Feature Aggregation Tree: Capture Temporal Motion Information for Action Recognition in Videos

Bing Zhu(✉)

Beijing Laboratory of Intelligent Information Technology,
School of Computer Science, Beijing Institute of Technology (BIT),
Beijing 100081, People's Republic of China
zhubing@bit.edu.cn

Abstract. We propose a model named Feature Aggregation Tree to capture the temporal motion information in videos for action recognition. Feature Aggregation Tree constructs a logical motion sequence by considering the concrete semantics of features and mining feature combinations in a video. It will save different feature combinations and then use the bayesian model to calculate the conditional probabilities of frame-level features based on the previous features to aggregate features. It doesn't matter about the length of the video. Compared with the existing feature aggregation methods that try to enhance the descriptive capacity of features, our model has the following advantages: (i) It considers the temporal motion information in a video, and predicts the conditional probability by using the bayesian model. (ii) It can deal with arbitrary length of the video, rather than uniform sampling or feature encoding. (iii) It is compact and efficient compared to other encoding methods, with significant results compared to baseline methods. Experiments on the UCF101 dataset and HMDB51 dataset demonstrate the effectiveness of our method.

Keywords: Action recognition · Feature learning
Feature aggregation

1 Introduction

Human action recognition [1] is one of the fundamental researches in the field of computer vision, which has great significance and application prospects in video retrieval, video recommendation and video surveillance. In recent years, many researches mainly focuse on two aspects. One is how to extract a more discriminative spatio-temporal description for the video. The other is how to aggregate frame-level features to a video-level feature, which gives more attention to efficient feature organization strategies.

J.-H. Lai et al. (Eds.): PRCV 2018, LNCS 11258, pp. 316–327, 2018.
https://doi.org/10.1007/978-3-030-03338-5_27

In terms of feature description, most of the existing video feature representations for action recognition are mainly learned by two different types of networks: one is two-stream network [2,3] and the other is 3D convolutional neural network [4–6]. The trend of networks is to learn better video features which can capture both spatial and temporal information in videos. And we need a strategy to handle long videos with arbitrary frames, which can aggregate frame-level features to a representation for the whole video.

In terms of feature aggregation, one strategy is selecting a key frame or several key frames to represent the entire action video [7–9]. This strategy can achieve satisfactory results when a video contains only one action instance, but it is not so useful in the videos containing multiple categories action instances. Another common strategy is to encode frame features, such as vectors of locally aggregated descriptors (VLAD) [10], fisher vectors (FV) [11,12] and bag of words (BoW) [13,14]. While these strategies cannot capture the temporal information of the entire video. In addition, in the neural network methods, the temporal pooling operation is usually used to compress the features of a video [3,15,16], e.g. the mean and the max pooling. There are also some recent works trying to modify the traditional pooling strategies to further improve the recognition performance, such as adascan [17] and ActionVLAD [18], which attaches frame features to different wight values. However, the pooling strategies don't consider the order of frames, which ignore the temporal information. Besides the CNNs, the LSTM network is also considered to use attention mechanism to learn the weight of different each frame [19–21]. But because of the complexity of the training process, LSTM doesn't become a mainstream method.

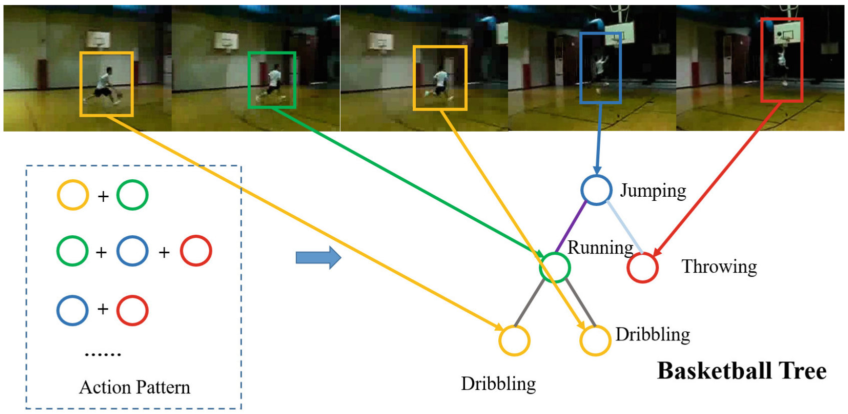


Fig. 1. We propose Feature Aggregation Tree to represent actions in videos. For example, Action "Basketball" can be grouped by "running", "dribbling", "jumping" and "throwing". We construct a "basketball" tree to record action primitives nodes and excavate the action pattern between two action primitives.

To better understand what an action is performing, temporal information is as important as spatial information. However, no matter hand-crafted features or deep features, are all frame-level, which don't make full use of the temporal information of the entire video. To better use the temporal information, we need to understand the component of actions firstly. The hierarchical definition proposed by Moeslund [22] divides the actions into three levels, e.g. the lower level definition is "action primitive", the middle level definition is "action" which is an ordered combination of primitives, while the concept of high-level definition of "behavior" is the logical combination of actions, which is a complex advanced semantics. Taking the action of basketball as an example, shown in Fig. 1, playing basketball can be broken down into several action primitives like "running", "dribbling", "jumping" and "throwing"and these primitives are organized in a temporal order. Actions have different meanings in different orders, such as "running-throwing" means playing basketball, while "running-jumping" means high jump or long jump. And these temporal information involved in the patterns will be helpful in action recognition. The method ActionVLAD with the similar idea proves the effectiveness. In this work, we propose a novel method named Feature Aggregation Tree (FA-Tree) to learn video features for action recognition, which is based on the knowledge of frequent patterns and association rules in the field of data mining [23].

The main contribution of this work is that we propose a novel FA-Tree for action recognition, which has the following advantages: (i) The method treats frame-level features as action primitives, and aggregate them into action patterns. Taking the temporal information of primitives into account, FA-Tree organizes patterns with different orders to better represent a complete action, and then calculate the precise conditional probability of an action. (ii) The method can deal with arbitrary length of the video, rather than uniform sampling or feature encoding. (iii) The model is compact and efficient, and has achieved good results on two datasets.

2 Related Work

Action Feature Representation. In recent years, more and more researchers want to extract more discriminative features to represent a video, which should contain temporal information as well as spatial information. Some hand-crafted traditional features [1,24,25] are proposed from 2D to 3D, and their description ability has been significantly improved. It is worthy mentioning that Wang et al. [26] proposed improved Dense Trajectories (iDT), which is the best hand-crafted feature at present but it is computationally intensive. Simonyan and Zisserman [3] proposed the two-stream network, which decomposed a video into appearance and motion streams, and trained two networks respectively. Considering that the input of 2D convolutional neural networks is always an image so it lacks the temporal information, the 3D neural network uses the video segment as the input [4–6].

Video Feature Aggregation. One approach is to select a key frame or a key segment to replace the entire video when predicting the action category. Cao et al. [7] extracted the key frame with manifold learning based on the optical flow graph for action recognition. Liu et al. [8,9] used supervised learning and unsupervised clustering methods to extract key segments in action videos.

Another approach is feature encoding. Some methods use the bag of words model (BoW) [14] to extract some local spatio-temporal descriptor, and encode them into dictionaries to make templates [13,15,27,28]. Latev et al. [27] described a video with BoW that encoded HoG and HoF features. Ji et al. [5] also used BoW in their method. Similar to BoW are the methods such as VLAD [10,18] and Fisher Vector [11–13]. Wang et al. [15] proposed the improved Dense Trajectories(iDT) approach, which combined dense trajectories, histogram by using Fisher Vector to encode. By combining iDT [26] features and Fisher Vector [29] algorithm, Peng et al. [13] discussed fusing first and then encoding or encoding first and then fusing, and finally found the latter method is better. Tang et al. [30] proposed a more flexible approach using a variable duration HMM [31] that factored each video into latent states with variable durations.

Now the popular strategy in the neural network is to compress the information of different frames in a video into a fixed summary vector by using pooling operation [3,4,15,16]. The mean pooling and the max pooling are common choices, i.e. taking average or maximum values of each feature vector, such as C3D [4] adopts the average value of each feature in every dimension. However, these pooling methods consider each frame equally, which is not robust to the noisy information. As there may be some noisy frames in the video, these noisy frames will cause some losses and ultimately lead to error judgments. Some recent works try to modify pooling strategy for action recognition, such as ActionVLAD [18] and adascan [17].

Frequent Pattern Tree. Our Feature Aggregation Tree, which want to mine action pattern in a video, is inspired by Frequent Pattern Tree. Han et al. [32] introduced the Frequent Pattern Tree structure for storing crucial information about mining frequent patterns in transaction and time-series databases. They also developed the FP-Growth algorithm for efficient and scalable mining on both long and short frequent patterns. Chang et al. [33] proposed an incremental data mining algorithm based on FP-Growth using the concept of heap tree to address the issue of incremental updating of frequent itemsets. Aditya and Pradana [34] leveraged the FP-Growth algorithm to find the customer buying habits on market basket in organic medicine store. Dharmaraajan and Dorairangaswamy [35] utilized the FP-Growth algorithm to classify user behavior in identifying the patterns of the browsing and navigation data of web users.

3 Approach

In this section, we will describe the details of Feature Aggregation Tree. As is outlined in Fig. 2, we extract frame-level features by the C3D network and then regard these features as action primitives, which are the results of the softmax

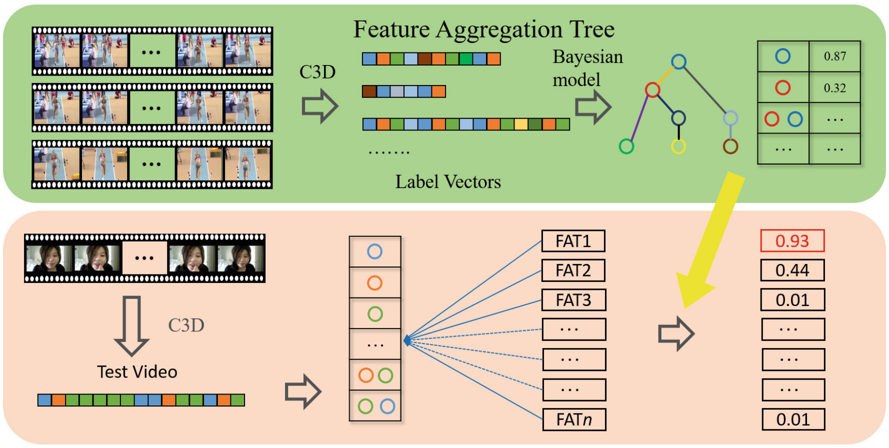


Fig. 2. When constructing the Feature Aggregation Tree (above), we extract the frame-level features by C3D network, and than get feature labels to make up label vectors. We use these label vectors to consturct Feature Aggregation Tree model and calculate the probabilities of each node and each pattern. When testing (below), we predict the test video label by matching each FA-Tree and calculating the probability.

layer. The next step is aggregating primitives into patterns to construct Feature Aggregation Tree. In a FA-Tree, each node corresponds to the conditional probability that the node appears, which we use the bayesian model to calculate. In the following we first describe how to construct Feature Aggregation Tree (Sect. 3.1) and then discuss the strategies for calculate the probability of each action pattern (Sect. 3.2).

3.1 How to Construct FA-Tree

In this part, we will give some definitions about the FA-Tree firstly. Specifically, devide the entire dataset D into different subsets, such as $D = \{S_1, S_2, \cdots, S_k\}$. Videos in every subset S_i have the same category label i, i is from 1 to k. And every subset S_i will generate one FA-Tree. Let $S_i = \{v_1, v_2, \cdots, v_j\}$, where v means a label vector, as every video corresponds to a label vector by C3D network, and j is the number of videos in subset S_i. Just like what is shown in Fig. 2. For each video, each frame in the video is regarded as an element in the label vector. Here we name one label in the vector as "item", and two different label pair as "pattern". Item set is $F = \{f_1, f_2, \cdots, f_m\}$ and pattern set is $P = \{p_1, p_2, \cdots, p_n\}$.

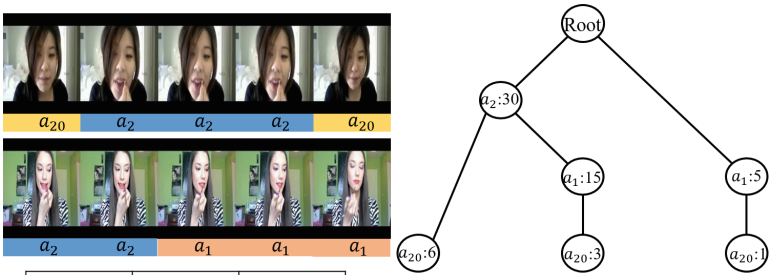
The first step is using the unique operation to deal with the same consecutive items. Because in our approach, we just consider different item pairs to mine association rules. The second step is to set support and confidence thresholds. Because there will be some noisy labels in the vector after the softmax layer, we set the minimum item support threshold (MIST) to remove these noisy labels when the frequence of one item is lower than the threshold. The other threshold is named minimum pattern confidence threshold (MPCT), which is set to choose the root node of a FA-Tree. When we construct a FA-Tree, the root node must be the actual label of this category. So we need to sort items by MIST, and MPCT determines that in top 0.05 or 0.1 rate of all items, we can choose the actual label as the root node. In Sect. 4.2, the data were uniformly sampled in experiments to help set the thresholds.

In addition, when constructing a FA-Tree, we have fully considered the temporal information in a video. Because in the processing step, we have not changed the positions of items. So the remaining items are organized in the order as the original video. The construction of a FA-Tree is divided into three steps. First, those items whose frequence is higher than MPCT are selected as the root node. Second, each label vector is divided into patterns to generate frequent pattern set. Third, for each root node, we connect the items that appear before root node in the left branch, and those after the root node in the right branch. The specific algorithm is shown as below.

Algorithm 1. Pseudo-code of the Construction of Feature Aggregation Tree

```
Input: Action label vector subset $S_i = \{v_1, v_2, \cdots, v_j\}$, $MIST$, $MPCT$
Output: Feature Aggregation Tree $FA - Tree$
1  Scan $S_i$ once. Collect items higher than $MIST$ to group $F$. Construct
   the pattern set $P$. Sort $F$ by support frequence in the descending order,
   and choose items higher than $MPCT$ to be the Root of a FA-Tree ;
2  Scan the pattern set $P$;
3  for each vector in $V_j$ do
4      for each pattern in $P$ do
5          if item $p$ appears before Root then
6              if Root has a left child node $p$ then
7                  the frequence of $p$ add 1;
8              else
9                  reach to the left child node of Root recursively, create a
                   new node $p$, and let its frequence be 1, linked to its
                   parent node and recorded in the list;
10         else
11             the same step as before except right instead of left;
12     if there is no $p$ in the pattern then
13         create new Root and repeat step 2
14 final ; return $FA - Tree$;
```

Given a simple FA-Tree as an example in Fig. 3. The letter ‘a’ means ‘action’ while the subscript of ‘a’ is the result of the softmax layer. The item set is $\{a_2, a_1, a_{20}\}$ and the pattern set is $\{[a_{20}, a_2], [a_2, a_1], [a_2, a_{20}], [a_1, a_{20}]\}$. When the root is a_2, we make a_{20} to be its left child node and a_1 to be its right child node. When we extract the pattern $[a_2, a_{20}]$, we find a_2 already has the right child node, so let a_{20} be the right child node of a_1.



a_{20} a_2 a_2 a_2 a_{20}
a_2 a_2 a_1 a_1 a_1
Root
a_2:30
a_1:15
a_1:5
a_{20}:6
a_{20}:3
a_{20}:1

Pattern set	Initial weight	Weight
$[a_{20}, a_2]$	0.24	0.3
$[a_2, a_1]$	0.6	0.65
$[a_2, a_{20}]$	0.12	0.04
$[a_1, a_{20}]$	0.04	0.01

Item set	Initial weight	Weight
a_2	0.5	0.78
a_1	0.33	0.15
a_{20}	0.17	0.07

Fig. 3. We construct a simple AF-Tree as an example. Initial weights are calculated by the statistical approach while weights are calculated by the bayesian model.

FA-Tree is a highly compressed structure that stores all the information of action patterns, and the memory space occupied by FA-Tree is proportional to the depth and width of the tree. For the depth of the tree, it generally depends on the complexity of label vectors, as well as the quality of classifier. For example, the more chaotic the label vector is, the deeper the tree will be. The width of the tree indicates that there are not only one root node. FA-Tree is compact because the size of the tree is usually much smaller than the original label vector set.

3.2 How to Design FA-Tree Probability Estimation

After constructing a FA-Tree, we initialize the weights of each node with the simple statistical approach. The definition of weights is shown as below. For each single item, its weight means the probability that it belongs to one action. For each pattern, its weight means the probability product of a two-item combination and this combination belongs to one action. The weight can be thought as the contribution of items and patterns to the whole video. However, simple statistical approach can not get precise weights in our experiments, Table 1. So we use the bayesian model to predict weights, the formula is shown as:

$$P(C_{cls}|l_i) = \frac{P(l_i|C_{cls})P(C_{cls})}{P(l_i)},$$

where C_{cls} means the category of one action, l_i means the i^{th} label in a video label vector. $P(l_i)$ is given by softmax classifier. $P(C_{cls})$ and $P(l_i|C_{cls})$ are calculated by data statistics. So we can update $P(C_{cls}|l_i)$ to have a more precise weight. And the weight will be saved in the node of a FA-Tree.

The FA-Tree is used to compute the probability of the whole video by finding the matched patterns in the test video. We will introduce the probability formula for calculating the video probability, which is as follows:

- Set the node weight parameter μ, pattern weight parameter γ.
- The patterns extracted from a label vector has N nodes and M patterns, referred to as p_{node} and $p_{pattern}$ respectively.

The probability that a test label vector passed by a FA-Tree can be expressed as:

$$P(v, FA - Tree) = \sum_{i=1}^{N} \mu_i p_i^{node} + \sum_{j=1}^{M} \gamma_j p_j^{pattern} + c,$$

where v represents the test video label vector;μ and γ are the weight parameters corresponding to p; c is a penalty, which plays a similar role as bias.

As is shown in Fig. 3, we get the initial weights with the statistical approach. Given that this AF-Tree belongs to action “ApplyLipstick”, which is label 2. If the assumption is $P(C_{cls=2}) = 0.76$, and $P(a_2), P(a_1), P(a_{20})$ are given by softmax, we can calculate the weights as the figure.

4 Experiments

4.1 Dataset

UCF101. UCF101 [36] is a dataset which is cut from real action videos in YouTube. It contains a total of 101 action categories and 13320 videos. We use split 1 for the experiment, including 9537 training videos and 3783 test videos, whose total hours up to 27 h.

HMDB51. HMDB51 [37] is collected from a variety of sources, most of which come from movies, and a small percentage from public databases such as Prelinger files, YouTube and Google Video. The dataset contains 6849 segments, which are divided into 51 action categories with at least 101 segments for each category.

4.2 FA-Tree Construction

In the experiment, we use the first split of HMDB51 dataset to show the process of parameter setting. Each video is divided into segments with the length of 16 frames and 50% overlap between segments. We use these video segments as the input of the 3D convolutional neural network [4] and we will get the classification result of each feature after the softmax layer. Therefore, for each action video, we can get a label vector which is made up by some different labels. We accumulate all the vectors of the same category action in one subset.

To get the item set and pattern set, first, we select 80% training data to predict the remaining 20% and we repeat this step 5 times. We randomly select some data in HMDB51, and finally select about 4500 videos to construct Feature Aggregation Trees. When observing these label vectors, some noisy data need to be removed. We just set the threshold MIST, and items below the thresholds are all excluded. The MIST is set to be 0.05 and the MPCT is set to be 0.1.

In the process of probability estimation, we set c as a penalty coefficient which is shown in the formula of the Sect. 3.2. We also test whether we should set the penalty factor c, which is shown in Table 1. The table (left) records accuracies without the bayesian model and the penalty coefficient c. While the first three columns in the table (right) record accuracies without the bayesian model but with c. And the last column in the table (right) records accuracies with the bayesian model and c.

Table 1. Accuracy (%) comparison between FA-Tree with PN (right) and without PN (left) on the HMDB51 dataset

	Rank-1	Rank-2	Rank-3
Split 1	56.9	69.9	75.0
Split 2	53.3	67.5	72.3
Split 3	55.4	69.2	74.7

	Rank-1	Rank-2	Rank-3	Bayesian
Split 1	57.0	69.8	75.4	67.7
Split 2	53.5	68.2	73.8	63.4
Split 3	55.5	69.7	74.8	66.8

When experimenting on the HMDB51 dataset, if we only use the C3D features and all weight of items and patterns are initialized, we can calculate the accuracy of Rank-2 is 69.80%. This shows that the Feature Aggregation Tree can really capture the latent motion information in the video. The reason why these segments can not achieve the highest score is that the predictions are mainly limited to using only the simple softmax. So after using the bayesian model we get the result lower than Rank-2 but higher than Rank-1.

4.3 FA-Tree Comparison Experiment

In this part, we consider Fisher Vector [11,13] and VLAD [10] to be the baseline method. In addition, we also consider the mean pooling and the max pooling, as well as RNN-FV [38] and ST-VLMPF [39]. The experimental results are in Table 2.

The result of FA-Tree is better than the baseline methods, which proves the effectiveness of our method. It is worthy mentioning that, compared with the improvement on the UCF101 dataset, the result is more obvious on the HMDB51 dataset because the labels in the UCF101 dataset are more ordered. However in the HMDB51 dataset, the FA-Tree can find enough action patterns from chaotic labels to represent the actions and ultimately improve the accuracy.

Table 2. Accuracy (%) comparison between mean pooling, max pooling and FA-Tree on the UCF101 dataset and the HMDB51 dataset

Strategies	UCF101	HMDB51
iFV [11]	79.8	49.0
VLAD [10]	81.4	49.1
RNN-FV [38]	82.3	52.9
Mean pooling	82.7	51.6
Max pooling	83.3	52.5
ST-VLMPF [39]	86.2	56.3
FA-Tree	**86.9**	**66.2**

4.4 Comparison with the State-of-the-Art

In Table 3, we show a comparison of our FA-Tree with the state-of-the-art methods on both datasets. Our method with MIFS feature achieves 94.6% on the UCF101 dataset and 74.2% on the HMDB51 dataset.

Table 3. Accuracy (%) comparison of our method with the state-of-the-art methods

Approach	UCF101	HMDB51
Wang et al. [26]	85.9	57.2
Tran et al. [4]	82.6	52.5
Simonyan et al. [3]	88.0	59.4
Peng et al. [13]	87.9	61.1
Wang et al. [16]	90.3	63.2
Wang et al. [2]	94.2	69.4
Kar et al. [17]	93.2	66.9
Girdhar et al. [18]	93.6	69.8
Duta et al. [39]	93.6	69.5
Our Method + MIFS [40]	**94.6**	**74.2**

5 Conclusion

We propose a novel model - the Feature Aggregation Tree to capture the temporal motion information in action videos. The FA-Tree connects frame-level features with the specific meanings of action primitives, and mines action patterns in the action sequence. We use the bayesian model to calculate the conditional probability of patterns. The experimental results on the UCF101 dataset and HMDB51 dataset demonstrate the effectiveness of our method.

References

1. Laptev, I., Lindeberg, T.: On space-time interest points. Int. J. Comput. Vis. **64**(2–3), 107–123 (2005)
2. Wang, L., et al.: Temporal segment networks: towards good practices for deep action recognition. ACM Trans. Inf. Syst. **22**(1), 20–36 (2016)
3. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: International Conference on Neural Information Processing Systems, pp. 568–576 (2014)
4. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks, pp. 4489–4497 (2014)
5. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 221–231 (2013)
6. Varol, G., Laptev, I., Schmid, C.: Long-term temporal convolutions for action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **PP**(99), 1 (2016)
7. Cao, X., Ning, B., Yan, P., Li, X.: Selecting key poses on manifold for pairwise action recognition. IEEE Trans. Ind. Inform. **8**(1), 168–177 (2012)
8. Liu, L., Shao, L., Zhen, X., Li, X.: Learning discriminative key poses for action recognition. IEEE Trans. Cybern. **43**(6), 1860–1870 (2013)
9. Jiang, Z., Lin, Z., Davis, L.S.: Recognizing human actions by learning and matching shape-motion prototype trees. IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 533–547 (2012)
10. Jegou, H., Douze, M., Schmid, C., Perez, P.: Aggregating local descriptors into a compact image representation, pp. 3304–3311 (2010)
11. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_11
12. Sydorov, V., Sakurada, M., Lampert, C.H.: Deep fisher kernels - end to end learning of the fisher kernel GMM parameters, pp. 1402–1409 (2014)
13. Peng, X., Wang, L., Wang, X., Qiao, Y.: Bag of visual words and fusion methods for action recognition: comprehensive study and good practice. Comput. Vis. Image Underst. **150**(C), 109–125 (2016)
14. Li, F.F., Perona, P.: A Bayesian hierarchical model for learning natural scene categories, pp. 524–531 (2005)
15. Wang, H., Dan, O., Verbeek, J., Schmid, C.: A robust and efficient video representation for action recognition. Int. J. Comput. Vis. **119**(3), 219–238 (2016)
16. Wang, L., Qiao, Y., Tang, X.: Action recognition with trajectory-pooled deep-convolutional descriptors, pp. 4305–4314 (2015)
17. Kar, A., Rai, N., Sikka, K., Sharma, G.: AdaScan: adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3376–3385 (2017)
18. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: ActionVLAD: learning spatio-temporal aggregation for action classification, pp. 3165–3174 (2017)
19. Sharma, S., Kiros, R., Salakhutdinov, R.: Action recognition using visual attention. arXiv preprint arXiv:1511.04119 (2015)
20. Ng, Y.H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification, vol. 16, no. 4, pp. 4694–4702 (2015)

21. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description, pp. 677–691 (2015)
22. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. Comput. Vis. Image Underst. **104**(2), 90–126 (2006)
23. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, Data Mining Concepts Models Methods & Algorithms, 2nd edn, vol. 5, no. 4, pp. 1–18 (2011)
24. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition, pp. 357–360 (2007)
25. Kläser, A., Marszalek, M., Schmid, C.: A spatio-temporal descriptor based on 3D-gradients. In: British Machine Vision Conference 2008, Leeds, September 2008
26. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: IEEE International Conference on Computer Vision, pp. 3551–3558 (2014)
27. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies, pp. 1–8 (2008)
28. Wang, H., Ullah, M.M., Klaser, A., Laptev, I., Schmid, C.: Evaluation of local spatio-temporal features for action recognition. In: BMVC 2009-British Machine Vision Conference, p. 124:1. BMVA Press (2009)
29. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization, pp. 1–8 (2007)
30. Tang, K., Fei-Fei, L., Koller, D.: Learning latent temporal structure for complex event detection, pp. 1250–1257 (2012)
31. Vezzani, R., Baltieri, D., Cucchiara, R.: HMM based action recognition with projection histogram features. In: Ünay, D., Çataltepe, Z., Aksoy, S. (eds.) ICPR 2010. LNCS, vol. 6388, pp. 286–293. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17711-8_29
32. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov. **8**(1), 53–87 (2004)
33. Chang, H.-Y., Lin, J.-C., Cheng, M.-L., Huang, S.-C.: A novel incremental data mining algorithm based on FP-growth for big data. In: 2016 International Conference on Networking and Network Applications (NaNA), pp. 375–378. IEEE (2016)
34. Aditya, P.: Market basket analysis using FP-growth algorithm in organic medicine store. Skripsi, Fakultas Ilmu Komputer (2016)
35. Dharmaraajan, K., Dorairangaswamy, M.: Analysis of FP-growth and Apriori algorithms on pattern discovery from weblog data. In: IEEE International Conference on Advances in Computer Applications (ICACA), pp. 170–174. IEEE (2016)
36. Soomro, K., Zamir, A.R., Shah, M.: UCF101: a dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
37. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition, pp. 2556–2563 (2011)
38. Lev, G., Sadeh, G., Klein, B., Wolf, L.: RNN fisher vectors for action recognition and image annotation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 833–850. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_50
39. Duta, I.C., Ionescu, B., Aizawa, K., Sebe, N., et al.: 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), pp. 3205–3214. IEEE (2017)
40. Lan, Z., Lin, M., Li, X., Hauptmann, A.G., Raj, B.: Beyond Gaussian pyramid: multi-skip feature stacking for action recognition, pp. 204–212 (2015)