Efficient dynamic texture classification with probabilistic motifs

Luong Phat Nguyen, Julien Mille, Dominique H Li, Donatello Conte, Nicolas Ragot

Efficient Dynamic Texture Classification with Probabilistic Motifs

Luong Phat Nguyen*, Julien Mille*†, Dominique H. Li*, Donatello Conte* and Nicolas Ragot*
* Laboratoire d'Informatique Fondamentale et Appliquée de Tours
Université de Tours, Tours 37000, France
† INSA Centre Val de Loire, Blois 41000, France

***Abstract*—We propose to tackle dynamic texture video classification as a pattern mining problem. In a nutshell, videos are represented by frequent sequences of representative patches. Firstly, we use a Gaussian Mixture Model to make the clustering of patches from training videos. Secondly, a soft assignment is used as an encoding method to construct sequences of probability vectors (*p-sequences*) representing sequences of spatio-temporal patches. Thirdly, for each class, we mine meaningful motifs appearing inside the training *p-sequences* by means of an adapted data mining approach. Finally, feature vectors are constructed from the mined motifs, using the probabilistic support, which quantifies the match between the *p-sequences*, of the video to be classified, and the key-motifs of the classes. Experimental results and analysis for dynamic texture classification on benchmark datasets (*i.e.* UCLA, Traffic) show the interest of the proposed method.**

I. Introduction

Dynamic textures are repetitive spatio-temporal patterns characterized by non-rigid and complex motions [1]. Extraction of spatio-temporal features and patterns may allow the characterization and classification of dynamic texture videos. In this paper, we are interested in creating a new representation of dynamic textures and measure its robustness by means of video classification, with a view to provide more insight on discriminant features than what can be expected from deep learning approaches. Let us first review existing texture video classification methods, *i.e.* hand-crafted features extraction and deep learning methods.

Early hand-crafted spatio-temporal feature-based methods depend on optical flow [2]–[5]. For example, in the work of [5], a motion/no-motion map is built and combined with mixed-state statistical models. However, optical flow-based methods, which lend themselves to the extraction of smooth motion fields, and thus might not represent dynamic textures well, which are usually made by chaotic motions in several directions. The time-evolving appearance of textures was explicitly modeled in [1], relying on a Linear Dynamical System (LDS). The method uses the model parameters as the input features. Another well-known family of hand-crafted features are spatio-temporal extensions of the Local Binary Pattern (LBP) method, such as CVLBP [6] or 2 dimensional LBP computed with 3 orthogonal planes (LBP-TOP) [7]. [8] come up with a combination of Gaussian filters and LBP patterns, where LBP descriptors are calculated from both blurred volumes and 3D difference of Gaussians volumes. Recently, a feature called momental directional pattern (MDP) [9] has been developed.

Since the breakthrough of AlexNet [10], many deep learning approaches have been developed for dynamic texture classification. Some works are based on information that is purely spatial [11], where 2D convolution filters are applied to each frame of a video. In the work of [11], two-level strategy is proposed: utilizing the transfer learning to extract mid-level features and forming a video feature representation by concatenating the mean vector and the diagonal entries of the co-variance matrix of the mid-level features. But such method neglects temporal regularity. In [12], feature extraction on 3 orthogonal planes based on convolutional neural networks is used. It achieves good results on many dynamic texture benchmark datasets. [13] introduce a learning-free ConvNet, operating on oriented 3^{rd} order Gaussian-based filtering, to extract features, fed to a classifier.

Beside machine learning approaches, data mining techniques, especially sequential pattern mining methods, have also been used to tackle various computer vision problems, such as image classification [14]–[16] or action recognition [17]–[19]. For example in [14], a data mining algorithm extracts frequent item patterns, which are used to create a bag-of-visual-words representation [20], for both unsupervised image ranking and supervised image classification. In [20], frequent patterns are used as mid-level features for image classification. For video data, [18] propose a pattern-growth mining method to extract sequential patterns. These patterns are then used as a bag-of-words to construct feature histograms, to classify human actions. The presented methods use a hard assignment from input observations to items and patterns. This can lead to unstable patterns and sensitive decisions, which is not suited to the fuzzy nature of dynamic textures. In order to avoid such problems, [19] propose a framework for action recognition where patterns are sequences of soft-assignments, rather than items themselves.

In this paper, we propose a novel representation of dynamic texture data, with video classification as final purpose, based on data mining. The proposed framework uses a sequential pattern mining algorithm to discover frequent sequences of image patches, that are referred to as key motifs, inspired from [19]. In the training phase, first, a set of symbols, corresponding to representative image patches, is extracted by unsupervised clustering. These symbols are the main

modes of patch distribution. Then, for each video class, a set of key motifs is constructed. This extraction handles both deterministic sequences (hard-assigned) and probabilistic sequences (soft-assigned). Sets of key motifs of the different classes are merged into a global one, in which each class is fairly represented. In the inference phase, probabilistic soft-assigned sequences are extracted from the video to be classified. The match between these probabilistic sequences and the key motifs of the unified set, makes a feature vector fed to the final classifier. The experimental analysis studies the use of this motif-based representation for video classification with an extensive parameter study, considering the impact on the number of key-motifs and the impact on classification performance.

We emphasize the fact that the proposed method does not follow the trend of convolutional neural networks, and more generally deep learning approaches. Representing dynamic textures using deep learning can be done by considering the learned features that travel along the successive layers of a convolutional and/or recurrent neural network. Since deep nets achieve very good performances, these features maps do represent well the spatio-temporal patterns that occur in dynamic textures. However, not every deep learning feature is discriminant. Most of deep nets have important redundancy, and interpretating features is intermediate layers is excessively hard, at least because of the quantity. We believe that alternative ways of extracting and representing discriminant image/video patterns, by means of sequential data mining here, should be explored in order to head towards explainable and interpretable machine learning. The code is publicly available at: https://github.com/lphatnguyen/proba_seq_mining.

II. Proposed method

Consider the task of video classification in C classes. In this section, we introduce an efficient framework which is based on a novel mining method of *p-sequences*, which are sequences of probability vectors. The proposed method is divided into three main stages:

- Unsupervised clustering of patches using a Gaussian mixture model, and construction of *p-sequences*.
- Mining key motifs from *p-sequences* for each class, thanks to sequential pattern mining.
- Constructing feature vectors which are based on probabilistic supports of the union of key motifs and classification using a SVM with χ^2 kernel.

A. Patch extraction and Gaussian mixture clustering

In this part of the paper, a simple method for patch extraction is proposed. From a video $\mathbf{V} \in \mathbb{R}^{T\times H\times W}$, multiple non-overlapping patches of size $\sigma \times \sigma$ are extracted. From the training dataset, we build a set $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1,...,|\mathcal{X}|}$, where $\boldsymbol{x}_i \in \mathbb{R}^{\sigma^2}$ is a flat vector representation of the i^{th} patch. Theses patches are used for calculating clusters. Beside K-Means clustering, Gaussian mixture models (GMM) are often used as clustering methods in computer vision to create dictionaries of visual symbols. Parameters of the GMM are estimated using the Expectation -Maximization (EM) method. Once they are set, the probability of a given patch can be easily calculated.

By definition, a Gaussian mixture (GM) is a combination of a finite number of K Gaussian distributions $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ where $k = 1, 2, ...K$ [21]. The Gaussian mixture model is made up by mixture weights $\pi_k \in \mathbb{R}$, means $\boldsymbol{\mu}_k \in \mathbb{R}^{\sigma^2}$ and covariances $\boldsymbol{\Sigma}_k \in \mathbb{R}^{\sigma^2\times\sigma^2}$. The Probability Density Function (PDF) at point $\boldsymbol{x}$ is:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \Sigma_{k=1}^{K}\pi_k\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k),\ 0 \leq \pi_k \leq 1,\ \Sigma_{k=1}^{K}\pi_k = 1$$

where $\boldsymbol{\theta}$ is the collection of all parameters of the GM (mixture weights, means and covariances). The expectation maximization algorithm (EM) is used as a learning method to update the parameters in the GM model [21]. The means computed by the EM algorithm form the dictionary of K symbols. The clustering is done on the whole training dataset $\mathcal{X}$, regardless of video classes.

After having learned the parameters of the model, a soft assignment is used rather than a hard assignment. In other words, a data point is represented by a vector of posterior probabilities rather than an assignment to a cluster. For a given data point $\boldsymbol{x}$, the posterior probability $p(k|\boldsymbol{x})$, *i.e.* the probability that $\boldsymbol{x}$ belongs to the k^{th} cluster, is

$$p_k(\boldsymbol{x}) = p(k|\boldsymbol{x}) = \frac{\pi_k\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\Sigma_{j=1}^{K}\pi_j\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \quad (1)$$

Using Eq (1), the soft assignment of a patch of size $\sigma \times \sigma$ is represented by a probability vector $\boldsymbol{p} = [p_1, p_2, ..., p_K] \in [0, 1]^K$. At a given spatial position, a sequence of consecutive patches forms a spatio-temporal patch of size $T \times \sigma \times \sigma$. Applying Eq (1) for each 2D patch of that sequence and each cluster k, the p-sequence $\boldsymbol{P}$ of length T is built, such as: $\boldsymbol{P} = \langle \boldsymbol{p}^1, \boldsymbol{p}^2, ..., \boldsymbol{p}^T \rangle$ where each $\boldsymbol{p}^j$ $(j = 1, 2, ..., T)$ is a probability vector in $[0, 1]^K$. Then, $\mathcal{P}$ is the set of *p-sequences* to be mined, $\mathcal{P} = \{\boldsymbol{P}_1, ..., \boldsymbol{P}_{|\mathcal{P}|}\}$.

B. Mining key motifs

Motif mining is a method to discover relevant subsequences as patterns in a sequential database. Many early motif mining methods are based on the *Apriori* algorithm for databases where items are certain. Such algorithm can be considered as a method of pattern expansion. The main idea is that subsequences of frequent motifs can also be frequent. Algorithm 1 demonstrates how super-sequence patterns can be generated from shorter mined patterns. In other words, from patterns containing l symbols, referred to as l-motifs, candidate patterns of $l + 1$ symbols are generated. For a particular case, an item can be considered as a 1-motif composed of one symbol in the dictionary.

Unlike in itemized sequence pattern mining, our data to be mined (*p-sequences*) are uncertain (probabilistic). To do so, we follow the mining procedure proposed in [19]. Algorithm 2 describes the motif-mining algorithm applied in our case for each class c, where each element $t^i, i = 1, ..., |\boldsymbol{T}|$ of the candidate motif $\boldsymbol{T}$ is a symbol corresponding to a cluster $k = 1, ..., K$ of the learned GM model. Notice that the

Algorithm 1: Expansion Algorithm

Input: $\mathcal{T}^l = \{\boldsymbol{T}^l_1, \boldsymbol{T}^l_2, ..., \boldsymbol{T}^l_{|\mathcal{T}^l|}\}$: set of $l-$motifs
Output: $\mathcal{T}^{l+1}$: set of $l+1-$motifs

$\mathcal{T}^{l+1} \longleftarrow \emptyset$
for $i = 1, 2, ..., \left|\mathcal{T}^l\right|$ **do**
  $tail \longleftarrow \boldsymbol{T}^l_i(2 : \left|\boldsymbol{T}^l_i\right|)$
  for $j = 1, 2, ..., \left|\mathcal{T}^l\right|$ **do**
    $head \longleftarrow \boldsymbol{T}^l_j(1 : \left|\boldsymbol{T}^l_j\right| - 1)$
    if $tail = head$ **then**
      $\mathcal{T}^{l+1} \longleftarrow \mathcal{T}^{l+1} + concat(\boldsymbol{T}^l_i, \boldsymbol{T}^l_j(\left|\boldsymbol{T}^l_j\right|)$
    end
  end
end

Algorithm 2: Key motifs Mining Algorithm

Input: $\mathcal{P}^c$: *p-sequences* set of class c, ϵ: support threshold
Output: Mined key motifs of the class c – $\mathcal{T}^c$

$\mathcal{T}^1 \longleftarrow \{1 - \text{motifs}\}$
$l \longleftarrow 2$
while $\mathcal{T}^{l-1} \neq \emptyset$ **do**
  $\mathcal{T}^l \longleftarrow$ expand$(\mathcal{T}^{l-1})$
  for $i = 1, 2, ..., \left|\mathcal{T}^l\right|$ **do**
    support $\longleftarrow 0$
    for $j = 1, 2, ..., |\mathcal{P}^c|$ **do**
      support $\longleftarrow$ support + $\eta\left(\boldsymbol{T}^l_i, \boldsymbol{P}^c_j\right)$
    end
    if $\frac{support}{|\mathcal{P}^c|} \leq \epsilon$ **then**
      Remove $\boldsymbol{T}^l_i$ from $\mathcal{T}^l$
    end
  end
  $l \longleftarrow l + 1$;
end

expansion step (Algorithm 1) is called at line 4. Algorithm 2 depends on the computation of the probabilistic support $\eta(\boldsymbol{T}, \boldsymbol{P})$. This support measures how well the motif $\boldsymbol{T}$ is matched to a *p-sequence* $\boldsymbol{P}$. The calculation of the probabilistic support searches for a mapping table M where each $M(i) \in \{1, 2, ..., |\boldsymbol{P}|\}$ represents a row i of the computed dynamic table. The probabilistic support is computed as

$$\eta(\boldsymbol{T}, \boldsymbol{P}) = \max_{M} \prod_{i}^{|\boldsymbol{T}|} p_{t^i}^{M(i)},$$
$$\text{Subject to:} M(i) < M(i+1), M(i+1) - M(i) \leq g. \quad (2)$$

where g is the maximum gap constraint and $p_{t^i}^{M(i)}$ is the probability, at temporal position $M(i)$, of symbol t^i. Considering that the candidate motif $\boldsymbol{T}$ has $|\boldsymbol{T}|$ elements and the *p-sequence* $\boldsymbol{P}$ has T probability vectors, as described in [19], the probabilistic support function $f(\boldsymbol{T}(1 : |\boldsymbol{T}|), \boldsymbol{P}(1 : T))$ measures the probability of matching between $\boldsymbol{T}(1 : |\boldsymbol{T}|) = \langle t^i \rangle_{i=1,2,...,|\boldsymbol{T}|}$ and $\boldsymbol{P}(1 : T) = \langle \boldsymbol{p}^j \rangle_{j=1,2,...,T}$. At a given position i in $\boldsymbol{T}$ and j in $\boldsymbol{P}$, the probabilistic support function $f(\boldsymbol{T}(1 : i), \boldsymbol{P}(1 : j))$ is calculated using equation 3.

$$f(\boldsymbol{T}(1 : i), \boldsymbol{P}(1 : j)) = p^j_{t^{(i)}} \times \max_{k \in \{j-g, ..., j-1\}} f(\boldsymbol{T}(1 : i-1), \boldsymbol{P}(1 : k)) \quad (3)$$

which can be implemented easily using dynamic programming. When finished constructing the dynamic table, the probabilistic support is the maximum value of the last row of the table: $\eta(\boldsymbol{T}, \boldsymbol{P}) \longleftarrow \max_{j \in \{1, ..., |\boldsymbol{P}|\}} f(\boldsymbol{T}(1 : |\boldsymbol{T}|), \boldsymbol{P}(1 : j))$.

C. Classification

A video sequence contains multiple non-overlapping video subsequences of size $T \times H \times W$. Each video subsequence consists of several non-overlapping spatio-temporal patches (represented by the $\boldsymbol{P}$ using Eq. 1). In this part, a bag-of-motifs method is used from the mined motifs of all classes. This process is divided into 3 steps: (a) uniting mined motifs of each class, (b) constructing probability histograms for each video sample and (c) classification learning. After having mined sequential motifs using Algorithm 2, a union of the motifs of all considered classes is made. The union set of mined motifs of all classes is $\mathcal{T}_\cup = \bigcup_{c=1}^{C} \mathcal{T}_c$. The previous mining of key motifs on a per-class basis ensures that, in this union, each class is sufficiently represented. If key motifs had been extracted regardless of classes, we could end up with a set where some classes would be much more represented than other ones when dealing with unbalanced datasets. Moreover, making this union allows us to define a global bag of motifs that will take into account the relative representation of each motif in each class, including the fact that some motifs could be ambiguous (*i.e.* represent different classes).

A bag-of-feature vector is generally a histogram that counts the number of occurrences of each symbol in a sample [22]–[24]. However, in our case, a feature vector is built based on the probabilistic supports. For a given test p-sequence $\boldsymbol{P}$, a feature vector $\boldsymbol{f}$, is made up of the probabilistic supports between each key motifs of the union set $\mathcal{T}_\cup$ and $\boldsymbol{P}$. Hence, we have $\boldsymbol{f} \in \mathbb{R}^{|\mathcal{T}_\cup|}$. To train the SVM classifier and at test time, we use feature vectors of video subsequences, instead of feature vectors of individual p-sequences. A feature vector of a video subsequence is calculated using the average of the $\boldsymbol{f}$'s of the p-sequences in this video subsequence. For the classification step, an SVM classifier based on the χ^2 kernel is used [25].

D. Differences with Wang et al. [19]

There exists several differences between our method and [19]. Since the application is different, we use image instead of 3D body joint coordinates. Second, we extract symbols by fitting a GMM, while [19] use the activated simplices method [26]. Third, contrary to what is done in [19], the mining is done with a fixed threshold ϵ. Indeed, thanks

to the modification mentioned above in Algorithm 2, there is no need to change ϵ during the mining process in order to obtain a given number of key-motifs. Experiments show that this threshold does not play a crucial role for accuracy but only for compactness of the model (number of extracted motifs). Lastly, [19] classify a test example by assigning it to the class which has the key motifs leading to the highest probabilistic supports. We found that characterizing a test example with the probabilistic support of all key motifs to be a more discriminant representation for classification.

E. Additional study on FFT magnitude patches

In our research, we are interested in exploring the magnitude values of the Fourier transforms of the texture input patch instead of the gray-scale patches. In other words, the Fourier transform is applied for a single patch but not the entire frame. The final Fourier transform magnitude patch $\boldsymbol{x}^{FT}$ is computed using the following procedure: (1) each gray-scale patch is transformed into the frequency domain using the FFT algorithm which is widely used for computing the Fourier transform of a patch $\boldsymbol{x}^{FT} \in \mathbb{C}^{\sigma \times \sigma}$; (2) a top left quadrant of the FFT patch is taken as it is symmetric along the horizontal and vertical axis, so the FFT patch has a size of $\mathbb{C}^{\frac{\sigma}{2} \times \frac{\sigma}{2}}$; (3) the magnitude of the FFT patch is computed as $\boldsymbol{x}^{FT} \in \mathbb{R}^{\frac{\sigma}{2} \times \frac{\sigma}{2}}$. Then, the key motif mining approach is used similarly on FFT patches instead of gray-scale patches.

III. Experiments

In this section, we are presenting the results obtained by our proposed method on different benchmark datasets (Traffic, UCLA) while comparing them to those of state-of-the-art methods.

A. Datasets and protocols

Traffic dataset [27] consists of 254 video samples. The sequences are recorded with a resolution of 320×240 with a temporal length between 42 and 52 frames, at 10 fps. Each video sample is then spatially resized to 80×60 and cropped to a size of 48×48 over the area where the motion is the most prominent. There are 3 classes in this database: heavy, medium and light traffic. For this dataset, a 4-fold evaluation protocol is used [27]. The average score of the 4-fold is recorded as the final result.

UCLA dataset [1] consists of 50 classes, where each class contains 4 sequences. As the result, the dataset has 200 dynamic textures sequences in total. Each original sample is captured with 75 frames, each of size 160×110. A slightly modified version of UCLA dataset is often utilized for dynamic texture classification where the original samples into sub-sequences of size 48×48. Three popular challenges of this dataset [9], [28], [29] are often used for classification. In the **50-class** (4-fold) configuration, a quarter of the data in each class is addressed as testing set and the remaining for the learning. The experiment is repeated 4 times and the average score of the 4 folds is reported as the final result. As for **9-class** configuration, a reorganization of the original 50-class configuration is made into 9 classes of *boiling water, fire, flower, fountains, plants, sea, smoke, water and waterfall*. Half the videos in each class are randomly selected for training and the remaining for testing. This step is repeated 20 times and the mean score of all the repetitions is recorded as the final result. In the **8-class** configuration, following the 9-class configuration, the *plants* class is omitted since the number of samples is much greater than those of other classes. The same evaluation metric as the 9-class configuration is used in order to evaluate this sub-dataset.

B. Experimental setup

As mentioned above, depending on the dataset and the configuration, a N-fold validation is performed. Each fold is composed of different set of videos. In our experiment, a video sample of each dataset is divided into multiple non-overlapping sequences of temporal lengths of 1 second. For the traffic database, a video subsequence contains 10 frames. For the UCLA dataset, a video subsequence has 15 frames. The final classification result for each test video sample is the majority vote of classified subsequences in the video sequence, which means the final result of a video is the majority class assigned to the subsequences of the video sample.

For the learning of the GM model, 4 patches in each video frame from training set are randomly selected. The GM model is initialized randomly. To assess the stability of our method against the choice of symbols (that depends on these random selections), the impact on the results is evaluated over 8 runs for each split of the N-fold validation.

The minimum support threshold ϵ (algorithm 2) is an important parameter. Therefore, an analysis could be useful to study the impact of such parameter ϵ on the number of mined motifs. Consequently, during the 8 runs, an analysis of parameters is also conducted to evaluate the impact on both accuracy and the number of motifs. This process is done for each set of parameters on the Traffic dataset and UCLA dataset. This analysis is only done on the gray-scale patches.

In the training and testing stage, the spatio-temporal patches are extracted at every possible position with a non overlapping condition. For each dataset and their configurations, both gray-scale patches and FFT patches are evaluated.

C. Experimental results

Traffic database: Looking at figure 1, the best results are obtained with a patch size of 8 (compared to 12). This is the most important parameter. Figure 1 shows that the number of motifs generated from codebook of sizes 20 and 50 are almost the same. Furthermore, the performances based on these codebook sizes are very close. For the minimum support threshold, good results can be achieved with a minimum support between 0.01 and 0.07. However, the number of mined key-motifs increases as the minimum support threshold decreases. Therefore, rather than using 500 or 600 key motifs with ϵ of 0.01, high results (eg. 96,56% of accuracy) can be achieved by using approximately 100 key motifs with an $\epsilon = 0.05$. Column 5 of table II shows the performance of

TABLE I
COMPARISON OF FEATURE VECTOR DIMENSION FOR DYNAMIC TEXTURE CLASSIFICATION.

Datasets / Method	UCLA 50-4fold	UCLA 9-class	UCLA 8-class	Traffic
3D-OFT [30]	290	290	290	–
HLBP [31]	1536	1536	1536	–
MEWLSP [28]	1536	1536	1536	–
RI-VLBP [32]	16384	16384	16384	16384
LBP-TOP [7]	768	768	768	768
CDT-TOP [33]	75	75	75	75
CNDT [34]	144	420	336	336
DM [29]	169	169	169	297
MDP [9]	3880	3880	3880	–
Gray-scale	651	3610	1196	118

TABLE II
COMPARISON OF RECOGNITION RATES (%) ON THE UCLA DATASET AND TRAFFIC DATABASE.

Datasets / Method	UCLA 50-4fold	UCLA 9-class	UCLA 8-class	Traffic
KDT-MD [35]	97.5	–	–	–
DFS [36]	89.5	–	–	–
CFV [37]	–	85.10	85.00	–
3D-OFT [30]	87.10	97.23	99.50	–
HLBP [31]	95.00	98.30	97.50	–
MEWLSP [28]	96.50	98.50	98.04	–
GoogleNet [12]	99.50	98.35	99.02	–
AlexNet [12]	99.50	98.05	98.48	–
CVLBP [6]	93.00	96.90	95.65	–
RI-VLBP [32]	77.50	96.30	91.96	93.31
LBP-TOP [7]	95.00	96.00	93.67	93.70
CDT-TOP [33]	95.00	96.33	93.41	93.70
MDP [9]	100.00	98.90	98.70	–
V-BIG [8]	99.50	97.95	97.50	–
CNDT [34]	95.00	95.61	94.32	96.46
DM [29]	98.50	97.80	96.22	96.60
Gray-scale	98.50*	96.69*	95.45*	96.56*
FFT	99.50	96.55	95.54	98.44

Note: '–' indicates that the result is not available.
'*' indicates that the result is averaged over 8 runs.

the proposed method for the Traffic database. The proposed method outperforms most of existing methods on this dataset and it is even on par with the diffusion-based method [29] with only 0.04% less (96.60% and 96.56%) with the average accuracy over 8 runs. Comparing to the diffusion-based model methods [33], [34], our motif-based method shows better results by at least 1%. Looking at the results obtained by the LBP-based methods [7], [32], it classifies the dataset much better by up to almost 3% being more compact in terms of feature vector size, which can be seen with Table I (16384 [32] and 768 [7] to 118 with our approach). In addition, the result obtained by using a quadrant of FFT image as input is

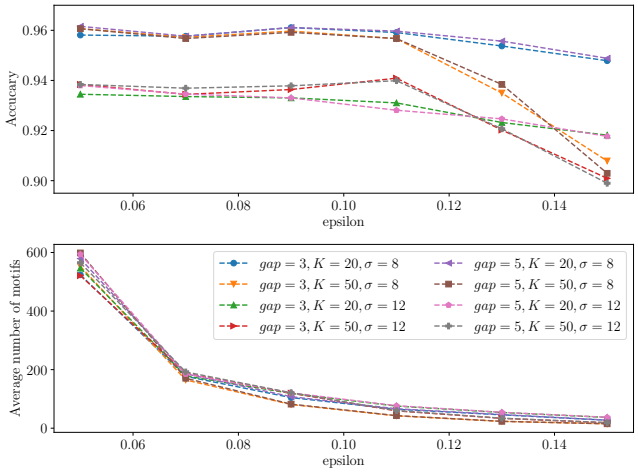

Fig. 1. An analysis of the stability of the method in terms of minimum support threshold and the number of mined motifs for the traffic dataset.

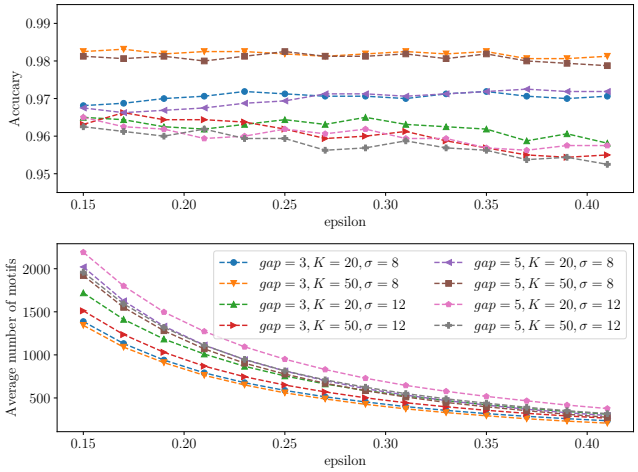

Fig. 2. An analysis of the stability of the method in terms of minimum support threshold and the number of mined motifs for the UCLA 50-class dataset.

very high with 98.44% of accuracy (table II). It is currently the highest score for this dataset. This result shows the efficacy of the proposed method for the Traffic dataset.

UCLA 50-class: Best results are obtained with the patch size of 8 (figure 2). But this time, the accuracy is better with 50 clusters than with 20 clusters. This can be explained by the complexity of textures occurring in the videos. For the minimum support, good results are achieved with ϵ between 0.19 and 0.3. In figure 2, the averaged highest result is obtained with only 651 motifs. From table II, the proposed method achieves 98.50% of the classification rate on this subset as the average score over 8 runs. It performs on par with the diffusion-based model (DM) [29] and outperforms other recent existing methods for example HLBP [6] or MEWLSP [28] as well as many other hand-craft methods. Nonetheless, in comparison with the DL method, the mining of key motifs

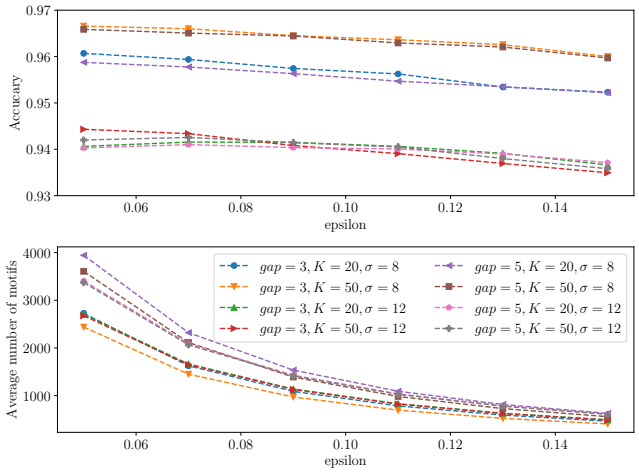

Fig. 3. An analysis of the stability of the method in terms of minimum support threshold and the number of mined motifs for the UCLA 9-class dataset.

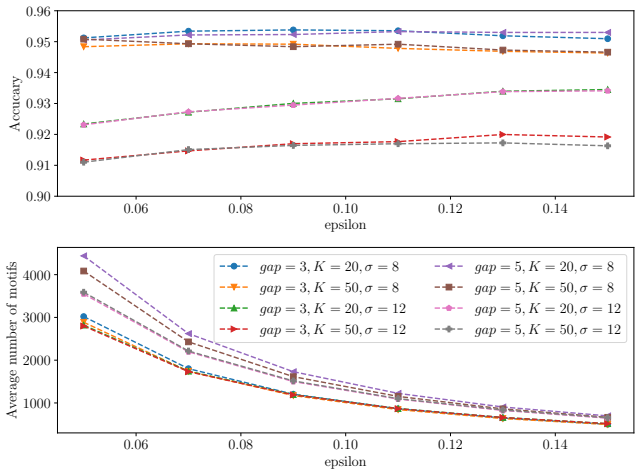

Fig. 4. An analysis of the stability of the method in terms of minimum support threshold and the number of mined motifs for the UCLA 8-class dataset.

method is just 1% less than the existing DT-CNN models [12]. Furthermore, in comparison to a recent LBP-based method, MDP [9] which scores 100% of accuracy for this subset, our method is just 1.5% less, with a higher compacity from 3880 to 651 features (see Table I). When FFT patches are used, the result (99.50%) increases by 1% in terms of accuracy and even on par with the DL approaches.

UCLA 9-class and 8-class: Figures 3 and 4 illustrate the parameters analysis of the UCLA 9-class and UCLA 8-class schemes. The tendency of classification rates is approximately the same as the UCLA 50-class and Traffic dataset as good results are obtained with ϵ ranging in the middle of the analyzing interval. The number of motifs begins to converge as ϵ passes 0.15. From table II, our model outperforms some of the other classical computer vision methods of about 1% or even better than CFV [37] of about 10% for the 9-class configuration. Nevertheless, it cannot achieve as good results as DL models [12] or a recent vision-based model of MDP [9] which are at 98.35% and 98.90% respectively. In the 8-class configuration, our method reaches a result of 95.45% of accuracy. This result is higher than many methods such as CDT-TOP [33], CVLBP [6] or CFV [37]. Nonetheless, the best results for this configuration is obtained with GoogleNet [12] with an accuracy of almost 100%. Plus, using only one quadrant for this scheme also results in good result (96.55%) but not as high as with grayscale images as inputs. Moreover, UCLA 8-class with FFT patches helps improving the performance by only 0.1% (95.54% in accuracy) comparing with the grayscale patches. Overall, the results, even if not the best, are satisfactory considering both accuracy and compacity (Table I).

In general, the performance of the proposed approach is comparable to the state-of-the-arts methods. Looking at the overall accuracy of the experiments, different sets of parameters are tested in order to measure the impact of each parameter for each module. Overall, it can be seen that the two most important parameters in the approach is the patch size σ and the number of cluster K. A smaller patch size of 8 with a greater number of cluster of 50 seems to perform best. Moreover, a maximum gap constraint also contributed to the performance of the approach. A smaller gap seems to generate less motifs than a bigger gap (a maximum gap constraint of 3 and 5) but with slightly better accuracy. This means that the set of extracted key motifs using a gap of 3 are more meaningful and more discriminant than with a gap of 5. As for the minimum support threshold (directly related to the number of motifs), increasing gradually this value can help us reach an optimal value and achieve highest result. This helps to remove some redundant motifs extracted from the mining process and to keep only the meaningful motifs. Passing this optimal value, the system begins to remove some discriminant motifs between classes and the accuracy decreases.

IV. Discussion and Future Works

Experimental classification results show that our key motifs-based representation is relevant for dynamic textures. The proposed method performs on par existing work, including deep learning approaches.

For further perspectives, mining motifs using p-sequences can be effective but not all mined motifs are useful for visual representation of video data. Therefore, motif filtering could be applied in order to extract only useful motifs. This addition should help speeding up the classification step and reduce the number of elements in the feature vector. One way to do it is to compute an entropy-like value of each motif. This value can be computed using the probabilistic support computed using p-sequences in each class and the whole dataset. If the entropy of the motif is low, the extracted motif is meaningful and represents well the input video. On the other hand, if the computed motif entropy is high, the motif appears in every class with an equal distribution to the data in every class. Such motifs with high entropy should be eliminated from the motif set.

REFERENCES

[1] S. Soatto, G. Doretto, and Ying Nian Wu, "Dynamic textures," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2, 2001, pp. 439–446 vol.2.

[2] R. C. Nelson and R. Polana, "Qualitative recognition of motion using temporal texture," *CVGIP: Image understanding*, vol. 56, no. 1, pp. 78–89, 1992.

[3] R. Péteri and D. Chetverikov, "Dynamic texture recognition using normal flow and texture regularity," in *Iberian Conference on Pattern Recognition and Image Analysis*, 2005, pp. 223–230.

[4] Z. Lu, W. Xie, J. Pei, and J. Huang, "Dynamic texture recognition by spatio-temporal multiresolution histograms," in *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)-Volume 1*, vol. 2, 2005, pp. 241–246.

[5] T. Crivelli, B. Cernuschi-Frias, P. Bouthemy, and J.-F. Yao, "Motion textures: modeling, classification, and segmentation using mixed-state markov random fields," *SIAM Journal on Imaging Sciences*, vol. 6, no. 4, pp. 2484–2520, 2013.

[6] D. Tiwari and V. Tyagi, "Dynamic texture recognition based on completed volume local binary pattern," *Multidimensional Systems and Signal Processing*, vol. 27, no. 2, pp. 563–575, 2016.

[7] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[8] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and N. S. Vu, "Volumes of Blurred-Invariant Gaussians for Dynamic Texture Classification," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11678 LNCS, pp. 155–167, 2019.

[9] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and X. S. Nguyen, "Momental directional patterns for dynamic texture recognition," *Computer Vision and Image Understanding*, vol. 194, p. 102882, 2020.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[11] X. Qi, C.-G. Li, G. Zhao, X. Hong, and M. Pietikäinen, "Dynamic texture and scene classification by transferring deep image features," *Neurocomputing*, vol. 171, pp. 1230–1241, 2016.

[12] V. Andrearczyk and P. F. Whelan, "Convolutional neural network on three orthogonal planes for dynamic texture classification," *Pattern Recognition*, vol. 76, pp. 36–49, 2018.

[13] I. Hadji and R. P. Wildes, "A spatiotemporal oriented energy network for dynamic texture recognition," in *IEEE International Conference on Computer Vision*, 2017, pp. 3066–3074.

[14] S. Nowozin, K. Tsuda, T. Uno, T. Kudo, and G. Bakir, "Weighted substructure mining for image analysis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[15] J. Yuan, Y. Wu, and M. Yang, "Discovery of collocation patterns: from visual words to visual phrases," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[16] J. Yuan, M. Yang, and Y. Wu, "Mining discriminative co-occurrence patterns for visual recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 2777–2784.

[17] A. Gilbert, J. Illingworth, and R. Bowden, "Fast realistic multi-action recognition using mined dense spatio-temporal features," in *IEEE 12th International Conference on Computer Vision*, 2009, pp. 925–931.

[18] S. Nowozin, G. Bakir, and K. Tsuda, "Discriminative subsequence mining for action classification," in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[19] C. Wang, Y. Wang, and A. L. Yuille, "Mining 3d key-pose-motifs for action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2639–2647.

[20] B. Fernando, E. Fromont, and T. Tuytelaars, "Mining mid-level features for image classification," *International Journal of Computer Vision*, vol. 108, no. 3, pp. 186–203, 2014.

[21] F. Nielsen, "K-mle: A fast algorithm for learning statistical mixture models," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 869–872.

[22] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 2001.

[23] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, IEEE International Conference on*, vol. 3, 2003, pp. 1470–1470.

[24] D. Lo, H. Cheng, J. Han, S.-C. Khoo, and C. Sun, "Classification of software behaviors for failure detection: A discriminative pattern mining approach," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, p. 557–566.

[25] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International journal of computer vision*, vol. 103, no. 1, pp. 60–79, 2013.

[26] C. Wang, J. Flynn, Y. Wang, and A. Yuille, "Recognizing actions in 3d using action-snippets and activated simplices," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[27] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 846–851.

[28] D. Tiwari and V. Tyagi, "Dynamic texture recognition using multiresolution edge-weighted local structure pattern," *Computers & Electrical Engineering*, vol. 62, pp. 485 – 498, 2017.

[29] L. C. Ribas, W. N. Goncalves, and O. M. Bruno, "Dynamic texture analysis with diffusion in networks," *Digital Signal Processing*, vol. 92, pp. 109–126, 2019.

[30] Y. Xu, S. Huang, H. Ji, and C. Fermüller, "Scale-space texture description on sift-like textons," *Computer Vision and Image Understanding*, vol. 116, no. 9, pp. 999 – 1013, 2012.

[31] D. Tiwari and V. Tyagi, "A novel scheme based on local binary pattern for dynamic texture recognition," *Computer Vision and Image Understanding*, vol. 150, pp. 58 – 65, 2016.

[32] G. Zhao and M. Pietikäinen, "Dynamic texture recognition using volume local binary patterns," in *Dynamical Vision*. Springer Berlin Heidelberg, 2007, pp. 165–177.

[33] W. N. Gonçalves and O. M. Bruno, "Dynamic texture analysis and segmentation using deterministic partially self-Avoiding walks," *Expert Systems with Applications*, vol. 40, no. 11, pp. 4283–4300, 2013.

[34] W. N. Gonçalves, B. B. Machado, and O. M. Bruno, "A complex network approach for dynamic texture recognition," *Neurocomputing*, vol. 153, pp. 211–220, 2015.

[35] A. B. Chan and N. Vasconcelos, "Classifying video with kernel dynamic textures," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–6.

[36] Y. Xu, Yuhui Quan, H. Ling, and H. Ji, "Dynamic texture classification using dynamic fractal analysis," in *International Conference on Computer Vision*, 2011, pp. 1219–1226.

[37] Y. Wang and S. Hu, "Chaotic features for dynamic textures recognition," *Soft Computing*, vol. 20, no. 5, pp. 1977–1989, 2016. [Online]. Available: http://dx.doi.org/10.1007/s00500-015-1618-4