A Temporal Order Modeling Approach to Human Action Recognition from Multimodal Sensor Data

Jun Ye, University of Central Florida
Hao Hu, University of Central Florida
Guo-Jun Qi, University of Central Florida
Kien A. Hua, University of Central Florida

From wearable devices to depth cameras, researchers have exploited various multimodal data to recognize human actions for applications, such as video gaming, education, healthcare, etc. While there have been many successful techniques in the literature, most of the current approaches have focused on statistical or local spatio-temporal features and do not explicitly explore the temporal dynamics of the sensor data. However, human action data contain rich temporal structure information that can characterize the unique underlying patterns of different action categories. From this perspective, we propose a novel temporal order modeling approach to human action recognition. Specifically, we explore subspace projections to extract the latent temporal patterns from different human action sequences. The temporal order between these patterns are compared and the index of the pattern that appears first is used to encode the entire sequence. This process is repeated multiple times and produces a compact feature vector representing the temporal dynamics of the sequence. Human action recognition can then be efficiently solved by the nearest neighbor search based on the Hamming distance between these compact feature vectors. We further introduce a sequential optimization algorithm to learn the optimized projections that preserve the pairwise label similarity of the action sequences. Experimental results on two public human action datasets demonstrate the superior performance of the proposed technique in both accuracy and efficiency.

CCS Concepts:•**Computing methodologies** → **Activity recognition and understanding;**

Additional Key Words and Phrases: Human action recognition, temporal order modeling, optimization, multimodal sensor data

ACM Reference Format:
Jun Ye, Hao Hu, Guo-Jun Qi, and Kien A. Hua, 2016. A Temporal Order Modeling Approach to Human Action Recognition from Multimodal Sensor Data. *ACM Trans. Multimedia Comput. Commun. Appl.* 1, 1, Article 1 (January 2016), 23 pages.
DOI: 0000001.0000001

1. INTRODUCTION

As sensor technology continues to develop, commodity sensing devices such as smart wristbands and depth cameras become more and more popular in one's daily life. These sensors are able to provide continuous multimodal data streams that can be utilized in a wide range of applications including healthcare, surveillance, video gaming and education. For example, miniature inertial sensors like the accelerometer and the gyroscope are integrated into smartphones and wristbands to detect various human daily activities such as walking and climbing steps [Lara and Labrador 2013]. Depth sensors such as Microsoft Kinect are exploited to recognize bodu gestures as a natural user in-

Author's addresses: J. Ye, H. Hu, G. Qi, and K. A. Hua, Computer Science Department, University of Central Florida, 4000 Central Florida Blvd, Orlando, FL 32816. Corresponding authors: Jun Ye (jye@cs.ucf.edu) and Guo-Jun Qi (guojun.qi@ucf.edu).

DOI: 0000001.0000001

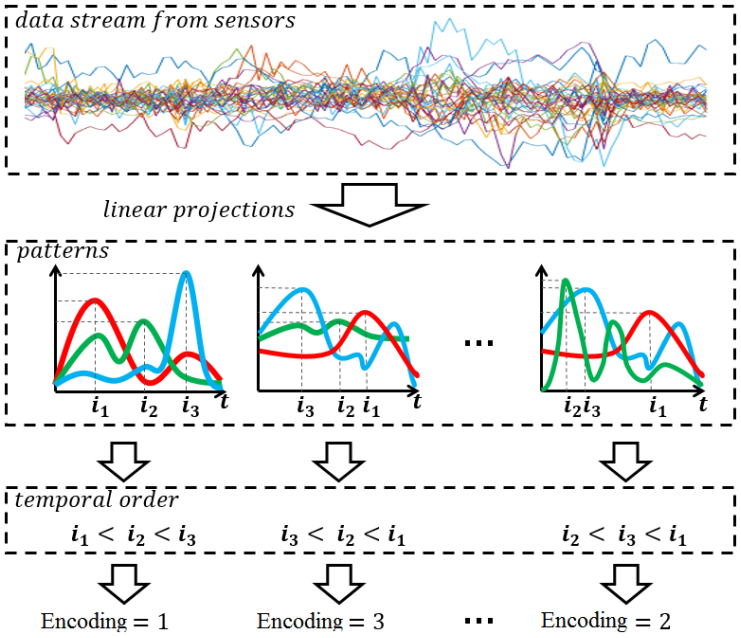


Fig. 1: A Diagram of the temporal order modeling approach.

terface of human-computer interactions [Rahman et al. 2010]. Multisensor multimedia monitoring systems have been studied and assessed for human-centric event detection [Hossain et al. 2011]. One of the key technologies behind these applications is human action recognition, which directly determines the quality of the user experience therein.

Although there has been intensive research on human action recognition in recent decades, it still remains a challenging problem due to the complexity of the spatio-temporal patterns inside the human action data streams as well as the highly noisy nature of the sensor data. Most of the existing approaches have focused on the video data and employ the local spatio-temporal features [Laptev 2005; Yang et al. 2012; Li et al. 2010], and cannot handle the complex temporal dynamics inside human action data. As an example, some actions may consist of visually similar postures but in different temporal orders, such as “put on a hat” and “take off a hat”. The above methods may be confused by these human actions because no global temporal information is explored. Some other methods such as Dynamic Temporal Warping [Müller and Röder 2006] and Temporal Pyramid [Wang et al. 2012] address these issues from the perspective of temporal modeling. However, they normally suffer from the temporal misalignment between sequential data of varied lengths due to the temporal translation and execution rate variations in human action data[Zhao et al. 2013]. The expensive computational overhead also makes these methods prohibited from large scale scenarios.

Human action sequence in the form of multimodal sensor data contain rich temporal patterns that can be used to distinguish between different action categories. In this article, we propose a novel approach from the perspective of temporal order modeling of the latent patterns inside the human action sequence. Specifically, a group of linear projections are used to map the original sensor data into different subspaces. Each subspace represents a certain latent pattern of the human action. We then compare

the temporal orders of the occurrences of these latent patterns and encode the entire sequence by the index of the pattern that comes first. This process is repeated multiple times and eventually yields a highly compact feature vector that is able to characterize the unique temporal patterns of the human action data. We name the feature vector as Temporal Order of Patterns (TOP). An example is illustrated in Figure 1, where three linear projections are used for the temporal order comparison. The index of the firstly occurring pattern is used as the encoded feature and a compact vector is generated through multiple iterations. The human action recognition problem can then be efficiently resolved by nearest neighbor search based on the Hamming distance between the compact TOP feature vectors. Therefore, the proposed algorithm can provide an efficient solution to real-world large-scale applications. Since we only utilize the temporal order information rather than the absolute time each pattern occurs at, the proposed algorithm is insensitive to temporal translation or execution variations.

In our previous work [Ye et al. 2015a], a temporal order-preserving video hashing technique based on random projections was proposed for 3D human action video retrieval. The current article proposes supervised optimization algorithms to learn projections of stronger discriminative power and investigates its performances in human activity recognition from multimodal sensor streams.

Our main contributions are summarized as follows:

— We propose a novel temporal order modeling approach that encodes the temporal order of latent patterns inside human action data. The resultant TOP feature vector forms a compact representation of the entire data sequence and is able to characterize the unique temporal structure of the human action sequence;
— We provide optimization algorithms that learn the optimal projections and enhance the discriminative power of the feature representation from the temporal order modeling; and
— We evaluate the performance of the proposed technique in two human activity datasets that are built on different sets of sensor signals. Experimental results show that the temporal order modeling approach can achieve state-of-the-art results with higher efficiency.

The remainder of this article is structured as follows. Section 2 briefly reviews the literature in the related area. In Section 3, the temporal order modeling approach is presented and analyzed. We further introduce the optimization algorithm for the temporal order modeling in Section 4. Performance studies are presented in Section 5, and finally, we conclude the article in Section 6.

It is worth noting that although we focus on human action recognition from sensor data, the proposed temporal order modeling approach could potentially be used as a generic solution to other recognition problems on sequential data that contains rich temporal patterns.

2. RELATED WORKS

Most exising works of human action recognition have focused on video data. Many approaches adopt histogram-based features to represent the distribution of the spatio-temporal patterns in human action data. Space-time interest points (STIP) [Laptev 2005] and cuboid [Dollár et al. 2005] are two representative local feature detectors to interpret a variety of human actions. Normally, the Bag-of-Words (BoW) feature representation is employed to learn the dictionary of spatio-temporal visual words and supervised learning methods, such as SVM, are then used to classify these feature vectors into different action categories. Besides the BoW-style methods, other features, such as the motion energy, have also been investigated for human action recognition. In [Bobick and Davis 2001], the motion energy image and motion history image were

constructed as a temporal template and the Hue moments were extracted to describe the human activity. Zhang *et.al.* [Zhang et al. 2015] proposed to jointly exploit the motion descriptor based on the combination of gradient information and optical flow with an effective representation based on sparse coding, for the purpose of detecting the most salient part of a human activity pattern.

Recent advances in depth cameras (e.g. Kinect) provided an affordable solution to access depth data from indoor environments and attracted huge interest in 3D human action recognition. A great number of approaches were proposed using depth data. [Gupta et al. 2013] proposed a human pose descriptor by computing the average normalized depth value on a dense grid over the 3D point cloud of the human body. In [Li et al. 2010], the bag of 3D points from the depth maps was sampled and projected onto three orthogonal 2D planes. An action graph was then employed to model the dynamics of the actions. Oreifej and Liu [Oreifej and Liu 2013] proposed the 4D normals from the surface of the 3D point cloud and introduced the histogram of oriented 4D normals (HON4D) feature descriptor.

With the success of skeleton joints estimation from the depth images [Shotton et al. 2013], joint-based methods gained huge popularity in 3D human action recognition. [Xia et al. 2012] proposed a view-invariant feature from the skeleton joints by coordinates transformation. Instead of directly using the joint locations, [Wang et al. 2012] and [Zhao et al. 2013] computed the normalized pair-wise joint distances from all pairs of skeleton joints and demonstrated good discriminative power. The joint-based features can be further quantized into code words, and histogram of 3D joints (HOJ3D) [Xia et al. 2012] and histogram of visual words [Zhu et al. 2013] were employed to describe the action sequences.

Most of the above methods adopt the histogram-based feature representations without considering the temporal information of the sequence data. On the contrary, temporal modeling-based methods have been proposed to model the temporal structures of human actions in a holistic way. Graph model-based methods such as Hidden Markov Model (HMM) [Lv and Nevatia 2006] and Conditional Random Field (CRF) [Han et al. 2010] have been proposed to interpret the dynamics of the human actions. Motion template-based approaches [Müller and Röder 2006; Zhao et al. 2013] employed Dynamic Time Warping (DTW) to compute the Euclidean distance between video data of varied lengths. Temporal Pyramid [Wang et al. 2012; Oreifej and Liu 2013] was developed to capture the temporal structure of the sequence by uniformly subdividing the action sequence into partitions. Spatio-temporal feature descriptors were then applied to each partition. This deterministic segmentation method may suffer from the misalignment of action sequences of varied length. To address this issue, an adaptive temporal pyramid [Yang and Tian 2014] was proposed by subdividing the sequence according to the motion energy. Hu, *et. al.* also adpot the Temporal Fourier Pyramid to interpret the temporal structure of the RGBD stream and further proposed a joint learning model to explore teh shared and feature-specific components between a collection of heterogeneous features [Hu et al. 2015]. In general, the aforementioned temporal modeling methods suffer from temporal misalignment due to factors of temporal translation, motion scales and execution rate variations of the human action data[Zhao et al. 2013]. Temporal modeling still remains to be a challenging problem for human action recognition.

More recently, Recurrent Neural Networks especially the Long Short-Term Memory Machines (LSTM) [Hochreiter and Schmidhuber 1997] have proven to be strong in capturing the temporal dynamics of time series. For example, a hierarchical architecture of RNN has been proposed to accumulate the results from multi-layer RNNs and eventually make the decision by a single-layer perceptron [Du et al. 2015]. A hybrid end-to-end deep architecture, DeepConvLSTM, built from Convolutional Neural Net-

Table I: Definitions of notations.

Notation	Definition
$\mathbf{X} \in \mathbb{R}^{D \times T}$	Sequential data collected from a sensor
T	Length of the sensor data
D	Dimension of the sensor data
$\mathbf{W} \in \mathbb{R}^{D \times K}$	Linear projections
K	Encoding size of temporal order modeling
L	Feature length of temporal order modeling
M	Size of mini-batch in each epoch of the optimization
N	Number of samples in the training set
s_{ij}	Pairwise label similarity between $\mathbf{X}_i$ and $\mathbf{X}_j$
w_{ij}	Weight of training pair $\mathbf{X}_i$ and $\mathbf{X}_j$ for Boosting

works (CNN) and the LSTM Recurrent Neural Networks has been proposed to model the temporal dynamics of human activities from wearable sensor signals [Ordóñez and Roggen 2016]. LSTM encodes the temporal structures of activity patterns in their memory states, but how the memory states can be mapped to temporal order of patterns is unclear, which limits their ability in fully capturing dynamic structures behind human activities.

In addition to the above vision-based methods, data from other sensors have also been explored for human action recognition. For example, inertial sensor signals, including the accelerometer [Preece et al. 2009; He and Jin 2009] and gyroscope [Altun et al. 2010], have been widely used for daily human activity recognition. Physiological signals such as heart rate, skin temperature and EEG have also been considered in a few works [Yin et al. 2008]. Statistical features in the time and frequency domains, such as min, max, mean, variance, skewness, kurtosis, Fourier Transform coefficients and Wavelet Transform coefficients to name a few, are normally extracted from the above signals [Altun et al. 2010]. Both unsupervised methods and supervised methods, such as KNN, SVM, decision tree and neural networks, have been employed to classify human activity data. Most of the above sensor signal-based methods address the classification problem based on statistical features. Although some methods achieve impressive performances when classifying daily human activities like "walking" and "lying down", they may be confused by human activities of similar statistical patterns but different temporal patterns. Our technique significantly differs from these methods by considering the internal temporal structure inside the sensor signals.

3. TEMPORAL ORDER MODELING

3.1. Latent Patterns

Suppose a sequential data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]$ of length T, where each point of data $\mathbf{x}_t \in \mathbb{R}^D$ is a D-dimensional sensor signal collected at time t ($1 \leq t \leq T$). Without lost of generality, $\mathbf{X}$ can be any sequential data such as time series and videos. In order to model the temporal characteristics of $\mathbf{X}$, we project $\mathbf{X}$ into a new sequence $\mathbf{s} = \mathbf{w}^T\mathbf{X}$, where $\mathbf{w} \in \mathbb{R}^D$. The projection $\mathbf{w}$ can be interpreted as forming a linear subspace for a latent pattern of the sequential data. Hence, the inner product $\mathbf{w}^T\mathbf{x}_t$ represents the confidence score that $\mathbf{x}_t$ contains this particular latent pattern.

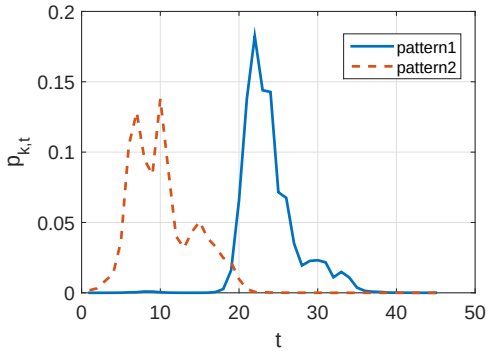


Fig. 2: Example of the distribution of $p_{k,t}$ of two patterns.

3.2. Temporal Order Encoding

We propose to model the temporal characteristics of sequential data by encoding the temporal order of the latent patterns inside them. To determine the temporal order of the latent patterns, we must first locate the moment they appear.

Suppose K latent patterns are used to encode the sensor data. We use the softmax function to model the probability that pattern k ($1 \leq k \leq K$) appears at time t:

$$p_{k,t} = \frac{e^{\alpha \mathbf{w}_k^{\mathrm{T}} \mathbf{x}_t}}{\sum_{t'=1}^{T} e^{\alpha \mathbf{w}_k^{\mathrm{T}} \mathbf{x}_{t'}}}, \tag{1}$$

where α is a rescale factor that scales the distribution of $p_{k,t}$ as illustrated in Figure 2. As can be seen from the figure, different latent patterns may exhibit different probability distributions if the projections are properly optimized. For example, the peak of p_2 occurs before that of p_1 in the example of Figure 2, which means pattern 2 appears earlier than pattern 1 in this sequential data. It is this intrinsic temporal order relationship that we wish to encode to model the temporal characteristics of the human action data. We further define the expectation of the time pattern k occurs by

$$u_k = \sum_{t=1}^{T} \frac{t}{T} p_{k,t}. \tag{2}$$

We call u_k the expected moment of pattern k as it gives the expectation of the normalized time (from 0 to 1) of the occurrence of pattern k given the probability $p_{k,t}$ at each point of time. We then encode $\mathbf{X}$ by the index of the pattern with the smallest expected moment among all K patterns.

$$c = \arg \min_{1 \leq k \leq K} u_k. \tag{3}$$

Take the two latent patterns in Figure 2 as an example, since the expectation of t in p_2 is smaller than that in p_1. The temporal order encoding process produces a feature value of 2, indicating that pattern 2 happens earlier than the other patterns.

As aforementioned, K projections $\mathbf{W} \in \mathbb{R}^{D \times K}$ are used to encode one feature dimension each time. The above process iterates L times and eventually generates an L-length K-ary feature vector $\mathbf{c} = [c_1, c_2, \ldots, c_L]$. Each element in $\mathbf{c}$ represents a temporal order encoding process between K projections. We refer to this feature as the Temporal Order of Patterns (TOP). K and L are the encoding size and the feature length of the

TOP, which are two crucial parameters controlling the performance of the above temporal order modeling approach. As an example, when $K = 2$, the pairwise comparison is made between two latent patterns (as illustrated in Figure 2). When a larger encoding size is adopted, a higher order comparison is performed among multiple latent patterns and more discriminative information can be encoded into the TOP feature. Similarly, a larger L may also bring in more information by investigating more latent patterns. However, it may also introduce redundant encoding to the feature vector and increase the computational overhead. More details of the impacts of K and L will be discussed in the experiments.

3.3. Discussions

To give an intuitive understanding of the latent temporal patterns for human action recognition, we show the following running examples from the experiments in the MSRActionPairs Dataset [Oreifej and Liu 2013]. Each subfigure in Figure 3 shows the cuves of probability $p_{k,t}$ of three latent patterns in different videos. The first row are from three videos of the action "put up a box", the second row shows three videos from the action "put down a box". All six videos are using the same set of three linear projections. It is clear that, pattern 2 (dash line) always appears earlier than the other patterns in the videos of "put up a box" (first row); while pattern 3 (dotted line) always appears earlier than the other patterns in all three videos of "put down a box" in the second row. Therefore, the temporal order of these latent patterns from the same set of projections carries discriminative information capable to characterize similar actions and distinguish different actions. It is also worth noting that, the peaks of different patterns in Figure 3 appear at different temporal locations of the video due to the execution variations and the temporal translations of the human action data [Zhao et al. 2013]. The proposed temporal order modeling approach is insensitive to these variations by exploiting the relatively orders of the latent patterns and produces reliable feature representations of the human action sequences. In addition, the modeling algorithm yields a highly compact feature vector from the original video. Hamming distance can then be utilized to compute the similarity between two videos. Thus, action recognition can be efficiently solved by nearest neighbor search based on the TOP feature vector.

4. LEARNING OPTIMIZED PROJECTIONS

In this section, we introduce a supervised optimization algorithm that learns the optimized projections and increases the discriminative power of the resultant TOP feature vector. We first formulate the optimization problem and then present the detailed learning algorithm.

4.1. Problem Formulation and Objective

The overall goal of optimization is to learn a set of projections so that sequential data of the same categories can have similar temporal order with respect to the corresponding latent patterns. In other words, their resultant TOP feature vectors should have a small Hamming distance. As aforementioned, the temporal order encoding is based on the $\arg\min$ that is discontinuous and non-differentiable. In order to address this challenge, we reformulate the problem as follows.

Suppose $\mathbf{u} = [u_1, u_2, \ldots, u_K]$ are the expected moments of K latent patterns from projections $\mathbf{W}$ in a sequential data. In order to investigate the temporal order of the K latent patterns inside the sequence, we use a second softmax function to compute the probability of pattern k coming first among the K patterns in a sequence. The

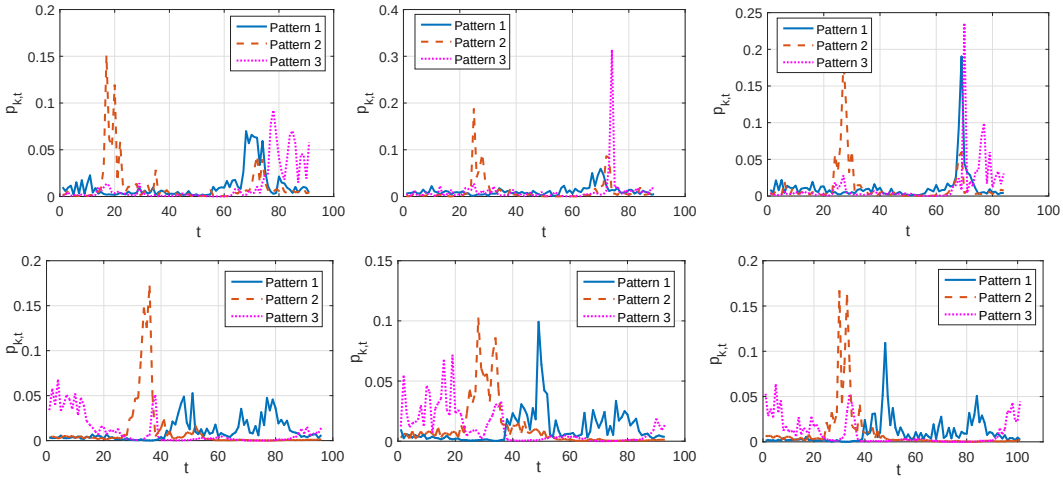


Fig. 3: Running examples of latent patterns from the same set of linear projections on two different categories of human action sequences. The first row are from the videos of "Put up a box", the second row are from videos of "put down a box." Results are from the experiments on the MSRActionPairs Dataset.

probability h_k is defined as

$$h_k = \frac{e^{(1-u_k)}}{\sum_{k'=1}^{K} e^{(1-u_{k'})}}$$

Given the above probability, we can then define a metric d_{ij} to evaluate whether the two sequential data i and j have similar temporal order with respect to the same set of projections $\mathbf{W}$ according to the following sum-product rule,

$$d_{ij} = \sum_{k=1}^{K} h_k^{(i)} h_k^{(j)}, \tag{4}$$

where $h_k^{(i)}$ and $h_k^{(j)}$ denote the probability of pattern k coming first among K patterns in sequence i and sequence j, respectively. It's clear that the product of the two softmax represents the probability that pattern k occurs first in both sequences. Therefore, the summation of $h_k^{(i)} h_k^{(j)}$ over all K patterns gives the overall probability of these two sequential data having the same temporal order with respect to $\mathbf{W}$. In this way, the $\arg\min$ is replaced by the similarity probability d_{ij}. The higher d_{ij}, the more likely sequential data i and j are similar regarding their temporal orders. The following l_2 loss function can then be defined to measure the error incurred by the temporal order encoding process,

$$\ell(d_{ij}, s_{ij}) = (d_{ij} - s_{ij})^2, \tag{5}$$

where s_{ij} is the pairwise label similarity. s_{ij} equals 1 when sequence i and j have the same label, and 0 otherwise. Thus, the overall learning objective is to find the optimized projections $\mathbf{W}$ by minimizing the total loss over all training pairs

$$\mathcal{L}(\mathbf{W}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \ell(d_{ij}, s_{ij}) + \lambda \|\mathbf{W}\|_F^2, \tag{6}$$

where N is the number of data in the training set. Note that $\mathbf{W}$ factors into the above objective function because d_{ij} is function of $\mathbf{W}$. Note that l_2 regularizer is introduced to reduce the overfitting. λ is a non-negative balancing coefficient to trade off between the l_2 loss and the l_2 regularizer on the projection matrix $\mathbf{W}$.

4.2. Optimization to the Temporal Order Modeling

Since we use softmax to model the probability of a latent pattern that comes first, the loss function in Eq.(6) is not convex. Nevertheless, $\ell(d_{ij}, s_{ij})$ is directly differentiable with respect to $\mathbf{W}$. Therefore, we adopt the stochastic gradient decedent approach to optimize $\mathbf{W}$ in an online fashion. Specifically, the gradient $\nabla_{\mathbf{W}}\mathcal{L}$ can be computed by the partial derivative of the total loss $\mathcal{L}$ with respect to each of the K projections in $\mathbf{W}$, $\nabla_{\mathbf{W}}\mathcal{L} = [\frac{\partial O}{\partial \mathbf{w}_1}, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_2}, \ldots, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_K}]$, where $\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k}$ can be computed as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k} = \sum_{i,j} \frac{\partial \ell}{\partial \mathbf{w}_k} = \sum_{i,j} 2(d_{ij} - s_{ij}) \frac{\partial d_{ij}}{\partial \mathbf{w}_k} + 2\lambda \mathbf{w}_k. \tag{7}$$

Since d_{ij}, u_k and $p_{k,t}$ are all functions of $\mathbf{w}_k$, it is straightforward to compute $\frac{\partial d_{ij}}{\partial \mathbf{w}_k}$ according to Eq.(1), Eq.(2) and Eq.(4). Thus $\mathbf{W}$ can be updated according to

$$\mathbf{W} \leftarrow \mathbf{W} - \eta \nabla_{\mathbf{W}}\mathcal{L}, \tag{8}$$

where η is the learning rate controlling the step size of each epoch. Figure 4 shows an example of the training objective through multiple epochs to learn a set of K projections $\mathbf{W}$ that generates one dimension of the L-length TOP feature vector. As can be seen, the learning process converges after 50 epochs.

The above learning procedure learns a set of K projections $\mathbf{W}$ to produce one dimension of the TOP feature vector. We repeat the above procedure L times and eventually learns L sets of projections which can produce an L-length K-ary TOP feature vector. The convex-concave nature of the objective function suggests there are no global minima, but multiple local minima. Taking advantage of this, the random initialization of $\mathbf{W}$ in each of the L iterations can be optimized to various local minima and eventually produces a complementary representation of the temporal characteristics of the sequential data. The pseudo code of the above learning algorithm is presented in Algorithm 1.

Algorithm 1 Optimization Algorithm

Input: training data $\chi = \{X_i\}_{i=1}^{N}$, K, L.
for $l = 1$ to L **do**
  Randomly initialize $\mathbf{W}_l = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K] \in \mathbb{R}^{D \times K}$;
  repeat
    Randomly select a mini-batch of training pairs;
    for $k = 1$ to K **do**
      Compute $\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k}$ over a mini-batch of χ by Eq.(7);
    end for
    Compute the gradient $\nabla_{\mathbf{W}}\mathcal{L} = [\frac{\partial \mathcal{L}}{\partial \mathbf{w}_1}, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_1}, \ldots, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_K}]$;
    Update $\mathbf{W}_l$ by $\mathbf{W}_l \leftarrow \mathbf{W}_l - \eta \nabla_{\mathbf{W}}\mathcal{L}$;
  until Convergence
end for
return the optimized projections $[\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L]$ for an L-dimensional TOP feature vector.

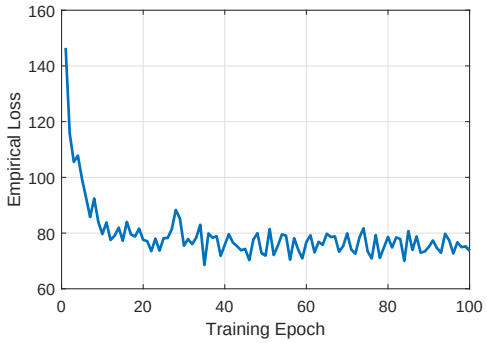


Fig. 4: Training loss over a mini-batch of pairwise training data through multiple epochs. Training loss is for one dimension of the TOP feature. Result is collected from the experiment on the MSRActionPairs Dataset [Oreifej and Liu 2013].

4.3. Sequential Learning with Boosting

The learning algorithm presented in the previous subsection learns L sets of projections $[\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L]$ independently. Although it can take advantage of multiple local minima and produce complimentary feature representations of the temporal patterns, it may also result in redundant projections corresponding to the same local minima. As a result, the discriminative power of the TOP feature vector may be compromised especially when L is large. To address this issue, we explore the Adaboost [Freund et al. 1999] algorithm to learn L sets of projections sequentially. In general, each set of K projections can be considered as a weak learner in the Adaboost framework and is trained sequentially. The subsequent projections attach greater importance to the "misclassified" training pairs by the previous projections. In the end, each set of projections will be trained based on different weights of the training pairs and achieve complimentary discriminative power to each other.

Specifically, denote w_{ij} as the weight assigned to a pair of training data X_i and X_j. All weights are initialized to be 1. When the lth set of K projections $\mathbf{W}_l$ are trained, the total training loss can be computed with respect to the pairwise weight as

$$\epsilon = \frac{\sum_{i,j} w_{ij}(d_{ij} - s_{ij})^2}{N(N-1)/2}, \tag{9}$$

where d_{ij} is the probability of data i and j having similar temporal order as defined in Eq.(4), s_{ij} is the pairwise label similarity and N is the size of the training set. It is clear that ϵ is normalized between 0 and 1. We use another weight θ_l to indicate the significance of projections $\mathbf{W}_l$. θ_l is computed as $\theta_l = 0.5 \log \frac{1-\epsilon}{\epsilon}$ and is further used to update the weight between each pair of training data by

$$w_{ij} = \begin{cases} w_{ij} e^{\theta_l (d_{ij} - s_{ij})^2}, & \theta_l \geq 0 \\ w_{ij} e^{-\theta_l (d_{ij} - s_{ij})^2}, & \textit{otherwise.} \end{cases} \tag{10}$$

The "misclassified" training pairs will be assigned larger weights so that following iterations can attach greater importance to these "misclassified" training pairs. w_{ij} is further normalized to ensure the weights of all training pairs add up to one throughout the Boosting. The objective in Eq.(6) is then modified into Eq.(11) to consider the

impact of the weights of the training pairs.

$$\mathcal{L}(\mathbf{W}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} w_{ij} \ell(d_{ij}, s_{ij}) + \lambda \|\mathbf{W}\|_F^2. \tag{11}$$

The partial derivative with respect to $\mathbf{w}_k$ in Eq.(7) is also updated accordingly:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k} = \sum_{i,j} 2w_{ij}(d_{ij} - s_{ij}) \frac{\partial d_{ij}}{\partial \mathbf{w}_k} + 2\lambda \mathbf{w}_k. \tag{12}$$

The pseudo code of the Sequential Learning Algorithm with Boosting is presented in Algorithm 2. As demonstrated in the pseudo code, the sequential learning process is implemented through updating the weight of each training pair so that following projections can attach greater significance to the "misclassified" pairs by the previous projections. Such a strategy makes the optimization of the projections $\mathbf{W}$ less isolated and more complementary to each other, and therefore addresses the issue of redundant projections corresponding to the same local minima. We will validate the effectiveness of the Sequential Learning Algorithm with Boosting in the performance study.

Algorithm 2 Sequential Learning Algorithm with Boosting

Input: training data $\chi = \{X_i\}_{i=1}^N$, K, L.
Initialize pairwise weight $w_{ij} = 1$ for all pairs of (X_i, X_j)
for $l = 1$ to L **do**
  Randomly initialize $\mathbf{W}_l = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K] \in \mathbb{R}^{D \times K}$;
  repeat
    for $k = 1$ to K **do**
      Compute $\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k}$ over a mini-batch of χ by Eq.(12);
    end for
    Compute the gradient $\nabla_{\mathbf{W}} \mathcal{L} = [\frac{\partial \mathcal{L}}{\partial \mathbf{w}_1}, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_1}, \ldots, \frac{\partial \mathcal{L}}{\partial \mathbf{w}_K}]$;
    Update $\mathbf{W}_l$ by $\mathbf{W}_l \leftarrow \mathbf{W}_l - \eta \nabla_{\mathbf{W}} \mathcal{L}$;
  until Convergence
  Compute error ϵ of $\mathbf{W}_l$ over the entire training set by Eq.(9);
  Compute weight θ_l of $\mathbf{W}_l$ and update each pairwise weight w_{ij} by Eq.(10);
end for
return the optimized projections $[\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L]$ for an L-dimensional TOP feature.

4.4. Complexity Analysis

Given a set of K projections $\mathbf{W}$, the cost of encoding one dimension of the TOP feature is $O(TDK) + O(TK) + O(\log K)$. The first component accounts for the cost of the linear projection as well as the softmax. The second component is the cost of finding the expected moments of the K latent patterns. The third component denotes the cost of finding the first occurring pattern. Considering K is typically small (from 2 to 8), the complexity is then $O(TD)$ which is linear with respect to the size of the input data.

Next, we analyze the complexity of the learning process. Let's consider the complexity of the mini-batch update in one epoch. Suppose the mini-batch contains M training pairs. The complexity at each pair of the training data consists of the following major components: the cost for computing the probability d_{ij} in Eq.(4) has a complexity of $O(KTD)$, and the cost for computing K partial derivatives $\frac{\partial d_{ij}}{\partial \mathbf{w}_k}$ is $O(K^2TD)$. Putting

it all together, the total cost to obtain $\nabla_{\mathbf{W}}$ over the mini-batch is $O(MK^2TD)$. Again, since K is typically a small number, the overall complexity for updating one epoch over the mini-batch is linear with respect to the number of training pairs in the mini-batch and the size of the input data.

5. PERFORMANCE STUDY

In this section, we extensively evaluate the performance of the proposed temporal order modeling approach on two multimodal sensor human action datasets and compare it with state-of-the-art results.

5.1. Datasets

UCI Daily and Sports Activities: The UCI Daily and Sports Activities Dataset [Altun et al. 2010] contains 19 daily and sports activities including *sitting, standing, lying, climbing stairs, walking, running, rowing, jumping, playing basketball*, etc. Each of the 19 actions is performed by 8 subjects (4 female and 4 male). Five body-worn miniature inertial and magnetic sensor units are placed on the body such that there is one on each knee, one on each wrist, and one on the chest. Each sensor unit has a 3-degree of freedom (DOF) triaxial accelerometer, a 3-DOF triaxial gyroscope and a 3-DOF magnetometer, and can produce 9 signals at a time. In total, 45 signals are available from the 5 sensor units located over the human body to collect the acceleration, rate of turn and Earth-magnetic field data. All sensor units are calibrated to acquire data at 25 Hz sampling frequency. The activities listed above are performed by each of the 8 subjects for 5 minutes and are divided into 5-second segments. In total, there are 9120 segments, each of which is a 45-dimensional multimodal sequential data containing 125 data points. The UCI Daily and Sports Activities Dataset is one of the largest sensor stream datasets for human activities in terms of number of signals and activity categories [Lara and Labrador 2013].

MSRActionPairs: The MSRActionPairs dataset [Oreifej and Liu 2013] consists of 12 action types performed by 10 subjects. Each subject performs every action three times. The actions are captured by the Microsoft Kinect depth camera [Zhang 2012]. The dataset provides a good variety of multimodal data streams including RGB and depth streams, as well as the coordinates of detected human skeleton joints. The actions are grouped into 6 pairs of similar actions which have exactly the same body postures but different temporal orders. For example, “pick up” and “put down,” “push a chair” and “pull a chair.” The actions in a pair are easy to confuse with one another, making the dataset very challenging. It is a very suitable dataset to validate the effectiveness of the proposed temporal modeling approach for human action recognition.

It is worth noting that the above two datasets are built on significantly different data sources: the first one is composed of inertial and magnetic data while the second one is image-based. We choose these two datasets to demonstrate that the proposed technique can work as a general solution to different data sources for human action recognition.

5.2. Experiment settings

Considering the compact nature of the TOP feature generated by the temporal modeling approach, we adopt KNN to classify the human action sensor data based on the Hamming distance. Hyper parameters, such as the rescale factor α, size of mini-batch and learning rate, are tuned based on the cross-validation on the training set.

Specifically, the results reported in the experiments are obtained using the following parameters. The initial learning rate for the UCI Daily and Sports Activities Dataset is set to 1. The learning rate will decay by 10% every 10 epochs. The pair-wise training

data are paired from the training samples in the training set, we use 20% of the same-label training pairs and 5% of the different-label training pairs. Mini-batch size is 798. The initial learning rate for the MSRActionPairs Dataset is set to 0.25 and will decay by 5% every 10 epochs. During the optimization, all same-label training pairs and 25% different-level training pairs are used. Mini-batch size is 60.

5.3. Results on UCI Daily and Sports Activities

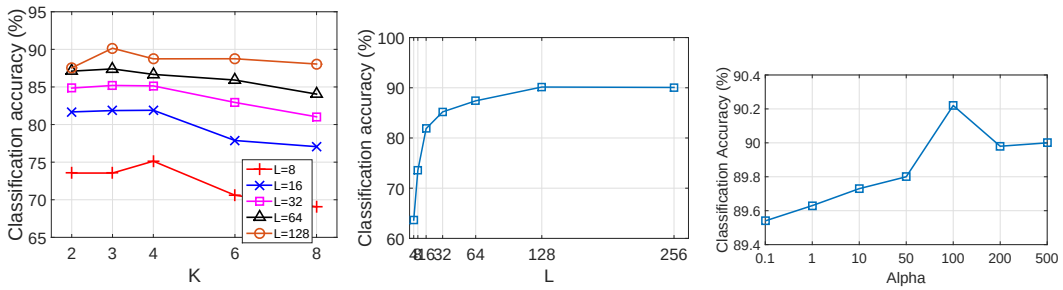
In the experiments, we follow the same leave-one-subject-out (L1O) cross-validation technique as in [Altun et al. 2010] so that the sensor data of 7 of the subjects are used for the training, and the data of the remaining subject are used in turn for validation. This process is repeated 8 times such that data of each subject is used exactly once for the validation. The final classification accuracy is estimated by the average results over the above subject-based L1O strategy.

As illustrated in Figure 1, the original sensor streams are highly noisy. To address this challenge, a preprocessing step is adopted to the original sensor data. Specifically, we use a sliding window to scan the data stream. The *mean*, *min*, *max* and *variance* are computed for each dimension of the original data stream inside the sliding window. As a result, four 45-dimensional sequential data are generated from the raw sensor data as the input of the temporal modeling algorithm. In the experiment, the sliding window size and the overlap step are set to 10 and 2, respectively. These two hyper parameters are also tuned based on the cross validation on the training set. Each of the 4 sequential data from the preprocessing step is fed to the temporal order modeling algorithm and yields its own TOP feature vector. During the classification, the Hamming distance based on the above 4 TOP feature vectors are fused by the average, and the final result is estimated by the KNN based on the fused Hamming distance.

5.3.1. Impact of Encoding Size, Feature Length and Rescale Factor. The encoding size K and the feature length L are two critical parameters to the performance of the temporal order modeling approach. We first evaluate the impact of K and L on the classification accuracy. In the experiment, we test different combinations of K values and L values. K is set to 2, 3, 4, 6 and 8, and L is set to 8, 16, 32, 64 and 128. Experimental results are shown in Figure 5a. We observe that a moderate K (3 or 4) can produce higher accuracy than the pairwise order comparison ($K = 2$). Such an observation demonstrates that the temporal order modeling algorithm can benefit from the comparison between a larger number of latent patterns since more discriminative information can be encoded. However, since only partial ordering (the first coming pattern) is encoded for the sake of compactness, the amount of information encoded becomes less and less when K increases. This may explain why the accuracy starts to drop when K continues to increase to 6 and higher. Therefore, the trade-off between the amount of encoded information and the compactness of the TOP feature must be considered.

We further fixed K to the optimal value 3 and investigate the impact of the TOP feature length L on the performance. Experimental results are shown in Figure 5b. As can been seen, the classification accuracy increases significantly as the feature length L goes up from 4 to 32. A 4-dimensional TOP feature vector can already achieve an accuracy around 65%. The accuracy continues to grow slowly as L goes up to 128 and remains stable after that. Such behavior demonstrates our earlier discussions that longer feature length may also bring redundant information, and the performance gain becomes marginal. Considering the trade-off between the accuracy and the computational overhead, we set L to 128 in the following experiments.

The rescale factor α in Eq.(1) is also an important hyper parameter. To assess the impact of α, we fix K and L to 3 and 128, respectively, and run experiments on different α ranging from 0.1 to 500. Figure 5c shows that the classification accuracy increase as



(a) Impact of the encoding size. (b) Impact of the TOP feature length (K is set to 3). (c) Impact of the rescale factor α (K is set to 3, L is set to 128).

Fig. 5: Impact of encoding size, feature length and rescale factor α on the performance in the UCI Daily and Sports Activities Dataset.

α increases and reaches the peak when α reaches 100. When α continues to increase, the performance start to drop. Such a behavior can be explained as follows. Wearable sensor signals are normally fluctuating between a narrow dynamic range. The resultant projection score appears flat throughout the sequential data. A reasonably large rescale factor α can magnify the disparity of projection scores through time and eventually makes the curve of probability $p_{k,t}$ in Eq.(1) less flat, which is beneficial to produce a more informative expected moment to encode the temporal pattern of the sequence. Therefore, a higher α can produce higher classification accuracy. Nevertheless, an excessively large α like 500 will make the largest $p_{k,t}$ dominate the entire sequence, and always fix the expected moment at the time where the peak of $p_{k,t}$ appears, which is less reliable then the expected moment. That's why the performance starts to drop when α is too large.

5.3.2. Impact of Number of Nearest Neighbors in KNN. We use KNN as the classifier to predict the label of the sensor data in the testing set. The number of the top nearest neighbors used in KNN has a big impact on the final classification accuracy. We run experiments on different numbers of nearest neighbors and report the results in Figure 6. As the figure shows, the KNN classifier achieves higher accuracy when a small number of nearest neighbors are adopted (around 7 to 13) which demonstrates that the TOP feature by the temporal order modeling method can well preserve the pairwise similarity of the original sensor data.

5.3.3. Effectiveness of Optimization and Boosting. In order to validate the effectiveness of the optimization algorithm as well as the Boosting algorithm, we implement three version of the temporal order modeling algorithm and compare their performances:

(1) Temporal Order Modeling with random projections
(2) Temporal Order Modeling with optimized projections
(3) Temporal Order Modeling with optimized projections and Boosting

Table II shows that temporal order modeling with the optimized projections has achieved higher classification accuracy than the random projection-based method; while the sequential learning with Boosting has further improved the performance of the optimized projections by 2%. Therefore, the effectiveness of the optimization algorithm as well as the Adaboost integration have been validated.

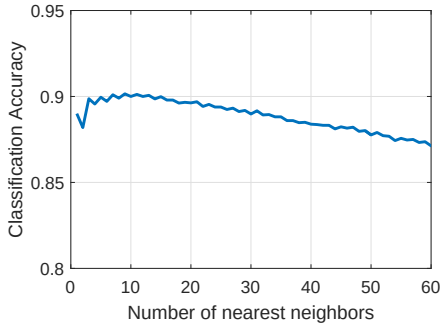


Fig. 6: Impact of the number of nearest neighbors in KNN on the performance.

5.3.4. Confusion Matrix. The confusion matrix of the classification results on the 19 action categories are shown in Figure 7. We can see that the proposed temporal order modeling approach can successfully address the confusions between most of the action classes. Even those actions with minimal movements such as "lying on back" (A3) and "lying on the right" (A4) can be accurately distinguished from each other. We do see significant confusions between "walking on a treadmill in flat" and "walking on a treadmill with 15^o inclined position". These two actions are very similar by themselves and are extremely difficult to tell apart from one another. Thus there are not very many differences in their temporal patterns to be leveraged by the algorithm.

5.3.5. Compare with State-of-the-art Results. We compare the performance of the temporal order modeling approach with state-of-the-art results on this dataset. In [Altun et al. 2010] and [Barshan and Yüksek 2014], several state-of-the-art results based on the same subject-based leave-one-out strategy have been reported (listed in Table II). As opposed to the proposed method, they extract the global statistical features over the entire sensor stream and do not leverage temporal patterns over time. Specifically, 26 features are extracted from each signal, including the min, max, mean, variance, skewness, kurtosis, auto-correlation sequence and the Discrete Fourier Transform coefficients. A 1170-dimensional feature vector is then produced to represent the original sequential data. Several supervised and unsupervised algorithms, such as Dynamic Time Warping, Naive Bayes, Least Squares Method, Artificial Neural Networks and Support Vector Machines, have been adopted to predict the labels of the testing samples. As shown in Table II, the proposed temporal modeling algorithm with KNN achieves 90.15% classification accuracy, which is higher than most of the state-of-the-art results. The only method having a slightly higher accuracy than ours is the Artificial Neural Networks by using the WEKA machine learning toolbox [Barshan and Yüksek 2014]. Whereas the proposed temporal order modeling only exploits the unsupervised K Nearest Neighbor Search with Hamming distance, which demonstrates the advantage of leveraging temporal order information for human activity recognition with sensor data.

We note that end-to-end deep neural networks have been successfully employed in various pattern recognition tasks and have refreshed state-of-the-art results by a significant margin. Therefore, we also compare our work with a DeepConvLSTM nodel [Ordóñez and Roggen 2016] consisting of Convolutional Neural Networks (CNN) and Long Short-Term Memory Recurrent Neural Networks (LSTM) with more than one million parameters. DeepConvLSTM takes the raw sensor data as input without any feature extraction and directly outputs estimation of the label by softmax. As shown in

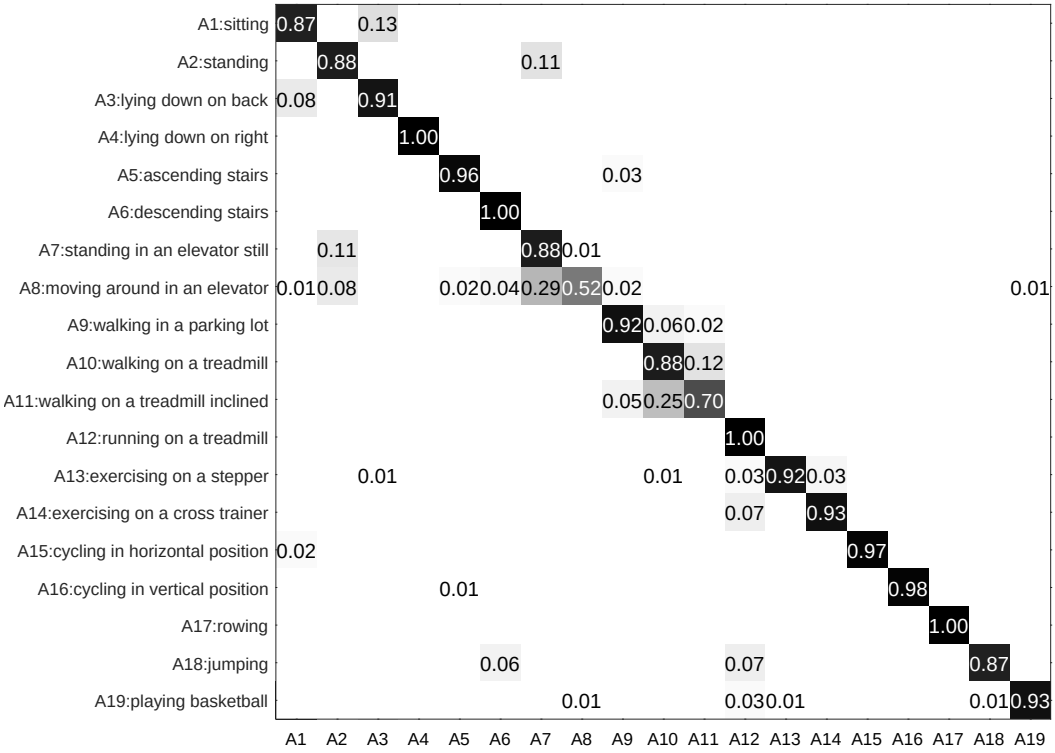


Fig. 7: Confusion matrix of the classification results in the UCI Daily and Sports Activities Dataset.

Table II, DeepConvLSTM achieved 91.8% classification accuracy and outperform our method and all the other state-of-the-art results. It is worth noting that our temporal order modeling approach only exploits linear projections and the KNN classifier and cannot compare with the nonlinear deep model like DeepConvLSTM in terms of the model complexity and number of parameters. However, our method can still achieve a comparable performance with respect to DeepConvLSTM. In addition, our method can also handle sequential data of varied length. Nevertheless, DeepConvLSTM only accepts fix-length input sequences due to its fixed 1D convolution structure.

5.3.6. Comparison of the Computational Cost. In order to validate the efficiency of the proposed technique, we compute the execution time of the temporal order modeling with optimized projections and Boosting and compare it with some of start-of-the-art methods listed in Table II. In the experiment, we report the total execution time of preprocessing, feature extraction, training and classification. Time is measured in unit of second. Non-deep methods are running in Matlab 2015b on a desktop PC with Intel i7-4770 quad-core 3.4GHz CPU and 32G RAM. DeepConvLSTM is running on Theano with Lasagne package [Dieleman et al. 2015]. An nVidia GTX 660 graphics card is used to accelerate the training and testing of DeepConvLSTM. For a fair comparison,

Table II: Comparison with state-of-the-art results in the UCI Daily and Sports Activities Dataset.

Reported Methods	*Accuracy*
Bayesian Decision Making [Altun et al. 2010]	75.8%
Dynamic Time Warping [Altun et al. 2010]	85.2%
Least Squares Method [Altun et al. 2010]	85.3%
K Nearest Neighbors [Altun et al. 2010]	86.0%
Support Vector Machines [Barshan and Yüksek 2014]	89.9%
Artificial Neural Networks [Barshan and Yüksek 2014]	91.0%
DeepConvLSTM [Ordóñez and Roggen 2016]	**91.8**%
Our Methods	
Temporal Order Modeling (random projection) + KNN	87.36%
Temporal Order Modeling (optimized projection) + KNN	88.14%
Temporal Order Modeling (optimized projection w/ Boosting) + KNN	90.15%

Table III: Comparison of the execution time breakdown in the UCI Daily and Sports Activities Dataset. Time is measured in unit of second. $K = 3$, $L = 128$ for temporal order modeling and the sequential learning with Boosting is used to learned the optimized projections.

Methods	*Preprocessing*	*Temporal modeling / feature extraction*	*Training*	*Classification*
Statistical + DTW	-	16.40	-	728.7
Statistical + KNN	-	16.40	-	0.41
Statistical + SVM	-	16.40	0.82	0.22
DeepConvLSTM (GPU)	0.55	-	7448	1.56
DeepConvLSTM (CPU)	0.55	-	77608	33.90
Our method	5.01	10.30	11454	3.24

we also run the DeepConvLSTM with pure CPU and compare the training and testing time. Experimental results are summarized in Table III. It takes 10.3 second to compute the TOP feature vectors for all the 9120 segmenets in the dataset which is highly efficient. Even consider the overhead of preprocessing, the proposed temporal order modeling method is still faster than the statistical feature extraction in [Altun et al. 2010]. Regarding the classification time, it takes only 3.24 seconds by KNN with TOP feature to classify all the samples in the testing set which is significantly more efficient than DTW. We note that SVM is much faster than all the other methods in testing. This is due to the nature of the SVM, which only needs to investigate support vectors when making decisions. The training process of DeepConvLSTM takes 77608 seconds to converge using CPU, and the testing process takes 33.90 seconds, which are 7 times and 10 times slower than the temporal order modeling methods with KNN. We see that GPU acceleration can achieve more than $10X$ speedup in both training and testing for DeepConvLSTM. However, for a fair comparison, the proposed temporal order modeling method has a clear advantage of the computational cost.

5.4. Results on MSRActionPairs

In the experiments, we follow the same cross-subject training/testing data split policy as in [Oreifej and Liu 2013; Yang and Tian 2014] that the first five subjects are used for testing and the remaining five are used for training. As opposed to the sensor signals in the UCI Daily and Sports Activities dataset, the data in the MSRActionPairs dataset are in the form of an RGBD image sequences. To better leverage the image data, we extract the following three features from each RGBD image before the temporal order modeling:

— **3D Joint Coordinates:** 3D coordinates of the 20 joints of the skeleton.
— **3D Joint Offset:** offset of each of the 20 joints between two consecutive frames [Zhu et al. 2013].
— **Histogram of Velocity Component:** histogram of the velocity components of the point cloud [Ye et al. 2015c].

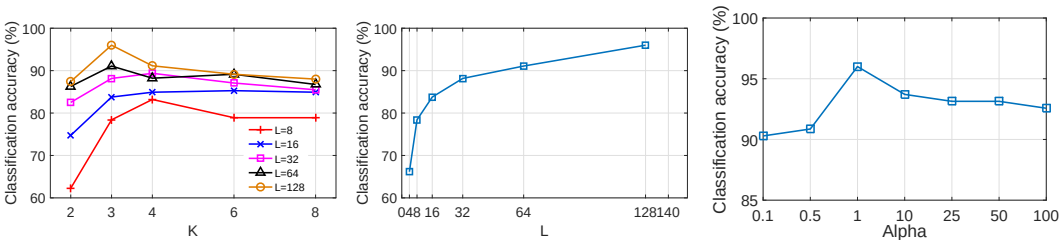
The above three features are extracted from each image frame of the RGBD sequences and are used as the input of the temporal order modeling approach in order to produce their own compact TOP feature vectors. Similar to the UCI dataset, the Hamming distance based on the three TOP feature vectors are fused, and the KNN classifier is adopted to predict the labels of the testing data based on the fused Hamming distance.

5.4.1. Impact of Encoding Size, Feature Length and Rescale Factor. Following the same strategy as in the previous experiments, we first evaluate the impact of the encoding size K on the performance while fixing the TOP feature length L to different values (e.g. 8, 16, 32, 64, 128). Results are shown in Table 8a. Similar to the results on the UCI dataset, a moderate K can achieve higher classification accuracy. Although the full ordering is encoded when K is set to 2, a small number of latent patterns makes the temporal order encoding less discriminative. A large K can enrich the TOP feature vector by investigating more latent patterns at a time. Nevertheless, more and more information are discarded as K increases due to the first-take-all encoding scheme. As a result, the overall performance of the TOP feature also decreases when K keeps increasing.

We further fix K to the optimal value of 3 and evaluate the impact of the feature length L on the performance. As shown in Figure 8b, the accuracy increases as L increases. However, the performance gain gets more and more marginal when L is over 32. Again, this is because of the redundant information introduced by the long TOP feature vector. Considering the trade-off between the accuracy and the computational overhead, we fix K and L to 3 and 128 respectively in the following experiment.

We also evaluate the impact of the rescale factor α on the performance and show the results in Figure 8c. The classification accuracy increases as α increases, and reaches the highest performance when α is set to 1. After that, the performance starts to drop when α continues to increase. Such a behavior is consistent with the results in the UCI dataset. A reasonably larger α can help improve the discriminative capability of the TOP feature by reshaping the curve of $p_{k,t}$. Whereas, when α is too large, the largest $p_{k,t}$ will dominate the entire sequence and compromise the discriminative power of the TOP feature vector.

5.4.2. Impact of Number of Nearest Neighbors in KNN. The classification results based on different numbers of nearest neighbors used in KNN is shown in Figure 9. The best classification accuracy is achieved when the top $15 \sim 25$ nearest neighbors are used to predict the label of the testing data. This demonstrates that the resultant TOP feature can well preserve the pairwise label similarity in the top nearest neighbors.



(a) Impact of the encoding size. (b) Impact of TOP feature length (K is fixed to 3). (c) Impact of the rescale factor α (K is set to 3, L is set to 128).

Fig. 8: Impact of the encoding size, TOP feature length and rescale factor α on the performance in the MSRActionPairs Dataset.

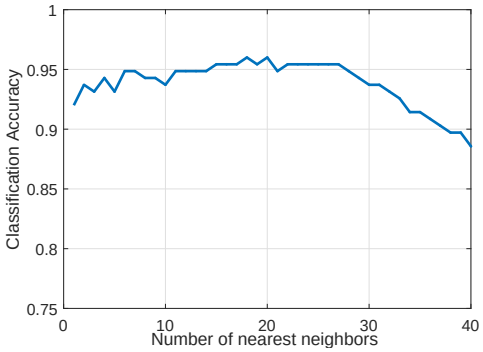


Fig. 9: Impact of number of nearest neighbors in KNN on the performance.

5.4.3. Effectiveness of Optimization and Boosting. To validate the effectiveness of the optimization algorithm as well as the sequential learning with Boosting, we further compare the performances of the random projection-based temporal order modeling with the optimized projection-based methods. Table IV shows that the optimization algorithm outperforms the random projection-based modeling algorithm by more than 4% in terms of the classification accuracy; the sequential learning with Boosting further increases the performance of the optimization algorithm by 4%, and achieves 96% classification accuracy. Such results demonstrate the effectiveness of the optimization algorithm and the sequential learning with Boosting algorithm.

5.4.4. Confusion Matrix. The confusion matrix of the classification results on the 12 action types is illustrated in Figure 10. It shows that the algorithm can perfectly distinguish the differences between the majority of the action pairs. The only exceptions are between the pair "pickUpBox" and "putDownBox," and the pair "liftBox" and "placeBox."

5.4.5. Comparison with State-of-the-Art Results. We compare the performance of the temporal order modeling approach with state-of-the-art methods on RGBD human action recognition. Results are listed in Table IV. The temporal order modeling approach with the KNN classifier can achieve state-of-the-art accuracy. The only methods having a slightly higher accuracy than ours are the Histogram of 4D Normals [Oreifej and Liu 2013] and the Super Normal Vector [Yang and Tian 2014]. However, both of them are

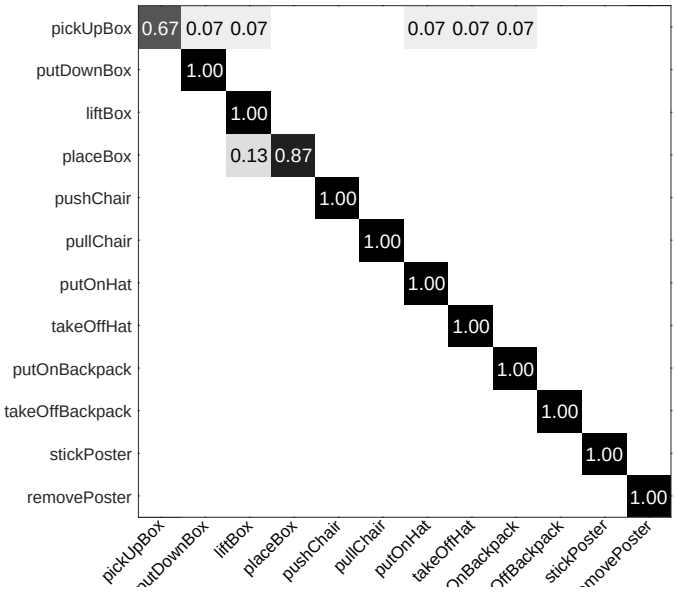


Fig. 10: Confusion matrix of classification results in the MSRActionPairs Dataset.

Table IV: Comparison with state-of-the-art results in the MSRActionPairs Dataset.

Reported Methods	*Accuracy*
Depth Motion Maps [Yang et al. 2012]	66.11%
Skeleton + LOP + Pyramid [Wang et al. 2012]	82.22%
WTA Hash + DTW [Ye et al. 2015b]	91.88%
DeepConvLSTM [Ordóñez and Roggen 2016]	93.18%
HON4D [Oreifej and Liu 2013]	96.67%
Super Normal Vector [Yang and Tian 2014]	**98.89**%
Our Methods	
Temporal Order Modeling (random projection) + KNN	87.89%
Temporal Order Modeling w/o Boosting + KNN	92.00%
Temporal Order Modeling w/ Boosting + KNN	96.00%

highly specialized methods relying on sophisticated features such as 4D surface normals or the 3D hypersurface polynormals of the point cloud. They may not be applied to more general problems. On the contrary, the proposed temporal order modeling approach is a generalized solution that does not rely on any particular feature, which demonstrates the advantages of the proposed method. We also run the experiment with the DeepConvLSTM model. DeepConvLSTM is an end-to-end deep learning method which doesn't require any feature extraction. Therefore, we concatenate the 3D coordinates of the 20 joints of the skeleton in each frame and form the 60-dimensional input data. Videos in the MSRActionPairs Dataset have different length. Since DeepCon-

Table V: Comparison of the execution time breakdown in the MSRActionPairs Dataset. $K = 3$, $L = 128$ for TOP feature. Time is measured in unit of second.

Methods	*Preprocessing*	*Temporal modeling / feature extraction*	*Training*	*Classification*
Super Normal Vector	4520.1	8281.4	1445.7	1.63
DeepConvLSTM (CPU)	1.25	-	2237	8.26
DeepConvLSTM (GPU)	1.25	-	243	0.45
Our method	952.5	3.39	398.0	0.11

vLSTM only accepts fixed-size input, we further normalized the sequential data into 100-frame-length by linear interpolation. A min-max normalization is used to scale all the data into $[0, 1]$. Different from the results in the UCI dataset, DeepConvLSTM only achieves 93.18% accuracy which is roughly 3% lower than the proposed temporal order modeling method. One of the possible explanations is, DeepConvLSTM is specially designed for the 1D time series and doesn't leverage the visual clues in the RGBD video data. As a result, it has limited capability in distinguish different human actions in videos. On the contrary, the proposed temporal order modeling algorithm shows good performance in both the sensor dataset and RGBD video dataset, and can handle videos of varied length.

5.4.6. Comparison of the Computational Cost. We also evaluate the computational cost of the proposed temporal order modeling with optmized projections and Boosting, and compare the execution time with that of state-of-the-art approaches. Comparison results are summarized in Table V. It is clear that the proposed temporal order modeling is significantly more efficient than the Super Normal Vector [Yang and Tian 2014] with respect to the execution time in preprocessing, feature extraction, training and classification. Specifically, it takes only 3.39 seconds to compute the TOP feature vectors for all samples in the testing set, and it takes only 0.11 seconds to classify them by KNN based on the Hamming distance. It is worth noting that video data has significantly higher semantic visual information than wearable sensor signals. Therefore, we employ several visual features in the preprocessing stage to better represent the raw image data and facilitate the temporal order encoding atop. We note that the preprocessing stage takes even more time than the training stage. This is because we use the Histogram of Velocity Component feature [Ye et al. 2015c] to represent the RGBD frame which is particularly computational expensive. The cost can be reduced if other feature is employed. As can be seen, it takes 398 seconds to learn the optimized linear projections by the sequential Learning with Boosting, which is significantly faster than the training time of Super Normal Vector method. Finally, we compare the training and classification time with DeepConvLSTM model. It is clear that, the proposed temporal order modeling algorithm requires significantly shorter training and testing time than DeepConvLSTM using CPU and achieves comparable training time to DeepConvLSTM using GPU.

6. CONCLUSIONS

Human action data in terms of multimodal sensor data contain rich temporal structure information that can characterize the unique underlying patterns of different action types. Compared with the spatial and statistical features for human action recognition, such temporal patterns are relatively under-researched. In this article, we propose a

novel temporal order modeling approach as a general solution to human action recognition. Specifically, we explore subspace projections to extract the latent temporal patterns in the sequential data. The temporal order between these patterns is compared and the index of the pattern that appears first is encoded to form a compact feature vector. We further learn the optimized projections by minimizing the empirical loss based on the objective on the training data. An Adaboost algorithm is then integrated into the learning process to reduce the redundant information between each dimension of the feature vector. Finally, KNN is used to efficiently solved the human action recognition based on the Hamming distance between these compact vectors. We evaluate the performance of the proposed algorithms on two public human action datasets consisting of various multimodal sensor streams. The results show the outstanding performance of the proposed technique in both accuracy and efficiency.

ACKNOWLEDGMENTS

This material is based upon work partially supported by NASA under Grant Number NNX15AV40A. Any opinions, findings, and conclusion or recommendation expressed in this materials are those of the authors and do not necessarily reflect the views of NASA.

REFERENCES

Kerem Altun, Billur Barshan, and Orkun Tunçel. 2010. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition* 43, 10 (2010), 3605–3620.

Billur Barshan and Murat Cihan Yüksek. 2014. Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units. *Comput. J.* 57, 11 (2014), 1649–1667.

Aaron F. Bobick and James W. Davis. 2001. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 23, 3 (2001), 257–267.

Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, J Kelly, and others. 2015. Lasagne: First Release. *Zenodo: Geneva, Switzerland* (2015).

Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. 2005. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. 65–72.

Yong Du, Wei Wang, and Liang Wang. 2015. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1110–1118.

Yoav Freund, Robert Schapire, and N Abe. 1999. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence* 14, 771-780 (1999), 1612.

Raj Gupta, Alex Yong-Sang Chia, and Deepu Rajan. 2013. Human activities recognition using depth images. In *Proceedings of the 21st ACM international conference on Multimedia*. 283–292.

Lei Han, Xinxiao Wu, Wei Liang, Guangming Hou, and Yunde Jia. 2010. Discriminative human action recognition in the learned hierarchical manifold space. *Image and Vision Computing* 28, 5 (2010), 836–849.

Zhenyu He and Lianwen Jin. 2009. Activity recognition from acceleration data based on discrete consine transform and SVM. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. IEEE, 5041–5044.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

M Anwar Hossain, Pradeep K Atrey, and Abdulmotaleb El Saddik. 2011. Modeling and assessing quality of information in multisensor multimedia monitoring systems. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 7, 1 (2011), 3.

Jian-Fang Hu, Wei-Shi Zheng, Jianhuang Lai, and Jianguo Zhang. 2015. Jointly learning heterogeneous features for RGB-D activity recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5344–5352.

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

Oscar D Lara and Miguel A Labrador. 2013. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys & Tutorials* 15, 3 (2013), 1192–1209.

Wanqing Li, Zhengyou Zhang, and Zicheng Liu. 2010. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops, 2010 IEEE Computer Society Conference on*. 9–14.

Fengjun Lv and Ramakant Nevatia. 2006. Recognition and segmentation of 3-d human action using hmm and multi-class adaboost. In *Computer Vision–ECCV 2006*. 359–372.

Meinard Müller and Tido Röder. 2006. Motion templates for automatic classification and retrieval of motion capture data. In *Proceedings of the 2006 ACM SIGGRAPH*. 137–146.

Francisco Javier Ordóñez and Daniel Roggen. 2016. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* 16, 1 (2016), 115.

Omar Oreifej and Zicheng Liu. 2013. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Computer Vision and Pattern Recognition, 2013 IEEE Conference on*. 716–723.

Stephen J Preece, John Yannis Goulermas, Laurence PJ Kenney, and David Howard. 2009. A comparison of feature extraction methods for the classification of dynamic activities from accelerometer data. *Biomedical Engineering, IEEE Transactions on* 56, 3 (2009), 871–879.

Abu Saleh Md Mahfujur Rahman, M Anwar Hossain, and Abdulmotaleb El Saddik. 2010. Spatial-geometric approach to physical mobile interaction based on accelerometer and IR sensory data fusion. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 6, 4 (2010), 28.

Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. 2013. Real-time human pose recognition in parts from single depth images. *Commun. ACM* 56, 1 (2013), 116–124.

Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. 2012. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition, 2012 IEEE Conference on*. 1290–1297.

Lu Xia, Chia-Chih Chen, and JK Aggarwal. 2012. View invariant human action recognition using histograms of 3D joints. In *Computer Vision and Pattern Recognition Workshops, 2012 IEEE Computer Society Conference on*. 20–27.

Xiaodong Yang and Yingli Tian. 2014. Super normal vector for activity recognition using depth sequences. In *Computer Vision and Pattern Recognition, IEEE Conference on*.

Xiaodong Yang, Chenyang Zhang, and YingLi Tian. 2012. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *Proceedings of the 20th ACM international conference on Multimedia*. 1057–1060.

Jun Ye, Hao Hu, Kai Li, Guo-Jun Qi, and Kien A Hua. 2015a. First-Take-All: Temporal Order-Preserving Hashing for 3D Action Videos. *arXiv preprint arXiv:1506.02184* (2015).

Jun Ye, Kai Li, and Kien A Hua. 2015b. WTA Hash-Based Multimodal Feature Fusion for 3D Human Action Recognition. In *2015 IEEE International Symposium on Multimedia (ISM)*. IEEE, 184–190.

Jun Ye, Kai Li, Guo-Jun Qi, and Kien A Hua. 2015c. Temporal Order-Preserving Dynamic Quantization for Human Action Recognition from Multimodal Sensor Streams. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*. ACM, 99–106.

Jie Yin, Qiang Yang, and Jeffrey Junfeng Pan. 2008. Sensor-based abnormal human-activity detection. *Knowledge and Data Engineering, IEEE Transactions on* 20, 8 (2008), 1082–1090.

Bo Zhang, Nicola Conci, and Francesco GB De Natale. 2015. Segmentation of Discriminative Patches in Human Activity Video. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 12, 1 (2015), 4.

Zhengyou Zhang. 2012. Microsoft kinect sensor and its effect. *MultiMedia, IEEE* 19, 2 (2012), 4–10.

Xin Zhao, Xue Li, Chaoyi Pang, Xiaofeng Zhu, and Quan Z Sheng. 2013. Online human gesture recognition from motion data streams. In *Proceedings of the 21st ACM international conference on Multimedia*. 23–32.

Yu Zhu, Wenbin Chen, and Guodong Guo. 2013. Fusing Spatiotemporal Features and Joints for 3D Action Recognition. In *Computer Vision and Pattern Recognition workshops, 2013 IEEE Conference on*.

Received July 2016; revised Sept 2016; accepted Dec 2016