A Novel Distribution-Embedded Neural Network for Sensor-Based Activity Recognition

Hangwei Qian[1,2,3] , **Sinno Jialin Pan**[1] , **Bingshui Da**[1] and **Chunyan Miao**[1,2]
[1]School of Computer Science and Engineering
[2]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly
[3]Interdisciplinary Graduate School
Nanyang Technological University, Singapore
qian0045@e.ntu.edu.sg {sinnopan, da0002ui, ascymiao}@ntu.edu.sg

Abstract

Feature-engineering-based machine learning models and deep learning models have been explored for wearable-sensor-based human activity recognition. For both types of methods, one crucial research issue is how to extract proper features from the partitioned segments of multivariate sensor readings. Existing methods have different drawbacks: 1) feature-engineering-based methods are able to extract meaningful features, such as statistical or structural information underlying the segments, but usually require manual designs of features for different applications, which is time consuming, and 2) deep learning models are able to learn temporal and/or spatial features from the sensor data automatically, but fail to capture statistical information. In this paper, we propose a novel deep learning model to automatically learn meaningful features including statistical features, temporal features and spatial correlation features for activity recognition in a unified framework. Extensive experiments are conducted on four datasets to demonstrate the effectiveness of our proposed method compared with state-of-the-art baselines.

1 Introduction

Human activity recognition is an important technique with a wide range of real-world applications, e.g., assisted living, personalized health monitoring, security and smart homes [Janidarmian *et al.*, 2017; Bulling *et al.*, 2014; Lara and Labrador, 2013]. The recognition of human activities has generally been approached in two categories based on different types of involved sensors, namely ambient-sensor-based and wearable-sensor-based. The former utilizes devices fixed in locations of interest, such as video cameras or WiFi access points attached to a fixed desk or wall. The latter refers to sensors attached to a human body [Chen *et al.*, 2012]. In this work, we focus on wearable-sensor-based activity recognition scenarios since wearable on-body sensors alleviate environment constraints and are free from privacy issues (take cameras as example, all the behaviours of participants are recorded and are easily recognized by others, while signals of wearables are not visible) [Yang *et al.*, 2015]. In these scenarios, the common practice is that the continuous multivariate time series data is partitioned into segments first, and each segment is assigned a specific activity. Feature extraction is commonly conducted on each segment to extract discriminative features. The extracted features are then fed into a classifier or fully-connected layers to recognize different activities [Hammerla *et al.*, 2016].

Existing feature extraction approaches can be classified into two categories: feature-engineer-based and deep-learning-based. The approaches of the former category aim to extract various aspects of information underlying each sensor-reading segment, such as statistical information [Janidarmian *et al.*, 2017], meta information, e.g., overall shape and spatial information [Lara and Labrador, 2013; Hammerla *et al.*, 2013; Lin *et al.*, 2007]. These approaches usually require domain knowledge to manually design proper features for specific applications, which is labor-intensive and time consuming. To overcome the limitations of feature-engineering-based approaches, Qian *et al.* [2018] proposed the SMM$_{AR}$ method to automatically extract all orders of moments as statistical features by using kernel embedding technique of distributions. However, SMM$_{AR}$ fails to extract temporal and spatial information from the segments of sensor readings, which is important for recognizing activities.

The approaches of the latter category aim to design deep neural networks to extract temporal and/or spatial features from the segments of sensor readings automatically [Wang *et al.*, 2017]. Different types of neural networks have been proposed to extract different kinds of information [Hammerla *et al.*, 2016]. Basically, deep feed-forward networks (DNNs) are used to extract higher-level features without taking temporal or spatial information into consideration. Convolutional neural networks (CNNs) are used to extract locally translation invariant features with respect to the precise location or precise time of occurrence of certain pattern within a data segment [Zeng *et al.*, 2014; Yang *et al.*, 2015; Ignatov, 2018]. Recurrent neural networks (RNNs) are suitable for exploiting the temporal dependencies within the activity sequence. The state-of-the-art for sensor-based activity recognition are basically combinations of these three types of base models [Morales and Roggen, 2016]. Though deep-learning-based approaches are able to learn powerful fea-

tures to represent temporal and/or spatial information underlying sensor data, they fail to capture statistical information, such as different orders of statistical moments, which has proven to be useful for activity recognition [Qian *et al.*, 2018; Qian *et al.*, 2019].

In this paper, we propose a novel Distribution-Embedded Deep Neural Network (DDNN) for wearable-sensor-based human activity recognition. The main novelty of our network lies in that we encode the idea of kernel embedding of distributions into a deep architecture, such that besides temporal and spatial information, all orders of statistical moments can be extracted as features to represent each segment of sensor readings, and further used for activity classification in an end-to-end training manner. Compared with the SMM$_{AR}$ method [Qian *et al.*, 2018], which also makes use of the kernel embedding technique to extract statistical features for sensor data, our proposed DDNN is capable of learning more powerful features beyond statistical features. In addition, SMM$_{AR}$ assumes that all activities are segmented beforehand, while DDNN relaxes the perfect-segmentation assumption by simply using sliding windows, which makes DDNN more practical for real-world scenarios. Moreover, SMM$_{AR}$ uses a single kernel to embed distributions, which may be sensitive to the parameter settings of the kernel, while DDNN uses a deep neural network to approximate the feature map of the kernel, which is more flexible as the parameters of the deep neural network are learned from the data.

In summary, our contributions are two-fold:

- Our proposed DDNN is a unified end-to-end trainable deep learning model, which is able to learn different types of powerful features for activity recognition in an automated fashion.
- Extensive experiments are conducted on several benchmark datasets to demonstrate the superior performance of our proposed DDNN.

2 Related Work

Feature-engineering-based machine learning methods. These methods include PCA, LDA, basis transform coding (wavelet transform and Fourier transform) and handcrafted statistical features of raw signals including orders of moments (mean, variance, skewness, etc), median, etc [Janidarmian *et al.*, 2017]. SMM$_{AR}$ method [Qian *et al.*, 2018] automatically extracts all orders of moments as statistical features by using kernel mean embedding technique. Besides statistical features, several methods treat extra meta information as extra structural features. For instance, ECDF method preserves the overall shape and spatial information of time series data [Hammerla *et al.*, 2013]; SAX method transforms continuous data into symbolic representations [Lin *et al.*, 2007].

Deep learning methods. Deep learning methods can substitute for the manual feature design and extraction procedure. The first deep learning method on activity recognition applies Restricted Boltzmann Machines (RBMs) to compare with manual features [Plötz *et al.*, 2011]. CNNs are the most widely used frameworks in this field [Zeng *et al.*, 2014; Yang *et al.*, 2015; Ignatov, 2018]. Yang *et al.*[2015] customized CNNs along temporal dimension of activity data to extract salient patterns of sensor signals at different time scales. Besides, temporal dependencies in time-series data are proven to be beneficial for activity recognition as well. DeepConvLSTM model [Morales and Roggen, 2016] applies two Long Short-Term Memory (LSTMs) layers on top of the abstract feature representations extracted by four convolutional layers. There are also research works to jointly learn shallow features by traditional methods and deep features by deep models [Ravì *et al.*, 2017; Ignatov, 2018]. There are also attempts of combinations of shallow classifiers with deep learned features. Hammerla *et al.* [2016] provided systematic comparisons on the performance of state-of-the-art deep learning methods with DNNs, CNNs and RNNs on activity recognition problems, especially various LSTMs.

Statistical features. Kernel methods have been well studied during the past decades, with the ability to learn nonlinear transformations of input data as implicit features, and of learning nonlinear classifiers as well [Smola *et al.*, 2007]. Recently, Muandet *et al.* [2017] illustrated the power of feature embedding on image classification, and Qian *et al.* [2018] investigated the similar technique on wearable-sensor-based activity recognition, with more reasonable and meaningful explanations on the extracted features. Similar idea has also been applied to the generative adversarial networks (GANs) with a different motivation of matching statistical features to enable the network to generate more realistic synthetic samples [Li *et al.*, 2015; Li *et al.*, 2017].

3 Preliminaries

The idea of extracting infinite number of statistical features may sound counter-intuitive, thus here we briefly introduce the kernel mean embedding technique [Smola *et al.*, 2007; Muandet *et al.*, 2017; Qian *et al.*, 2018].

Given a sample $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ drawn from a probability distribution $\mathbb{P}$, where $\mathbf{X}$ represents a set consisting of n elements $\mathbf{x}_i \in \mathbb{R}^{d\times 1}$. The technique of kernel embedding for representing an arbitrary distribution is to introduce a mean map operation $\mu(\cdot)$ to map instances to a RKHS (Reproducing Kernel Hilbert Space) $\mathcal{H}$, and to compute their mean in the RKHS as $\boldsymbol{\mu}_{\mathbb{P}} := \mu(\mathbb{P}) = \mathbb{E}_{\mathbf{x}\sim\mathbb{P}}[\phi_k(\mathbf{x})] = \mathbb{E}_{\mathbf{x}\sim\mathbb{P}}[k(\mathbf{x},\cdot)]$, where $\phi_k : \mathbb{R}^d \to \mathcal{H}$ is defined by a kernel $k(\cdot,\cdot)$. Here ϕ_k is a feature mapping function that extracts high-dimensional or even infinite-dimensional features from d-dimensional data space to Hilbert space $\mathcal{H}$. If the condition $\mathbb{E}_{\mathbf{x}\sim\mathbb{P}}(k(\mathbf{x},\mathbf{x})) < \infty$ is satisfied, then $\boldsymbol{\mu}_{\mathbb{P}}$ is also an element in $\mathcal{H}$. It has been proven that if the kernel $k(\cdot,\cdot)$ is characteristic, then the mapping $\mu : \mathcal{P} \to \mathcal{H}$ is injective [Sriperumbudur *et al.*, 2009]. The injectivity indicates an arbitrary probability distribution $\mathbb{P}$ is uniquely represented by an element in a RKHS through the mean map. In practice, one can use an unbiased empirical estimation to approximate the mean map $\hat{\boldsymbol{\mu}}_{\mathbb{P}} = \frac{1}{n}\sum_{i=1}^n \phi_k(\mathbf{x}_i) = \frac{1}{n}\sum_{i=1}^n k(\mathbf{x}_i,\cdot)$. Though in theory, the dimension of $\hat{\boldsymbol{\mu}}_{\mathbb{P}}$ is potentially infinite, by using the kernel trick, the inner product of two probability distributions in a RKHS can be computed efficiently through a kernel function associated to the RKHS, $\langle \hat{\boldsymbol{\mu}}_{\mathbb{P}_x}, \hat{\boldsymbol{\mu}}_{\mathbb{P}_z} \rangle = \tilde{k}(\hat{\boldsymbol{\mu}}_{\mathbb{P}_x}, \hat{\boldsymbol{\mu}}_{\mathbb{P}_z}) = \frac{1}{n_x n_z}\sum_{i=1}^{n_x}\sum_{j=1}^{n_z} k(\mathbf{x}_i, \mathbf{z}_j)$, where $\tilde{k}(\cdot,\cdot)$ is a linear kernel defined in the RKHS, n_x and

n_z are the sizes of the samples $\mathbf{X}$ and $\mathbf{Z}$ drawn from $\mathbb{P}_x$ and $\mathbb{P}_z$, respectively. In general, $\tilde{k}(\cdot,\cdot)$ can be a nonlinear kernel defined as $\tilde{k}(\hat{\boldsymbol{\mu}}_{\mathbb{P}_x}, \hat{\boldsymbol{\mu}}_{\mathbb{P}_z}) = \langle \psi(\hat{\boldsymbol{\mu}}_{\mathbb{P}_x}), \psi(\hat{\boldsymbol{\mu}}_{\mathbb{P}_z}) \rangle$, where $\psi(\cdot)$ is the associated feature mapping of the nonlinear kernel $\tilde{k}(\cdot,\cdot)$. These equations indicate that kernels can be applied on top of kernels. It is also possible to extract explicit low-dimensional feature maps $\mathbf{z}$ to substitute for the above implicit features by Random Fourier Features [Rahimi and Recht, 2007]: $k(\mathbf{x},\mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \approx \mathbf{z}(\mathbf{x})^\top \mathbf{z}(\mathbf{x}')$, where the inner product of explicit feature maps can uniformly approximate the kernel values without the kernel trick.

4 The Proposed DDNN Model

4.1 The Overall Model

Activity recognition is challenging as it is affected by many factors, i.e., dynamic spatial-temporal correlations and varying patterns of activities conducted by multiple participants. Based on the above motivation, we design an end-to-end trainable neural network structure for human activity recognition problem. Our proposed model has three main modules to learn feature representations for human activity recognition:

- Statistical module f_1: this module aims to learn all orders of moments statistics as features in an automated fashion.
- Spatial module f_2: this module aims to learn correlations among sensors placements.
- Temporal module f_3: this module aims to learn temporal sequence dependencies along the time scale.

By stacking the above learned features together and forming a unified architecture, we can build a trainable model for activity recognition. The overall illustration of the proposed model is shown in Figure 1.

In our problem setting of activity recognition, the streams of multivariate sensor readings are partitioned by fixed-size sliding window with length L. We randomly split activities into training set $\{(\mathbf{X}_i, y_i)\}_{i=1}^n$, validation set $\{(\mathbf{X}_j, y_j)\}_{j=1}^m$ and test set $\{\mathbf{X}_t\}_{t=1}^p$, where each activity $\mathbf{X}_i = [\mathbf{x}_{i1} \ \ldots \ \mathbf{x}_{iL}] = [\mathbf{x}_i^1 \ \ldots \ \mathbf{x}_i^d]^T \in \mathbb{R}^{d\times L}$, and $y_i \in \{1 \ldots n_c\}$ with n_c denoting the number of predefined activity categories. Here each column $\mathbf{x}_{ij} \in \mathbb{R}^{d\times 1}$ is a vector of signals received from d sensors at j-th timestamp, and each row $(\mathbf{x}_i^r)^T \in \mathbb{R}^{1\times L}$ represents the signals recorded by r-th sensor within the current sliding window.

Note that in this work, we simply concatenate these three modules' learned features $[f_1(\mathbf{X}_i), f_2(\mathbf{X}_i), f_3(\mathbf{X}_i)]$ before feeding into fully-connected layers. However, it is possible to explore more complex and interleaved ways to connect these modules depending on different scenarios. For instance, one possible choice is $f_1([f_2(\mathbf{X}), f_3(\mathbf{X}_i)])$, with which statistical features are learned on top of the features extracted by other two modules. This is actually a generalized way of learning features, i.e., $[f_2(\mathbf{X}), f_3(\mathbf{X})]$ is considered as a special type of data transformation of raw data $\mathbf{X}_i$, while $f_1(\mathbf{X}_i)$ learns features directly from raw data. It is also possible to build a deeper model with these three modules as atom building blocks.

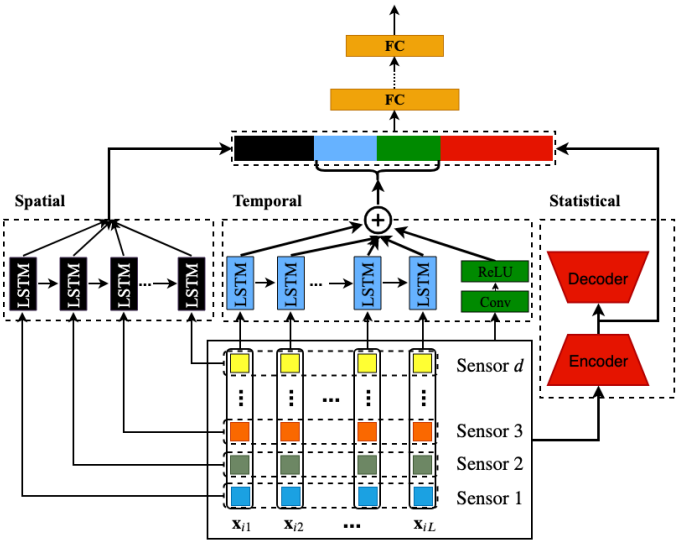

Figure 1: Illustration of the proposed DDNN architecture. The input to the network consists of a data sequence $\mathbf{X}_i = [\mathbf{x}_{i1} \ \ldots \ \mathbf{x}_{iL}] = [\mathbf{x}_i^1 \ \ldots \ \mathbf{x}_i^d]^T \in \mathbb{R}^{d\times L}$ extracted from d sensors and partitioned by sliding window approach with length L. From left to right, there are three modules for extracting spatial, temporal and statistical features respectively. Note that the input data format for these modules are different. Spatial correlations among sensors whose signals are represented as row vectors $\{(\mathbf{x}_i^r)^T\}_{r=1}^d$ are learned by LSTMs. Temporal dependencies are extracted from column vectors $\{\mathbf{x}_i^j\}_{j=1}^L$ by both LSTMs and CNNs (we will explain later why CNNs extract temporal dependencies instead of spatial correlations). Statistical module take the matrix form data $\mathbf{X}_i$ as inputs of autoencoder. All the learned features are then concatenated into a single feature vector, which is input to the fully-connected layers.

4.2 Statistical Module

Inspired by SMM$_{AR}$ [Qian *et al.*, 2018], we aim to learn statistical features automatically by a deep learning model. One disadvantage of SMM$_{AR}$ is that the learned features are limited by a *fixed* Gaussian kernel $k(x,x') = \exp(-\gamma\|x-x'\|^2)$ with *fixed* γ, hence parameter tuning of proper bandwidth for kernel is required in advance. Here we aim to learn statistical features from multiple kernels without manual parameter tuning. This statistical module can be seamlessly combined with other modules to form a unified deep learning architecture which can be trained and optimized.

First, we aim to design a neural network f_1 to learn the statistical feature mapping $\phi_{f_1}(\cdot)$ automatically, i.e.,

$$f_1(\mathbf{X}_i) = \phi_{f_1}(\mathbf{X}_i). \tag{1}$$

However, the desired ϕ_{f_1} takes the matrix as input, while ϕ_k works for vectorial input. To address this issue, we take the average of feature mapping within each sliding window as

$$\phi_{f_1}(\mathbf{X}_i) = \frac{1}{L}\sum_{j=1}^{L} \phi_k(\mathbf{x}_{ij}). \tag{2}$$

Second, we expect f_1 is able to learn the best kernel automatically from different possible characteristic kernels $k \in \mathcal{K}$.

$$f_1^*(\mathbf{X}_i) = \max_{f_1} \phi_{f_1}(\mathbf{X}_i) = \max_{k \in \mathcal{K}} \frac{1}{L} \sum_{j=1}^{L} \phi_k(\mathbf{x}_{ij}). \quad (3)$$

Note that the learned features $f_1^*(\mathbf{X}_i)$ are in vectorial form. As mentioned in Section 3, the prerequisite of expressive feature extraction is the characteristic property of kernels, i.e., the feature mapping $f_1(\cdot)$ should be injective (not necessarily invertible). To make the neural network injective, there should be another function or neural network f_1^{-1} such that $f_1^{-1}(f_1(\mathbf{X}_i)) = \mathbf{X}_i$ for all possible $\mathbf{X}_i$'s. Therefore, as suggested in [Li *et al.*, 2017], we utilize an autoencoder to guarantee the injectivity of the feature mapping.

To be specific, an autoencoder includes an encoder f_e, and a decoder f_d, where the encoder is used to map the input sequence to a fixed-length vector, then the decoder is used to unroll this vector to sequential outputs and try to reconstruct the input data of the encoder. In our scenario, the encoder is the desired f_1 module, and $f_d = f_1^{-1}$. Though both of our proposed model and the model in [Li *et al.*, 2017] utilize an autoencoder to make sure the injectivity of neural networks, the motivations are quite different. We utilize the autoencoder as feature learner for classifying activity classes, while in their model, the autoencoder works for hypothesis testing, i.e., to make generated synthetic samples as indistinguishable from true samples as possible.

The standard loss function of the autoencoder tries to minimize the reconstruction error $\ell_{\text{ae}} = \|x - f_d(f_e(\tilde{x}))\|$ between inputs x and outputs $\tilde{x}$, but it is insufficient for statistical feature learning. We further use an extra loss function based on MMD distance to force the autoencoder to learn good feature representations of inputs:

$$\text{MMD}_k(\mathbf{X}_p, \mathbf{X}_q) = \left\| \frac{1}{n_p} \sum_{i=1}^{n_p} (\phi_k(\mathbf{x}_i)) - \frac{1}{n_q} \sum_{j=1}^{n_q} (\phi_k(\mathbf{x}_j)) \right\|_2$$
$$= \sqrt{\frac{1}{n_p^2} \sum_{i,i'} k(\mathbf{x}_i, \mathbf{x}_{i'}) - \frac{2}{n_p n_q} \sum_{i,j} k(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{n_q^2} \sum_{j,j'} k(\mathbf{x}_j, \mathbf{x}_{j'})}$$

where n_p and n_q are the numbers of timestamps of two activities $\mathbf{X}_p$ and $\mathbf{X}_q$, respectively. The resultant MMD loss function on the autoencoder is as follows:

$$\ell_{\text{MMD}}(\mathbf{X}_i, f_d(f_e(\mathbf{X}_i))) = \frac{1}{L} \left\| \sum_{j=1}^{L} f_e(\mathbf{x}_{ij}) - f_e(f_d(f_e(\mathbf{x}_{ij}))) \right\|_2.$$

Note that by taking f_e and f_d to be the identity function, ℓ_{MMD} is reduced to ℓ_{ae}, where the mean vector (1st order moment) difference between inputs and outputs of the autoencoder is calculated. Our choices for f_e and f_d in the proposed deep learning model aim to match higher order moments statistical features. Therefore, this loss function forces the hidden representations of autoencoder to successfully convey sufficient information of desired statistics to the decoder.

4.3 Spatial Module

Convolutional layers in CNNs are firstly designed for the image-based problems. The standard CNNs are able to extract spatial-invariant features with a kernel filter running over the images or videos. However, the current so-called CNNs for human activity recognition tasks are actually not truly on spatial dependencies. Usually the wearable sensor data is in 1-dimension, where the so-called spatial CNNs are actually along the temporal aspect [Hammerla *et al.*, 2016; Morales and Roggen, 2016]. There are also attempts to force the multiple 1-dimensional data of different sensor channels into a virtual image, then standard CNNs can be applied [Yang *et al.*, 2015].

Our viewpoint of spatial correlations are different from the previous work. We try to capture the spatial correlations between sensors attached to the human body. From our point of view, the signals of a certain sensor are inevitably affected by the attached locations on the human body or joints. Imagine a participant is walking, with sensors attached to his upper arm, lower arm and legs. It is common that when the right leg of the participant is on the front, the right arms are waved to the opposite direction at the same time. Also the movements of upper arm and lower arm are constrained by the joints of the human body. Therefore we aim to model such kinds of spatial correlations of the sensors, which is usually ignored in the literature. As illustrated in Figure 1, the input data $\mathbf{X}_i$ in the sliding window is treated as d row vectors $[\mathbf{x}_i^1 \ ... \ \mathbf{x}_i^d]^T$, each of which associated to a single sensor. A LSTM is connected with each sensor data, and hence the dependencies between sensors are learned to form a spatial feature vector.

4.4 Temporal Module

In order to exploit the temporal dependencies within each activity, we utilize both CNNs and LSTMs as building blocks of temporal module. As discussed in previous subsection, CNNs with 1-D filters are applied on each channel $\{\mathbf{x}_i^r\}_{r=1}^d$ of sensor data $\mathbf{X}_i$. By applying the filter to go through different regions of the input, it is then able to detect the local salience patterns of the signals. Note that CNNs are applied along the temporal dimension, thus it is able to learn the temporal dependencies. Besides, LSTMs are connected to temporal data $\{\mathbf{x}_{ij}\}_{j=1}^L$ to learn temporal information as well. Specifically, we choose LSTMs instead of RNNs due to the diminishing gradients problem. LSTMs are designed to have more dynamic and flexible memory cells through gating mechanism, which enables LSTMs to learn temporal relationships on longer time scales. The outputs of CNNs and LSTMs are concatenated into a single vector to represent temporal features.

5 Experiments

Dataset. We conduct experiments on four sensor-based activity datasets. The overall statistics information of datasets are listed in Table 1. The Daphnet Gait dataset (DG) [Bächlin *et al.*, 2010] corresponds to a medical application and records activities from 10 participants affected with Parkinson's Disease, aiming to detect freezing of gait incidents. The data is segmented by sliding window of 1 second duration and 50% overlap. The Opportunity dataset (OPPOR) [Chavarriaga *et al.*, 2013] comprises 17 mid-level gesture classes conducted in an ambient-sensor home environment together with 19 on-body sensors. These gestures are short in duration and non-

repetitive. Null class data exists in the dataset indicating transitions of two adjacent activities. The UCIHAR dataset [Anguita *et al.*, 2012] collects six activities (walking, walking upstairs, walking downstairs, sitting, standing, laying) carried out with a group of 30 volunteers within an age range of 19-48 years. The PAMAP2 dataset [Reiss and Stricker, 2012] includes 12 different physical activities (household activities and exercise activities) which are performed by 9 subjects wearing 3 inertial measurement units. These activities are prolonged and repetitive, typical for systems aiming to characterize energy expenditure.

5.1 Experimental Setup

All these datasets have class imbalance problem, especially OPPOR and DG. Therefore, in our experiments, we set the probability of an activity being chosen in a training epoch to be the inverse of the number of the certain activity. We follow the experimental setup in [Hammerla *et al.*, 2016]. Micro-F1 (miF) and weighted macro-F1 (maF) are selected as performance measure. 77 out of 113 features are used for OPPOR, with run 2 from subject 1 as validation set, runs 4 and 5 from subject 2 and 3 as test set and the rest as training set. Sliding windows of 1 second duration with 50% overlap is applied. For PAMAP2, 12 protocol activities are studied, with data downsampled to 33Hz. Sliding window length is 5.12 seconds with 1 second as step size. Runs 1 and 2 for subject 5 are used as validation set, and runs 1 and 2 for subject 6 are used as test set, with the rest being training set. The raw data of DG is downsampled to 32Hz as well. Sliding window duration is 1 second with half overlap. We use subject 9's first run as validation set, subject 2's runs 1 and 2 as test set with the rest being training set. The UCIHAR has been preprocessed and segmented by data provider beforehand, where the raw data is randomly partitioned into two sets, where 70% of the volunteers generated training data and 30% the test data. The sensor signals were pre-processed by applying noise filters and then sampled in sliding windows of 2.56 second (128 readings). Data normalization is conducted on all datasets. For our architecture, we utilize 4 linear layers with ReLU attached after each linear layer as encoder and decoder's architecture. Both LSTMs in spatial and temporal module have l layers of LSTMs with h-dimensional hidden representations, where $l \in \{1, 2, 3\}$ and $h \in \{32, 64, 128, 256, 512, 1024\}$. Four convolutional layers with filter size $(1, 5)$ are utilized in the temporal module, with ReLUs and max pooling layers attached after each convolutional layer. All feature vectors are concatenated into a single vector before feeding into three fully-connected layers. The batch size is set to 64, and the maximum training epoch is 100. Adam optimizer is used for training with learning rate 10^{-3} and weight decay 10^{-3}. All experiments are run on a Tesla V100 GPU. [1]

Baselines. We compare our proposed model with baseline methods as well as state-of-the-art methods. Due to the fact that feature-engineering-based machine learning methods are hard to scale up, in this paper we mainly compare our proposed DDNN model with deep learning based methods.

- DDNN$-f_1$: the proposed deep model without the statistical module. This baseline is set to investigate the efficacy of the statistical module.
- DDNN$-f_2$: the proposed deep model without the spatial module. This baseline is set to investigate the efficacy of the spatial module.
- CNN_Yang [Yang *et al.*, 2015]: a state-of-the-art CNN-based model with 3 convolutional layers. We follow the architecture in the paper and reproduce the model.
- DeepConvLSTM [Morales and Roggen, 2016]: a state-of-the-art model with 4 convolutional layers and 2 LSTM layers. We also follow the architecture and reproduce the model.
- DNN: 5-layer linear transformation with ReLU activation function.
- CNN: 4-layer CNNs with kernel size $(1, 5)$ with ReLU activation function and max pooling layer attached to the output of each CNN.
- LSTM: 2-layer LSTMs with the dimension of hidden representation in the range $\{32, 64, 128, 256, 512\}$.
- LSTM-f, LSTM-S, b-LSTM-S: state-of-the-art LSTMs variants to capture temporal sequences information. Results are directly from [Morales and Roggen, 2016].

5.2 Experimental Results and Analysis

The results of the proposed method and baselines on 4 datasets are listed in Table 2. The best performance for each evaluation metric is highlighted in bold. Our proposed DDNN has achieved the best performance on all datasets, except for the maF of OPPOR. These results indicate that our proposed model is capable of learning powerful various features for classification of activity recognition with more discriminative power.

Impact of spatial and statistical module. Remarkably, the performances of DDNN are consistently better than those of DDNN$-f_1$ and DDNN$-f_2$ on all datasets. This favorably validates our motivation that statistical features and spatial features are beneficial to the deep learning models besides the widely used temporal features in existing literature.

Robustness of the proposed DDNN. One interesting finding is that our proposed model is more robust than other baselines. For instance, LSTM-related methods are obviously inferior on DG and UCIHAR, and CNN-based models are much worse than other baselines in OPPOR. One possible reason may lie in the unified framework of DDNN, where different aspects of features are learned together. It is reasonable that the contributions of different features on the classification performance are task-dependent, i.e., the importance of statistical module f_1 and spatial module f_2 varies in datasets since each dataset has unique characteristics on properties.

Parameters' sensitivity. Another aspect of robustness is found during parameter tuning, where DDNN is less sensitive to the changes of parameters. For example, when we set the number of LSTM layers to be $\{1, 2, 3\}$, and LSTMs hidden representation dimensions to be $\{32, 64, 128, 256, 512\}$,

[1] Code of the proposed DDNN is available at https://github.com/Hangwei12358/IJCAI2019_DDNN.

Datasets	# train	# val.	# test	# sw	# Feature	# Class	Frequency	# Subjects
OPPOR	715,785	32,224	121,378	30	113	18	30	4
UCIHAR	941,056	NA	377,216	128	9	6	50	30
DG	312,970	37,122	30,188	32	9	2	100	10
PAMAP2	473,447	90,814	83,366	170	52	12	100	9

Table 1: The overall information of the four datasets. Note that "# train", "# val." and "# test" refer to total number of training, validation and test samples, respectively."#sw" denotes the sliding window length used in the experiments. UCIHAR is preprocessed and segmented beforehand by the data provider, which does not contain validation set.

	DG		OPPOR		UCIHAR		PAMAP2	
Methods	miF	maF	miF	maF	miF	maF	miF	maF
DDNN	**92.59**	**91.61**	**83.66**	86.01	**90.53**	**90.58**	**93.23**	**93.38**
DDNN$-f_1$	91.38	90.67	81.27	84.51	89.96	89.93	87.49	86.84
DDNN$-f_2$	89.67	88.97	77.96	82.27	88.60	88.58	89.37	89.43
CNN_Yang	87.96	86.65	9.98	2.95	88.12	88.11	70.17	70.46
DeepConvLSTM	87.21	84.28	75.47	78.92	89.05	89.07	84.31	82.73
DNN	88.91	86.47	77.05	80.25	87.65	87.72	80.31	79.82
CNN	89.23	88.85	10.66	3.56	86.66	86.77	89.75	89.72
LSTM	88.34	86.93	63.17	69.92	74.52	74.75	90.38	90.29
LSTM-f$*$	67.3	-	67.2	90.8	-	-	92.9	-
LSTM-S$*$	76.0	-	69.8	91.2	-	-	88.2	-
b-LSTM-S$*$	74.1	-	74.5	**92.7**	-	-	86.8	-

Table 2: Overall comparison results on the four datasets (unit: %). Note that the results of baselines with $*$ are directly copied from [Morales and Roggen, 2016].

the performance difference of DDNN is only roughly several percentage, while other models' performance gap is larger. We also investigate the dimensions of hidden representations in the autoencoder of statistical module ranging from $0.5d$ to $10d$ with d indicating the number of dimensions of raw data. Empirically, higher dimensional hidden representations actually hinder the performance of deep model, while the dimensions lower than $4d$ does not affect the performance drastically. We also investigate the weights on the added loss function ℓ_{MMD} for statistical module. We conduct experiments with various weights put on the loss function. As illustrated in Figure 2, the performance is steady (ranging from 0.88 to 0.9) within the weight ranging from 10^{-4} to 10^{1}, but when the weights are larger than 10^{1}, the performance degrades drastically. The reason may be the large weights on the ℓ_{MMD} leads to less contribution of the rest two modules (temporal and spatial), which affects the final performance.

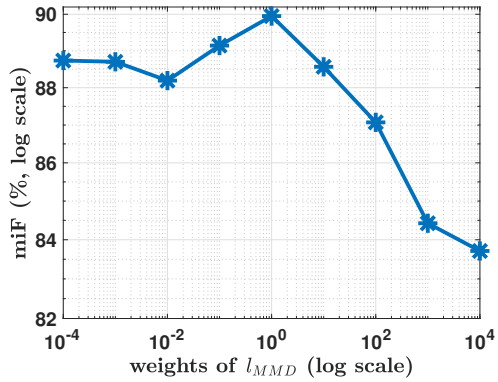

Figure 2: Illustration of performance difference with different weights put on the loss function ℓ_{MMD}.

6 Conclusion and Future Work

In this paper, we propose a novel architecture for wearable-sensor-based activity recognition tasks. Our proposed DDNN model is able to automatically learn three types of features: 1) statistical features, 2) spatial correlations among sensors, and 3) temporal features. Extensive experiments with analysis are conducted to compare with state-of-the-art methods. Experimental results demonstrate the superior efficacy of the proposed model. In the future, we plan to extend DDNN to semi-supervised setting where the number of labeled training data is limited.

Acknowledgments

This research is partially supported by the Singapore Ministry of Health under its National Innovation Challenge on Active and Confident Ageing (NIC Project No. MOH/NIC/HAIG03/2017). Sinno J. Pan thanks the support from the NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020, and Singapore MOE Tier-1 grant 2018-T1-002-143.

References

[Anguita *et al.*, 2012] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz.

Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *IWAAL*, pages 216–223. Springer, 2012.

[Bächlin *et al.*, 2010] Marc Bächlin, Meir Plotnik, Daniel Roggen, Inbal Maidan, Jeffrey M. Hausdorff, Nir Giladi, and Gerhard Tröster. Wearable assistant for parkinson's disease patients with the freezing of gait symptom. *IEEE Trans. Information Technology in Biomedicine*, 14(2):436–446, 2010.

[Bulling *et al.*, 2014] Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv.*, 46(3):33:1–33:33, 2014.

[Chavarriaga *et al.*, 2013] Ricardo Chavarriaga, Hesam Sagha, Alberto Calatroni, Sundara Tejaswi Digumarti, Gerhard Tröster, José del R. Millán, and Daniel Roggen. The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters*, 34(15):2033–2042, 2013.

[Chen *et al.*, 2012] Liming Chen, Jesse Hoey, Chris D. Nugent, Diane J. Cook, and Zhiwen Yu. Sensor-based activity recognition. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 42(6):790–808, 2012.

[Hammerla *et al.*, 2013] Nils Y. Hammerla, Reuben Kirkham, Peter Andras, and Thomas Ploetz. On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution. In *ISWC*, pages 65–68, 2013.

[Hammerla *et al.*, 2016] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. In *IJCAI*, pages 1533–1540. IJCAI/AAAI Press, 2016.

[Ignatov, 2018] Andrey Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Appl. Soft Comput.*, 62:915–922, 2018.

[Janidarmian *et al.*, 2017] Majid Janidarmian, Atena Roshan Fekr, Katarzyna Radecka, and Zeljko Zilic. A comprehensive analysis on wearable acceleration sensors in human activity recognition. *Sensors*, 17(3):529, 2017.

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

[Li *et al.*, 2015] Yujia Li, Kevin Swersky, and Richard S. Zemel. Generative moment matching networks. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1718–1727. JMLR.org, 2015.

[Li *et al.*, 2017] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. MMD GAN: towards deeper understanding of moment matching network. In *NIPS*, pages 2200–2210, 2017.

[Lin *et al.*, 2007] Jessica Lin, Eamonn J. Keogh, Li Wei, and Stefano Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Min. Knowl. Discov.*, 15(2):107–144, 2007.

[Morales and Roggen, 2016] Francisco Javier Ordóñez Morales and Daniel Roggen. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[Muandet *et al.*, 2017] Krikamol Muandet, Kenji Fukumizu, Bharath K. Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends in Machine Learning*, 10(1-2):1–141, 2017.

[Plötz *et al.*, 2011] Thomas Plötz, Nils Y. Hammerla, and Patrick Olivier. Feature learning for activity recognition in ubiquitous computing. In *IJCAI*, pages 1729–1734, 2011.

[Qian *et al.*, 2018] Hangwei Qian, Sinno Jialin Pan, and Chunyan Miao. Sensor-based activity recognition via learning from distributions. In *AAAI*, 2018.

[Qian *et al.*, 2019] Hangwei Qian, Sinno Jialin Pan, and Chunyan Miao. Distribution-based semi-supervised learning for activity recognition. In *AAAI*, 2019.

[Rahimi and Recht, 2007] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.

[Ravì *et al.*, 2017] Daniele Ravì, Charence Wong, Benny Lo, and Guang-Zhong Yang. A deep learning approach to on-node sensor data analytics for mobile or wearable devices. *IEEE J. Biomedical and Health Informatics*, 21(1):56–64, 2017.

[Reiss and Stricker, 2012] Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. In *ISWC*, pages 108–109. IEEE Computer Society, 2012.

[Smola *et al.*, 2007] Alexander J. Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *ALT*, pages 13–31, 2007.

[Sriperumbudur *et al.*, 2009] Bharath K. Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Gert R. G. Lanckriet, and Bernhard Schölkopf. Kernel choice and classifiability for RKHS embeddings of probability distributions. In *NIPS*, pages 1750–1758, 2009.

[Wang *et al.*, 2017] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *CoRR*, abs/1707.03502, 2017.

[Yang *et al.*, 2015] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiaoli Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *IJCAI*, pages 3995–4001. AAAI Press, 2015.

[Zeng *et al.*, 2014] Ming Zeng, Le T. Nguyen, Bo Yu, Ole J. Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *MobiCASE*, pages 197–205. IEEE, 2014.