PAPER

Efficient Action Spotting Using Saliency Feature Weighting

Yuzhi SHI[†], **Takayoshi YAMASHITA**[†], **Tsubasa HIRAKAWA**[†], **Hironobu FUJIYOSHI**[†], **Mitsuru NAKAZAWA**[††], **Yeongnam CHAE**[††], ***and*** **Björn STENGER**[††],

SUMMARY Action spotting is a key component in high-level video understanding. The large number of similar frames poses a challenge for recognizing actions in videos. In this paper we use frame saliency to represent the importance of frames for guiding the model to focus on keyframes. We propose the frame saliency weighting module to improve frame saliency and video representation at the same time. Our proposed model contains two encoders, for pre-action and post-action time windows, to encode video context. We validate our design choices and the generality of proposed method in extensive experiments. On the public SoccerNet-v2 dataset, the method achieves an average mAP of 57.3%, improving over the state of the art. Using embedding features obtained from multiple feature extractors, the average mAP further increases to 75%. We show that reducing the model size by over 90% does not significantly impact performance. Additionally, we use ablation studies to prove the effective of saliency weighting module. Further, we show that our frame saliency weighting strategy is applicable to existing methods on more general action datasets, such as SoccerNet-v1, ActivityNet v1.3, and UCF101.

key words: *Action spotting, SoccerNet-v2, Frame saliency*

1. Introduction

Action spotting is a key component for understanding video content. Detecting actions in each frame provides rich information for video understanding applications and has application in video search, content moderation, and video summarization. In this paper we are interested in the task of action spotting, which can be used for video editing. Editing raw footage of sports games by hand to generate shorter summaries for broadcast is time-consuming. By spotting game-specific actions of interest, these can be selected specifically, summarizing the highlights of a particular game.

Action spotting is defined as the temporal localization of special actions within videos. Prior work has mainly focused on the question of how to model the temporal context by evaluating cost functions and different model structures. For example, new loss functions have been explored in CALF [2], where a loss function is proposed that weights features based on their temporal context. Frames that are temporally distant from an action are given a high cost, while closer frames are given a lower cost. The recent work NetVLAD++ [3] has shown the benefit of using two different encoders for pre-action and post-action temporal windows.

A less explored research direction is how to understand video content efficiently by focusing on keyframes. Videos contain many similar frames, which add little information to the task at hand. Much like a storyboard or cartoon strips, actions and their context can be expressed in far fewer pictures, if these contain sufficient information. To locate keyframes in videos, we use saliency to represent the importance of frames based on the similarity of frame features and take the frames with high saliency as keyframes. In this paper, we propose the frame saliency weighting module to calculate a saliency score for each frame and use these as feature weights for action spotting. Different from traditional saliency calculation methods, the proposed method improves video representation and frame saliency based on frame features for better performance and efficient inference. We show that this module has significant benefits in terms of guiding the model to extract meaningful representations. Experiments demonstrate that this frame saliency computation, based on frame similarity within temporal windows, is efficient and highly effective at improving video representation, while being significantly less complex than models using self-attention such as transformers [4]. As an additional benefit, focusing on the salient features allows the model to understand videos with less parameters and reduce the model size over 90% while maintaining high action spotting precision.

On SoccerNet-v2 [1], the proposed model reaches an 57.3% Average-mAP for action spotting, an absolute improvement of +3.9% with respect to the current state of the art. We use confidence scores and saliency scores to analyze the the performance of the proposed model in detail. By

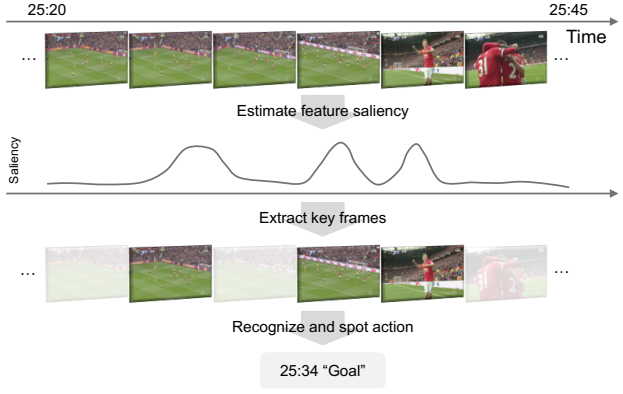

Fig. 1: **Action Spotting**. The action can be recognized from a single frame showing a cheering player. The proposed method estimates frame saliency in order to focus on discriminative keyframes for efficient action spotting.

Manuscript received January 1, 2015.
Manuscript revised January 1, 2015.
DOI: 10.1587/trans.E0.??.1

[†]All frames are taken from video footage in [1].

using multiple feature extractors and computing embedded features from these, the Average-mAP further increases to 75.0%. In extensive experiments, we validate the model structure, feature extractors, and hyper parameter choices. To highlight its generality and versatility, we show how our method can be used for video understanding tasks on the other public datasets and improve existing models.

In summary, (1) we propose the frame saliency weighting module to focus on salient information in videos by using temporal saliency weighting of feature vectors within temporal windows, (2) we conduct extensive experiments to analyze the performance of action spotting using saliency, (3) the proposed model achieves state-of-the-art performance in SoccerNet-v2 [1] on the task of action spotting, and improve existing methods on other tasks of video understanding.

2. Related Work

Video Understanding. Video understanding aims to extract the necessary information from videos that may contain many different types of actions. Several subtasks can be defined as action recognition [5], action detection [6], and video classification [7], etc. Action recognition typically targets atomic actions in short video clips. Action detection aims to find the time range in which an action takes place. However, the start and end times of an action are sometimes ambiguous and depend on the annotation in a specific dataset. For example, the 'soccer goal' action may only include the scene of kicking ball and the ball flying into the goal, or may include player running and cheering scenes. Video classification is the task of predicting a label that is relevant to the video.

Action Spotting. In the action spotting task, one or more action classes are predicted in every frame, including a background class when no specific action is detected. Example applications include action search, or the generation of highlight videos based on the estimated timestamps for actions of interest. Action spotting in sports videos is a challenging task owing to the large amount of data to be processed, rapid scene changes and the imbalance of action class labels. Previous work proposed several methods for action spotting in soccer videos. A regression and masking approach for soccer videos (RMS-Net) were introduced in [8], which ignores pre-action data and focuses on post-action frames during training. Recent work introduces a context-aware loss function, weighting frames at different temporal distances from the ground truth timestamp, as distant, just before and just after an action occurs [2]. The recent NetVLAD++ model proposes an architecture with two NetVLAD [9] modules to learn a context-sensitive vocabulary for past and future temporal context [3]. This model has shown to achieve high action spotting accuracy on SoccerNet-v2. Few attempts have been made to focus on the important frames which are related to actions for efficient video understanding. We explicitly address this problem by estimating the saliency of each frame, reducing the weight of redundant frames, and increasing the weight of keyframes that are important.

Frame Extraction. The purpose of frame extraction is to use as few video frames as possible to represent as much video content as possible. There are many previous work related to frame extraction, which select keyframes from video to generate video summaries or improve the performance of video understanding tasks [10], [11]. SCSampler [10] introduces a lightweight "clip-sampling" model that can efficiently identify the most salient temporal clips within a long video. Different with them, we identify the most salient frames within each video clips and improve the video representation at a finer granularity. SMART [11] proposed a method that, instead of selecting frames by considering one at a time, considers them jointly. Their frame selection module is independent of the backbone model, resulting in a complex training process and an increase in model parameters. While our model could locate keyframes and improve the performance at the same time.

Datasets. Many video datasets have been created for action detection [12] and classification of actions [13] and activities [7]. Large-scale datasets for specific sports have been compiled, such as GolfDB [14] and MLB-YouTube [15]. A dataset of 222 soccer videos was released in [16]. SoccerDB[17] merged a subset of 270 games from SoccerNet-v1 with 76 soccer games. SoccerNet-v1 [18] contains three action labels, card, goal and substitution. The recently extended dataset, SoccerNet-v2[1] includes over 300 thousand additional annotations and proposed novel tasks required for the automatic production of soccer broadcast videos. It contains 765 hours of footage of 500 soccer games and includes 17 different game-related action labels.

3. Methodology

The videos contain many similar frames in temporal segments (chunks) that are redundant for action spotting. Additionally, when an action occurs, the content of video changes, and the changed frame is different from others. Such frames appear when actions occur, consequently they should be focused on. The proposed method explicitly addresses the two facts. Considering that simply removing similar frames would reduce the duration of actions and change the content of video, we introduce an efficient frame saliency weighting module, which reduces the weight of redundant information and highlights keyframes that are important for the action spotting task. Action spotting models benefit from such distinct frames which have low inter-frame similarity and high saliency.

3.1 Frame Saliency Weighting

To focus on keyframes efficiently, we weight features with saliency, which is used to represent the importance of frames. The formula of computing a weighted frame feature is as follows.

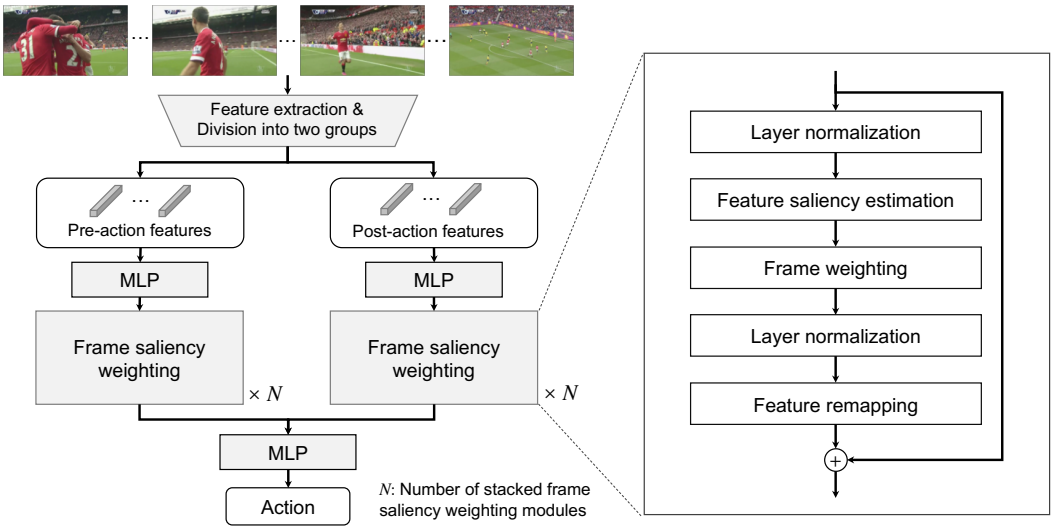

Fig. 2: **Proposed Network**. We propose a new model with two encoders. An encoder is stacked by frame saliency weighting modules. A frame saliency weighting module consists of a frame saliency estimator, feature weighting and remapping.

$$\boldsymbol{k}_i = \sum_{j=1}^{N_f} \frac{e^{-(s_{i,j}-\theta)}}{\sum_{m=1}^{N_f} e^{-s_{m,j}}} \boldsymbol{f}_i, \tag{1}$$

where $\boldsymbol{f}_i$ means the i^{th} feature in a chunk, $\boldsymbol{k}_i$ is the weighted feature vector of the i-th frame, N_f is the number of frames in a chunk, and θ is a hyper parameter to adjust feature weights. When not explicitly stated, the value of θ is 0. By weighting features with saliency, model could locate keyframes and improve video representation efficiently. When action happens, the frame and frame feature would change. Consequently, we consider that locating distinct frames according to the similarity of frame features is an efficient way to detect actions. $s_{i,j}$ represents the similarity of i-th and j-th frame features. The calculation of $s_{i,j}$ is as follows.

$$s_{i,j} = \boldsymbol{f}_i^{\mathrm{T}} \boldsymbol{f}_j. \tag{2}$$

If $s_{i,j}$ is high, we consider the i-th and j-th frames similar and the content of them redundant.

To further improve the representation, we subsequently apply feature remapping using two fully connected layers and a ReLU layer.

$$\boldsymbol{k}_{\mathrm{remap}} = \max\left(0, \boldsymbol{k}\boldsymbol{W}_1 + \boldsymbol{b}_1\right)\boldsymbol{W}_2 + \boldsymbol{b}_2 + \boldsymbol{f}, \tag{3}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are the weights and $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are the biases of the fully connected layers, respectively. $\boldsymbol{f}$ means the features of all frames in a chunk and $\boldsymbol{k}$ represents the weighted features. This step improve the representation ability of the model via mapping features into a higher-dimensional space and projecting them back into the original space. Finally, we use a residual structure to solve the gradient degradation problem and use two layer-normalization to make training more stable.

3.2 Model Architecture

The structure of the proposed model is shown in Figure 2. When an action occurs, the scenes before and after the action are generally different, especially in soccer videos. In order to locate actions by detecting scene changing before and after an action, we split video clip features into two parts of the same length, pre-action and post-action, and use two encoders to process each part, respectively. Given an image feature dimension L and the number of frames in a chunk N_f, the shape of the output of the feature extractor is $N_f \times L$. We use a fully connected layer to reduce the feature dimensions for efficiency and for a fair comparison among different feature extractors, which output features with different length. The size of the matrix output by the fully connected layer is $N_f \times E$, where E is the output dimension of the fully connected layer. The input of the remapping module is an $N_f \times E$ matrix, which is expanded to an $N_f \times 2E$ matrix by a fully connected layer. After a ReLU activation layer, it uses another fully connected layer to remap the features to a $N_f \times E$ matrix. In this manner, the shape of input data of frame saliency module is the same with the output data. Therefore an encoder could contains N frame saliency weighting module. In the last Multilayer Perceptron (MLP) module, we concatenate the output of two encoders and average them in the temporal direction. Finally, we use a softmax activation layer, a fully connected layer, and a sigmoid layer to output the action classification result.

3.3 Implementation Details

We use a binary cross-entropy loss, which we optimize using an Adam optimizer, an initial learning rate of 0.003. We set the batch size to 64 and the chunk size to 15 seconds. We carry out non-maximum suppression (NMS) [18] over

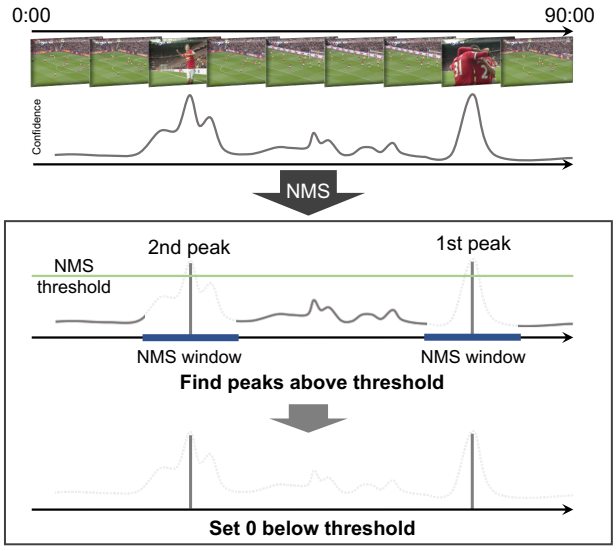

Fig. 3: **NMS Process**. To improve spotting results, we use non-maximum suppression to handle prediction. NMS process is used on confidence of a whole video each class.

a 30 second window with a threshold 0. We use the model obtaining the highest mAP value on the validation dataset as the final model and use it to evaluate on the test dataset.

NMS. To reduce the false detection rate, low-confidence predictions are filtered using non-maximum suppression (NMS), which is the same as in previous works [1]–[3], [18]. The NMS process is shown in Figure 3. For each class, we find the first peak of confidence from all frames and set the confidence of the rest frames in NMS window to 0. Then, we find the next peak and perform the same process until all peaks above the NMS threshold are found. Finally, we set the confidences below the threshold to 0.

4. Experiments

In experiments we evaluate the performance of the proposed model. We visualize the confidence graph as well as the saliency graph to analyze the results, and conduct extensive experiments to analyze the influence of chunk size and the number of encoders, etc. To evaluate the effectiveness of the frame saliency weighting module, we compare the proposed models with fixed saliency and learnable saliency. Further, to evaluate the generality of the proposed method, we use the proposed module to improve the video representation for action spotting, temporal action proposal generation, and video classification using SoccerNet-v1, Activity-v1.3, and UCF101, respectively.

Dataset. We use the public SoccerNet-v2 dataset [1], which contains video footage and annotations of 500 soccer games to train and evaluate our method. We split the dataset into training, validation, and testing sets following the same procedure as the original paper (300, 100, and 100 games, respectively). The frame rate of videos is 2 frames for each second. The ground truth for each frame is a label vector. The label vector contains 17 different action labels as well as a label for the background. In the training process, videos are divided into chunks, and each chunk is annotated by a label vector. The action label in label vector is set as 1 if the corresponding action occurs in the chunk and other labels are set as 0. If none of the 17 actions appears in the chunk, then the background label is set to one. If there are a goal action and two play out of ball actions happen in the chunk, the labels of goal and play out of ball are 1 and other labels are 0 in the label vector.

Video Encoding. The feature extractor maps every video frame to a feature vector. We evaluated three different feature extractors, ResNet-152 [19], ResNet-152 with PCA, and embedding features [20]. The ImageNet [21] pretrained ResNet-152 model extracts 2048-dimensional features for each frame. The feature of ResNet-152 with PCA are further reduced to 512 dimensions using PCA. Embedded features [20] are computed by passing frames to five networks pretrained on other datasets, and then embed all features to obtain an 8576-dimensional feature vector. In advance, features are computed for all frames, where frames are scaled to 224×224 pixels before passing them to feature extractors, respectively.

Evaluation Protocol. Action spotting task require models to output a multi-label classification prediction for each frame. If the distance between the ground truth timestamp and the predicted timestamp is less than Δ seconds, the prediction is considered positive. Δ is a threshold ranging from 5-60 seconds using a 5 second step size. We calculate the average precision (AP) for each action class and each Δ. The mean average precision (mAP) score is calculated by taking the mean AP over all classes with Δ. The Average-AP is the average of 12 AP values calculated over 12 tolerances Δ for each class. The Average-mAP metric is the average of 12 mAP values calculated over 12 tolerances Δ.

4.1 Action Spotting

Table 1 shows the results of our proposed model as well as the models from the literature. We achieve 57.3% Average-mAP, representing an absolute increase of 3.9% over the previous state of the art. NetVLAD [9] and MaxPool pooled features regardless of their temporal relationship between frames. The NetVLAD++ model uses two NetVLAD pooling modules to handle the first and second half of every chunk, respectively. However, it does not consider the importance of frames within each half chunk, treating them equally. In contrast, the proposed model uses the saliency of frames as feature weights to focus on keyframes and learn the importance of each frame.

Performance decreases when increasing the stride value from 1 to 20 frames, however, with an Average-mAP of 53.1% it is an absolute 5% higher than NetVLAD++. This means that our model can run the inference at a significantly higher speed for a similar average mAP value. Additionally, the model still outperforms NetVLAD++ in terms of Average-mAP at one tenth of the model size. Consequently,

Table 1: **Action Spotting Results.** Evaluation results in terms of Average-mAP, where available, on the SoccerNet-v2 dataset with two different stride values during testing. N is the number of frame saliency weighting modules and E is the feature dimension output by the first fully connected layer.

Model	Feature Extractor	Stride 1	Stride 20	Size/Param
NetVLAD	ResNet+PCA	31.4	-	7.50MB/0.66M
CALF	ResNet+PCA	40.7	-	6.64MB/0.58M
MaxPool	ResNet+PCA	18.6	-	0.11MB/0.01M
NetVLAD++	ResNet+PCA	50.7	46.7	7.50MB/0.66M
NetVLAD++	ResNet	53.4	48.1	19.50MB/1.70M
Ours [$N = 1, E = 512$]	ResNet+PCA	**56.0**	**51.0**	27.24MB/2.38M
Ours [$N = 1, E = 64$]	ResNet	54.0	49.5	**1.93MB/0.17M**
Ours [$N = 2, E = 256$]	ResNet	56.9	53.0	18.19MB/1.59M
Ours [$N = 1, E = 512$]	ResNet	**57.3**	**53.1**	66.29MB/3.16M

the frame saliency weighting module is an elegant and efficient structure for action spotting.

Confidence Score. To analyze the experiment results, we visualize the results via using a confidence score graph, shown in Figure 4. The confidence score is the prediction of the model before processed by NMS. From (a) and (d) in Figure 4, we observe that for the *goal* action label, confidence is high after the action itself, corresponding to scenes of celebration by players and fans. *Offside* actions typically include scenes where one of the referee raises a flag. In Figure 4 (b), the scene where a referee raising the flag occurs for 6 seconds after the offside action itself. In Figure 4 (e), no referee scene appears, and the confidence of an offside action is low. Figure 4 (c) shows a *substitute* action, and confidence is high for the frames 10 seconds before and after the action, as the camera typically focuses for several seconds on the player running to the sideline and the substitute player entering the field. The confidence of the substitution action is close to zero after a substitution as shown in Figure 4 (f). Confidence score could indicate the scenes related to specific actions and the reason for the prediction results.

Saliency Score. To explore whether the saliency score is critical for action spotting, we plot saliency score graphs of six samples. The saliency score is calculated by the last frame saliency estimation module in the model. We set the stride length to 15 seconds in inference, equal to the chunk size used in training and concatenate the saliency scores in every chunk to obtain the saliency of all frames. From Figures 5 (a) and (d), we observe that high-saliency scores appear at the time when a referee appears or players fall to the ground, which are salient features to spot a *foul* action. For *goal* actions, the model focuses on the frames in which the ball enters the goal. Because the saliency score is high on such frames, shown in Figures 5 (b) and (e). In Figures 5 (c) and (f), the frames in which a referee raises a yellow flag have high saliency scores, which are focused when locating off-side action. From these results, we can observe that the saliency score of the frames that are related to actions is high. Therefore the frame saliency weighting module can focus on action-related frames and improve video representation by using the saliency scores as feature weights. Furthermore, the saliency score can help us to interpret the prediction results and help the proposed model be applied in the areas where interpretable results are required. The frame-level saliency scores are helpful to improve model performance.

4.2 Influence of Chunk Size and Number of Encoders

We analyze the effect of the number of encoders, as well as the effect of the chunk size using the Average-mAP metric. The results are presented in Table 2.

Chunk Size. Because our model is trained for recognizing action in every chunk, the chunk size is a necessary hyper parameter. If the chunk size is too large, a chunk would contain multiple actions which can affect each other and make the training challenging. If there is at most one action within a chunk, the adjacent chunks do not affect each other. The model needs to memorize $C + 1$ types of patterns, where C is the number of action classes in the dataset. If at most two actions occur in a chunk, the model needs to memorize $C^2 + 1$ types of patterns. On the other hand, if the chunk size is too small, video chunks where no action happens would increase and result in a more unbalanced dataset. Therefore, selecting an appropriate chunk size is important for action spotting. As shown in Table 2, 15 seconds is an appropriate chunk size, which is also reported in NetVLAD++[3]. We consider that the appropriate chunk size is determined by duration of frames which are related to specific actions.

Number of Encoders. When an action occurs, the scene of the video would change. For example, before a goal action, players are running to the goal, while after a goal is scored, players celebrate and assemble. A model with two encoders is able to detect such scene changes easily, by learning from pre-action and post-action scenes, respectively. The model with two encoders performs best, as shown in Table 2. Models with three or four encoders increase the complexity of model structure while not further improving accuracy. We hypothesize that when increasing the number of encoders with a fixed chunk size, the temporal segments feeded into each encoder will be smaller and they will contain less temporal information. It increases the difficulty of capturing temporal correlations within frames in a chunk.

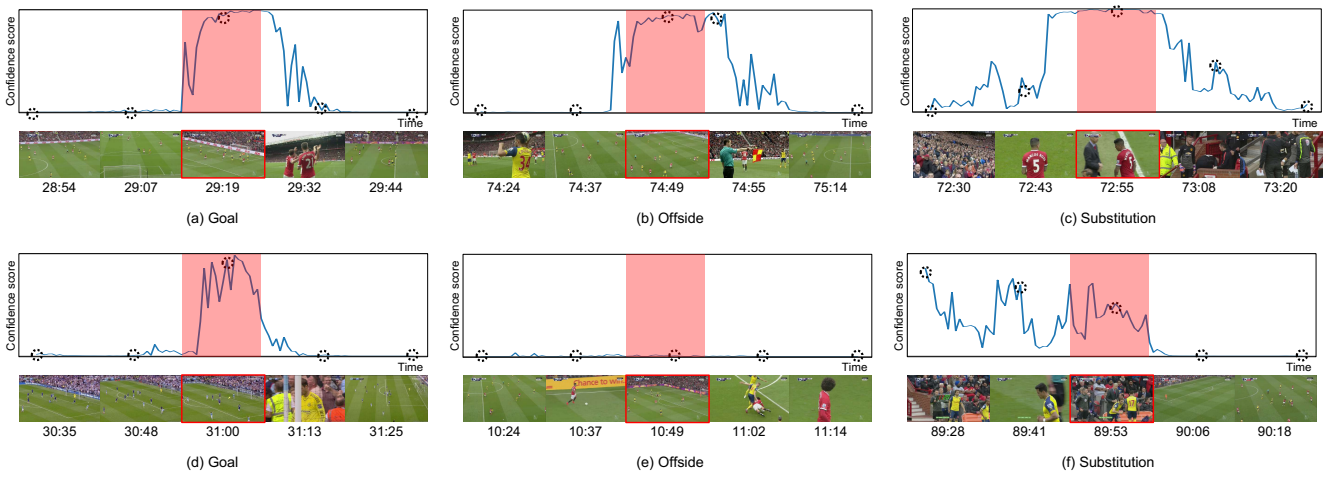

Fig. 4: **Confidence Score Examples**. Labeled actions from the SoccerNet-v2 dataset [1] are shown in frames marked by red boxes. The confidence scores for the adjacent frames are marked by circles in the graph. The translucent red rectangle indicates the range within 5 seconds from the ground truth timestamp.

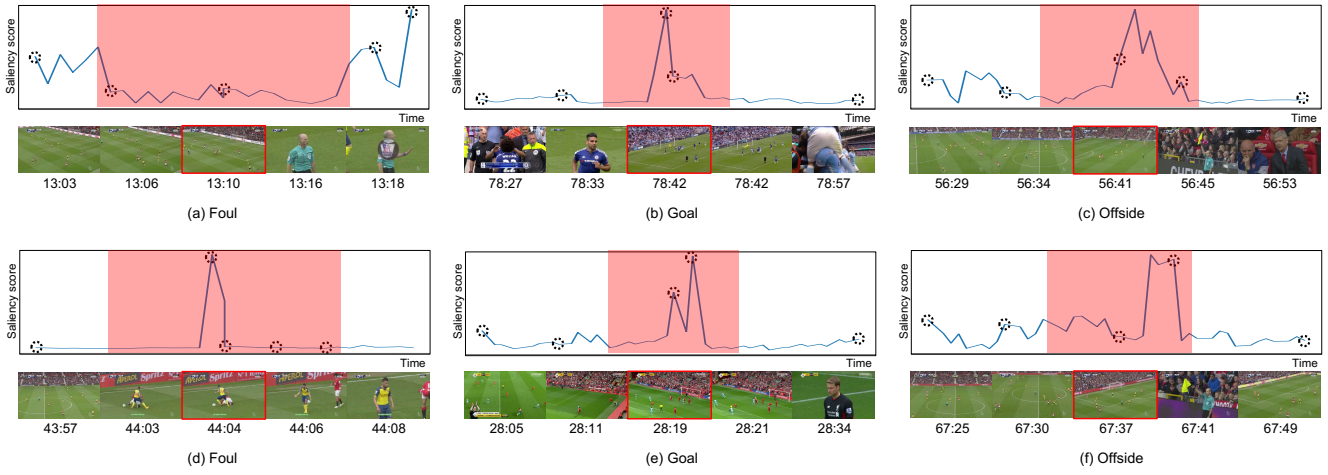

Fig. 5: **Saliency Score Examples**. Labeled actions from the SoccerNet-v2 dataset [1] are shown in frames marked by red boxes. The saliency scores for the adjacent frames are marked by circles in the graph. The translucent red rectangle indicates the range within 5 seconds from the ground truth timestamp.

Table 2: **Comparison of Chunk Sizes and Numbers of Encoders**. We evaluate the Average-mAP for chunk sizes from 10 to 30 seconds, while changing the number of encoders from 1 to 4, using ResNet-152 as the feature extractor. The highest value was obtained for a model using two encoders and a chunk size of 15 seconds.

Chunk size (s)	Number of encoders			
	1	2	3	4
10	50.98	54.92	54.96	54.30
15	52.90	**57.32**	55.51	55.47
20	50.81	54.97	53.69	54.01
25	48.12	53.30	53.03	51.41
30	45.69	50.99	49.55	49.52

4.3 Influence of Model and Feature Extractor

We analyze the effect of the model and feature extractor, and compare the proposed method with self-attention using the Average-mAP metric. The results are presented in Tables 3 and 4.

Model. We compare our proposed method with NetVLAD++ [3] and a transformer-based model [4] using different feature extractors. As shown in Table 3, our proposed method achieves the best performance. NetVLAD++ leverage two NetVLAD encoders to learn temporal information, however it could not capture the importance of each frame. The transformer-based model only have one encoder, which leads to the difficulty of locating the frame changes before and after actions.

Feature Extractor. The embedding features extracted by five models [20] greatly improve the performance of all methods, highlighting the importance of efficient video representations for the action spotting task. For any feature

Table 3: **Comparison of Models and Feature Extractors** in terms of Average-MAP. Our proposed method achieves the best performance in all choices of the feature extractor.

Feature Extractor	Transformer	NetVLAD++	Ours
ResNet+PCA	47.8	50.1	**56.0**
ResNet	48.3	53.4	**57.3**
Embedding	73.8	74.1	**75.0**

Table 4: **Comparison with Self-attention**. ResNet-152 is used as feature extractor. We develop two self-attention based models with an encoder and two encoders respectively.

Model	Average-mAP	Size (MB)
Self-attention (1 encoder)	48.3	97.2
Self-attention (2 encoders)	53.0	249.0
Ours (2 encoders)	**57.3**	**66.3**

extractor, our proposed method reaches a better result than other models. We consider that the proposed method would improve the presentation of video features by increasing the weights of keyframes.

Comparison with Self-attention Model. In order to compare with self-attention, we develop two self-attention based models with a transformer encoder and two transformer encoders, respectively. Our method reaches a higher Average-mAP, as presented in Table 4. We consider that it is because the frame saliency weighting module does not have three fully connected layers to calculate the vectors K, Q, V in a transformer encoder [4]. The calculation of K, Q, V changes the mapping results of the feature extractor and reduces the ability of capturing the similarity among frames. In addition, the three fully connected layers increase the model parameters. Different from them, our model learns to calculate the saliency of each frame based on the similarities of the learnable frame features.

4.4 Comparison with Frame Extraction Methods

To further demonstrate the effectiveness of our method, we leverage three frame extraction methods to select frames before feeding features into the proposed model, and compare with different methods. Keyframe extraction is an efficient method used to clearly express the important contents of a video file by extracting a set of representative frames and removing the duplicated ones. The techniques of keyframe extraction can be classified into three main classes: sampling-based, shot based and clustering-based techniques [22]. We develop these methods, and the implementation details are as follows. The sampling-based method randomly selects 10 frames from every chunk containing 30 frames. The shot-based method splits every video chunk into several shots, and select the center frame in each shot as a keyframe. The cluster-based method clusters all 30 frames in every chunk into 10 frames. The model without keyframe extraction module achieve the best performance, as shown in Table 5. We consider that selecting frames would change the original temporal information and influence the performance of action spotting. Compared to them, our method can focus on the keyframes and learn to locate keyframes based on the inter-frame similarity and annotation labels automatically without changing the original temporal relationship. Therefore, the proposed method achieves the best performance.

Table 5: **Comparison with Keyframe Extraction Methods**. We use Average-mAP on SoccerNet-v2 as the metric and ResNet152 with PCA as the feature extractor. The chunk size is 15 seconds, and frame rate is 2.

Method	Average-mAP
Sampling-based	49.0
Shot-based	38.0
Cluster-based	46.9
Ours	**56.0**

Table 6: **Comparison of Fixed Saliency and Variable Saliency**. In the model with fixed saliency, we set 1.0 to all saliency scores to ignore the influence of saliency.

Model	Average-mAP
Fixed saliency	49.5
Variable saliency	**56.0**

Table 7: **Comparison NetVLAD++ with and without proposed method**. NetVLAD++(*) denotes NetVLAD++ with our saliency weight module. Training time is the time used for obtaining the highest mAP on the validation dataset.

Model	Average-mAP	Training time(Second)
NetVLAD++	53.4	368.9
NetVLAD++(∗)	**54.5**	**292.7**

4.5 Importance of Saliency

We compare the performance of our method with fixed and learnable saliency on SoccerNet-v2 dataset to analyze the influence of the proposed saliency weighting module. The features are extracted using ResNet with PCA. To remove the influence of saliency score, we fix all saliency scores in the pretrained model as 1, as a result, the weight of every frame is equal, and the influence of saliency score can be ignored. Without saliency weighting, the average mAP of the model drops to 49.5%, a decrease by absolute 6.5% as shown in Table 6. The saliency weighting module improves the Average-mAP for action spotting by weighting frames using learnable saliency scores.

4.6 Generality of Saliency

4.6.1 Improvement of NetVLAD++

To prove that the frame saliency weighting module could find the keyframes and improve feature representation by

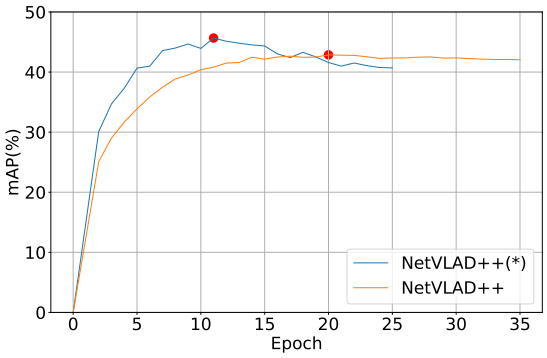

Fig. 6: **Change of mAP with respect to Epochs on the Validation Dataset.** The changes of NetVLAD++ and NetVLAD++(∗) during training are shown respectively.

weighting features through saliency scores, we develop a model by adding a frame saliency weighting module before the NetVLAD++ model, denoted by NetVLAD++(∗). The learnable frame saliency weighting module could be seen as a preprocessing module. As shown in Table 7, by using the frame saliency weighting module to improve frames, the performance of NetVLAD++ is improved by 1.1%. Additionally, by using frame saliency, the training time is reduced. As presented in Figure 6, NetVLAD++ obtains the highest mAP at epoch 20 using 368.9 seconds, whereas NetVLAD++(∗) obtains the highest mAP at epoch 11 with 292.7 seconds. The frame saliency module gives larger weights to keyframes, which encourages the model to focus on important frames and understand video content from keyframes effectively.

4.6.2 Evaluation on Other Datasets.

To demonstrate the applicability of the proposed model to different public datasets and different tasks, we report the evaluation results on three additional datasets, SoccerNet-v1 [18], ActivityNet v1.3 [12], and UCF101 [23].

SoccerNet-v1 is the predecessor of SoccerNet-v2, released by the same authors. It has 6,637 temporal annotations, including three classes of actions (goal, card, and substitution). SoccerNet-v1 contains fewer annotations and thus longer intervals between actions than SoccerNet-v2. There are no 120-second frame intervals containing more than five actions in the SoccerNet-v1 dataset. On the contrary, SoccerNet-v2 has up to 14 actions in 120-second intervals. ActivityNet v1.3 [12] is a large-scale dataset consisting of 19,994 videos with 200 activity classes for action recognition, temporal action proposal generation and detection. The UCF101 dataset consists of 13,320 video clips, which are classified into 101 categories. All the videos are collected from YouTube and have a fixed frame rate of 25 FPS with the resolution of 320×240.

Experiments were conducted for three video understanding tasks, action spotting, temporal action proposal generation, and action recognition.

Table 8: **Results on SoccerNet-v1.** We use Average-mAP to evaluate all methods.

Method	Feature Extractor	Average-mAP
NetVLAD	ResNet+PCA	49.7
CALF	ResNet+PCA	62.5
NetVLAD++	ResNet+PCA	61.1
Ours	ResNet+PCA	**65.0**
Ours	ResNet	**68.1**

Table 9: **Results on ActivityNet v1.3** [12]. Evaluation results in terms of AUC score for temporal action proposal generation. The results are obtained on fully-supervised training. SSTAP (+saliency) use frame saliency weighting module to improve video presentation before the SSTAP model.

Method	AUC
SSTAP	67.5
SSTAP (+saliency)	**67.6**

SoccerNet-v1. In this experiment, we report Average-mAP for evaluation. The results on SoccerNet-v1 are shown in Table 8. Consistent with the results on SoccerNet-v2, the proposed method achieves the highest Average-mAP with 68.1% using ResNet features, an increase of +7.0% over NetVLAD++. An interesting observation is that in contrast to the SoccerNet-v2 dataset, CALF [2] outperforms NetVLAD++ on the SoccerNet-v1 dataset. The loss function in CALF defines six temporal segments around each ground-truth action to include temporal relationships. When actions occur frequently in a dataset such as SoccerNet-v2, their temporal segments maybe overlap, resulting in decreasing performance. Therefore, CALF is better suited for datasets with fewer annotations and longer intervals between actions, such as SoccerNet-v1.

ActivityNet v1.3. Using ActivityNet v1.3, we evaluate methods by the performance of temporal action proposal generation. To adapt our approach to the task, we add our saliency weighting module as the first module in SSTAP [24], a recent method based on self-supervised learning for the detection of temporal actions, and develop the SSTAP (+saliency) model. We measure the average recall (AR) for different average numbers of proposals (AN) as AR@AN, and calculate the area under the AR *vs.* AN curve (AUC) as the metric on ActivityNet v1.3, where AN varies from 0 to 100. The results are shown in Table 9. SSTAP with our added saliency module improves the AUC score by 0.1%, confirming the effectiveness of the saliency module for the task of generating temporal action proposals and improving video features. In addition, using the proposed module, the importance of frames could be provided to interpret the prediction of existing models.

UCF101 To confirm the effectiveness of the proposed model for action recognition, we develop another model by

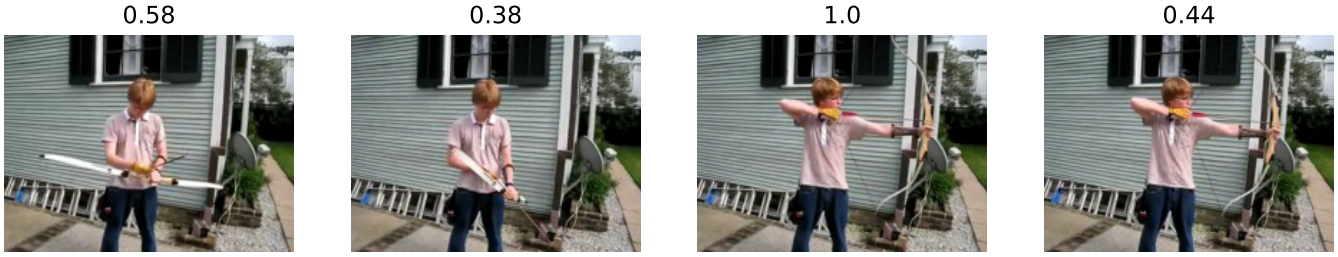

Fig. 7: **Saliency Score of the Archery Action**.

Table 10: **Action Recognition on UCF101.** SlowFast(*) is the combination model of SlowFast and the proposed model.

Model	Accuracy
SlowFast	85.4
SlowFast(∗)	**88.6**

combining the proposed model with SlowFast [25], denoted by SlowFast(*), and compare it with SlowFast using the UCF101 [23] dataset. The SlowFast is pre-trained on Kinetics 400 dataset [26]. As shown in Table 10, the performance is improved by 3.2% by using the proposed model as another branch. The proposed model could capture important information that the SlowFast might miss.

To evaluate the interpretability of the proposed method, the proposed method generates some saliency scores of the archery action using the video from the UCF101 dataset, as shown in Figure 7. The durations of the videos from the UCF101 dataset are short and each video only contains one type of action. Therefore most of the frames in the videos are related to the action label. The saliency scores of these videos are higher than other dataset. The saliency score is high on the third frame, where a man is holding a bow and an arrow than on other frames. The frame is related to the archery action. Therefore, the proposed model could recognize the action by locating keyframes.

5. Conclusion

In this paper we addressed the problem of redundant information in videos by focusing on keyframes. We proposed the saliency weighting module which weights frames according to feature similarity to reduce the influence of redundant frames and improve frame saliency and representation at the same time. We analyzed the performance of our proposed model and showed that the proposed model achieved state-of-the-art accuracy on SoccerNet-v2, obtaining 57.3% Average-mAP. Additionally, our proposed model obtains the 75.0% Average-mAP using embedding features. Furthermore, we indicate that our saliency weighting module can be effectively applied to existing video understanding methods.

References

[1] A. Deliège, A. Cioppa, S. Giancola, M.J. Seikavandi, J.V. Dueholm, K. Nasrollahi, B. Ghanem, T.B. Moeslund, and M. Van Droogenbroeck, "SoccerNet-v2 : A dataset and benchmarks for holistic understanding of broadcast soccer videos," CVPRW, pp.4508–4519, 2021.

[2] A. Cioppa, A. Deliège, S. Giancola, B. Ghanem, M. Van Droogenbroeck, R. Gade, and T.B. Moeslund, "A context-aware loss function for action spotting in soccer videos," CVPR, pp.13126–13136, 2020.

[3] S. Giancola and B. Ghanem, "Temporally-aware feature pooling for action spotting in soccer broadcasts," CVPRW, pp.4490–4499, 2021.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L.u. Kaiser, and I. Polosukhin, "Attention is all you need," NeurIPS, pp.5998–6008, 2017.

[5] M.E. Kalfaoglu, S. Kalkan, and A.A. Alatan, "Late temporal modeling in 3d cnn architectures with bert for action recognition," ECCV, pp.731–747, 2020.

[6] R. Dai, S. Das, L. Minciullo, L. Garattoni, G. Francesca, and F. Bremond, "PDAN: Pyramid dilated attention network for action detection," In Winter Conference on Applications of Computer Vision, pp.2970–2979, 2021.

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," CVPR, pp.1725–1732, 2014.

[8] M. Tomei, L. Baraldi, S. Calderara, S. Bronzin, and R. Cucchiara, "RMS-Net: Regression and masking for soccer event spotting," ICPR, pp.7699–7706, 2021.

[9] A. Relja, G. Petr, T. Akihiko, P. Tomas, Josef, and Sivic, "NetVLAD: Cnn architecture for weakly supervised place recognition," CVPR, pp.5297–5307, 2016.

[10] B. Korbar, D. Tran, and L. Torresani, "Scsampler: Sampling salient clips from video for efficient action recognition," CoRR, 2019.

[11] S.N. Gowda, M. Rohrbach, and L. Sevilla-Lara, "Smart frame selection for action recognition," AAAI, 2021.

[12] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," CVPR, pp.961–970, 2015.

[13] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," CVPR, pp.6299–6308, 2017.

[14] W. McNally, K. Vats, T. Pinto, C. Dulhanty, J. McPhee, and A. Wong, "Golfdb: A video database for golf swing sequencing," CVPRW, pp.2553–2562, 2019.

[15] A. Piergiovanni and M.S. Ryoo, "Fine-grained activity recognition in baseball videos," CVPRW, pp.1740–1748, 2018.

[16] J. Yu, A. Lei, Z. Song, T. Wang, H. Cai, and N. Feng, "Comprehensive dataset of broadcast soccer videos," Multimedia Information Processing and Retrieval, pp.418–423, 2018.

[17] Y. Jiang, K. Cui, L. Chen, C. Wang, and C. Xu, "Soccerdb: A large-scale database for comprehensive video understanding," Proceedings of the 3rd International Workshop on Multimedia Content Analysis in Sports, pp.1–8, 2020.

[18] S. Giancola, M. Amine, T. Dghaily, and B. Ghanem, "SoccerNet: A scalable dataset for action spotting in soccer videos," CVPRW, pp.1824–1834, 2018.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," CVPR, pp.770–778, 2016.

[20] X. Zhou, L. Kang, Z. Cheng, B. He, and J. Xin, "Feature combination

meets attention: Baidu soccer embeddings and transformer based temporal detection," arXiv:2106.14447, 2021.

[21] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," CVPR, pp.248–255, 2009.

[22] B.O. Sadiq, B. Muhammad, M.N. Abdullahi, G. Onuh, A.A. Muhammed, and A.E. Babatunde, "Keyframe extraction techniques: A review," ELEKTRIKA-Journal of Electrical Engineering, vol.19, no.3, pp.54–60, 2020.

[23] K. Soomro, A.R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," arXiv preprint arXiv:1212.0402, 2012.

[24] X. Wang, S. Zhang, Z. Qing, Y. Shao, C. Gao, and N. Sang, "Self-supervised learning for semi-supervised temporal action proposal," CVPR, pp.1905–1914, 2021.

[25] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," Proceedings of the IEEE/CVF international conference on computer vision, pp.6202–6211, 2019.

[26] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp.6299–6308, 2017.