IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 June 2024, accepted 27 June 2024, date of publication 1 July 2024, date of current version 9 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3421666

RESEARCH ARTICLE

Dynamic Texture Classification Using AutoEncoder-Based Local Features and Fisher Vector Encoding

ZHE LI[1], XIAOCHAO ZHAO[1], TIANFAN ZHANG[1], XIAO JING[2], WEI SHI[1], AND QIAN CHEN[1]
[1]School of Computer and Information Science, Hubei Engineering University, Xiaogan 432000, China
[2]School of Cybersecurity, Northwestern Polytechnical University, Xi'an 710072, China

Corresponding author: Xiaochao Zhao (xczhao@hbeu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 72002067, in part by the Natural Science Foundation of Hubei Province under Grant 2020CFB497 and Grant 2022CFB516, and in part by the Humanities and Social Science Research Project of Ministry of Education of China under Grant 22YJCZH242.

ABSTRACT Dynamic texture classification has been widely studied because of its applications in various computer vision tasks. The key to classifying dynamic textures lies in describing them, *i.e.*, extracting features from them. A variety of traditional dynamic texture descriptors have been carefully designed in many research studies. And some researchers directly use pre-trained deep models for feature extraction. However, training a deep model from scratch for dynamic texture description is rarely explored due to the lack of a large-scale dynamic texture dataset. In this paper, we propose to train a deep model on existing small-scale dynamic texture datasets for feature extraction. We first randomly sample a number of 3D cubes from each training video. Then a simple AutoEncoder network is trained with the cubes, and the encoder will serve as a local feature extractor. The features extracted from all the training cubes are used to fit a Gaussian mixture model, which will later be used for Fisher vector encoding. Finally, given a video, we densely sample cubes, feed them into the encoder, and encode the output local features into a global feature vector using the learned Gaussian mixture model. The proposed method is evaluated on three benchmark datasets with various evaluation protocols and its effectiveness is verified by the obtained competitive results.

INDEX TERMS Dynamic texture, feature extraction, AutoEncoder, Gaussian mixture model, fisher vector encoding.

I. INTRODUCTION

Dynamic texture (DT) is the extension of static texture in the temporal domain [1]. DTs are videos that contain repetitive patterns in both the spatial and temporal domain. Typical DT examples are sea waves, running water, and swaying trees. DT classification has been extensively studied in the last two decades because of its broad application in the field of computer vision, such as traffic monitoring [2], fire detection [3], crowd management [4], and facial analysis [5], [6], [7].

Similar to static texture classification, the key for conducting an effective classification on a set of DTs is feature extraction. However, DT classification is more challenging because both static and dynamic patterns in DT can be easily influenced by the changes of illumination, viewpoint, scale and rotation, which makes it hard to extract stable and discriminative features [8], [9]. Many DT descriptors have been reported in the literature. According to whether a learning process is involved and what type of learning process it is, DT descriptors can be roughly grouped into three classes: handcrafted, learning-based, and deep-learning-based. Some typical and representative research studies in each class are briefly reviewed below.

Handcrafted DT descriptors are carefully designed by researchers without involving any learning process. Early methods [10], [11], [12] focus on the motion patterns and estimate optical flow information, calculating statistics from

The associate editor coordinating the review of this manuscript and approving it for publication was Dian Tjondronegoro.

which DT features are derived. Optical flow features are usually unstable and are rarely utilized nowadays. Another line of research in this class is to use the simple but effective local binary pattern (LBP) [13]. Zhao and Pietikainen [14] extended LBP into a spatio-temporal descriptor called volume LBP (VLBP) for DT description. Later, they treated a DT video as three image sequences, from each of which LBP features were extracted and concatenated for DT description (called LBP on three orthogonal planes, LBP-TOP) [5]. The success of VLBP and LBP-TOP has inspired many research works [6], [15], [16], [17], [18]. However, the high performance of LBP-based methods is usually obtained at the cost of high dimensionality, which may limit their application in some scenarios. Fractal structure, a geometrical property, is also utilized for DT description [19], [20], [21], [22], [23]. The design of handcrafted DT descriptors generally requires strong prior knowledge and they do not adapt well to new data [24].

Learning-based DT descriptors refer to those using traditional feature learning techniques. As a DT is considered to be generated by a linear dynamical system (LDS) [1], [25], the estimated system parameters are used for DT description [25], [26], [27], [28], [29], [30], [31], [32], [33], [34]. However, a DT video is too complex to be precisely modeled by a linear system and the performance is limited. Incorporating a learning process to improve the representative power of LBP has also been explored, such as optimizing LBP structures [35], [36], salience analysis [37], filter learning [38], [39], [40], [41], and dictionary learning [42], [43], [44], [45], [46]. Involving a traditional learning process does improve the performance for DT classification. But there generally exists a set of key parameters needed to be carefully tuned such that good performance can be obtained, making these methods complex.

Deep-learning-based DT descriptors are those that learn features from data by applying deep learning techniques. Most methods in this class make use of the high representational power of deep networks (*e.g.*, VGGNet [47], AlexNet [48], and GoogleNet [49]) that are pre-trained on external large-scale datasets [50], [51], [52], [53]. Methods in this class generally outperform those in the other two classes. But it remains unclear whether a network can be trained directly with DT datasets and used for DT classification.

According to the above review and discussion, it is obvious that learning-based DT descriptors, especially those deep-learning-based ones, are more preferable in terms of performance. But there is an interesting question of why the researchers choose to borrow the representational power from pre-trained networks, rather than training one from scratch. One answer is that existing DT datasets are insufficient for the application of deep learning techniques. We disagree with it and argue that deep learning techniques can be directly applied to existing DT datasets, which is partly supported by another data-limited task of diagnosing faults of aero-engine bearings [54], [55]. To further verify our point, we will train a simple network from scratch and utilize it for DT description.

In this paper, we propose a DT descriptor on the basis of AutoEncoder [56] and Fisher vector (FV) encoding [57] (denoted as AE-FV). Specifically, we first randomly sample a small number of local sub-videos from each training DT video, resulting in a training dataset. Then a simple AutoEncoder network is trained on this dataset. Subsequently, all the randomly sampled sub-videos are fed into the encoder of the trained AutoEncoder and their outputs are utilized to fit a Gaussian mixture model (GMM). To extract features from a given DT, we densely sample local sub-videos from it and feed them into the encoder, of which the outputs are aggregated into a global DT feature vector by applying FV encoding with the trained GMM. The proposed AE-FV is extensively evaluated on three benchmark DT datasets with various evaluation protocols. The obtained results outperform many traditional methods and some deep-learning-based ones, thereby showing its effectiveness for DT description. The contributions of this work are threefold: 1) the practicability of training a network from scratch for DT description is verified to a certain extent; 2) as the local feature extractor of a bag-of-words (BoW) model, even a simple deep model can effectively improve performance; 3) the dimensionality of our AE-FV descriptor is no more than 3000, which is lower than many state-of-the-art methods.

The rest of this paper is organized as follows. Section II reviews related work. Section III describes the proposed AE-FV in detail. Section IV presents details about experiments. Section V concludes this paper.

II. RELATED WORK

In this section, we mainly focus on DT descriptors reported in the last decade, most of which will be compared with the proposed method in terms of DT classification in Section IV. Those methods that are already introduced in Section I are skipped.

A. HANDCRAFTED

Unlike those early methods that simply use statistics of optical flow for DT description, Nguyen et al. proposed to combine motion and appearance, and designed two DT descriptors, *i.e.*, features derived from directional trajectories based on motion angle patterns (FD-MAP) [58] and directional dense trajectory patterns (DDTP) [59]. Both of them provide relatively good performance.

Besides the above mentioned methods, most DT descriptors in this class are variants of LBP. Sun et al. [60] incorporated lacunarity analysis with local ternary pattern (LTP) [61], resulting in a descriptor called LTP-Lac. Tiwari and Tyagi [16], [17] proposed to improve the performance of VLBP and LBP-TOP by introducing additional information about the values of local central pixels, the magnitudes of local pixel differences, and local contrast. Later, they applied Weber's law on LBP patterns (WLBPC) [62], and also proposed the edge-weighted local structure pattern (EWLSP) descriptor [63] that adaptively determined the local threshold. On the basis of VLBP, Zhao et al. [6] proposed

to count the number of ones in a binary code as a new feature code such that including more pixels would not significantly increase feature length. And they also made use of additional information in a way similar to [16] and [17]. By modifying the binary encoding scheme in LBP-TOP, Nguyen et al. proposed a series of descriptors, which are completed local structure pattern on three orthogonal planes (CLSP-TOP) [64], complete statistical adaptive pattern (CSAP-TOP) [65], hierarchical local pattern (HILOP) [66], Rubik Gaussian-based pattern (RUBIG) [67], and momental directional pattern (MEMDP) [68]. Moreover, they also tried to extract LBP features from Gaussian-filtered data or Gaussian gradients [18], [69], [70], [71].

As DTs contain repetitive patterns in the spatio-temporal domain, a group of research works utilize fractal analysis to capture such properties for DT description. Xu et al. [20] built a DT descriptor called dynamic fractal spectrum (DFS). Dubois et al. [72] tried to capture such self-similarity information via 2D+T curvelet transform. Quan et al. [23] proposed to use lacunarity, a specialized concept in fractal geometry, to form a DT descriptor named spatio-temporal lacunarity spectrum (STLS).

B. LEARNING-BASED

Although LDS is the original concept in the field of DT analysis, a few LDS-based DT descriptors are reported recently [32], [33], [34], [73]. Mumtaz et al. [73] first extracted motion features from DTs and then organized them into a tree structure called bag-of-systems tree (BoST) for DT description. On the basis of LDS, Wang and Hu [32] built a chaotic feature vector with four informative components. Wang et al. [34] designed a LDS-based codebook using extreme learning machine. Wei et al. [33] proposed a LDS-based sparse coding framework to construct a dictionary for DT description. Besides the above LDS-based methods, graph theory is also utilized to model DTs [74], [75], [76]. Gonçalves et al. [74] built a complex network based on the Euclidean distance between related pixels for DT description (DT-complex). Ribas and Bruno proposed the diffusion-based DT features (DT-diffusion) [75] and applied deterministic partially self-avoiding walk to extract features from graphs of DTs (DT-DPS) [76].

There are also methods trying to combine LBP with various learning techniques for performance enhancement. Arashloo and Kittler [38] modified LBP-TOP by replacing local pixel differences with filter responses at multiple scales (MBSIF-TOP), in which the filters were learned using independent component analysis. Similarly, Zhao et al. [39] adopted PCA for filter learning and proposed a DT descriptor called multiscale PCA-based feature on three orthogonal planes (MPCAF-TOP). Later, Arashloo et al. [40] learned multiple sets of filters via PCA and organized them into a network for DT description, resulting in a DT descriptor called PCANet-TOP. Instead of extracting features separately from three orthogonal planes, Zhao et al. [41] proposed learning 3D filters and directly extracting features from DT videos.

Some other methods discard the binary encoding scheme and directly employ dictionary learning on local sub-videos. Quan et al. proposed orthogonal tensor dictionary learning [43] and equiangular kernel dictionary learning [44] for DT description. But how they selected items from the learned dictionary remains unknown. Jansson and Lindeberg [77] utilized space-time separable kernels to extract spatio-temporal receptive field (STRF) responses, on which PCA-based dimension reduction was conducted. Zhao et al. [45] applied 3D random projection to extract local features, then a GMM was learned and used as a dictionary for FV encoding. Similarly, Xiong et al. [46] proposed to learn ICA-based filters for local feature extraction, and then learned a GMM.

C. DEEP-LEARNING-BASED

Tran et al. [50] trained a 3D convolutional neural network (C3D) on a large set of videos and used it as a DT feature extractor. Qi et al. [51] and Hong et al. [52] fed selected DT frames into the VGGNet [47] to extract frame-based features, from which statistical features and Fisher vectors were respectively built for DT description. Andrearczyk and Whelan [53] viewed a DT video as three image sequences and fed each image into AlexNet [48] or GoogleNet [49], from which the outputs were aggregated into a feature vector (denoted as DT-CNN). With the strong representational power of models pre-trained on large-scale datasets, these methods generally provide better performance than those in the other two classes. Zrira et al. [78] proposed to train a deep belief network with extracted LBP features. However, this method was only tested on one dataset and had no advantage over many handcrafted features. Additionally, two learning-free networks [79], [80] reported good performance for DT classification. Hadji and Wildes [79] used a set of pre-defined 3D Gaussian third-order derivative filters as convolution layers and Junior et al. [80] used random filters. Methods in this class generally outperform those in the other two classes.

III. THE PROPOSED DT DESCRIPTOR

In this section, we describe the proposed AE-FV in detail. Our AE-FV is a two-stage DT description method. The first stage is to construct the dictionary from training data and the second one is to encode a given DT into a feature vector with the learned dictionary. Fig. 1 illustrates the processing framework of AE-FV.

A. DATA PREPROCESSING

As existing DT datasets are generally insufficient to train a deep model directly, we adopt the concept of BoW and randomly sample a number of sub-videos from each training DT, generating a relatively large number of samples for training a deep model.

Given a training DT of size $X \times Y \times T$ ($X \times Y$ represents the frame size and T is the number of frames), N sub-videos of size $l \times l \times l$ are randomly sampled. We then vectorize these sub-videos and normalize all the pixels to the range [0, 1] by

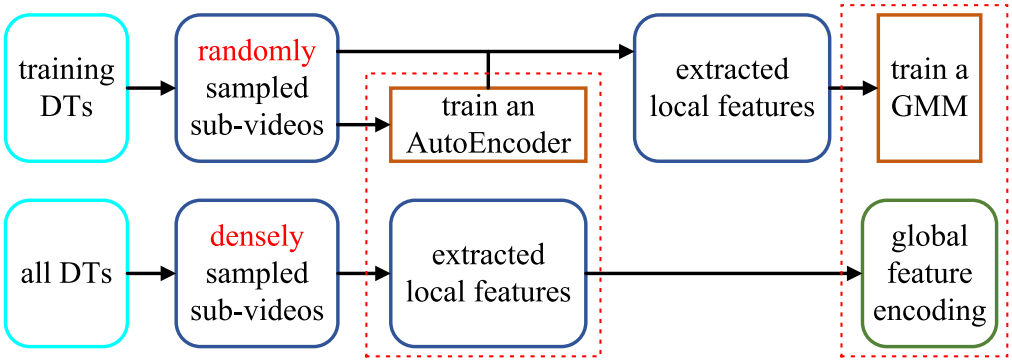

FIGURE 1. Flowchart of the proposed AE-FV. The top flow is the training process and the bottom one is the feature extraction process.

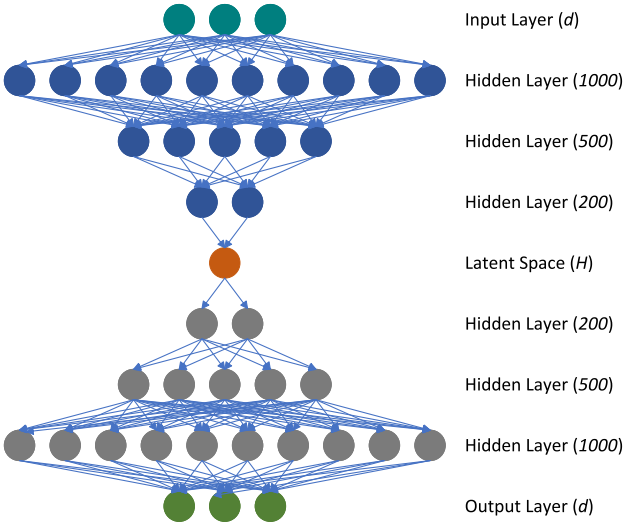

FIGURE 2. Structure of our AutoEncoder network. The layers in blue are the encoder while those in gray are the decoder.

dividing them by 255, resulting in a set of vectors $\{\mathbf{x}_i\}_{i=1}^{N}$ ($\mathbf{x}_i \in \mathbf{R}^D, D = l^3$). By setting the value of N appropriately, a dataset sufficient for the application of deep learning techniques can be obtained.

B. AutoEncoder TRAINING

After obtaining a large set of vectors corresponding to sub-videos, an AutoEncoder network is trained on them. An AutoEncoder is a type of neural network designed to map the given input to its essential features, which allows the original input to be most accurately reconstructed. The essential features are usually low-dimensional and referred to as latent variables. An AutoEncoder consists of three parts: the encoder, the latent space, and the decoder. The encoder compresses a given input to its latent variables in the latent space. The decoder reconstructs the input data from the latent variables. As an unsupervised learning technique, the training of an AutoEncoder does not require labels. The loss function typically measures the mean squared error between the original input and the reconstructed output. Given these properties, AutoEncoder is highly suitable for feature learning.

We design a simple AutoEncoder to learn low-dimensional representations for local sub-videos. Its structure is illustrated in Fig. 2. In the encoding part, a D-dimensional input is first mapped to a 1000-dimensional vector, which is then gradually compressed into an H-dimensional vector in the latent space. In the decoding part, the latent vector is mapped reversely to reconstruct the input. Assuming there are M training DTs, we denote the reconstruction of $\{\mathbf{x}_i\}_{i=1}^{MN}$ as $\{\hat{\mathbf{x}}_i\}_{i=1}^{MN}$, where N represents the number of sampled sub-videos within each training DT. The loss function is defined as

$$loss = \frac{1}{MN}\sum_{i=1}^{MN}\sum_{d=1}^{D}(x_{id} - \hat{x}_{id})^2. \tag{1}$$

The network is trained for 20 epochs using the Adam optimizer and the learning rate is 0.001. Once the training is completed, the encoder is utilized as a local feature extractor. The features extracted from all the training sub-videos are denoted as $\{\mathbf{e}_i\}_{i=1}^{MN}$ ($\mathbf{e}_i \in \mathbf{R}^H$).

C. GMM TRAINING

After obtaining the local features from all the training data, we further need to construct a dictionary that will be used to aggregate all the local features from each sample into a global feature vector, as is done in many BoW methods. The two most used dictionaries are GMM and K-means Clusters. As GMM can capture second-order information, we choose it as our dictionary and conduct FV encoding in the next section.

GMM is a generative model and assumes that samples are generated by multiple Gaussian distributions. Therefore, a GMM can be estimated by applying the expectation-maximization algorithm on the training feature set $\{\mathbf{e}_i\}_{i=1}^{MN}$. The model is comprised of K Gaussian distributions, which are denoted as $\Theta = (\mu_k, \sigma_k, \pi_k : k = 1, \ldots, K)$. μ_k, σ_k and π_k respectively correspond to the mean, covariance matrix and weight of the kth distribution. When a feature vector is fed to the kth Gaussian component, the output is the probability that the sample is generated by this component. So far, the training process of our AE-FV is completely introduced and it is summarized as Algorithm 1.

D. FEATURE ENCODING

In Section III-B and Section III-C, we have obtained two dictionaries, *i.e.*, the encoder of our AutoEncoder network and the GMM. Now we use them to generate a global feature vector for each DT. Here, we employ dense sampling instead of the random sampling scheme used in Section III-A. Given a DT of size $X \times Y \times T$, sub-videos of size $l \times l \times l$ are densely sampled, generating a set of local samples $\{\mathbf{x}_p\}_{p=1}^{P}$, where $P = (X - 2\lfloor \frac{l}{2} \rfloor) \times (Y - 2\lfloor \frac{l}{2} \rfloor) \times (T - 2\lfloor \frac{l}{2} \rfloor)$. Subsequently, each of $\{\mathbf{x}_p\}_{p=1}^{P}$ is fed to the encoder, generating a set of latent vectors $\{\mathbf{e}_p\}_{p=1}^{P}$ ($\mathbf{e}_p \in \mathbf{R}^H$), which are used as the extracted local features of the given DT. Finally, we conduct FV encoding with the learned GMM such that all the P local descriptors are aggregated into a global feature vector to represent the given DT.

Algorithm 1 Training Process of AE-FV

Input: Training set of M DTs $\mathbb{X}_{tra} = \{\mathbf{X}^1, \ldots, \mathbf{X}^M\}$
Parameter: N, l, H, K
Output: An AutoEncoder and a GMM

$\mathbf{S} = \emptyset$
for m = *1 to* M **do**
  Randomly sample N $l \times l \times l$ sub-videos from $\mathbf{X}^m$
  $\{\mathbf{x}_i\}_{i=1}^{N} \leftarrow$ Vectorize and normalize them
  $\mathbf{S} \leftarrow \mathbf{S} \cup \{\mathbf{x}_i\}_{i=1}^{N}$
end
Train an AutoEncoder (the latent size is H) with $\mathbf{S}$
$\mathbf{E} = \emptyset$
for j = *1 to* MN **do**
  Encode $\mathbf{x}_j \in \mathbf{S}$ into $\mathbf{e}_j$ with the encoder
  $\mathbf{E} \leftarrow \mathbf{E} \cup \mathbf{e}_j$
end
Estimate a GMM consisting of K components with E
return the AutoEncoder and GMM

For each local descriptor $\mathbf{e}_p$, its soft assignment weight to the kth Gaussian component is calculated by

$$q_k^p = \frac{\pi_k P_k(\mathbf{e}_p)}{\sum_{t=1}^{K} \pi_t P_t(\mathbf{e}_p)}, \tag{2}$$

where $P_k(\cdot)$ is the probability density function of the kth Gaussian distribution.

With all the local descriptors and their corresponding soft assignment weights, the partial derivatives of the kth Gaussian component of the GMM with respect to its mean and covariance are computed as

$$\mathbf{g}_k^u = \frac{1}{P\sqrt{\pi_k}} \sum_{p=1}^{P} q_k^p \left(\frac{\mathbf{e}_p - \mu_k}{\sigma_k} \right), \tag{3}$$

$$\mathbf{g}_k^v = \frac{1}{P\sqrt{2\pi_k}} \sum_{p=1}^{P} q_k^p \left[\left(\frac{\mathbf{e}_p - \mu_k}{\sigma_k} \right)^2 - 1 \right], \tag{4}$$

where $\mathbf{g}_k^u$ and $\mathbf{g}_k^v$ are respectively the first-order and second-order statistics of all the local descriptors to the kth component. Finally, all the $2K$ derivatives are concatenated to form the global descriptor $\mathbf{f} = [\mathbf{g}_1^{uT}, \mathbf{g}_2^{uT}, \ldots, \mathbf{g}_K^{u\,T}, \mathbf{g}_1^{vT}, \mathbf{g}_2^{vT}, \ldots, \mathbf{g}_K^{v\,T}]^T \in \mathbf{R}^{2KH}$. Additionally, the signed square-rooting and the L2 normalization are applied to $\mathbf{f}$ for feature enhancement [57]. The feature encoding process of our AE-FV is summarized as Algorithm 2.

IV. EXPERIMENTAL EVALUATION

In this section, the proposed AE-FV descriptor is evaluated on three benchmark DT datasets, *i.e.*, UCLA [25], DynTex [81], and DynTex++ [82]. To emphasize the representational power of DT descriptors, we intentionally choose the simple nearest neighbor (NN) classifier. The similarity metric used is Euclidean distance. The classification results of existing DT descriptors are directly quoted from the literature.

Algorithm 2 Feature Extraction Process of AE-FV

Input: Dataset of Q DTs $\mathbb{X}_{all} = \{\mathbf{X}^1, \ldots, \mathbf{X}^Q\}$
Parameter: The trained AutoEncoder and GMM
Output: Global feature vectors for each DT

$\mathbf{F} = [\,]$
for q = *1 to* Q **do**
  Densely sample P $l \times l \times l$ sub-videos from $\mathbf{X}^q$
  $\{\mathbf{x}_p\}_{p=1}^{P} \leftarrow$ Vectorize and normalize them
  for p = *1 to* P **do**
    Encode $\mathbf{x}_p$ into $\mathbf{e}_p$ with the encoder
  end
  for k=*1 to* K **do**
    $\mathbf{g}_k^u, \mathbf{g}_k^v \leftarrow$ Encode $\{\mathbf{e}_p\}_{p=1}^{P}$ with the kth Gaussian component
  end
  $\mathbf{f}_q = [\mathbf{g}_1^{uT}, \mathbf{g}_2^{uT}, \ldots, \mathbf{g}_K^{u\,T}, \mathbf{g}_1^{vT}, \mathbf{g}_2^{vT}, \ldots, \mathbf{g}_K^{v\,T}]^T$
  $\mathbf{f}'_q \leftarrow$ Apply signed square-rooting and L2 normalization to $\mathbf{f}_q$
  $\mathbf{F}[:, q] = \mathbf{f}'_q$
end
return $\mathbf{F}$

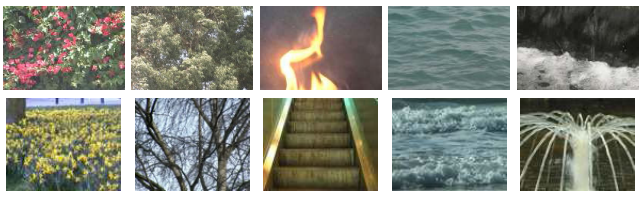

FIGURE 3. Key frames in some DTs from UCLA (top) and DynTex (bottom).

A. DATASETS

In this section, the three benchmark DT datasets and their corresponding evaluation protocols are introduced. As shown in Fig. 3, the key frames in several DTs from the UCLA dataset and DynTex dataset are presented. Before conducting experiments, all the RGB DT videos are converted to gray-scale ones.

1) UCLA

This dataset contains 200 videos belonging to 50 classes, each of which has 4 videos. The original video size is 160 × 110 × 75. We choose a preprocessed version, in which the video size is 48 × 48 × 75 and each DT only preserves key dynamical properties. On this dataset, there are 3 breakdowns with 4 evaluation protocols, which are described below.

a: 50-CLASS BREAKDOWN

This is the original setting and there are two protocols. The first protocol is the leave-one-out (LOO) classification, in which one DT is used as the test sample and the other DTs are used for training. This process is repeated 200 times, and the number of correctly classified DTs is counted. The other protocol is the 4-fold cross validation (4CFV), where all 200 DTs are divided into 4 groups (the division scheme is provided with the dataset). Each group contains one DT from each of the 50 classes. Each time, one group is used as the test set, and the other three groups are used for training. The four classification rates are finally averaged.

b: 9-CLASS BREAKDOWN

In the original 50-class breakdown, it can be observed that some DTs in different classes share the same semantics. Therefore, all the 200 DTs are semantically organized into 9 classes (smoke, fire, boiling, water, flower, sea, waterfall, fountain, and plant). In the experiment, half of the DTs in each of the 9 classes are used as a test set while the other half is for training. This process is repeated 20 times, and the 20 classification rates are averaged to obtain the final classification rate.

c: 8-CLASS BREAKDOWN

As the plant class in the 9-class breakdown has 108 DTs, resulting in an unbalanced data distribution, the experimental result may be biased. Therefore, to remedy this imbalance, the plant class is discarded, leaving the remaining 8 classes, which are referred to as the 8-class breakdown. Then, the half-and-half validation is repeated 20 times, and the classification rates are averaged.

2) DynTex

This dataset contains 679 DT videos of size 352 × 288 × 250. Four subsets of them are chosen for evaluation. The four subsets are introduced as follows.

a: DynTex35

This is the first edition of the DynTex dataset. There are 35 DT videos of size 400 × 300 × 250, each of which belongs to a class. According to [14], each video is segmented along three axes at the specific point ($x = 170$, $y = 130$, and $t = 100$) to generate 8 samples. With an additional segmentation at $t = 100$, there are a total of 10 DT samples per class. Among all the 350 DT samples, those of the same size are put into one group, resulting in 10 groups of DTs in total. In the experiment, each time one group is used as the test set while the other 9 groups are used for training. The 10 classification rates are finally averaged.

A few research studies have adopted the nearest class center (NCC) classifier on this subset. The training feature vectors that belong to the same class are averaged to represent the class center. Then a test feature vector is assigned to the class to which the vector's distance to the center is the minimum.

b: ALPHA

60 videos belonging to 3 classes are selected to form this subset. Each class contains 20 DTs.

c: BETA

162 videos are selected to form this subset. They are organized into 10 classes. The number of DTs in each class ranges from 7 to 20.

d: GAMMA

275 videos are selected to form this subset. They are also organized into 10 classes. The number of DTs in each class ranges from 7 to 38.

For the three subsets of Alpha, Beta, and Gamma, the classification task becomes increasingly challenging as the number of DTs increases. For evaluation, the LOO classification method is adopted.

3) DynTex++

As the videos in the DynTex dataset are generally influenced by multiple factors such as viewpoint, camera motion, background motion, zooming, *etc.* Ghanem and Ahuja [82] selected 345 DTs from DynTex and clipped them into 3600 DT videos, each of which was further filtered and preprocessed to contain only one DT pattern. All the 3600 DTs are grouped into 36 classes, and each class contains 100 DTs. In the experiment, the half-to-half validation scheme is adopted and repeated 10 times. The 10 classification rates obtained are finally averaged.

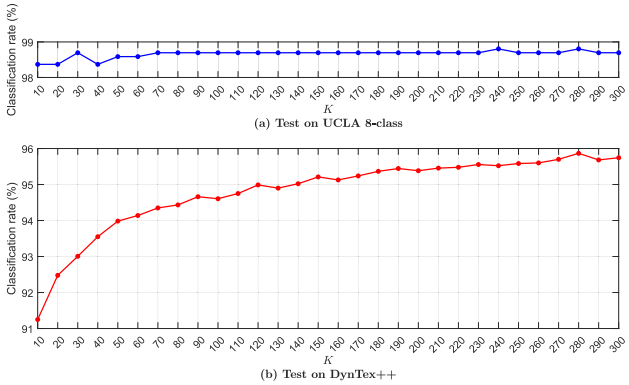

FIGURE 4. Performance of AE-FV with varying K on (a) the UCLA 8-class breakdown and (b) the DynTex++.

B. PARAMETER SETTING

The proposed AE-FV has 4 key parameters, which are the number (N) and size (l) of the sampled sub-videos, the length

TABLE 1. Performance comparison of the proposed AE-FV with other methods on the UCLA dataset.

Method	Classification Rate(%)				Feature
	50-class LOO	50-class 4CFV	9-class	8-class	Length
FD-MAP [60]	97.00	98.50	97.30	99.02	12,760
DDTP [61]	98.50	98.50	93.85	96.09	6768
DFS [20]	-	100^S	97.50^S	99.20^S	500
STLS [23]	-	99.50^S	97.40^S	99.50^S	1080
FoSIG [71]	99.50^S	100^S	98.95^S	98.59^S	1200
V-BIG [72]	99.50^S	99.50^S	97.95^S	97.50^S	2400
$HoGF^{2D}$ [18]	100^S	100^S	99.20^S	98.91^S	7200
$HoGF^{3D}$ [18]	100^S	100^S	99.25^S	99.57^S	9600
$DoDGF^{2D}$ [73]	100^S	100^S	99.25^S	99.13^S	4800
$DoDGF^{3D}$ [73]	100^S	100^S	99.55^S	99.57^S	7200
LTP-lac [62]	-	99.70^S	96.80^S	99.20^S	-
CVLBP [17]	-	93.00	96.90	95.65	512
novel LBP [16]	95.00	95.00	98.35	97.50	1536
CLSP-TOP [66]	99.00	99.00	98.30	97.06	1152
MEWLSP [65]	96.50	96.50	98.55	98.04	4608
WLBPC [64]	-	96.50	97.17	97.61	-
CVLBC [6]	98.50	99.00	99.20	99.02	11250
CSAP-TOP [67]	99.50	99.50	97.50	98.15	13200
HILOP [68]	99.50^S	99.50^S	97.80^S	96.30^S	5664
MEMDP [70]	100^S	100^S	98.90^S	98.70^S	3888
RUBIG [69]	100^S	100^S	99.20^S	99.13^S	21600
DBRF [9]	99.50	100	99.00	99.67	768
DT-Diffusion [77]	-	98.50	97.80	96.22	-
DT-DPS [78]	-	96.00	96.80	96.59	-
MBSIF-TOP [38]	99.50	99.50	98.75	97.80	6144
MPCAF-TOP [39]	99.50	99.50	99.15	98.26	3840
STRF N-jet [79]	-	100	99.00	99.10	-
OTDL [43]	-	98.50	97.50	97.00	2700
EKDL [44]	-	-	98.60^S	-	-
3DRF [45]	-	-	99.24	98.59	1000
B3DF_SMC [41]	99.50	99.50	98.85	98.15	65536
ICFV [46]	99.50	99.00	99.25	99.57	1600
DT-RNNs [55]	-	97.05	98.54	97.74	-
PCANet-TOP [40]	-	99.50	-	-	36864
DT-CNN-AlexNet [51]	-	99.50*	98.05*	98.48*	-
DT-CNN-GoogleNet [51]	-	99.50*	98.35*	99.02*	-
VLBP-DBN [80]	-	-	94.34*	-	-
LBP-TOP-DBN [80]	-	-	98.48*	-	-
AE-FV($K = 10$)	97.50	98.00	99.10	98.37	200
AE-FV($K = 20$)	97.50	98.00	99.15	98.37	400
AE-FV($K = 50$)	99.50	99.50	99.25	98.59	1000
AE-FV($K = 150$)	99.50	99.50	99.30	98.70	3000

Note: "-" means unavailable; "S" means SVM classifier; "$*$" means classification by network.

(H) of latent vectors, and the number (K) of components in the trained GMM. As our purpose is to verify that deep learning techniques could be applied on small datasets with the BoW model, we do not conduct experimental tests to find a parameter setting that gives favorable performance. Instead, we simply set $N = 500, l = 3, H = 10$ and we do not tune the structure of our AutoEncoder network for better performance.

As for the choice of K, using more Gaussian components in a GMM would generally improve performance, while the corresponding feature length also increases. To assess the influence of K on the performance of our AE-FV, we conduct two experimental tests respectively on the UCLA 8-class breakdown and the DynTex++ dataset, where the value of K ranges from 10 to 300 with a step size of 10. Results are shown in Fig. 4. For the less challenging UCLA dataset, increasing the value of K would not bring notable improvement. On the DynTex++ dataset, increasing the value of K would gradually improve performance. When $K >= 150$, the classification rates are above 95% and do not exceed 96% even when $K = 300$. Considering these results, a small K is suitable for simple DT datasets, while a relatively large K is needed for challenging DT datasets. To keep the feature length relatively short, we report results with $K \in \{10, 20, 50, 150\}$ in experiments.

C. COMPARATIVE EVALUATION

In this section, we evaluate the proposed AE-FV descriptor on the three DT datasets and compare its classification rates with those of many existing descriptors. The feature lengths of the various DT descriptors, if available, are also compared.

1) RESULTS ON THE UCLA DATASET

The results on the UCLA dataset are shown in Table 1. For the proposed AE-FV, it can be observed that increasing the number of Gaussian components does improve performance and also increase feature dimensionality. What is surprising is that even AE-FV with $K = 10$ can provide good performance

and outperform many existing descriptors on the 9-class and 8-class breakdowns, verifying that the 10-dimensional local descriptor produced by the encoder of our AutoEncoder is highly discriminative.

Most methods achieve a classification rate of 99.50% on the UCLA 50-class breakdown using the LOO scheme. Our AE-FV with $K = 50$ provides a classification rate of 99.50% and the descriptor has only 1000 dimensions. Some methods (HoGF [18], DoDGF [71], MEMDP [68], RUBIG [67]) achieve a classification rate of 100% by using the SVM classifier, making it unclear how much the descriptors themselves contribute to this performance. They use high-dimensional features (dimensionality ranges from 3888 to 21600) while ours is just 1000. A few other descriptors (FD-MAP [58], DDTP [59], novel LBP [16], MEWLSP [63], CVLBC [6]) achieve classification rates below 99.50% even with high-dimensional features. Considering complexity, dimensionality, and performance, the proposed AE-FV is more favorable than others. Moreover, we further check the only DT sample mis-classified by our AE-FV ($K = 50$). It turns out that a sample from the smoke-b class is incorrectly classified into the water-b-near class, as shown in Fig. 5. It can be observed that the two samples highly resemble each other.

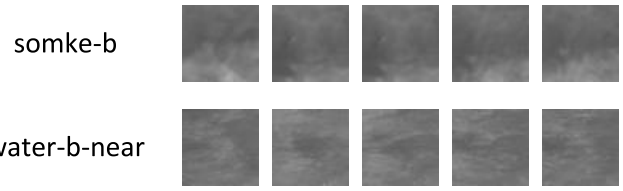

FIGURE 5. The first five frames from the mis-classified sample belonging to the smoke-b class (top) and those from its nearest neighbor in the training set (bottom). These two samples are found with AE-FV ($k = 50$).

As for the 4CFV scheme, the proposed AE-FV with ($K = 10$) provides a classification rate of 98% and outperforms most handcrafted descriptors as well as a few traditional learning-based ones (DT-Diffusion [75], DT-DPS [76], OTDL [43], DT-RNNs [80]) when adopting the NN classifier. When using $K = 50$, AE-FV gives a classification rate of 99.50%, which is only second to DBRF [9] and STRF N-jet [77]. However, DBRF carefully chooses the parameter setting by conducting experimental tests. STRF N-jet uses 16348-dimensional features and contains two stages of multi-scale filtering, followed by PCA. Two DT-CNN [53] methods with pre-trained model (AlexNet [48] and GoogleNet [49]) also give a classification rate of 99.50%. In aspects of complexity and dimensionality, the proposed AE-FV still has some superiority over other methods.

When using the 9-class breakdown, AE-FV with $K = 10$ gives a classification rate of 99.10% and outperforms many other methods, including the deep-learning-based ones, and some evaluated with the SVM classifier. If we increase the value of K, marginal improvement is observed. Among existing methods, the classification rates of CVLBC [6], MPCAF-TOP [39], 3DRF [45], and ICFV [46] are 99.20%, 99.15%, 99.24%, and 99.25%, respectively. If we set $K = 50$, the classification rate of our AE-FV is 99.25% and the descriptor has only 1000 dimensions. Fig. 6 illustrates the confusion matrix of AE-FV ($K = 10$) under this protocol. It indicates that our AE-FV cannot well distinguish smoke samples from water samples. Although some samples from the two classes are visually similar, future effort should be made to tackle such mis-classifications.

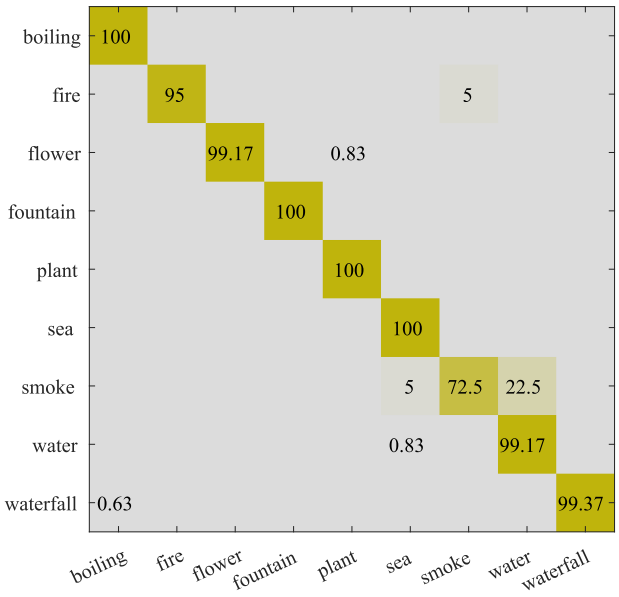

FIGURE 6. Confusion matrix of AE-FV ($K = 10$) on 9-class breakdown.

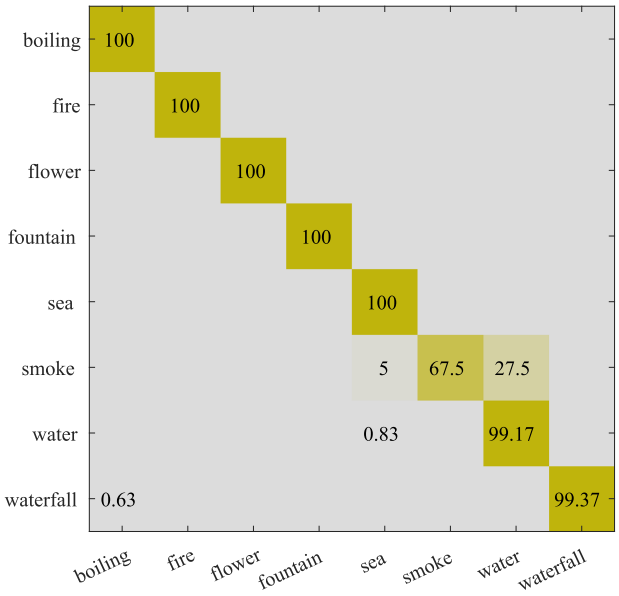

FIGURE 7. Confusion matrix of AE-FV ($K = 10$) on 8-class breakdown.

When evaluated on the 8-class breakdown, the classification rates of most methods decreased compared with those on the 9-class breakdown. It gets more challenging after removing the plant class. AE-FV with $K = 10$ gives a rate of 99.10% and increasing the value of K would only bring marginal improvement. Except for those using

TABLE 2. Performance comparison of the proposed AE-FV with other methods on the DynTex and DynText++ datasets.

Method	Classification Rate(%)					Feature
	DynTex35	Alpha	Beta	Gamma	DynTex++	Length
FD-MAP [60]	95.71	91.67	84.57	80.30	92.87	12,760
DDTP [61]	96.86	91.67	83.33	82.20	89.24	6768
STLS [23]	98.20^S	-	-	-	94.50^S	1080
FoSIG [71]	99.14^S	96.67^S	92.59^S	92.42^S	95.99^S	1200
V-BIG [72]	99.43^S	100^S	95.06^S	94.32^S	96.65^S	2400
$HoGF^{2D}$ [18]	99.71^S	100^S	97.53^S	96.59^S	97.19^S	7200
$HoGF^{3D}$ [18]	99.43^S	98.33^S	98.15^S	97.53^S	97.63^S	9600
$DoDGF^{2D}$ [73]	99.71^S	100^S	97.53^S	96.21^S	97.14^S	4800
$DoDGF^{3D}$ [73]	99.71^S	100^S	98.15^S	96.97^S	97.52^S	7200
LTP-lac [62]	97.90^S	89.60^S	80.90^S	79.90^S	94.80^S	-
novel LBP [16]	98.57	-	-	-	96.28	1536
CLSP-TOP [66]	97.71	95.00	90.12	89.39	93.73	1152
MEWLSP [65]	99.71	-	-	-	98.48	4608
WLBPC [64]	-	-	-	-	95.01	-
CVLBC [6]	98.86	-	-	-	91.31	11250
CSAP-TOP [67]	96.00	91.67	89.51	90.53	-	13200
HILOP [68]	99.71^S	96.67^S	91.36^S	92.05^S	96.21^S	5664
MEMDP [70]	99.71^S	96.67^S	96.91^S	93.94^S	96.03^S	3888
RUBIG [69]	98.86^S	100^S	95.68^S	93.56^S	97.08^S	21600
DBRF [9]	100	95.00	90.12	87.50	95.18	768
DT-Diffusion [77]	-	-	-	-	93.80	-
DT-DPS [78]	-	-	-	-	94.60	-
MBSIF-TOP [38]	98.61	90.00	90.70	91.30	97.17	6144
MPCAF-TOP [39]	-	-	-	-	96.52	3840
STRF N-jet [79]	-	100	93.80	91.20	-	-
EKDL [44]	-	-	-	-	93.40^S	-
3DRF [45]	99.43	98.33	89.51	89.77	94.80	1000
B3DF_SMC [41]	99.71	95.00	90.12	90.91	95.58	65536
ICFV [46]	99.71	100	92.59	90.91	93.02	1600
DT-RNNs [55]	-	-	-	-	96.51	-
PCANet-TOP [40]	-	91.67	90.12	89.39	-	36864
C3D [47]	-	100	99.38	96.97	-	-
st-TCoF [48]	-	98.33	98.15	98.11	-	8192
D3 [49]	-	100^S	100^S	98.11^S	-	-
DT-CNN-AlexNet [51]	-	100*	99.38*	99.62*	98.18*	-
DT-CNN-GoogleNet [51]	-	100*	100*	99.62*	98.58*	-
AE-FV($K = 10$)	98.57	91.67	84.57	81.06	91.25	200
AE-FV($K = 20$)	99.14	96.67	83.33	84.09	92.48	400
AE-FV($K = 50$)	99.43	96.67	89.51	87.12	93.98	1000
AE-FV($K = 150$)	99.43	98.33	93.83	89.77	95.21	3000

Note: "-" means unavailable; "S" means SVM classifier; "$*$" means classification by network.

the SVM classifier, FD-MAP [58], CVLBC [6], STRF N-jet [77], and DT-CNN-GoogleNet [53] give rates higher than 99% and show marginal superiority over our AE-FV, while DBRF [9] and ICFV [46] outperform our AE-FV by about 1%. However, FD-MAP [58] and CVLBC [6] use very high-dimensional features (more than 10000 dimensions); STRF N-jet [77] is very complex, and DT-CNN-GoogleNet [53] uses a model pre-trained on a large dataset; Both DBRF [9] and ICFV [46] need to tune parameters carefully. Compared with them, the proposed AE-FV does not tune parameters and, with low-dimensional features, provides a relatively high rate (slightly lower than 99%). The confusion matrix is shown in Fig. 7. The main problem is still the "smoke-water" mis-classifications.

FIGURE 8. First three frames of the mis-classified DT (top) and those of the corresponding nearest neighbors in the training set (bottom).

2) RESULTS ON THE DynTex DATASET

Results on this dataset are presented in Table 2. The DynTex35 subset is not challenging, as most methods achieve classification rates near or higher than 99%. While the proposed AE-FV with $K = 50$ gives a classification rate of 99.43%, MEWLSP [63], DBRF [9], B3DF_SMC [41], and ICFV [46] show a marginal improvement over AE-FV ($K = 50$), but their superiority is negligible. These methods either use high-dimensional features or require to careful parameter tuning for good performance. A few other descriptors that use

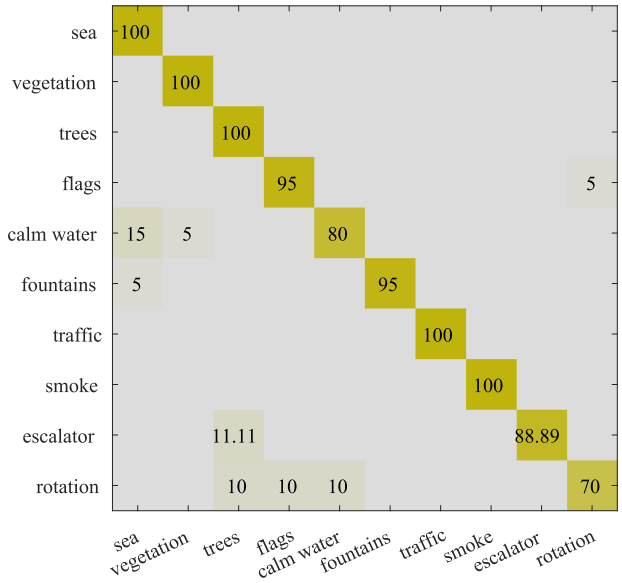

FIGURE 9. Confusion matrix of AE-FV ($K = 150$) on the Beta subset.

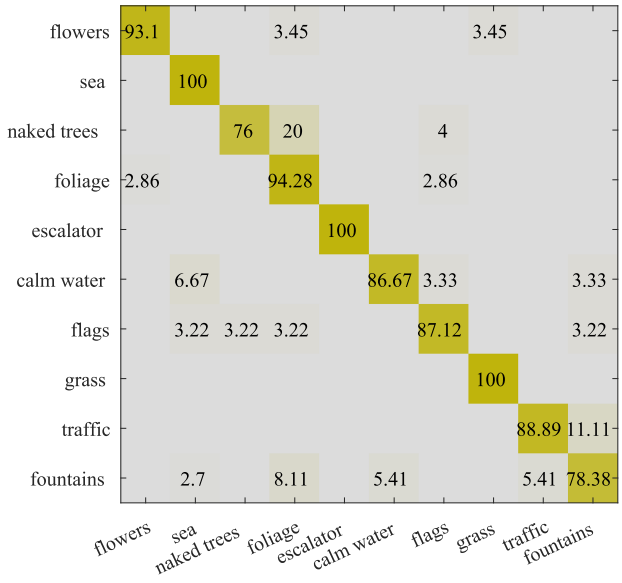

FIGURE 10. Confusion matrix of AE-FV ($K = 150$) on the Gamma subset.

the SVM classifier, also exhibit negligible superiority over our method.

The Alpha subset is a little challenging. The classification rates of many handcrafted descriptors are lower than 97%, even though some of them are evaluated with the SVM classifier. The proposed AE-FV ($K = 150$) gives a rate of 98.33%, which can only be considered acceptable. On the other hand, four deep-learning-based methods achieve excellent performance. The VGGNet-based st-TCoF [51] gives a rate of 98.33%, and is on par with our AE-FV ($K = 150$). The only DT mis-classified by AE-FV ($K = 150$) is shown in Fig. 8. The top DT is from the grass class while the bottom one is from the tree class. They are visually different. However, if we focus on the local areas, the grasses and small tree branches resemble each other and their motion patterns are also similar.

On the Beta subset, the proposed AE-FV ($K = 150$) achieves a rate of 93.88% and outperforms all the traditional methods using the NN classifier, as well as several evaluated with the SVM classifier. Deep-learning-based methods significantly outperform others. It can also be noticed that the classification rates on this subset are generally lower than those on the Alpha subset. This is because the Beta subset has 162 DTs from ten classes, in which both the visual and dynamical patterns are very complex. Fig. 9 shows the confusion matrix on this subset. Mis-classifications are mainly conducted for the three classes of calm water, escalator, and rotation.

The Gamma subset is constructed by introducing extra DTs into the Beta subset and thus is even more challenging. Even the deep-learning-based methods failed to provide a classification rate of 100%. The proposed AE-FV ($K = 150$) and all the traditional methods evaluated with the NN classifier achieve classification rates around 90%. However, HoGF^{2D} [18], HoGF^{3D} [18], DoDGF^{2D} [71], and DoDGF^{3D} [71] provide classification rates higher than 96%. We think the combination of high-dimensional features and the utilization of the SVM classifier contributes to such good performance. Except for C3D, the other deep-learning-based methods undoubtedly outperform all the traditional methods, demonstrating the strong representational power of deep models trained on large-scale datasets. Fig. 10 shows the confusion matrix on this subset. Only the DTs in the three classes of sea, escalator, and grass are 100% correctly classified. For other classes, about 10% of the DTs are mis-classified on average.

3) RESULTS ON THE DynTex++ DATASET

Results on this dataset are also shown in Table 2. The proposed AE-FV ($K = 150$) achieves a classification rate of 95.21%. Among traditional methods using the NN classifier, novel LBP [16], MEWLSP [63], MBSIF-TOP [38], MPCAF-TOP [39], B3DF_SMC [41], and DT-RNNs [80] outperform our method by 1.07%, 3.27%, 1.96%, 1.31%, 0.37%, and 1.3%, respectively. Except for MEWLSP [63] and MBSIF-TOP [38], the superiority of other three methods is marginal. MEWLSP [63] is learning-free and does not perform well on the UCLA dataset. MBSIF-TOP [38] uses high-dimensional features (6144 dimensions). For those methods that are evaluated with the SVM classifier and outperform our AE-FV, their advantages are generally less than 2%. And whether the descriptors contribute more than the SVM classifier remains unclear. Only DT-CNN-AlexNet [53] and DT-CNN-GoogleNet [53] are tested on this dataset. The results are respectively 98.18% and 98.58%, which are 3% higher than that of our AE-FV. Overall, AE-FV ($K = 150$) performs relatively well on this dataset. The class-specific classification rates are illustrated in Fig. 11. The classification rate can be as low as 73.2%, indicating that the DTs in some classes are difficult to distinguish from other DTs.

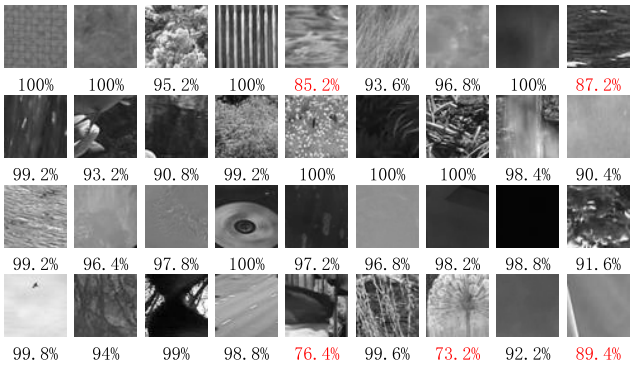

FIGURE 11. Class-specific classification rates by AE-FV ($K = 150$). Results lower than 90% are in red.

TABLE 3. Ablation study for AE-FV on UCLA 9-class breakdown.

Method	Classification Rate(%)			
	$K = 10$	$K = 20$	$K = 50$	$K = 150$
PCA+BoW	92.45	94.35	95.50	96.35
RP+BoW	92.00	95.25	96.15	96.15
AE+BoW	*95.45*	*96.85*	*97.70*	*97.90*
PCA+FV	99.20	99.20	99.20	99.25
RP+FV	**99.40**	**99.40**	99.10	99.15
AE+FV	99.10	99.15	**99.25**	**99.30**

Note: "AE" means AutoEncoder.

D. ABLATION STUDY

In order to verify the effectiveness of the two individual components (AutoEncoder for local feature extraction and GMM for local feature aggregation) of AE-FV, we design five variants for ablation analysis. As AE-FV first extracts local features and then aggregates those from the same DT into a global feature vector, we consider two other typical local feature extraction methods (PCA and random projection) and another feature encoding method (the classic BoW model). PCA is applied to the training patches and the top 10 eigenvectors are used as filters to extract features from patches. Random projection (RP) uses 10 randomly generated filters to extract local features. The sizes of PCA filters and random filters are equal to the sample patch size. And both the PCA-based local features and the random local features have 10 dimensions, which are the same as our AutoEncoder-based local features. The classic BoW model first applies K-means clustering to the training local descriptors, in which the number of cluster centers can be 10, 20, 50, and 150. Then, each of the local descriptors extracted from one DT is assigned to a cluster center according to its distance to each center. Finally, a histogram is built as the corresponding global feature vector.

We have three types of local features (AutoEncoder-based, PCA-based and RP-based) and two feature aggregation methods (FV and BoW). To make a fair comparison, all the three types of local features have 10 dimensions and the number of Gaussian components in GMM is set equal to the number of clusters in the classic BoW model. Specifically, we combine each of the three features with each of the two feature aggregation models, and compare the performance of the six combinations on the UCLA 9-class breakdown dataset in aspect of classification rate. The results are shown in Table 3. When using BoW, the superiority of AutoEncoder-based local features is fully demonstrated. AE+BoW outperforms PCA+BoW and RP+BoW by about 3%. AE+BoW's advantage over PCA+BoW and RP+BoW is above 1.5% for $K \in \{20, 50, 150\}$. When using the powerful FV encoding, the classification rates of all three feature types are above 99%. For $K \in \{10, 20\}$, PCA+FV and RP+FV slightly outperform AE+FV by about 0.1% and 0.3%, respectively. For $K \in \{50, 150\}$, AE+FV slightly outperforms PCA+FV and RP+FV. One strange thing is that the performance of RP+FV drops when increasing K from 20 to 50, implying that random features may not be very stable. Overall, local features extracted by a simple AutoEncoder network are more discriminative than PCA-based and RP-bade features when using BoW, and FV encoding can significantly improve performance for each of the three types of local features. Therefore, our AE-FV that encodes AutoEncoder-based local features via FV encoding, is indeed effective for the description of DTs.

TABLE 4. Runtime (in seconds) comparison of AE-FV, CVLBC, MPCAF-TOP and PCANet-TOP.

Method	Time
CVLBC [6]	1.38
MPCAF-TOP [39]	3.14
PCANet-TOP [40]	37.71
AE-FV ($K = 150$)	**0.99**

E. COMPUTATIONAL EFFICIENCY

It has been claimed that computational efficiency is as important as classification accuracy for a pattern recognition task [37]. Therefore, we need to measure the time cost (wall-clock time) of our AE-FV and compare it with those of three existing methods, *i.e.*, the learning-free CVLBC, the learning-based MPCAF-TOP and PCANet-TOP. A feature vector is extracted from a sample in the UCLA dataset by each of the four methods. And the elapsed time between feeding the sample into the program and obtaining the feature vector is used as the computation time. To achieve a stable time cost, we run each test 10 times and the computation times are averaged as the final time cost. To make a fair comparison, we use $K = 150$ for our AE-FV while the other three methods are tested with the parameter settings reported in the original literature. All the methods are implemented with MATLAB, and are executed on an idle server with dual E5-2680v4 CPUs and 192GB memory (no GPU is involved).

The results are shown in Table 4. All of CVLBC, MPCAF-TOP and PCANet-TOP inherit LBP's drawback that the sign of each feature code needs to be extracted. As there are plenty of feature codes, either counting or encoding the signs is time-consuming. Both MPCAF-TOP and PCANet-TOP extract features at multiple scales, which not only increases dimensionality, but also requires more processing time. On the other hand, our AE-FV

using 150 Gaussian components needs only 0.99 seconds to process a DT, which is more efficient than other three methods. Considering the results of the four methods in Table 1 and Table 2, only CVLBC and MPCAF-TOP respectively outperform our AE-FV ($K = 150$) on the UCLA 8-class dataset and the DynTex++ dataset by 0.32% and 1.31%, and these improvements are marginal. For other cases, our AE-FV ($K = 150$) is either on par with or better than the other methods. Therefore, we firmly believe that our AE-FV achieves a good balance between performance and complexity.

F. SUMMARY AND DISCUSSION

With an arbitrary parameter setting, the proposed AE-FV achieves good performance on the UCLA dataset and also works well on the DynTex++ dataset. For evaluation on the DynTex dataset, the performance is relatively good on the DynTex35 and Alpha subsets. However, on the Beta and Gamma subsets, the results of our AE-FV can only be considered acceptable and are far from satisfactory. We think the reason behind this situation are two-fold. One reason is that the Beta and Gamma subsets are naturally very challenging. The DTs in these two subsets may contain multiple motion patterns, and some DTs from different classes can appear visually similar in certain viewpoints or at a specific scale. The other reason is that the parameter setting may be inappropriate for the Beta and Gamma subsets. Firstly, DTs in Beta and Gamma subsets are large in size. But we still sample $3 \times 3 \times 3$ sub-videos from them, which may fail to capture the key dynamical patterns. Secondly, both intra-class and inter-class variations in the two subsets are large. But we only sample 500 sub-videos per DT for training and the local descriptor has only 10 dimensions, which may be insufficient to build a compact global descriptor.

V. CONCLUSION

In this paper, we adopt the bag-of-words model and directly apply deep learning techniques to small-scale DT datasets for classification. The proposed framework consists of a training process and a feature extraction process. In the training process, a small number of sub-videos are randomly sampled from each training DT, which are further normalized and vectorized. Then a simple AutoEncoder is trained with these data. Finally, a Gaussian mixture model is fitted with the encoded training data. The outputs of the training process are the AutoEncoder network and the Gaussian mixture model, which will be used as two dictionaries. In the feature extraction process, sub-videos are densely sampled and processed as in the training process. These sub-videos are then fed into the encoder of the AutoEncoder network, thereby generating a set of local descriptors. These local descriptors from the same DT are aggregated into a global feature vector via FV encoding with the Gaussian mixture model. Even with an arbitrary parameter setting, the proposed method still achieves good performance using the nearest neighbor classifier. When we use 50 Gaussian components, a good balance between performance and dimensionality is obtained. With feature vectors having 1000 dimensions, the classification rates on the UCLA dataset are around 99%. On the challenging DynTex dataset, we increase the number of Gaussian components to 150 (feature length increases to 3000) and our AE-FV outperforms most existing methods that are also evaluated with the nearest neighbor classifier. These experimental results demonstrate the effectiveness and efficiency of our AE-FV, verifying the possibility of directly applying deep learning techniques to small-scale datasets. As a few methods using high-dimensional features slightly outperform the proposed method, additional features such as image gradients could be utilized to enhance performance.

REFERENCES

[1] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Int. J. Comput. Vis.*, vol. 51, no. 2, pp. 91–109, 2003.

[2] A. B. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2005, pp. 771–776.

[3] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Çetin, "Computer vision based method for real-time fire and flame detection," *Pattern Recognit. Lett.*, vol. 27, no. 1, pp. 49–58, Jan. 2006.

[4] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–7.

[5] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 915–928, Jun. 2007.

[6] X. Zhao, Y. Lin, and J. Heikkilä, "Dynamic texture recognition using volume local binary count patterns with an application to 2D face spoofing detection," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 552–566, Mar. 2018.

[7] X. Li, X. Hong, A. Moilanen, X. Huang, T. Pfister, G. Zhao, and M. Pietikäinen, "Towards reading hidden emotions: A comparative study of spontaneous micro-expression spotting and recognition methods," *IEEE Trans. Affect. Comput.*, vol. 9, no. 4, pp. 563–577, Oct. 2018.

[8] T. T. Nguyen and T. P. Nguyen, "A comprehensive taxonomy of dynamic texture representation," *ACM Comput. Surv.*, vol. 55, no. 1, pp. 1–39, Jan. 2023.

[9] X. Zhao, F. Xu, Y. Ma, Z. Liu, M. Deng, U. S. Khan, and Z. Xiong, "Dynamic texture classification using directional binarized random features," *IEEE Access*, vol. 11, pp. 55895–55910, 2023.

[10] C.-H. Peh and L.-F. Cheong, "Synergizing spatial and temporal texture," *IEEE Trans. Image Process.*, vol. 11, no. 10, pp. 1179–1191, Oct. 2002.

[11] R. Péteri and D. Chetverikov, "Qualitative characterization of dynamic textures for video retrieval," in *Proc. Int. Conf. Comput. Vis. Graph. (ICCVG)*. Dordrecht, The Netherlands: Springer, 2006, pp. 33–38.

[12] Z. Lu, W. Xie, J. Pei, and J. Huang, "Dynamic texture recognition by spatio-temporal multiresolution histograms," in *Proc. 7th IEEE Workshops Appl. Comput. Vis. (WACV/MOTION)*, vol. 2, Jan. 2005, pp. 241–246.

[13] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[14] G. Zhao and M. Pietikinen, "Dynamic texture recognition using volume local binary patterns," in *Proc. Int. Workshop Dyn. Vis.* Berlin, Germany: Springer, 2005, pp. 165–177.

[15] E. Rahtu, J. Heikkilä, V. Ojansivu, and T. Ahonen, "Local phase quantization for blur-insensitive image analysis," *Image Vis. Comput.*, vol. 30, no. 8, pp. 501–512, Aug. 2012.

[16] D. Tiwari and V. Tyagi, "A novel scheme based on local binary pattern for dynamic texture recognition," *Comput. Vis. Image Understand.*, vol. 150, pp. 58–65, Sep. 2016.

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

[18] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Prominent local representation for dynamic textures based on high-order Gaussian-gradients," *IEEE Trans. Multimedia*, vol. 23, pp. 1367–1382, 2021.

[19] Y. Xu, Y. Quan, H. Ling, and H. Ji, "Dynamic texture classification using dynamic fractal analysis," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1219–1226.
[20] Y. Xu, Y. Quan, Z. Zhang, H. Ling, and H. Ji, "Classifying dynamic textures via spatiotemporal fractal analysis," *Pattern Recognit.*, vol. 48, no. 10, pp. 3239–3248, Oct. 2015.
[21] Y. Xu, S. Huang, H. Ji, and C. Fermüller, "Scale-space texture description on SIFT-like textons," *Comput. Vis. Image Understand.*, vol. 116, no. 9, pp. 999–1013, Sep. 2012.
[22] H. Ji, X. Yang, H. Ling, and Y. Xu, "Wavelet domain multifractal analysis for static and dynamic texture classification," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 286–299, Jan. 2013.
[23] Y. Quan, Y. Sun, and Y. Xu, "Spatiotemporal lacunarity spectrum for dynamic texture classification," *Comput. Vis. Image Understand.*, vol. 165, pp. 85–96, Dec. 2017.
[24] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "PCANet: A simple deep learning baseline for image classification?" *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5017–5032, Dec. 2015.
[25] P. Saisan, G. Doretto, Y. Nian Wu, and S. Soatto, "Dynamic texture recognition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. CVPR*, vol. 2, Dec. 2001, p. 58.
[26] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 846–851.
[27] A. B. Chan and N. Vasconcelos, "Classifying video with kernel dynamic textures," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–6.
[28] A. Mumtaz, E. Coviello, G. R. G. Lanckriet, and A. B. Chan, "Clustering dynamic textures with the hierarchical EM algorithm for modeling video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1606–1621, Jul. 2013.
[29] A. Ravichandran, R. Chaudhry, and R. Vidal, "View-invariant dynamic texture recognition using a bag of dynamical systems," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1651–1657.
[30] A. Ravichandran, R. Chaudhry, and R. Vidal, "Categorizing dynamic textures using a bag of dynamical systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 342–353, Feb. 2013.
[31] B. Ghanem and N. Ahuja, "Sparse coding of linear dynamical systems with an application to dynamic texture recognition," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 987–990.
[32] Y. Wang and S. Hu, "Chaotic features for dynamic textures recognition," *Soft Comput.*, vol. 20, no. 5, pp. 1977–1989, May 2016.
[33] X. Wei, Y. Li, H. Shen, F. Chen, M. Kleinsteuber, and Z. Wang, "Dynamical textures modeling via joint video dictionary learning," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2929–2943, Jun. 2017.
[34] L. Wang, H. Liu, and F. Sun, "Dynamic texture video classification using extreme learning machine," *Neurocomputing*, vol. 174, pp. 278–285, Jan. 2016.
[35] J. Ren, X. Jiang, and J. Yuan, "Dynamic texture recognition using enhanced LBP features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Nov. 2013, pp. 2400–2404.
[36] J. Ren, X. Jiang, J. Yuan, and G. Wang, "Optimizing LBP structure for visual recognition using binary quadratic programming," *IEEE Signal Process. Lett.*, vol. 21, no. 11, pp. 1346–1350, Nov. 2014.
[37] S. Hong, J. Ryu, and H. S. Yang, "Not all frames are equal: Aggregating salient features for dynamic texture classification," *Multidimensional Syst. Signal Process.*, vol. 29, no. 1, pp. 279–298, Jan. 2018.
[38] S. R. Arashloo and J. Kittler, "Dynamic texture recognition using multiscale binarized statistical image features," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2099–2109, Dec. 2014.
[39] X. Zhao, Y. Lin, and J. Heikkilä, "Dynamic texture recognition using multiscale PCA-learned filters," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4152–4156.
[40] S. R. Arashloo, M. Chehel Amirani, and A. Noroozi, "Dynamic texture representation using a deep multi-scale convolutional network," *J. Vis. Commun. Image Represent.*, vol. 43, pp. 89–97, Feb. 2017.
[41] X. Zhao, Y. Lin, L. Liu, J. Heikkilä, and W. Zheng, "Dynamic texture classification using unsupervised 3D filter learning and local binary encoding," *IEEE Trans. Multimedia*, vol. 21, no. 7, pp. 1694–1708, Jul. 2019.
[42] M. Harandi, C. Sanderson, C. Shen, and B. Lovell, "Dictionary learning and sparse coding on Grassmann manifolds: An extrinsic solution," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3120–3127.
[43] Y. Quan, Y. Huang, and H. Ji, "Dynamic texture recognition via orthogonal tensor dictionary learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 73–81.
[44] Y. Quan, C. Bao, and H. Ji, "Equiangular kernel dictionary learning with applications to dynamic texture analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 308–316.
[45] X. Zhao, Y. Lin, and L. Liu, "Dynamic texture recognition using 3D random features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2102–2106.
[46] Z. Xiong, F. Mo, X. Zhao, F. Xu, X. Zhang, and Y. Wu, "Dynamic texture classification based on 3D ICA-learned filters and Fisher vector encoding in big data environment," *J. Signal Process. Syst.*, vol. 94, no. 11, pp. 1129–1143, Nov. 2022.
[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 26th Annu. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[50] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.
[51] 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, Jan. 2016.
[52] S. Hong, J. Ryu, W. Im, and H. S. Yang, "D3: Recognizing dynamic scenes with deep dual descriptor based on key frames and key segments," *Neurocomputing*, vol. 273, pp. 611–621, Jan. 2018.
[53] V. Andrearczyk and P. F. Whelan, "Convolutional neural network on three orthogonal planes for dynamic texture classification," *Pattern Recognit.*, vol. 76, pp. 36–49, Apr. 2018.
[54] Z. Wang, Q. Luo, H. Chen, J. Zhao, L. Yao, J. Zhang, and F. Chu, "A high-accuracy intelligent fault diagnosis method for aero-engine bearings with limited samples," *Comput. Ind.*, vols. 159–160, Aug. 2024, Art. no. 104099.
[55] Z. Dong, D. Zhao, and L. Cui, "An intelligent bearing fault diagnosis framework: One-dimensional improved self-attention-enhanced CNN and empirical wavelet transform," *Nonlinear Dyn.*, vol. 112, no. 8, pp. 6439–6459, Apr. 2024.
[56] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2018, pp. 415–419.
[57] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Berlin, Germany: Springer, Sep. 2010, pp. 143–156.
[58] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and X. S. Nguyen, "Directional beams of dense trajectories for dynamic texture recognition," in *Proc. 19th Int. Conf. Adv. Concepts Intell. Vis. Syst. (ACIVS)*. Cham, Switzerland: Springer, Sep. 2018, pp. 74–86.
[59] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Directional dense-trajectory-based patterns for dynamic texture recognition," *IET Comput. Vis.*, vol. 14, no. 4, pp. 162–176, Jun. 2020.
[60] Y. Sun, Y. Xu, and Y. Quan, "Characterizing dynamic textures with space-time lacunarity analysis," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jun. 2015, pp. 1–6.
[61] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, Jun. 2010.
[62] D. Tiwari and V. Tyagi, "Improved Weber's law based local binary pattern for dynamic texture recognition," *Multimedia Tools Appl.*, vol. 76, no. 5, pp. 6623–6640, Mar. 2017.
[63] D. Tiwari and V. Tyagi, "Dynamic texture recognition using multiresolution edge-weighted local structure pattern," *Comput. Electr. Eng.*, vol. 62, pp. 485–498, Aug. 2017.
[64] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Completed local structure patterns on three orthogonal planes for dynamic texture recognition," in *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2017, pp. 1–6.

[65] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Completed statistical adaptive patterns on three orthogonal planes for recognition of dynamic textures and scenes," *J. Electron. Imag.*, vol. 27, no. 5, p. 1, Oct. 2018.
[66] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Dynamic texture representation based on hierarchical local patterns," in *Advanced Concepts for Intelligent Vision Systems*. Cham, Switzerland: Springer, 2020, pp. 277–289.
[67] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Rubik Gaussian-based patterns for dynamic texture classification," *Pattern Recognit. Lett.*, vol. 135, pp. 180–187, Jul. 2020.
[68] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and X. S. Nguyen, "Momental directional patterns for dynamic texture recognition," *Comput. Vis. Image Understand.*, vol. 194, May 2020, Art. no. 102882.
[69] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "Smooth-invariant Gaussian features for dynamic texture recognition," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 4400–4404.
[70] T. T. Nguyen, T. P. Nguyen, F. Bouchara, and N.-S. Vu, "Volumes of blurred-invariant Gaussians for dynamic texture classification," in *Computer Analysis of Images and Patterns*. Cham, Switzerland: Springer, 2019, pp. 155–167.
[71] T. T. Nguyen, T. P. Nguyen, and F. Bouchara, "A novel filtering kernel based on difference of derivative Gaussians with applications to dynamic texture representation," *Signal Process., Image Commun.*, vol. 98, Oct. 2021, Art. no. 116394.
[72] S. Dubois, R. Péteri, and M. Ménard, "Characterization and recognition of dynamic textures based on the 2D+T curvelet transform," *Signal, Image Video Process.*, vol. 9, no. 4, pp. 819–830, May 2015.
[73] A. Mumtaz, E. Coviello, G. R. G. Lanckriet, and A. B. Chan, "A scalable and accurate descriptor for dynamic textures using bag of system trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 697–712, Apr. 2015.
[74] W. N. Gonçalves, B. B. Machado, and O. M. Bruno, "A complex network approach for dynamic texture recognition," *Neurocomputing*, vol. 153, pp. 211–220, Apr. 2015.
[75] L. C. Ribas, W. N. Gonçalves, and O. M. Bruno, "Dynamic texture analysis with diffusion in networks," *Digit. Signal Process.*, vol. 92, pp. 109–126, Sep. 2019.
[76] L. C. Ribas and O. M. Bruno, "Dynamic texture classification using deterministic partially self-avoiding walks on networks," in *Proc. 20th Int. Conf. Image Anal. Process. (ICIAP)*. Cham, Switzerland: Springer, Sep. 2019, pp. 82–93.
[77] Y. Jansson and T. Lindeberg, "Dynamic texture recognition using time-causal and time-recursive spatio-temporal receptive fields," *J. Math. Imag. Vis.*, vol. 60, no. 9, pp. 1369–1398, Nov. 2018.
[78] N. Zrira, K. Mouhcine, I. Benmiloud, and E. H. Bouyakhf, "Dynamic texture-based scene classification using deep belief networks," in *Proc. Int. Conf. Learn. Optim. Algorithms, Theory Appl.*, May 2018, pp. 1–6.
[79] I. Hadji and R. P. Wildes, "A spatiotemporal oriented energy network for dynamic texture recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3085–3093.
[80] J. J. D. M. Sá Junior, L. C. Ribas, and O. M. Bruno, "Randomized neural network based signature for dynamic texture classification," *Expert Syst. Appl.*, vol. 135, pp. 194–200, Nov. 2019.
[81] R. Péteri, S. Fazekas, and M. J. Huiskes, "DynTex: A comprehensive database of dynamic textures," *Pattern Recognit. Lett.*, vol. 31, no. 12, pp. 1627–1632, Sep. 2010.
[82] B. Ghanem and N. Ahuja, "Maximum margin distance learning for dynamic texture recognition," in *Proc. 11th Eur. Conf. Comput. Vis. (ECCV)*. Berlin, Germany: Springer, 2010, pp. 223–236.

ZHE LI received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2018. He is currently an Associate Professor with the School of Computer and Information Science, Hubei Engineering University, Xiaogan, China. His recent research interests include data mining, and applications to crime prediction and urban computing.

XIAOCHAO ZHAO received the B.S. and Ph.D. degrees in software engineering from Hunan University, Changsha, China, in 2012 and 2018, respectively. Since 2019, he has been a Lecturer with Hubei Engineering University. His research interests include image processing and computer vision.

TIANFAN ZHANG received the M.S. degree from Wuhan University, in 2012, and the Ph.D. degree from Northwestern Polytechnical University, in 2021. He is currently an Associate Professor with the School of Mathematics and Statistics, Hubei Engineering University. His main research interests include big data and data governance, digital image processing, and artificial intelligence systems.

XIAO JING received the M.S. degree in electrical engineering from Wuhan University, Wuhan, China, in 2018. He is currently pursuing the Ph.D. degree in cybersecurity with Northwestern Polytechnical University, Xi'an, China. His recent research interests include data mining, and applications to text classification and social network analysis.

WEI SHI received the B.S. degree in computer science and technology and the M.S. and Ph.D. degrees in earth exploration and information technology from Yangtze University, in 2004, 2009, and 2021, respectively. Since 2022, he has been a Lecturer with Hubei Engineering University. His research interests include stochastic signal processing and machine learning.

QIAN CHEN received the master's degree in electronics and communication engineering from Wuhan University, Wuhan, Hubei, China, in 2020. She has been a Teacher with Hubei Engineering University, since 2022. Her research interest includes natural language processing.