Convolutional Neural Networks for Human Activity Recognition using Multiple Accelerometer and Gyroscope Sensors

Sojeong Ha, Seungjin Choi
Department of Computer Science and Engineering
Pohang University of Science and Technology
77 Cheongam-ro, Nam-gu, Pohang 790-784, Korea
{hsi0317,seungjin}@postech.ac.kr

***Abstract*—Human activity recognition involves classifying times series data, measured at inertial sensors such as accelerometers or gyroscopes, into one of pre-defined actions. Recently, convolutional neural network (CNN) has established itself as a powerful technique for human activity recognition, where convolution and pooling operations are applied along the temporal dimension of sensor signals. In most of existing work, 1D convolution operation is applied to individual univariate time series, while multi-sensors or multi-modality yield multivariate time series. 2D convolution and pooling operations are applied to multivariate time series, in order to capture local dependency along both temporal and spatial domains for uni-modal data, so that it achieves high performance with less number of parameters compared to 1D operation. However for multi-modal data existing CNNs with 2D operation handle different modalities in the same way, which cause interferences between characteristics from different modalities. In this paper, we present CNNs (CNN-pf and CNN-pff), especially CNN-pff, for multi-modal data. We employ both *partial weight sharing* and *full weight sharing* for our CNN models in such a way that *modality-specific characteristics* as well as *common characteristics* across modalities are learned from multi-modal (or multi-sensor) data and are eventually aggregated in upper layers. Experiments on benchmark datasets demonstrate the high performance of our CNN models, compared to state of the arts methods.**

I. Introduction

Human activity recognition has recently drawn extensive attention in various areas such as pervasive, mobile and context-aware computing [1], [2], [3], [4], due to the advancement of techniques, including smart mobile devices, wireless communication networks, and machine learning. The main ingredient for successful human activity recognition might be in learning *fruitful representations* of sensor signals (which corresponds to feature extraction) in such a way that the characteristics of pre-defined actions are well identified by classifiers in mobile device.

Earlier idea was to use hand-crafted features, which require prior knowledge on signals. Examples are statistical features such as mean, variance, and fast Fourier transform coefficients [5], [6], [7]. Data-driven methods, where features are learned from data without prior knowledge on signals, such as hidden Markov model (HMM) [8], support vector machine (SVM) [9], hidden conditional random fields (HCRF) [10], and convolutional neural network (CNN) [11], [12], [13], [14] are recently considered. Among these methods, CNN, which is one of deep learning methods, has established itself as a powerful technique, since representations learned by CNNs well capture local dependency and scale invariance in signals.

In most of existing work on CNNs for human activity recognition, 1D convolution operation is applied to individual univariate time series to capture local dependency along the temporal dimension of sensor signals [11], [12], [13]. In practice, however, multiple sensors yield multivariate time series, or even single 3-axis accelerometer produces 3-dimensional time series. Thus, it is desirable to consider spatial dependency among multiple sensors or across axes of accelerometers or gyroscopes, as well as local dependency along the temporal dimension. 2D kernel was, recently, employed in [14] to capture local dependency among the temporal dimension as well as across spatial locations. Moreover, 2D kernel reduces model size compared to 1D kernel, which is another important issues of CNN especially in mobile device because of limited memory size.

Here we additionally consider multi-modal sensors, meaning that different type of sensors, which have a potential to capture fruitful representations from various modalities. 2D kernel CNN [14] fits to multi-sensor uni-modal inputs, which can capture both dependencies (spatial and temporal dependencies), so that it achieves high performance with less number of parameters compared to 1D CNN. However for multi-modal data (accelerometer and gyroscope), [14] share weights for whole input signals in convolutional layer (full weights sharing), and extract same features without distinction of modalities, which might cause interferences between characteristics produced by accelerometers and gyroscopes for capturing modality-specific features.

In this paper, we present CNNs (CNN-pf and CNN-pff), especially CNN-pff, where 2D convolution and pooling operations are applied to multi-modal time series data. CNN-pf represents **CNN** models with **p**artial weight sharing in first convolutional layer and **f**ull weight sharing in second convolutional layer. CNN-pff represents **CNN** models with **p**artial and **f**ull weight sharing in first convolutional layer and

978-1-5090-0620-5/16/$31.00

full weight sharing in second convolutional layer. We employ *partial weight sharing* and *full weight sharing* in such a way that *modality-specific characteristics* as well as *common characteristics* across modalities are learned from multi-modal (or multi-sensor) data and are eventually aggregated in the upper layer. The model CNN-pf employs partial weight sharing in the lower layer, where different 2D convolution operation is applied through each modality to capture modality-specific features, followed by full weight sharing in the upper layer to aggregate them. The model CNN-pff employs partial weight sharing in the lower layer, where 2D convolution operation is applied to each modality to capture modality-specific features and at the same time is applied to whole input space to capture common features appearing across modalities, followed by full weight sharing in the upper layer to aggregate them.

The rest of this paper is organized as follows. In Section 2, we review CNN with full and partial weight sharing and previous CNN based human activity recognition approaches. In Section 3, we propose two models (named CNN-pf and CNN-pff) for multi-modal human activity recognition data. In Section 4, by comparing the existing methods (existing CNN models and a few state-of-the-art methods) to CNN-pf and CNN-pff on benchmark dataset [15], we demonstrate the high performance for multi-modal data and smaller model size of our models. Finally, conclusions are drawn in Section 5.

II. Related Work

A. Full weight sharing vs. partial weight sharing

Convolutional neural network (CNN) comprises an input layer, one or more convolutional and pooling layers, followed by one or more fully-connected layers [16]. The convolution operation is applied to either input nodes or hidden nodes in the previous layer, calculating a weighted sum of values of nodes within the convolution kernel (or mask), followed by adding a bias and passing it through an activation function to form a feature map. Full weight sharing enforces local filter weights to be tied and shared for all local subsets on which the convolution operates (see the left model in Fig. 1). On the other hand, partial weight sharing, which was originally developed for speech processing [17], allows local filter weights to be shared for only a particular set of local subsets in the input space. (see the right model in Fig. 1). Thus, multiple sets of local filter weights are used to capture local characteristics which might be different, depending on space or time, while a single set of local filter weights is applied in the case of full weight sharing.

B. CNN based human activity recognition

Recent applications of CNN to human activity recognition [12], [11], [13] employed 1D convolution and pooling operations along the temporal dimension of sensor signals, yielding a feature map k in the lth convolutional layer, denoted by

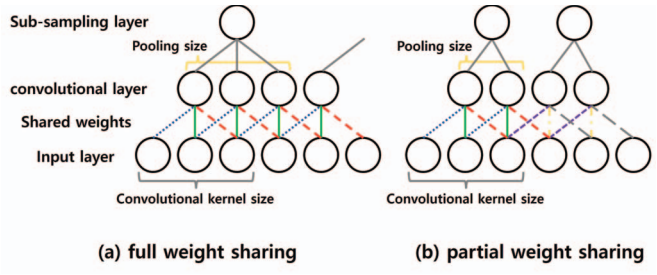


Fig. 1. The concepts of full weight sharing and partial weight sharing in CNN. Same colored lines: same weights

$z_{i,j}^{l,k}$ (i runs over a spatial domain and j runs over a temporal domain), which is computed as

$$z_{i,j}^{l,k} = \sigma\left(\sum_{k'=1}^{K'}\sum_{y=1}^{Y} w_{y,k'}^{l-1,k} z_{i,j+y-1}^{l-1,k'} + b^{l-1,k}\right),$$

where $\sigma(\cdot)$ is the rectified linear function (ReLU) [18], K' is the number of feature maps in the $(l-1)$-th layer, Y is the size of convolution kernel running over the temporal domain of a signal, $w^{l-1,k'} \in \mathbb{R}^{Y\times K'}$ is a local filter weight matrix, and $b^{l-1,k} \in \mathbb{R}$ is a bias. Then, the size of a feature map is reduced by subsampling in a pooling layer, where for instance, 1D max pooling operation takes feature maps in the previous layer as input to produce

$$z_{i,j}^{l,k} = \max_{(j-1)Q+1\le j'\le jQ} z_{i,j'}^{l-1,k},$$

where Q is the size of the pooling kernel.

In [12], full weight sharing was employed, where the same 1D convolution kernel was applied to 6 individual univariate time series, the first three of which were measured at one 3-axis accelerometer and the last three of which were recorded by one gyroscope. In [11], partial weight sharing was employed, where individual 1D convolution kernel was placed on each univariate time series corresponding to a signal of one axis in an accelerometer, and the feature maps were combined in the upper fully-connected layer. Multiple accelerometers signals were considered in [13], but as in [12], the same 1D convolution kernel was applied to each univariate time series produced by 3-axis accelerometers. Fig. 2 illustrates the existing CNN model where the full weight sharing is employed to handle multi-sensor data.

In the case where multiple sensors produce multivariate time series, 2D convolution and pooling operations were shown to be desirable to capture local dependency between sensors as well as local dependency between time steps [14]. However, only full weight sharing was employed in [14], placing the same 2D convolution kernel on sensor signals even from different modalities, which might yields conflicts of characteristics. The 2D convolution operation is performed by calculating a feature map $z_{i,j}^{l,k}$ as

$$z_{i,j}^{l,k} = \sigma\left(\sum_{k'=1}^{K'}\sum_{x=1}^{X}\sum_{y=1}^{Y} w_{x,y,k'}^{l-1,k} z_{i+x-1,j+y-1}^{l-1,k'} + b^{l-1,k}\right), \quad (1)$$

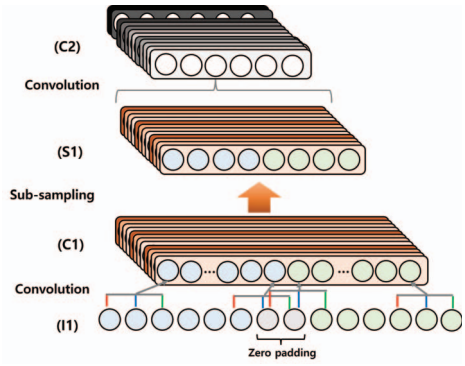


Fig. 2. The input layer reads accelerometer signals (blue nodes) and gyroscope signals (green nodes) with zero padded nodes placed between them. The same convolution kernel (either 1D or 2D) is sliding over whole inputs with weights tied to produce feature maps.

where X and Y are the size of 2D convolution kernel running over space and time, respectively, K' is the number of feature maps in the convolutional layer $(l-1)$, $w^{l-1,k'} \in \mathbb{R}^{X \times Y \times K'}$ is a local filter weight tensor, and $b^{l-1,k} \in \mathbb{R}$ is a bias. The 2D pooling operation produces:

$$z_{i,j}^{l,k} = \max_{\substack{(i-1)P+1 \le i' \le iP, \\ (j-1)Q+1 \le j' \le jQ}} z_{i',j'}^{l-1,k},$$

where P and Q are the size of he pooling kernel running over space and time, respectively.

CNNs are trained by the backpropagation algorithm, where in convolutional layers the chain rule requires the following derivatives for $z_{i,j}^{l,k} \ge 0$ and $z_{i-x+1,j-y+1}^{l,k} \ge 0$, respectively:

$$\begin{aligned} \frac{\partial \sigma(z_{i,j}^{l,k})}{\partial w_{x,y,k'}^{l-1,k}} &= z_{i+x-1,j+y-1}^{l-1,k'}, \\ \frac{\partial \sigma(z_{i-x+1,j-y+1}^{l,k})}{\partial z_{i,j}^{l-1,k'}} &= w_{x,y,k'}^{l-1,k}, \end{aligned}$$

in pooling layers, these derivatives are required to calculate:

$$\frac{\partial z_{i,j}^{l,k}}{\partial z_{i',j'}^{l-1,k}} = \begin{cases} 1, & \text{if } z_{i',j'}^{l-1,k} = \max\limits_{\substack{(i-1)P+1 \le i'' \le iP, \\ (j-1)Q+1 \le j'' \le jQ}} z_{i'',j''}^{l-1,k}, \\ 0, & \text{otherwise.} \end{cases}$$

Softmax function is used in the final layer for classification. The number of nodes in the final layer, denoted by N, is set to equal the number of pre-defined activity classes. The softmax function is applied to each node, producing a value in [0,1], which corresponds to class membership probability:

$$\text{softmax}(z_j) = \frac{\exp\{z_j\}}{\sum_{n=1}^{N} \exp\{z_n\}}, \quad \text{for } j = 1, \ldots, N.$$

III. Proposed Models

In this section we present our CNN models (CNN-pf and CNN-pff), especially CNN-pff, where modality-specific characteristics as well as common characteristics are integrated to handle multivariate time series measured at multiple sensors for human activity recognition. The overall structure and parameters of proposed CNN-pff used in this paper is illustrated in Fig. 3, where in the lower layer partial weight sharing is employed, placing separate convolution kernels on each modality as well as on the whole input space, and in the upper layer full weight sharing is employed to extract non-linear features from aggregated feature maps.

Fig. 5 illustrates the *input batch*, formed by segmenting sensor signals via sliding window, which is fed into the input layer of CNNs. A sliding window of size T is placed on each sensor signal measured by either accelerometers or gyroscopes, yielding a univariate time series of length T. For instance, three accelerometers and one gyroscope produce $3 \times 3 + 3 = 12$ univariate time series. Segmented sensor signals are stacked in row, yielding an input batch of size $C \times T$. The size of row, C, equals $|C_a| + |C_g| + |C_0|$, where $|\cdot|$ denotes the cardinality, C_a denotes the row index set for accelerometer sensor signals, C_g denotes the row index set for gyroscope signals, and C_0 is the index set for zero-padded rows. Row index runs in the order of C_a, C_0, and C_g. The number of zero-padded rows is set to one less than the vertical size of 2D convolution kernel. Zero padding is introduced to avoid interferences between accelerometers and gyroscopes when 2D convolution kernel is applied.

A. CNN-pf

Accelerometers and gyroscopes are heterogeneous sensors, so that these two sensor signals exhibit very different characteristics. In contrast to [14], CNN-pf employs two different convolution kernel sets in the lower layer, each of which is applied to accelerometer signals and gyroscope signals, with weights in each kernel tied (partial weight sharing), to capture modality-specific characteristics (see Fig. 4.(a)). Modality-specific feature maps are aggregated in the upper layer via full weight sharing.

In convolutional layers, we construct multiple feature maps, as in the standard CNN. The $|K_{pa}|$ feature maps are produced by applying the convolution operation to accelerometer signals and the $|K_{pg}|$ feature maps are constructed by applying the convolution operation to gyroscope signals. Note that K_{pa} and K_{pg} are index sets to represent feature maps from accelerometer signals and feature maps from gyroscope signals, respectively. We define the index set I_k to describe 2D convolution operations in the CNN-pf model.

$$I_k = \begin{cases} \{i \,|\, 1 \le i \le |C_a|\} & \text{if } k \in K_{pa}, \\ \{i \,|\, |C_a| + 1 \le i \le |C_a| + |C_g|\} & \text{if } k \in K_{pg}. \end{cases}$$

With this index set I_k, the first convolutional layer (C1) calculates feature maps as:

$$z_{i,j}^{2,k} = \sigma\left(\sum_{x=1}^{X} \sum_{y=1}^{Y} w_{x,y}^{1,k} z_{i+x-1,j+y-1}^{1} + b^{1,k} \right), \text{ for } i \in I_k. \quad (2)$$

Feature maps produced in the convolutional layer (C1) constitute the ones from accelerometer signals and the others

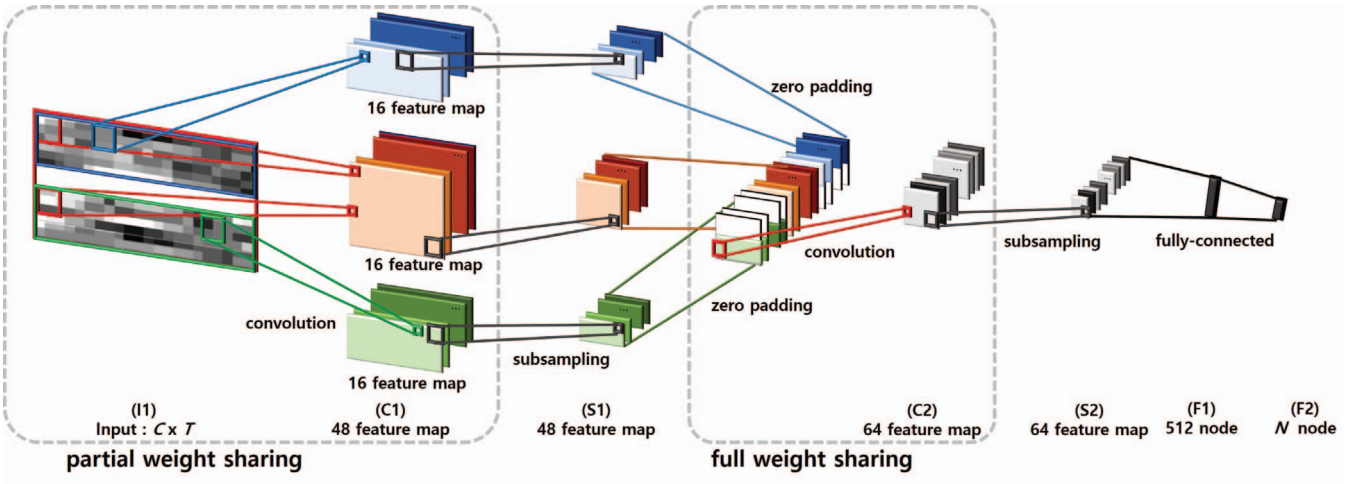
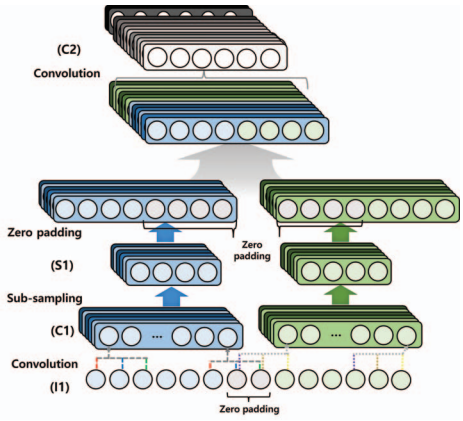
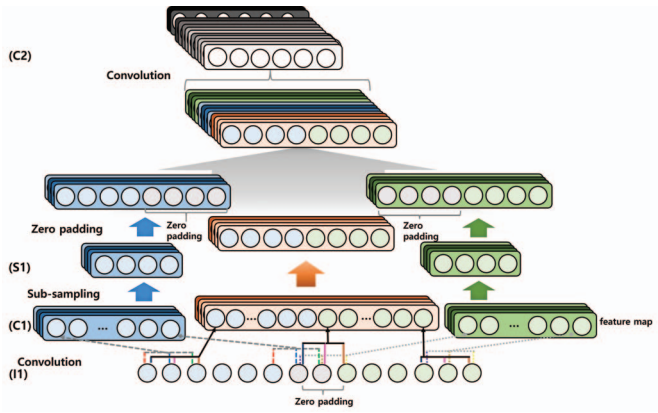


Fig. 3. The input layer places a window on multivariate time series recorded by accelerometers and gyroscopes, yielding M input batches of size $C \times T$, where the upper region corresponds to multivariate time series associated with accelerometer signals, the lower region corresponds to multivariate time series associated with gyroscope signals, and zero-padded rows are placed in the middle. The final layer classifies an input batch into one of N pre-defined actions. The size of convolution kernel in our experiment varies for layers: (I1) 3 x 3, (C1) 2 x 3, (S1) 3 x 5, (C2) 2 x 3.



(a) CNN-pf



(b) CNN-pff

Fig. 4. Our two CNN models are shown, including CNN-pf in the left and CNN-pff in the right. Both models capture modality-specific features in the lower layer, which are integrated in the upper layer. CNN-pff employs three different convolution kernel sets in the lower layer, each of which is applied to mode 1 (accelerometer), mode 2 (gyroscope), and whole input space, while CNN-pf considers only two different convolution kernels, each of which is applied to mode 1 and mode 2, respectively. Feature maps in both CNN-pf and CNN-pff are aggregated in the upper layer, which are followed by the fully-connected layers for classification.

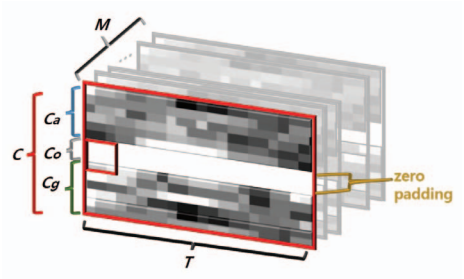


Fig. 5. Input batch of size $C \times T$ is formed by segmenting sensor signals via a sliding window.

from gyroscope signals. The naive aggregation of feature maps may cause conflicts between two different features, since the same positions in all feature maps are combined, regardless of modalities. Thus, a zero padding scheme is introduced, as shown in Fig. 6, such that integrated feature maps preserve features extracted by different modalities.

With the zero padding illustrated in Fig. 6, the first max pooling operation in the layer (S1) produces

$$z_{i,j}^{3,k} = \begin{cases} \max_{\substack{(i-1)P+1 \le i' \le iP, \\ (j-1)Q+1 \le j' \le jQ}} z_{i',j'}^{2,k} & \text{if } i' \in I_k, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

where P and Q represents the height and width of the pooling kernel.

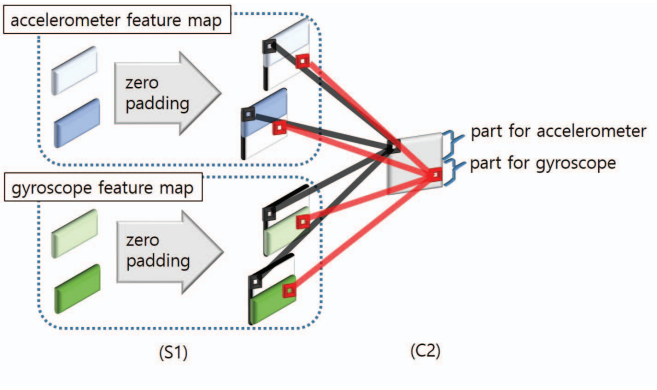


Fig. 6. CNN-pf's resizing feature maps techniques by applying zero padding on feature maps to avoid conflicts between features from two different modalities before they are aggregated in the upper layer.

With $i \in I_k$ for $\forall k$, the backpropagation algorithm using chain rule in convolutional layer (C1) requires the following derivatives for $z^{2,k}_{i,j} \geq 0$ and $z^{2,k}_{i-x+1,j-y+1} \geq 0$, respectively.

$$\frac{\partial \sigma(z^{2,k}_{i,j})}{\partial w^{1,k}_{x,y}} = z^{1}_{i+x-1,j+y-1}, \tag{4}$$

$$\frac{\partial \sigma(z^{2,k}_{i-x+1,j-y+1})}{\partial z^{1}_{i,j}} = w^{1,k}_{x,y}, \tag{5}$$

and in the layer (S1) with I_k requires

$$\frac{\partial z^{3,k}_{i,j}}{\partial z^{2,k}_{i',j'}} = \begin{cases} 1 \quad \text{if } z^{2,k}_{i',j'} = \max\limits_{\substack{(i-1)P+1 \leq i'' \leq iP, \\ (j-1)Q+1 \leq j'' \leq jQ}} z^{2,k}_{i'',j''}, \\ \qquad \& \; i' \in I_k \\ 0 \quad \text{otherwise.} \end{cases} \tag{6}$$

B. CNN-pff

The CNN-pf model extracts modality-specific features in the lower layer, as described in the previous section. However, two different modalities might share some common characteristics, unless they are completely orthogonal to each other. Earlier work with different models emphasized the importance of common characteristics as well as modality-specific features [19], [20], [21]. The CNN-pff model extracts modality-specific features as well as common features across modalities in the lower layer, employing the partial weight sharing where three different convolution kernel sets are applied to each modality as well as the whole input space (see Fig. 4). Three different multiple feature maps are constructed from accelerometer signals, gyroscope signals, and the whole input (accelerometer + gyroscope). We denote by K_{ca}, K_{cg}, and K_{cb} the index sets for feature maps constructed from accelerometer signals, gyroscope signals, and the whole input, respectively. To apply 2D convolution to the input batch, we define the index set I_k as

$$I_k = \begin{cases} \{i | 1 \leq i \leq |C_a|\}, & \text{if } k \in K_{ca}, \\ \{i | |C_a| + 1 \leq i \leq |C_a| + |C_g|\} & \text{if } k \in K_{cg}, \\ \{i | 1 \leq i \leq |C_a| + |C_g|\} & \text{if } k \in K_{cb}. \end{cases} \tag{7}$$

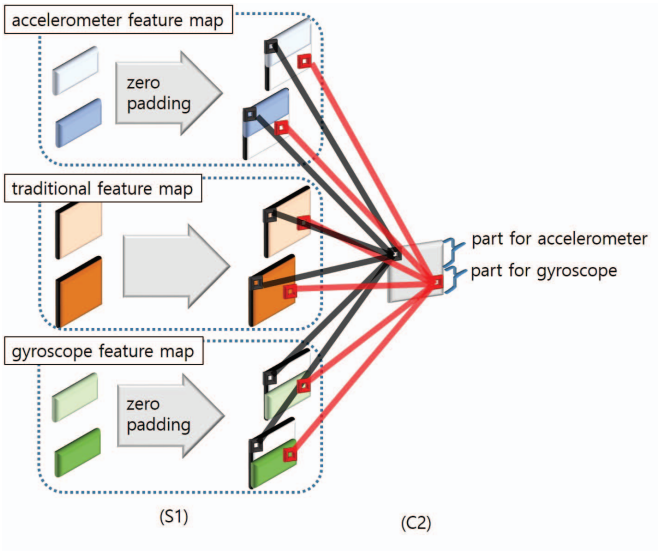


Fig. 7. CNN-pff's resizing feature maps techniques by applying zero padding on feature maps to avoid conflicts between features from different modalities and whole input space, before they are aggregated in the upper layer.

With this index set I_k, calculations in the layer (C1) are the same as (2). Zero-padding to avoid conflicts between different modalities, as illustrated in Fig. 7, is applied, before the convolution operation is carried out in the layer (C2). Gradient computations required for the backpropagation algorithm are the same as (4), (5), and (6), with the index set I_k in (7).

IV. Experiments

In experiments, we use the rectified linear units (ReLU) [18] as an activation function, which is defined by

$$\sigma(x) = \max(0, x),$$

where its gradient is computed as

$$\sigma'(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x \leq 0. \end{cases}$$

We also apply the dropout technique with probablity 0.5 on fully-connected layer for training our CNN models, then combine all learned weights as suggested in [22].

A. Dataset Description

Mhealth dataset [15] is about the 12 daily activities (Standing still, sitting and relaxing, lying down, walking, climbing stairs, waist bends forward, frontal elevation of arms, knees bending, cycling, jogging, running, jumping front and back) from 10 subjects. Activities are recorded with a sampling frequency of 50Hz from 4 different types of sensors: three 3-axis accelerometer sensors (placed on chest, right wrist, left ankle), two 3-axis gyroscope sensors (placed on right wrist, left ankle), three 3-axis magnetometer sensors (placed on chest, right wrist, left ankle), and one 2-lead Electrocardiogram sensor (placed on chest). We use three accelerometers (chest, right wrist, left ankle) and two gyroscopes (right wrist, left

TABLE I
CLASSIFICATION ACCURACY COMPARISON ON MHEALTH BENCHMARK DATASETS AMONG VARIOUS METHODS. THE HIGHEST CLASSIFICATION ACCURACY PER EACH SUBJECT IS MARKED IN BOLD AND THE SECOND HIGHEST CLASSIFICATION ACCURACY IS MARKED IN UNDERLINED.

Method	classification accuracy (%)										
	subject1	subject2	subject3	subject4	subject5	subject6	subject7	subject8	subject9	subject10	total
HMM	61.30	63.58	68.24	71.92	59.28	70.40	69.07	70.02	64.15	67.27	66.62
SVM	58.79	54.50	77.37	76.06	76.05	63.92	54.31	50.64	76.22	66.18	65.40
HCRF	62.86	74.04	81.07	67.10	73.36	70.83	54.22	49.91	81.99	70.61	68.60
1D CNN	81.99	86.29	**86.83**	**85.01**	85.98	88.92	**99.29**	95.29	98.85	**95.75**	90.43
2D CNN	84.68	80.81	83.99	84.07	87.98	87.97	89.87	96.34	**100**	95.66	89.14
CNN-pf	**87.19**	**87.66**	84.24	84.08	96.68	**90.44**	89.69	97.81	**100**	95.48	91.33
CNN-pff	85.45	87.06	83.48	82.95	**97.04**	89.77	98.76	**99.27**	**100**	95.66	**91.94**

ankle) sensors in our experiment. We segment data using a sliding window with a size of 60 samples with 50% overlap over time domain, which is sufficient to classify above 12 classes. In several changes of subject for test about 1100 windows (from one subject), and 10000 windows (from nine subjects except for test subject) are used to test and train respectively.

B. Comparison with Existing Methods

We compare our models with state-of-the-art methods which include:

- **Hidden Markov model (HMM)** [23]: Segmented signals using sliding windows are used as the input of HMM. We compare the results of each one-vs-all HMM models having 5 gaussian mixtures and 15 states, and select the highest scoring model.
- **Support vector machine (SVM)** [24]: The segmented raw signals are converted into input vectors. Input vectors are used as the input of SVM. One-vs-all SVM model is used through each activity class, and highest scoring model is selected.
- **Hidden conditional random fields (HCRF)** [10]: The segmented signals using sliding windows are used as the input of HCRF. We compare the results of each one-vs-all HCRF model, and select the highest scoring model. Each HCRF model is trained using three hidden states.
- **Convolutional neural networks with 2D kernel (2D CNN)** [14]: Details are explained in related work. Here, we consider CNN using 2D convolution kernel and 2D pooling kernel. We use seven-layer CNN described in Fig. 3 with full weight sharing.
- **Convolutional neural networks with 1D kernel (1D CNN)** [12] : Details are explained in related work. Here, we consider CNN using 1D convolution kernel and 1D pooling kernel with full weight sharing. We use seven-layer CNN described in Fig. 3, where only the kernel sizes (1 x n, n: same with $2D$ kernel size over time) are different.

C. The Number of Trainable Weights

CNN with 2D kernel in convolution and subsampling layer has much less number of trainable weights than 1D kernel case, where both CNN models have same structures (the number of layers, nodes in fully connected layers and kernels in convolutional layer and so on) except for different kernel size.

In our experiments, CNN with 1D kernel has 4445964 trainable parameter where size of input in (I1):15 x 60, convolution kernel in (C1): 48 x [1x3], pooling kernel in (S1): [1x3], convolution kernel in (C2): 64 x [1x5], pooling kernel in (S2): [1x3], the number of nodes in (F1): 512, the number of nodes in (F2): 12 (the number of activity class). Whereas CNN with 2D kernel has 938048 trainable parameter where input size with zero padding in (I1): 17 x 60, convolution kernel in (C1): 48 x [3x3], pooling kernel in (S1): [2x3], convolution kernel in (C2): 64 x [3x5], pooling kernel in (S2): [2x3], and the number of nodes in fully connected layers (F1, F2) are same as CNN with 1D kernel. Note that our proposed methods (CNN-pf, CNN-pff) have same number of trainable weights with standard 2D CNN.

After experiments of both CNN with above parameters, we find that 1D CNN's number of trainable weights is about 4.7 times the number of trainable weights of 2D CNN and our proposed methods.

D. Results

The final results of two proposed methods (CNN-pf, CNN-pff) and the state-of-the-art methods on mhealth benchmark datasets are represented in Table I. Our proposed methods have the better classification accuracy than the other comparable methods. Especially CNN-pff has the highest performance. By considering both experimental results and the number of trainable weights, proposed methods, which have much less number of parameters than 1D CNN, better classify multi-modal activity classes than existing CNN methods with 1D kernel.

E. CNN-pff for multi-modal data

We simulate the situation in which both modality-specific features and common features across modalities are needed to classify activities, so that CNN-pff have benefits compared to CNN with 2D kernel and CNN-pf. With a smart device which has 1 tri-axial accelerometer sensor and 1 tri-axial gyroscope sensor, we collect sensor signals for three different tasks: 1) keep motionless, 2) spin at the same spot, 3) move horizontally through floor. We apply 2D CNN, CNN-pf, and CNN-pff classifiers on these collected data, then compare

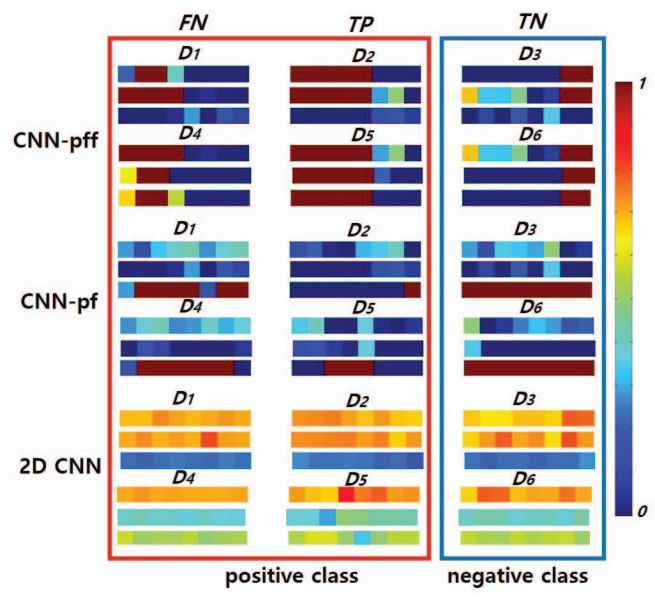


Fig. 8. (S2) layer's output (most discriminative top three feature maps) of 6 example signals in IV-E. Positive class: moving horizontally, negative class: spinning at the same spot. Naming of TP (true positive), TN (true negative), and F N (false negative) is based on classification results of 2D CNN and CNN-pf.

the performance. We set the number of nodes in (F1) as 256 and (F2) as 3 (the number of class), and the other parameters are same as Fig. 3. The classification accuracy of 2D CNN, CNN-pf is 79.73%, 66.55% respectively and CNN-pff's classification accuracy is 99.66%.

We compare the most discriminant top three feature maps in (S2) layer of six samples in Fig. 8 to visualize CNN-pff model's effectiveness for above data. We assume spinning at the same spot as negative class, and moving horizontally through floor as positive class. We choose six samples which are false negative (D_1, D_4), true positive (D_2, D_5), true negative (D_3, D_6) samples of 2D CNN and CNN-pf, and represent the most discriminant top three feature maps of (S2) layer through each classifiers. Note that all samples are correctly classified by CNN-pff. Classified results of CNN-pf and 2D CNN on six samples are same. The CNN-pff's feature maps of D_1 and D_4 are similar to true positive samples feature maps than feature maps of false positive samples. Where we use CNN-pf or 2D CNN classifier, although we select the most discriminant top three feature maps of (D_1, D_4), these feature maps are hardly similar to true positive samples' feature maps than true negative samples' feature maps.

V. Conclusions

In this paper we have presented CNNs (CNN-pf, CNN-pff), especially CNN-pff, for human activity recognition to handle multivariate time series data measured at multiple heterogeneous sensors. Partial and full weight sharing in the lower layers captured modality-specific characteristics as well as common characteristics across modalities respectively. After applying full and partial weight sharing, feature maps were resized by padding zeros, followed by aggregation of feature maps via full weight sharing in the upper layers. Experiments on benchmark datasets have showed the high classification accuracy of our CNNs, compared to existing CNNs and state-of-the-art methods. We have demonstrated that our CNNs achieve the high performance with much smaller number of parameters compared to CNNs with 1D kernels, and handle multi-modal data more efficiently compared to existing CNNs with 2D kernels.

Acknowledgments

This work was supported by the IT R&D Program of MSIP/IITP (B0101-15-0307, Machine Learning Center), National Research Foundation (NRF) of Korea (NRF-2013R1A2A2A01067464), and Samsung Electronics.

References

[1] T. Chouhury, S. Consolvo, B. Harrison, J. Hightower, A. LaMarca, L. LeGrand, A. Rahimi, A. Rea, G. Borriello, B. Hemingway, P. Klasnja, K. Koscher, J. A. Landay, J. Lester, D. Wyatt, and D. Haehnel, "The mobile sensing platform: An embedded activity recognition system," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 32–41, 2008.

[2] C. R. Wren and E. M. Tapia, "Toward scalable activity recognition for sensor networks," in *Locationa and Context Awareness LNCS*, vol. 3987. Springer, 2006, pp. 168–185.

[3] K. Van Laerhoven and K. Aidoo, "Teaching context to applications," *Personal and Ubiquitous Computing*, vol. 5, pp. 46–49, 2001.

[4] M. Stikic and B. Schiele, "Activity recognition from sparsely labeled data using multi-instance learning," in *Location and Context Awareness LNCS*, vol. 5561. Springer, 2009, pp. 156–173.

[5] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive Computing LNCS*, vol. 3001. Springer, 2004, pp. 1–17.

[6] T. Plötz, H. Y. Hammerla, and P. Olivier, "Feature learning for activity recognition in uniquitous computing," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Barcelona, Spain, 2011.

[7] A. Krause, D. P. Siewiorek, A. Smailagic, and J. Farringdon, "Unsupervised, dynamic identification of physiological and activity context in wearable computing," in *Proceedings of the IEEE International Symposium on Wearable Computers (ISWC)*, 2003.

[8] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden Markov model," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1992.

[9] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The jigsaw continuous sensing engine for mobile phone applications," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, Zurich, Switzerland, 2010.

[10] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell, "Hidden conditional random fields for gesture recognition," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, New York, NY, USA, 2006.

[11] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, and J. Zhan, "Convolutional neural networks for human activity recognition using mobile sensors," in *Proceedings of the Sixth International Conference on Mobile Computing, Applications and Services (MobiCASE 2014)*, Austin, Texas, USA, 2014.

[12] S. Duffner, S. Berlemont, G. Lefebvre, and C. Garcia, "3D gesture classification with convolutional neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Florence, Italy, 2014.

[13] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Buenos Aires, Argentina, 2015.

[14] S. Ha, J.-M. Yun, and S. Choi, "Multi-modal convolutional neural networks for activity recognition," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Hong Kong, 2015.

[15] O. Banos, R. Garcia, J. A. Holgado-Terriza, M. Damas, H. Pomares, I. Rojas, A. Saez, and C. Villalonga, "mHealthDroid: A novel framework for agile development of mobile health applications," in *Proceedings of the International Work-Conference on Ambient Assisted Living (IWAAL)*, 2014.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, 2012.

[18] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the International Conference on Machine Learning (ICML)*, Haifa, Israel, 2010.

[19] H. Lee and S. Choi, "Group nonnegative matrix factorization for EEG classification," in *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, Clearwater Beach, Florida, 2009.

[20] M. Salzmann, C. H. Ek, R. Urtasun, and T. Darrell, "Factorized orthogonal latent spaces," in *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, Sardinia, Italy, 2010.

[21] Y. Kang and S. Choi, "Restricted deep belief networks for multi-view learning," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, Athens, Greece, 2011.

[22] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, 2013.

[23] K. Murphy, "Hidden Markov model (HMM) toolbox for Matlab," 1998, available at http://www. ai. mit. edu/˜ murphyk/Software/HMM/hmm. html.

[24] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.