Universidad Carlos III de Madrid, the European Unions Seventh Framework Programme for research, technological development and demonstration under grant agreement n 600371, el Ministerio de Economa y Competitividad (COFUND2013-51509) and Banco Santander.

Data Statement

The datasets used for the evaluation of the proposed framework are from previously reported studies, which have been cited.

References

- Buch, S., Escorcia, V., Shen, C., Ghanem, B., Niebles, J.C., 2017. Sst: Single-stream temporal action proposals, in: *Computer Vision and Pattern Recognition (CVPR)*, 2017 IEEE Conference on, IEEE. pp. 6373–6382.
- Caba Heilbron, F., Carlos Niebles, J., Ghanem, B., 2016. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1914–1923.
- Chen, W., Xiong, C., Xu, R., Corso, J.J., 2014. Actionness ranking with lattice conditional ordinal random fields, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 748–755.
- Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection, in: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, IEEE. pp. 886–893.
- Dalal, N., Triggs, B., Schmid, C., 2006. Human detection using oriented histograms of flow and appearance, in: *European conference on computer vision*, Springer. pp. 428–441.
- Duchenne, O., Laptev, I., Sivic, J., Bach, F., Ponce, J., 2009. Automatic annotation of human actions in video, in: *Computer Vision, 2009 IEEE 12th International Conference on*, IEEE. pp. 1491–1498.
- Duta, I.C., Ionescu, B., Aizawa, K., Sebe, N., et al., 2017. Spatio-temporal vector of locally max pooled features for action recognition in videos, in: *30TH IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR 2017)*, IEEE. pp. 3205–3214.
- Escorcia, V., Heilbron, F.C., Niebles, J.C., Ghanem, B., 2016. Daps: Deep action proposals for action understanding, in: *European Conference on Computer Vision*, Springer. pp. 768–784.
- Gaidon, A., Harchaoui, Z., Schmid, C., 2014. Activity representation with motion hierarchies. *International journal of computer vision* 107, 219–238.
- Gao, J., Yang, Z., Sun, C., Chen, K., Nevatia, R., 2017. Turn tap: Temporal unit regression network for temporal action proposals. *arXiv preprint arXiv:1703.06189*.
- van Gemert, J.C., Jain, M., Gati, E., Snoek, C.G., et al., 2015. Apt: Action localization proposals from dense trajectories., in: *BMVC*, p. 4.
- Gkioxari, G., Malik, J., 2015. Finding action tubes, in: *Computer Vision and Pattern Recognition (CVPR)*, 2015 IEEE Conference on, IEEE. pp. 759–768.
- Gorban, A., Idrees, H., Jiang, Y., Zamir, A.R., Laptev, I., Shah, M., Sukthankar, R., 2015. Thumos challenge: Action recognition with a large number of classes, in: *CVPR workshop*.
- Herath, S., Harandi, M., Porikli, F., 2017. Going deeper into action recognition: A survey. *Image and vision computing* 60, 4–21.
- Jain, M., Van Gemert, J., Jégou, H., Bouthemy, P., Snoek, C., 2014. Action localization with tubelets from motion, in: *CVPR-International Conference on Computer Vision and Pattern Recognition*.
- Karaman, S., Seidenari, L., Del Bimbo, A., 2014. Fast saliency based pooling of fisher encoded dense trajectories, in: *ECCV THUMOS Workshop*, p. 5.
- Laptev, I., 2005. On space-time interest points. *International journal of computer vision* 64, 107–123.
- Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B., 2008. Learning realistic human actions from movies, in: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, IEEE. pp. 1–8.
- Oneata, D., Verbeek, J., Schmid, C., 2014. The lear submission at thumos 2014. *THUMOS14 Action Recognition Challenge*.
- Peng, X., Wang, L., Wang, X., Qiao, Y., 2016. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *Computer Vision and Image Understanding* 150, 109–125.
- Perronnin, F., Sánchez, J., Mensink, T., 2010. Improving the fisher kernel for large-scale image classification, in: *European conference on computer vision*, Springer. pp. 143–156.
- Sánchez, J., Perronnin, F., Mensink, T., Verbeek, J., 2013. Image classification with the fisher vector: Theory and practice. *International journal of computer vision* 105, 222–245.
- Schuld, C., Laptev, I., Caputo, B., 2004. Recognizing human actions: a local svm approach, in: *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, IEEE. pp. 32–36.
- Shou, Z., Wang, D., Chang, S.F., 2016. Temporal action localization in untrimmed videos via multi-stage cnns, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1049–1058.
- Singh, B., Marks, T.K., Jones, M., Tuzel, O., Shao, M., 2016. A multi-stream bi-directional recurrent neural network for fine-grained action detection, in: *Computer Vision and Pattern Recognition (CVPR)*, 2016 IEEE Conference on, IEEE. pp. 1961–1970.
- Sivic, J., Zisserman, A., 2003. Video google: A text retrieval approach to object matching in videos, in: *IEEE*. p. 1470.
- Soomro, K., Zamir, A.R., Shah, M., 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.
- Su, Y.C., Grauman, K., 2016. Leaving some stones unturned: dynamic feature prioritization for activity detection in streaming video, in: *European Conference on Computer Vision*, Springer. pp. 783–800.
- Sun, C., Shetty, S., Sukthankar, R., Nevatia, R., 2015. Temporal localization of fine-grained actions in videos by domain transfer from web images, in: *Proceedings of the 23rd ACM international conference on Multimedia*, ACM. pp. 371–380.
- Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M., 2015. Learning spatiotemporal features with 3d convolutional networks, in: *Computer Vision (ICCV)*, 2015 IEEE International Conference on, IEEE. pp. 4489–4497.
- Wang, H., Schmid, C., 2013. Action recognition with improved trajectories, in: *Computer Vision (ICCV)*, 2013 IEEE International Conference on, IEEE. pp. 3551–3558.
- Wang, L., Qiao, Y., Tang, X., 2014. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge* 1, 2.
- Yao, A., Gall, J., Van Gool, L., 2010. A hough transform-based voting framework for action recognition, in: *Computer Vision and Pattern Recognition (CVPR)*, 2010 IEEE Conference on, IEEE. pp. 2061–2068.
- Yu, G., Yuan, J., 2015. Fast action proposals for human action detection and search, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1302–1311.
- Yuan, J., Liu, Z., Wu, Y., 2009. Discriminative subvolume search for efficient action detection, in: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, IEEE. pp. 2442–2449.
- Yuan, J., Pei, Y., Ni, B., Moulin, P., Kassim, A., 2015. Adsc submission at thumos challenge 2015, in: *CVPR THUMOS Workshop*, p. 2.