Mining 3D Key-Pose-Motifs for Action Recognition

Chunyu Wang[1], Yizhou Wang[1], and Alan L. Yuille[2]

[1] Nat'l Eng. Lab. for Video Technology, Cooperative Medianet Innovation Center
Key Lab. of Machine Perception (MoE), Sch'l of EECS, Peking University, Beijing, 100871, China
{wangchunyu, Yizhou.Wang}@pku.edu.cn
[2]Department of Statistics, University of California, Los Angeles (UCLA), USA
Departments of Cognitive Science and Computer Science, John Hopkins University, Baltimore, MD 21218
yuille@stat.ucla.edu

Abstract

Recognizing an action from a sequence of 3D skeletal poses is a challenging task. First, different actors may perform the same action in various styles. Second, the estimated poses are sometimes inaccurate. These challenges can cause large variations between instances of the same class. Third, the datasets are usually small, with only a few actors performing few repetitions of each action. Hence training complex classifiers risks over-fitting the data. We address this task by mining a set of key-pose-motifs for each action class. A key-pose-motif contains a set of ordered poses, which are required to be close but not necessarily adjacent in the action sequences. The representation is robust to style variations. The key-pose-motifs are represented in terms of a dictionary using soft-quantization to deal with inaccuracies caused by quantization. We propose an efficient algorithm to mine key-pose-motifs taking into account of these probabilities. We classify a sequence by matching it to the motifs of each class and selecting the class that maximizes the matching score. This simple classifier obtains state-of-the-art performance on two benchmark datasets.

1. Introduction

Action recognition from RGB videos [10, 28, 26, 7, 22, 16] is an important task with many applications, such as intelligent surveillance, sports video analysis and human-computer interactions. Although this task has attracted a lot of attention, the recognition performance is unsatisfactory because of the considerable variations in the appearance and the scales of the people performing the same action. In addition, 2D images will be very dependent on viewpoint and the same action can vary as the perspective changes.

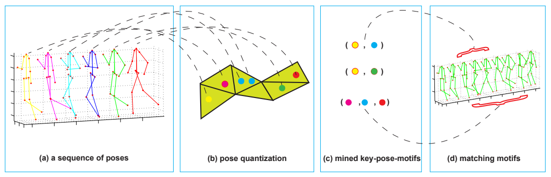

Figure 1: Overview: (a) a sequence of training poses. (b) the poses quantized by a simplicial model. (c) mined key-pose-motifs. (d) a sequence is classified by matching it to the motifs of each class.

The introduction of RGB-D cameras and the corresponding pose estimation algorithms [19] makes it possible to obtain 3D human poses and hence study pose-based action recognition. This leads to progress on the appearance and viewpoint variations [11] [25] [27] [23] but there are other challenges remaining unsolved. First, different actors often perform the same action in differing styles. Second, the 3D poses are sometimes inaccurate because they are usually estimated from noisy depth maps. The two challenges together cause large intra-class variations making two instances of the same class far apart. It is not plausible to compare two instances directly. Third, most benchmarks only provide only a few sequences for each action which makes training complex classifiers sensitive to over-fitting.

Psychological studies, however, show that humans can effortlessly recognize actions from pose sequences despite all these challenges [9]. Indeed some actions can be classified from a single key-pose [31]. This suggests that we can perform action classification using a set of key poses rather than using the whole pose sequence. The key poses are close but not necessarily adjacent in the sequences and hence allows short and variable gaps between the poses. These gaps help deal with style variations. Also the representation is robust to outlier poses as they have little effect on the representation as long as the key poses are accurate.

Inspired by the psychological studies, we propose to mine a set of *key-pose-motifs* for each action class. We define a motif to be a short sequence of poses which are nearby but not necessarily adjacent in the original sequences. A motif is called a key-pose-motif of a particular class if it appears in a sufficient number of sequences of that class. We mine several key-pose-motifs for each action class and classify an input sequence by finding the action class whose key-pose-motifs best match the sequence. Observe that this approach also has the ability to detect the start and finish of an action sequence by inspecting the matching results, although that is not the focus of this paper.

To mine the key-pose-motifs, we need to quantize the continuous pose sequences (by a dictionary) into discrete sequences where each pose of the sequence is represented by a symbol of the dictionary. We use clustering algorithm to learn the dictionary. We use the activated simplices method proposed in [24] to learn the dictionary. Each symbol in the dictionary is an activated simplex consisting of a set of bases which represents data by their convex combinations. Typically, each pose is quantized (represented) by the closest simplex. However, in our work, to reduce the influence of quantization error (e.g., two similar poses are quantized by different simplices), we use soft-quantization so that a pose is represented by several symbols with a probability for each (based on the distance to the symbol). Hence each pose is represented by a probability vector (its dimension is the same of as the number of symbols in the dictionary) and a sequence of poses is represented by a probability matrix where each column is the probability vector of the corresponding pose. Standard sequential pattern mining algorithms do not deal with this type of probabilistic data so we propose a novel, and efficient, algorithm to mine the key-pose-motifs which is one of our contributions.

We classify a test sequence by matching it to the key-pose-motifs of each class and select the class by maximizing the matching score. The classifier is interpretable because we can visualize the mined key-pose-motifs and the matched positions in the sequence. So when there are misclassifications, we can easily detect why and where the failures happen. In experiments we use action-units consisting of short sequences of poses (e.g., neighboring three poses compose an action-unit and the original pose sequence is transformed to an action-unit sequence). But for ease of exposition, we will describe our method using poses only (and introduce action-units in the experiment section).

The paper is organized as follows. Section 2 reviews related work. Sections 3, 4 and 5 describe the proposed action representation (action-snippets), the key-pose-motif mining algorithm and the action classification method respectively. The remainder of the paper is devoted to experiments followed by a conclusion of this work.

2. Related Work

In this section, we provide a brief overview of the related work on human pose based action recognition. We also review the existing sequential pattern mining algorithms and discuss how they differ from our method.

2.1. Human pose based on action recognition

We classify the existing work into three categories according to how temporal cues are modeled. The first class of work [8] [3] [23] ignores the temporal information and treat the poses in a sequence independently. They usually adopt "bag of poses" [8] [23] or majority voting [3] schemes for classification. At the other extreme, the second class of work [30] classifies pose sequences by modeling all poses in a sequence, for example by Hidden Markov Models [30] or by dynamic time warping [18]. Another line of work [25] [27] [12] encodes restricted temporal pose structures, for example, using temporal pyramid matching [27] [12] or by modeling neighboring pose changes [25].

Our proposed key-pose-motifs models the temporal structures of a set of key poses. However, our approach differs from [25] [27] [12] as key-pose-motifs are more flexible because they allow gaps between neighboring poses which makes them robust to speed variations. Besides the temporal structures are automatically learned from training sequences rather than manually designed.

2.2. Sequential pattern mining

Sequential pattern mining is the task of discovering frequent subsequences as patterns in a sequence database. There a lot of work [1, 20, 32] when the sequence database is certain (or deterministic) rather than probabilistic. The main challenge of sequential pattern mining is that it is computationally infeasible to examine all possible subsequences to determine the frequent ones, most early sequential pattern mining methods are based on A-prior algorithm[2] where the main idea is that the sub-sequences of frequent patterns are also frequent. So we can generate candidate longer frequent patterns from the shorter ones which can substantially reduce the search space to be examined. However, those methods can not deal with the situations where the sequences are uncertain. Consider the amount of noise in our data, it is important for us to use soft-quantization which means that standard mining methods do not apply.

There are some work [33] [13] which address the task when the database is uncertain. But the uncertainty in these work [33] [13] occurs at different levels as ours. Muhammad et al. [13] propose a method to mine sequential patterns when there is uncertainty about which sequence an item is associated with (but this is not equivalent to our problem). Zhao et al. [33] solve a similar problem as ours. However, it is impossible to incorporate the gap constraints using their framework.

3. Action Representation

We represent an action by a sequence of 3D poses $\{y^{(1)}, \cdots, y^{(m)}\}$ where each pose y is a high dimensional vector consisting of a set of body joint locations. This is a natural and intrinsic representation as it conforms to studies of how humans understand actions [4].

To mine key-pose-motifs, we first quantize the poses using a dictionary of discrete symbols. The classic quantization method is k-means which represents a pose by its nearest symbol in the vocabulary. However, small perturbations of poses, due to inaccuracies in poses, can cause the poses to be assigned to different symbols which will hurt classification performance.

We propose instead to use an alternative quantization method, namely the activated simplices method [24]. The simplicial model consists of a mixture of activated simplices $\mathbb{S} = \{s_1, \cdots, s_K\}$ where each simplex has several bases. A simplex represents poses by convex combinations of the bases which will form a convex hull. All the data which are close to the convex hull will be regarded as similar. For each action class, we learn a dictionary of activated simplices and combine them to make our dictionary of symbols. There are two reasons for choosing the simplicial model. First, it was shown to be robust to the inaccuracies in poses [24]. Slightly distorted poses will be projected to the same simplex. Second, the simplicial model is more semantically meaningful. In the simplicial model, semantically similar poses are projected to different positions of the same simplex and they all have small projection errors. This is because activated simplices are able to represent the local linearity of poses, as described in [24]. We will compare the two quantization methods in the experiment section.

To make the representation more robust to inaccurate poses, we use soft-quantization to assign each pose to all symbols. More precisely, We represent a pose using all of the K symbols in the dictionary and associate each symbol with a probability $\mathbf{p}_i$ which measures its distance to the pose $\mathbf{p}_i = \frac{e^{-\text{dist}(s_i,y)}}{\sum_{j=1}^{K} e^{-\text{dist}(s_j,y)}}$ ($dist(s_j, y)$ measures the distance of the point y to the convex hull formed by the activated simplex s_j. see [24] for the definition). Intuitively, small distances induce large probabilities and vice versa. Finally, each pose $y^{(i)}$in a sequence is represented by a probability vector $\mathbf{p}^{(i)} = [\mathbf{p}_1^i, \cdots, \mathbf{p}_K^i]^T$ and a sequence of poses is represented by a matrix $\mathbf{P} = (\mathbf{p}^{(1)}, \cdots, \mathbf{p}^{(m)})$.

4. Mining Key-pose-motifs

We propose an efficient algorithm to mine key-pose-motifs from the probability matrices, representing the soft-assignment of poses to symbols, as described above. We first give formal definitions of the terms which are going to be used in this paper.

Table 1: An example probabilistic sequence of length five which is defined on a vocabulary of five symbols (each column sums to one). The most probable sequence is (s_1, s_2, s_5, s_1, s_1) but there are 95 other possibilities.

time / symbols	1	2	3	4	5
s_1	0.7	0	0.3	0.9	1
s_2	0.1	0.8	0.0	0.1	0
s_3	0.1	0.1	0.1	0	0
s_4	0.1	0.0	0.2	0	0
s_5	0.0	0.1	0.4	0	0

Definition 4.1 (Deterministic sequence)**.** $T = (t^{(1)}, \cdots, t^{(m)})$ is a deterministic sequence of length m containing symbols chosen from the dictionary, i.e., $t^{(i)} \in \mathbb{S}$, at each of the m time-stamps.

Definition 4.2 (Probabilistic sequence)**.** $\mathbf{P} = (\mathbf{p}^{(1)}, \cdots, \mathbf{p}^{(m)})$ is a probabilistic sequence of length m where each item $\mathbf{p}^{(i)}$ is a K dimensional vector specifying the soft-assignment of the input pose $y^{(i)}$ to the K symbols in $\mathbb{S}$. For example, $\mathbf{p}_j^{(i)}$ represents the probability that the pose $y^{(i)}$ is s_j. Table 1 shows an example probabilistic sequence of length 5.

Definition 4.3 (Probabilistic support)**.** Given a deterministic sequence T and a probabilistic sequence $\mathbf{P}$, the probabilistic support from $\mathbf{P}$ to T is a scalar value η between zero and one which measures how well T can be matched to $\mathbf{P}$. The formal definition is given in section 4.1.

Definition 4.4 (Key-pose-motif)**.** A key-pose-motif of an action class is a *deterministic* sequence whose probabilistic support averaged over all (probabilistic) sequences from that class is larger than a threshold ϵ. A motif of length m is called an m-motif.

Method Overview: The task of key-pose-motif mining is to find out all the key-pose-motifs from a datasets of pose sequences. Each pose sequence is soft-quantized and hence is represented by a probability matrix. Algorithm 1 shows the algorithm. Initially, each symbol in the dictionary $\mathbb{S}$ is a candidate 1-motif. We compute the average probabilistic supports for the candidates and remove the ones whose average supports are smaller than the threshold ϵ. Then we expand the 1-motifs to get candidate 2-motifs and continue recursively to get higher-order motifs. We repeat the process until no larger motifs can be generated. There are two main components in the algorithm. The first computes the probabilistic supports for the candidate motifs which is discussed in section 4.1. The second expands the k-motifs to get candidate $k+1$-motifs which is discussed in section 4.2.

Algorithm 1 Key-pose-motif Mining Algorithm

```
$T^1$={1-motifs}
for ($k = 2$; $T^{k-1} \neq \emptyset$; $k$++) do
    $T^k$ = expand($T^{k-1}$)
    for ($i = 1$; $i \leq |T^k|$; $i$++) do
        support=0
        for ($j = 1$; $j \leq |D|$; $j$++) do
            support=support+$\eta(t_i^k, D_j)$
            If $\frac{\text{support}}{|D|} \leq \epsilon$
                $T^k \leftarrow T^k - \{t_i^k\}$
            endif
        end for
    end for
end for
```

4.1. Probabilistic Support

Unlike the existing sequential mining algorithms such as [20], the inputs are probabilistic sequences. It is non-trivial to decide whether a key-pose-motif appears in a probabilistic sequence. We propose to compute the *probabilistic support* of the motif in the sequence as follows.

Let a deterministic sequence (e.g., a candidate motif) be $T = (t^{(1)}, \cdots, t^{(m)})$ and a probabilistic sequence be $\mathbf{P} = (\mathbf{p}^{(1)}, \cdots, \mathbf{p}^{(n)})$. The length m of the deterministic sequence is usually much smaller than the length n of the probabilistic. The probabilistic support of $\mathbf{P}$ for T measures how well T can be matched to $\mathbf{P}$. Formally, we search for a mapping $M(i) \in \{1 \cdots n\}, i \in \{1 \cdots m\}$ which maps each item in T to an item (location) in $\mathbf{P}$ with the constraint $M(i) < M(i+1)$ and $M(i+1) - M(i) \leq g$. Here g is the maximum gap constraint which prevents neighboring poses in a motif from matching to positions which are far away from each other in the sequence.

We define the probabilistic support $\eta(T, \mathbf{P})$ as follows:

$$\eta(T, \mathbf{P}) = \max_M \prod_{i=1}^{m} \mathbf{p}_{t^{(i)}}^{(M(i))}$$
$$s.t. \quad M(i) < M(i+1), \quad M(i+1) - M(i) \leq g \tag{1}$$

We now show that this objective function can be optimized efficiently by dynamic programming. Let $T(1:n_1) = (t^{(1)}, \cdots, t^{(n_1)})$ and $\mathbf{P}(1:n_2) = (\mathbf{p}^{(1)}, \cdots, \mathbf{p}^{(n_2)})$. Let $f(T(1:n_1), \mathbf{P}(1:n_2))$ denote the probabilistic support of matching $T(1:n_1)$ to $\mathbf{P}(1:n_2)$ with the condition that $t^{(n_1)}$ is matched to $\mathbf{p}^{(n_2)}$. Hence $f(T(1:n_1), \mathbf{P}(1:n_2))$ can be computed by:

$$f(T(1:n_1),\mathbf{P}(1:n_2)) = \mathbf{p}_{t^{(n_1)}}^{(n_2)} \times \max_{i \in \{n_2-g, \cdots, n_2-1\}} f(T(1:n_1-1), \mathbf{P}(1:i)) \tag{2}$$

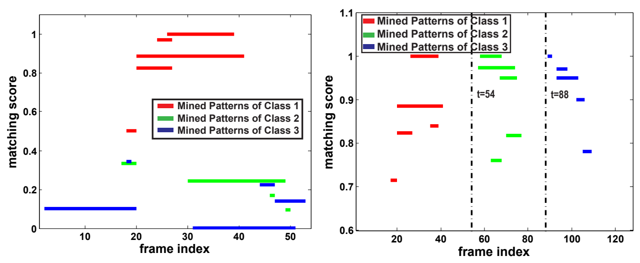

Figure 2: Left: matching the mined key-pose-motifs of Classes 1, 2 and 3 to a sequence of Class 1 (length 54). Each line segment defines the matching regions of each motif and its y-axis value gives the matching score. Right: a long sequence is composed by sequences from classes 1, 2 and 3, respectively (lengths 54, 34 and 40). We match the key-pose-motifs of the three classes to the long sequence. This shows that motifs can be used to roughly detect the starts and ends of actions.

Using Eq.(2), we can efficiently compute a probabilistic support matrix $f(,)$ of dimension $m \times n$. We iterate over the last row of the matrix and find out the maximum value which is $\eta(T, \mathbf{P})$.

4.2. The Expansion Algorithm

The expansion algorithm enables us to mine key-pose-motifs by efficiently searching over the enormous space of deterministic sequences. It exploits the fact that larger sequences are compositions of smaller sequences. Suppose a sequence $T = (t^{(1)}, \cdots, t^{(k+1)})$ is a key-pose-motif, then we can guarantee that its head and tail sub-sequences $T^{head} = (t^{(1)}, \cdots, t^{(k)})$ and $T^{tail} = (t^{(2)}, \cdots, t^{(k+1)})$ are also key-pose-motifs (note that, by equation (1), these subsequences must have higher support than the sequence and hence their supports will be above threshold). Unlike existing methods which do not have the maximum gap constraints [33], it is not guaranteed that all the other sub-sequences, e.g., $(t^{(1)}, t^{(3)}, \cdots, t^{(k)})$ are also key-pose-motifs. For example, suppose there is a sequence $(1, 2, 3, 4, 5)$ and the maximum gap is set to be one, then the sequence $(1, 3, 5)$ is supported by the input sequence but the sub-sequence $(1, 5)$ is not.

We derive our expansion algorithm based on these ideas. Let $FT^k = \{T_1^k, T_2^k, \cdots, T_{|FT^k|}^k\}$ denote the set of k-motifs mined in the last iteration. For each pair of motifs in FT^k, for example, T_1^k and T_2^k, we compare the last $k-1$ items of T_1^k with the first $k-1$ items of T_2^k. If they are all equal, then we generate a $k+1$-motif candidate T_1^{k+1} by concatenating the T_1^k with the last item of T_2^k.

Example 4.1 (Expansion)**.** Suppose we have three 3-motifs $FT^3 = \{(1,2,3), (2,3,4), (3,4,6)\}$. Then by joining $(1,2,3), (2,3,4)$ we obtain $(1,2,3,4)$ and by joining

Table 2: The state-of-the-art action recognition accuracies over all 252 splits on the MSR-Action3D Dataset.

Methods	Acc (%)	Year
HON4D [14]	82.15	2013
Tran [21]	84.54	2013
Wang [24]	88.10	2014
Du [6]	89.00	2015
Ours	**94.40**	2015

$(2,3,4),(3,4,6)$ we obtain $(2,3,4,6)$. Hence we get two 4-motifs $FT^4 = \{(1,2,3,4),(2,3,4,6)\}$.

Proposition 4.1. *The above expansion algorithm will not leave out any key-pose-motifs.*

Proof. If a sequence is a key-pose-motif, then its head and tail subsequences must also be key-pose-motifs by equation (1). It means that the two subsequences have already been mined. Then in the expansion stage, the two subsequences will generate the larger candidate key-pose-motif for sure.

5. Action Recognition: Inference Algorithm

We propose a simple classifier which works by matching a test sequence to the key-pose-motifs of each class. During the matching process each motif will get a probabilistic support, given by equation (1). We classify the sequence to be the class that gets the largest average probabilistic support over all motifs of that class. Ideally, a sequence of a particular class should have large supports for the key-pose-motifs mined for that class and small supports for the key-pose-motifs of other classes. Figure 2 shows an example.

The classifier is simple because it has no parameters. In addition, it is also interpretable. We can visualize the mined key-pose-motifs and the matched poses in a sequence. This helps us spot why and where failures may happen. Figure 2 shows that we can even use this model for action localization, i.e. to coarsely find the start and end of an action.

6. Experiments

We conduct experiments on four most popular action recognition benchmarks, i.e. the MSR-Action3D dataset [11], the UTKinect dataset [30], the MSR Daily Activity3D dataset [27] and the Florence dataset [17]. We first compare our method with the state-of-the-arts on the four datasets respectively. Then we present diagnostic analysis of the method on the MSR-Action3D dataset.

Action-units. In our experiment, we concatenate l consecutive poses together to form an *action-unit* $\hat{y}^{(i)} = [y^{(i)} \cdots y^{(i+l-1)}]$ and represent an action by a sequence of action-units: $A = \{\hat{y}^{(1)}, \cdots, \hat{y}^{(m-l-1)}\}$. Consecutive action-units have overlaps. The proposed mining and classification algorithms are readily applicable to the action-unit sequences. The activated simplices are also learned on action-units. The use of action-units is more robust to outlier poses because a single inaccurate pose will not significantly affect the action-unit. We evaluate the influence of this pre-processing in experiments.

Table 3: The state-of-the-art action recognition accuracies using single split on the MSR-Action3D Dataset.

Methods	Acc (%)	Year
Vemulapalli [23]	89.48	2014
Wang [25]	90.22	2013
Wang [24]	91.30	2014
Luo [12]	96.70	2013
Ours	**99.36**	2015

Table 4: Action recognition accuracies on the UTKinect Dataset using the "leave-one-sequence-out" criterion.

Methods	Acc (%)	Year
Maxime [5]	91.5	2014
Xia [30]	90.92	2012
Ours	**93.47**	2015

6.1. On The MSR-Action3D Dataset

The MSR-Action3D dataset provides 557 pose sequences of ten subjects performing 20 actions. There are about 50 frames in each sequence. This is a challenging dataset because first many actions in the dataset are similar and second the pose sequences of the same action can have large variations due to either 3D pose estimation inaccuracies and performing style variations.

While learning the simplicial model, we set the number of bases to be 400 by cross-validation. We obtain about 200 activated simplices whose average dimension is about five. While mining the key-pose-motifs, we decrease the minimum support threshold ϵ from 1 to 0 with the step size of 0.05 until we obtain about 50 key-pose-motifs for each class. The number 50 is set by cross validation.

Most existing works choose five subjects for training and the remaining five subjects for testing, e.g. in [11], and report the result based on a single split. However, it is shown in [15] that the way how the data are split (i.e. choosing which five subjects for training) can have large influence on the results. To make the results more comparable, in this work, we experiment with all 252 possible subject splits and report the average accuracy. Figure 4 shows the classification confusion matrix of a certain split.

Comparison with the State-of-the-arts. Table 2 compares our method with the state-of-the-art methods using

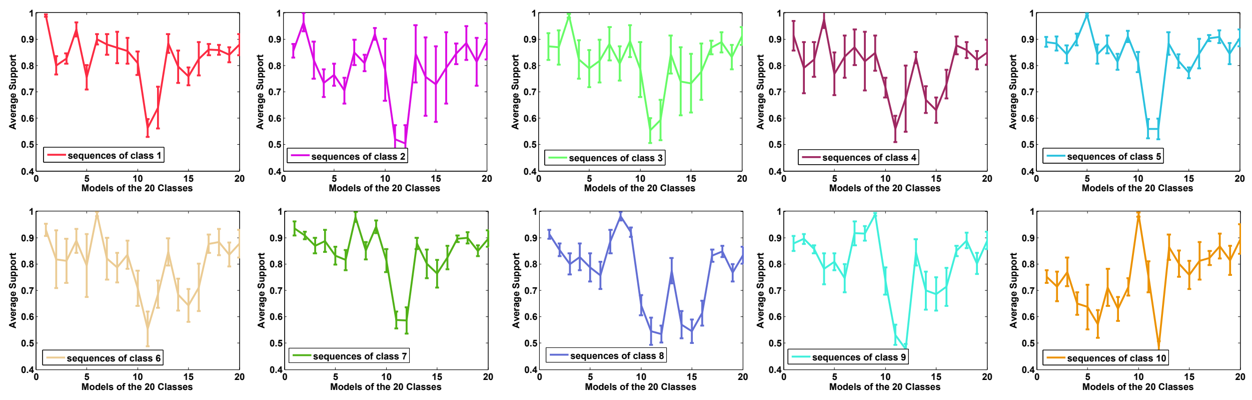

Figure 3: The x-axis is the index of the 20 classes of key-pose-motif-models and the y-axis is the average matching scores (standard deviations) which are computed when matching a set of test sequences of a particular class to those models. For example, in the first sub-figure, we match a set of sequences of class 1 to the 20 classes of models and obtain the average matching scores and standard deviations for each model. The motif model of class 1 gets the largest average score.

the protocol of "average over all splits". We can see that our method outperforms the state-of-the-art methods [14] [21] [24] [6]. In addition, our method is also the simplest in terms of both the features and the classifiers. *Note* that the method proposed in [24] uses the activated simplices as classifiers directly using a nearest-neighbor algorithm. We can see that we improve the performance by mining key-pose-motifs on activated simplices. Du et al. [6] use deep learning techniques to learn an end-to-end classifier which achieves the current best performance.

Since some methods only provide results for a single s-plit, we also provide these numbers. However, note that they are not directly comparable as the methods may choose different five subjects for training. The accuracies of our method using single split criterion: (1) the accuracy is 95.88% when we use subjects $1, 2, 3, 4, 5$ for training; (2) the accuracy is 97.44% when we use subjects $1, 3, 5, 7, 9$ for training; (3) the best accuracy of a single split is 99.36%. Table 3 shows the state-of-the-art results using the single split criterion.

Comparison with Baselines. The first baseline is the Direct Matching Method. Given a test sequence of action-units, we compute the dynamic time warping based matching scores between the sequence and the training sequences of all classes. The class that achieves the largest average matching score is the predicted class. The method achieves an accuracy of 88%. The result is not satisfactory which is mainly because it cannot deal with large intra-class variations effectively. The second baseline uses the proposed key-pose-motifs mining method. However, each action-unit is quantized into only one symbol (It is not a probabilistic representation). We name this method as the Deterministic Mining Method. The deterministic mining method achieves an accuracy of 91% which is lower than our method, but is already higher than the state of the arts. We also compared

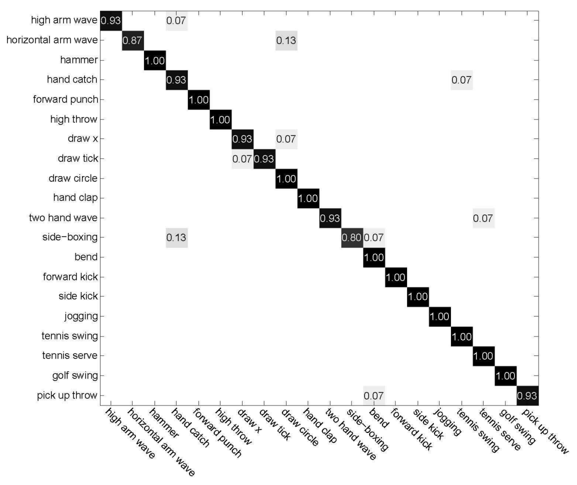

Figure 4: Confusion matrix on the MSR-Action3D dataset.

with a method which uses probabilistic key-pose-motif mining but uses k-means to obtain the probabilistic data. The method achieves an accuracy of 88.7% which is lower than ours. This is mainly because k-means based quantization is not meaningful in the sense that semantically close poses will have similar probabilistic representations.

6.2. On The UTKinect Dataset

The UTKinect dataset [30] was captured using a single stationary Kinect. There are ten action including walk, sit down, stand up, pick up, carry, throw, push, pull, wave hands, clap hands. There are ten subjects involved in the experiment with each subject performing each action twice. There are 199 sequences in total.

We learn 200 bases and end up with about 120 activated simplices. The number of bases in the simplices is five

Table 5: Action recognition accuracies on the MSR Daily Activity3D Dataset.

Methods	Acc (%)	Year	Features
Ours	**83.47**	2015	Skeletons
Actionlet [27]	68.00	2012	Skeletons
HON4D [14]	80.00	2014	depth
Actionlet [27]	85.75	2012	Skeletons + depth
Xia [29]	88.20	2013	Skeletons + depth

on average. We use the "leave-one-sequence-out" evaluation criterion where one sequence is used for testing and the rest of the sequences are used for training. We repeat the process for all sequences (199 in total) and report the average accuracy. Table 4 shows the results. We can see that our approach outperforms the state-of-the-arts. The main reason may be because our method suffers less from noisy poses because it is only the key poses rather than all poses are useful for classification. In other words, the model will be accurate as long as the key poses are accurate.

6.3. On the MSR Daily Activity3D Dataset

The Daily Activity3D dataset is captured by a Kinect device and it provides both depth map and 3D skeleton sequences. It includes 16 activities: drink, eat, read book, call cellphone, write on a paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lay down on sofa, walk, play guitar, stand up and sit down. For some actions, each subject performs them in both "sitting" and "standing" poses. In total, there are 320 sequences.

This is a rather challenging dataset and very few work have demonstrated good performance on it. In particular, the 3D joint locations are very noisy when the performer stands close to the sofa or sits on the sofa. Considering the large amount of noises in 3D skeletons, most methods [27, 29] combine both depth maps and 3D joint locations for action recognition. Note that the depth map is relatively more accurate in this dataset. Our method only uses 3D skeletons which is a more challenging problem.

We use the cross-subject evaluation method to compare our method with the state-of-the-arts. However, it is worth noting that the dataset doesn't specify which five subjects to use for training. We report the result when training on subjects 1-5. We also report the average recognition result over all 252 possible splits.

Table 5 shows the results on this dataset using a single split evaluation criterion. Our method outperforms the state-of-the-art methods [27, 29] which use only 3D skeletons or depth maps. Some methods which use more information (e.g., combine the 3D skeletons and depth maps) achieve slightly better performance than ours. The average recognition accuracy over all 252 splits for our method is 79%.

Table 6: Action recognition accuracy on the Florence dataset using leave-one-actor-out setting.

Methods	Accuracy (%)	Year
Lorenzo et al. [17]	82.15	2013
Raviteja et al. [23]	90.88	2014
Tran et al. [5]	87.04	2014
Our Approach	**92.25**	2015

The result is promising given the amount of noises in the dataset. To the best of our knowledge, no previous methods have reported the average results.

6.3.1 The Florence Dataset

The dataset [17] was captured using a Kinect camera at the University of Florence. It includes nine activities: wave, drink from a bottle, answer phone, clap, tight lace, sit down, stand up, read watch, and bow. During acquisition, ten subjects were asked to perform the above actions for two or three times. This resulted in a total of 215 activity samples. Following the data suggestion, we adopt a leave-one-actor-out protocol: we train the classifier using all the sequences from nine out of ten actors and test on the remaining one. We repeat this procedure for all actors and compute the average classification accuracy values of the ten actors.

We set the number of bases for each class to be 50 (450 in total) by cross-validation. Table 6 compares our method with the state-of-art methods on this dataset. Our approach achieves the highest recognition accuracy.

6.4. Diagnostic Analysis

Reasons behind the performance. We observe in experiments that given a test sequence of a certain class, the corresponding class of key-pose-motifs usually obtain very large probabilistic supports while other key-pose-motifs obtain small supports. See Figure 3. We can also tell from the figure which actions are easy to differentiate and which are not. For example, in the eighth sub-figure (row 2, column 3) of Figure 3, we can see that the classes of 7, 8 and 9 are ambiguous because they all get large supports when the test sequences are from class 8. The three classes are "draw x", "draw tick" and "draw circle" actions respectively which are in fact very similar.

Influence of the Parameters. We evaluate the three main parameters in the proposed method: the gap g in the maximum gap constraints, the number of mined key-pose-motifs for each class and the number of poses l in an action-unit. To save time, we only use the first ten splits out of the 252 splits and report the average recognition accuracy.

Figure 5 shows the influence of the gap constraints. We can see that there is large performance improvement by allowing gaps between consecutive poses. One of the reasons

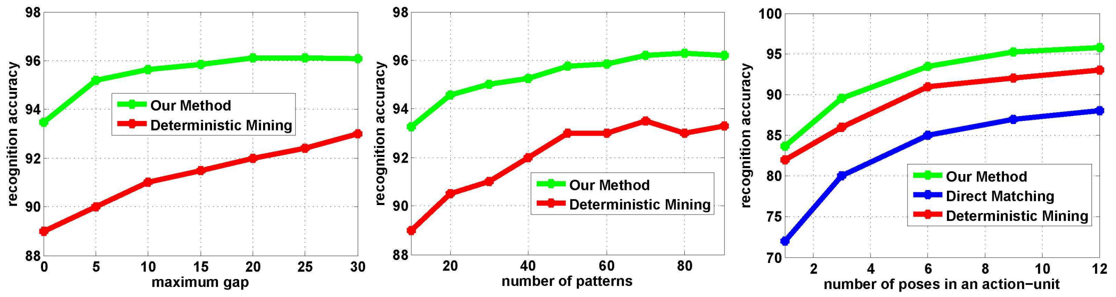

Figure 5: Influence of the maximum gap constraint, the number of mined key-pose-motifs in each class and the number of poses in an action-unit.

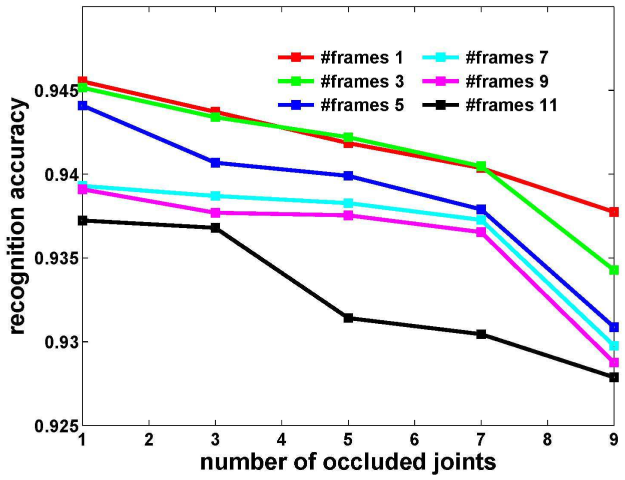

Figure 6: The influence of the number of occluded joints and the number of affected frames in a test sequence on action recognition accuracies on the MSR-action3D dataset. See section 6.1 for more details.

might be that it can deal with the speed variations among different sequences. In other words, the key poses might appear at various positions of different sequences and allowing gaps between consecutive poses helps reduce the influence of the variations. The best performance is achieved when the maximum gap is 20. After that, increasing the gap will degrade the performance a little bit. This may be because too large a gap encourages the model to find key-pose-motifs that are not meaningful.

We also evaluate the influence of the number of key-pose-motifs. We adjust the minimum probabilistic support threshold ϵ and obtain a desired number of key-pose-motifs. Figure 5 shows the result. First, using more key-pose-motifs will consistently improve the performance when the number is smaller than 80. This is reasonable because the model becomes more representative using more key-pose-motifs. However, when the number is too large, then many motifs which only appear in a few sequences are also mined which makes the model over representative. In other words, the motifs become less discriminative because they might be able to represent sequences of other classes well which degrades the action recognition performance.

The use of action-units improves the performance. See Figure 5. We can see that there is a large performance improvement by putting more poses in an action-unit. But the change is not significant after the number exceeds 9.

Robust to Occlusion. We now evaluate the robustness of the method to inaccurate poses which are mainly caused by occlusions. We synthesize a set of data by randomly perturbing the 3D poses in the MSR-Action3D dataset. In particular, we set some joint locations of several poses in a pose sequence as zero to simulate occlusion. We control the number of perturbed joints and frames. See Figure 6 for the results. In the worst case, when 9 joints (about 45%) of 11 frames (about 20%) are contaminated, the performance only drops by about 2%. The results justify that our method is robust to the occlusions.

7. Conclusion

We propose a simple and interpretable method for action recognition. By mining the key-pose-motifs which are not necessarily adjacent in the original sequence, we obtain a compact representation which is robust to intra-class variations. We evaluate the model on two benchmark datasets and show that it outperforms the state-of-the-arts. Moreover, the model is easy to interpret and we can spot where and why failures happen. In our future work, we would like to improve the discriminative power of the method by mining discriminative key-pose-motifs which can match to the sequences of its class very well but will not match to the sequences of other classes.

Acknowledgements: We thank for support from the following research grants: 973-2015CB351800, NSFC-61272027, NSFC-61231010, NSFC-61527804, NSFC-61210005, ONR grant N00014-15-1-2356, NSF Expedition NSF award CCF-1317376.

References

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, pages 3–14. IEEE, 1995.

[2] R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. In *VLDB*, volume 1215, pages 487–499, 1994.

[3] J. Ben-Arie, Z. Wang, P. Pandit, and S. Rajaram. Human activity recognition using multidimensional indexing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1091–1104, 2002.

[4] I. Bülthoff, H. Bülthoff, and P. Sinha. Top-down influences on stereoscopic depth-perception. *Nature neuroscience*, 1(3):254–257, 1998.

[5] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, and A. Del Bimbo. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. 2014.

[6] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*. IEEE, 2015.

[7] A. Efros, A. C. Berg, G. Mori, J. Malik, et al. Recognizing action at a distance. In *ICCV*, pages 726–733. IEEE, 2003.

[8] W. Gong, A. D. Bagdanov, F. X. Roca, and J. Gonzàlez. Automatic key pose selection for 3d human action recognition. In *Articulated Motion and Deformable Objects*, pages 290–299. Springer, 2010.

[9] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, 14(2):201–211, 1973.

[10] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2-3):107–123, 2005.

[11] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *CVPRW*, pages 9–14. IEEE, 2010.

[12] J. Luo, W. Wang, and H. Qi. Group sparsity and geometry constrained dictionary learning for action recognition from depth maps. In *ICCV*, pages 1809–1816. IEEE, 2013.

[13] M. Muzammal and R. Raman. Mining sequential patterns from probabilistic databases. In *Advances in Knowledge Discovery and Data Mining*, pages 210–221. Springer, 2011.

[14] O. Oreifej and Z. Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *CVPR*, pages 716–723. IEEE, 2013.

[15] J. R. Padilla-López, A. A. Chaaraoui, and F. Flórez-Revuelta. A discussion on the validation tests employed to compare human action recognition methods using the msr action3d dataset. *arXiv preprint arXiv:1407.7390*, 2014.

[16] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *CVPR*, pages 1242–1249. IEEE, 2012.

[17] L. Seidenari, V. Varano, S. Berretti, A. Del Bimbo, and P. Pala. Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses. In *CVPRW*, pages 479–485. IEEE, 2013.

[18] S. Sempena, N. U. Maulidevi, and P. R. Aryan. Human action recognition using dynamic time warping. In *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*, pages 1–5. IEEE, 2011.

[19] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[20] R. Srikant and R. Agrawal. *Mining sequential patterns: Generalizations and performance improvements*. Springer, 1996.

[21] Q. D. Tran and N. Q. Ly. Sparse spatio-temporal representation of joint shape-motion cues for human action recognition in depth sequences. In *Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF), 2013 IEEE RIVF International Conference on*, pages 253–258. IEEE, 2013.

[22] N. Vaswani, A. R. Chowdhury, and R. Chellappa. Activity recognition using the dynamics of the configuration of interacting objects. In *CVPR*, volume 2, pages II–633. IEEE, 2003.

[23] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *CVPR*, pages 588–595. IEEE, 2014.

[24] C. Wang, J. Flynn, Y. Wang, and A. L. Yuille. Recognizing actions in 3d by action-snippets and activated simplices. *AAAI*, 2016.

[25] C. Wang, Y. Wang, and A. L. Yuille. An approach to pose-based action recognition. In *CVPR*, pages 915–922. IEEE, 2013.

[26] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[27] J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*, pages 1290–1297. IEEE, 2012.

[28] D. Weinland, M. Özuysal, and P. Fua. Making action recognition robust to occlusions and viewpoint changes. In *Computer Vision–ECCV 2010*, pages 635–648. Springer, 2010.

[29] L. Xia and J. Aggarwal. Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera. In *CVPR*, pages 2834–2841. IEEE, 2013.

[30] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *CVPRW*, pages 20–27. IEEE, 2012.

[31] W. Yang, Y. Wang, and G. Mori. Recognizing human actions from still images with latent poses. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2030–2037. IEEE, 2010.

[32] M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine learning*, 42(1-2):31–60, 2001.

[33] Z. Zhao, D. Yan, and W. Ng. Mining probabilistically frequent sequential patterns in large uncertain databases. *Knowledge and Data Engineering, IEEE Transactions on*, 26(5):1171–1184, 2014.