VersaTL: Versatile Transfer Learning for IMU-based Activity Recognition using Convolutional Neural Networks

Mubarak G. Abdu-Aguye[1] and Walid Gomaa[1,2]
[1]*Computer Science and Engineering Department, Egypt-Japan University of Science and Technology, Egypt*
[2]*Faculty of Engineering, Alexandria University, Egypt*

Keywords: Activity Recognition, Transfer Learning, IMU, Convolutional Neural Networks, Deep Learning.

Abstract: The advent of Deep Learning has, together with massive gains in predictive accuracy, made it possible to reuse knowledge learnt from solving one problem in solving related problems. This is described as Transfer Learning, and has seen wide adoption especially in computer vision problems, where Convolutional Neural Networks have shown great flexibility and performance. On the other hand, transfer learning for sequences or timeseries data is typically made possible through the use of recurrent neural networks, which are difficult to train and prone to overfitting. In this work we present *VersaTL*, a novel approach to transfer learning for fixed and variable-length activity recognition timeseries data. We train a Convolutional Neural Network and use its convolutional filters as a feature extractor, then subsequently train a feedforward neural network as a classifier over the extracted features for other datasets. Our experiments on five different activity recognition datasets show the promise of this method, yielding results typically within 5% of trained-from-scratch networks while obtaining between a 24-52x reduction in the training time.

1 INTRODUCTION

Deep neural networks entered the mainstream with the seminal work of Hinton and Salakhutdinov (Hinton and Salakhutdinov, 2006). Since then, the general trend in machine learning research has been towards the adoption and analysis of deep methods as applied to different domains. This can be mainly attributed to the massive performance improvements gained through the use of deep neural networks (Jordan and Mitchell, 2015), with such methods defining the state of the art in several diverse applications today. Additionally, deep methods have, in some sense, refined and justified previous intuition and hypotheses regarding neural learning. Perhaps the most specific case of this is the notion of hierarchical representation as put forward in (Hinton and Salakhutdinov, 2006), (Bengio, 2012), where (deep) neural networks are believed to learn features of the data in an incremental, compositional manner. Some work done in visualizing the learned features (Zeiler and Fergus, 2014) and activation maps (Harley, 2015) in image-based Convolutional Neural Networks provide definitive evidence of this phenomenon.

A direct implication of this sort of hierarchical learning is that it enables the sharing or re-use of more general, lower-level knowledge gained from solving one problem in the solution of another problem sharing the same or similar lower-level bases/primitives. For instance, the problem of recognizing two different objects will have similar requirements at a lower level, e.g., recognizing boundaries or edges of the objects, up until some point where the visual characteristics are sufficiently different from the two objects (e.g. boxes and birds have radically different appearances when considered in their entirety, even though they are both composed of lines and curves). Therefore, it becomes theoretically possible to re-use this lower level knowledge in solving similar problems, therefore saving time, resources, and requiring much less training data than training a deep network from scratch. This is described as *transfer learning* (Pan et al., 2010) and is very commonly used in computer vision tasks (Oquab et al., 2014),(Karpathy et al., 2014).

For other types of data whose structures are not trivial to decompose intuitively, transfer learning has not gained much traction for varying reasons. Difficulties in dealing with varying data lengths, as opposed to standard approaches to same in image-based problems may be one cause of this. For sequence/time-series data, approaches to trans-

Abdu-Aguye, M. and Gomaa, W.
VersaTL: Versatile Transfer Learning for IMU-based Activity Recognition using Convolutional Neural Networks.
DOI: 10.5220/0007916705070516
In *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2019)*, pages 507-516
ISBN: 978-989-758-380-3

fer learning have naturally adopted recurrent architectures like Long Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) due to their sequence-modelling abilities and their ability to generate a fixed-size representation of their input data regardless of its length. However, such networks require significant effort to be properly initialized and trained (Salehinejad et al., 2018) and are often prone to overfitting (Zaremba et al., 2014). On the other hand, Convolutional Neural Networks have much more desirable properties, i.e., they are much more flexible and robust, and can match recurrent models even on sequential data (Yin et al., 2017), (Bai et al., 2018). Therefore transfer learning approaches relying on convolutional neural networks can be considered as viable alternatives to recurrent neural networks.

In this work we propose *VersaTL*, a transfer learning scheme for activity recognition data based on convolutional neural networks. This is aimed at relaxing the requirement for large amounts of training data usually required for training deep models from scratch. Additionally, this serves to significantly reduce the time required for the training process for both small and large datasets with minimal loss in predictive performance.We consider a fixed number of dimensions (i.e., number of axes/channels) for our data. In order to deal with varying input lengths, we incorporate a 1-D variant of Spatial Pyramid Pooling (He et al., 2014) so as to obtain a fixed length feature vector from the convolutional feature maps in our network. We evaluate our proposed technique on five different activity recognition datasets, as we consider that there are common patterns in the data collected during different activities regardless of the source of the data or the activity itself. We achieve promising results comparable to trained-from-scratch networks using this scheme with expectedly much lower training time. Additionally, we show that our method competes favorably against similar, extant methods.

The rest of this paper is structured as follows. In Section 2 we discuss other literature that are relevant to the works carried out in this paper. We provide a brief description of key concepts relevant to this work in Section 3. Section 4 describes the methodology proposed by *VersaTL* in terms of its implementation and considerations. In Section 5 we describe the datasets considered as well as the manner in which our proposed technique was tested. Section 6 gives the results of the experiments described in Section 5, as well as their consequent evaluation. In Section 7 we discuss additional experiments and their results, which further clarify our arguments and provide comparative information between our work and other work already in literature. We conclude the paper in Section 8 with a summary of our work and the outcomes we obtained, together with points for future consideration.

2 RELATED WORK

Here we discuss other works in literature which apply transfer learning to activity recognition. We group them into two main categories based on the adopted paradigm - *instance-based* methods, which aim to adapt samples from one domain for reuse in another domain, and *feature-based* methods, which aim to transfer feature-extraction knowledge from the source domain to the target domain.

2.1 Instance-based Transfer Learning

The work done in (Hu et al., 2011) is aimed towards the adaptation of activity instances collected in the performance of some task in the training of a classifier for another set of instances. They propose an algorithm to do this, which functions by utilizing pre-existing web data describing the two activity sets. Based on the web data, a similarity framework is built within which the instances for some activity in the source domain may be mapped to some other activity in the target domain with some confidence value. They achieve results comparable to traditional methods highlighting the effectiveness of their technique.

The authors in (Khan and Roy, 2017) proposed an instance-based transfer learning framework, where labelled samples from one set of activities (i.e source activity set/domain) are re-used/boosted as samples for matching/common activities in another activity set, based on a transferability metric. K-Means clustering is then used to derive ”anomalous” clusters of the unseen/uncommon activities in the target activity set. A classifier is then trained on samples from the common and uncommon activities, the output of which governs how new instances are classified. Specifically, new samples are either matched using a classifier trained on the boosted samples or matched using the anomalous clusters previously defined, depending on if the discriminatory classifier classifies the sample as belonging to a common or uncommon activity. The authors consider three datasets and report up to 85% recall in inter-dataset testing, where the target dataset contains previously unseen activities.

In contrast, the work carried out in this paper is geared towards feature-based transfer learning,

which we believe is more flexible and does not depend on domain interrelationships or the availability of suitable samples from the source domain.

2.2 Feature-based Transfer Learning

In (Khan and Roy, 2018), the authors propose the transfer of feature-extraction knowledge from one dataset to another (i.e source to target). They trained a Deep Sparse Autoencoder (DSAE) on pre-extracted feature vectors and use a part of the encoder portion of the trained network as a feature extractor. Three Support Vector Machine classifiers are then trained, one on features extracted from source-domain samples using the DSAE, another on features extracted from the target domain using the DSAE and the last trained on low-level features also extracted from the target domain. Decision fusion techniques are then used to combine the outputs of the SVMs to yield a final decision. The authors also compare their method to two state-of-the-art transfer-learning based classifiers and report better performance than both of them.

In (Chikhaoui et al., 2018), a transfer learning scheme utilizing a convolutional neural network (CNN) is proposed. The authors pretrain a CNN-based classifier, and transfer the learned weights to a new, similarly-structured network for the target domain. Further training on the target domain data is then carried out with a view to fine-tuning the new network, without changing/updating the transferred weights. The authors investigate the performance of their method on three datasets and investigate its efficacy across different users, device placements and sensor modalities. They report F1-scores of up to 0.94 in some of the evaluated scenarios.

As compared to (Khan and Roy, 2018), VersaTL *does not make any assumptions about activity interrelationships between the source and target domains as it aims for generality. Also, it obviates the need for any feature pre-extraction due to the use of the convolutional frontend, permitting the use of the raw timeseries directly. Furthermore, it does not impose any requirements or strictures on the classification methodology.*

Additionally, VersaTL *differs from (Chikhaoui et al., 2018) which requires the use of samples with fixed lengths. We allow for both fixed and varying-length samples, which are a reality in the domain of activity recognition. We also design our network to be more compact. As a result, our network is simpler as it has much fewer weights and therefore requires much less training data. This permits the use of small datasets as we demonstrate in the following sections.*

3 BACKGROUND

3.1 Spatial Pyramid Pooling

Spatial Pyramid Pooling (SPP) was first introduced by He et al. in (He et al., 2014). It can be considered as a pooling method for yielding fixed-length outputs from feature maps in convolutional neural networks, which traditionally rely on fixed-size (image) inputs. These fixed-size inputs are usually obtained by cropping or resizing the inputs, which affect network accuracy on varying-size input (images) due to accidentally-induced geometric deformations. By eliminating this requirement, spatial pyramid pooling makes it possible to train convolutional networks on varying-size inputs, which was also found to boost the predictive performance of such networks since the features learned become scale and deformation-invariant and can therefore be considered to be robust.

Traditional pooling methods such as max and average pooling use fixed-length pooling windows. As a result the size of their outputs depend on the size of their input feature maps. SPP computes the size of its pooling windows *adaptively* based on the size of the input feature maps. Specifically, for a given pooling size s, SPP computes a pooling window size and the necessary amount of padding *so as to divide the input feature map(s) into s regions*. Subsequently, it performs a pooling operation (max or average) over each (spatial) region, generating one feature per region. In this manner it generates the specified number (as given by s) of output features per feature map.

In order to gain the invariance described previously, SPP is typically performed for different pooling sizes. In practice this involves the division of the input feature maps into the number of regions specified by each pooling size, and yields one feature per region. Therefore, the total number of features generated is the sum of the features generated per pooling size considered. In (He et al., 2014), a pooling size of 1 was always considered in addition to other pooling sizes. SPP with a pooling parameter/size of 1 can be considered to be a global pooling operation i.e., it performs pooling over the entire input feature map, yielding a single feature. In this way it can be seen that global pooling operations are particular cases of SPP where the pooling parameters/sizes are set to 1. Thus the use of SPP (and including 1 in the pooling sizes to be considered) encompasses global pooling and extends it to arbitrary scales as well. This allows for the maintenance of multiscale information obtained from the input implicitly, preserving local and global characteristics of the input regardless of its size.

Without a loss of generality, SPP can also be ap-

plied to 1-D inputs such as timeseries data. In this scenario the adaptive pooling window itself becomes 1-dimensional and its length is computed in terms of the length of the sequence. The input series is then padded (if necessary) and subdivided into chunks as described previously, and then the pooling operation (max or average) is performed over each chunk, yielding one feature per chunk.

In the current work SPP parameters/sizes are described using a hyphenated notation, e.g., $a-b-c$. This notation implies that the input sample is divided into a regions, and then region-wise pooling is performed yielding a features. This is repeated for size b, yielding b features and size c yielding c features. Therefore for SPP parameters $a-b-c$, the total number of features computed per input feature map is $a+b+c$, e.g., $4-2-1$ yields $4+2+1=7$ features per feature map.

4 PROPOSED METHODOLOGY

The architecture of *VersaTL* is shown in Figure 1. We begin with a convolutional neural network, consisting of a feature extraction portion based on two convolutional layers followed by a classification portion based on a three-layer feedforward neural network. We use a dilated convolution in the second filter, which allows for a wider receptive field for convolutional filters while utilizing fewer weights than a regular convolutional window. The output of the convolutional filters depends on the length of the input data. This is not directly suitable for the classification portion, which requires a fixed-length feature vector as input. To work around this limitation, we include a spatial pyramid pooling layer (He et al., 2014) which converts the varying-length filter outputs into a fixed length vector. The use of spatial pyramid pooling is influenced by its ability to summarize the generated feature maps over large and small (time) scales, thereby preserving local and global characteristics of the extracted features in a compact manner. As such, we adopt this type of pooling rather than a global average or max-pooling so as to maintain some of the temporal information/variations of the activity signals while keeping the network size small. We use the hyperbolic tangent ("tanh") activation function in the convolutional filters as we found it to yield superior performance relative to the other activation functions. Additionally, we include batch normalization layers in between the feedforward layers in the classification portion. This was found to significantly improve the network performance and convergence time.

Given a parent dataset, we train the model in an end-to-end manner, i.e., we train both the feature extraction and classification portions as a single unit using cross entropy loss and the Adam optimizer. After this training, we consider the feature extraction portion to be suitably discriminative for feature extraction purposes. For any other dataset, the feature extraction portion is used as a black box to generate fixed-length feature vectors describing the input. The generated vectors can then be used with any other classifier, although in this work we focus on using a feedforward network tailored to the number of classes in the (new) dataset for classification. The feedforward network is then trained with some portion of the generated vectors (collected in mini-batches) used as training examples, and the remaining data is used to evaluate the performance of the proposed method. This approach has been adopted in this work as it is virtually identical to the transfer learning scheme used in computer vision tasks, which has shown itself to yield excellent results.

Details of our experimental evaluations are provided in the following section.

5 EXPERIMENTAL METHODOLOGY

5.1 Datasets Considered

In order to obtain performance estimates for our proposed method, we consider five publicly-available activity recognition datasets. The datasets chosen have different sample counts and lengths and generally consist of different activity sets. In addition, they generally come from different devices with different characteristics (e.g. placement, make and model of the device, etc.). Although the datasets are comprised of different modalities (e.g accelerometer, gyroscope, magnetometer, orientation, etc.) we consider the accelerometer and gyroscope readings only. This is because these readings are available across all the datasets and therefore allow for straightforward transfer learning from one dataset to another.

5.1.1 Gomaa-1 Dataset

The Gomaa-1 dataset (Gomaa et al., 2017) consists of 603 varying-length samples collected from three subjects, using an Apple SmartWatch series 1 at a sampling rate of 50Hz. It contains 14 activities of daily living.

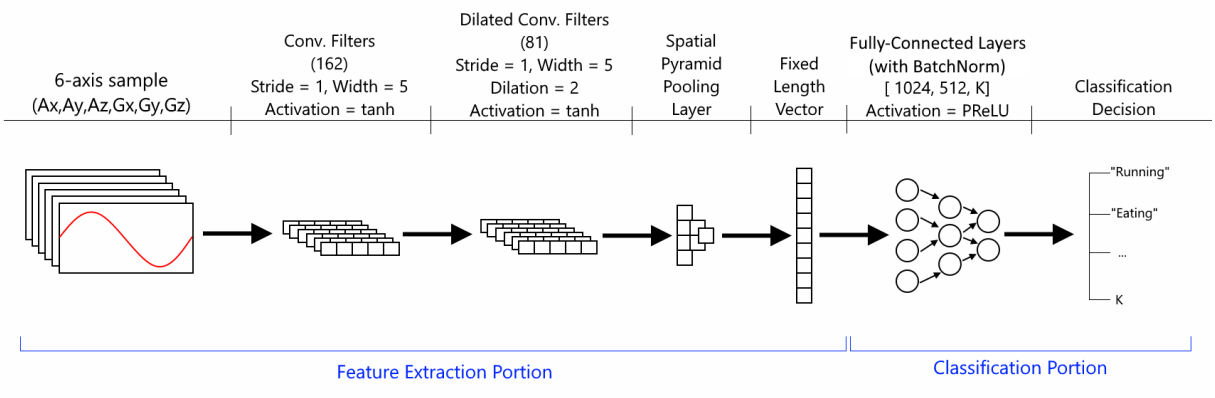

Figure 1: Proposed System Architecture.

5.1.2 Human Activity and Postural Transitions (HAPT)

The human activity and postural transitions (HAPT) dataset (Reyes-Ortiz et al., 2016) consists of 1,214 varying-length samples collected from thirty subjects, using Android smartphones at a sampling rate of 50Hz. It contains 6 activities and 6 postural transitions.

5.1.3 Daily and Sports Activities

The Daily and Sports activities (Altun et al., 2010) dataset consists of 9,120 fixed-length samples collected from 8 subjects. The samples were recorded using Xsens IMU units at a sample rate of 25Hz. It contains 19 activities.

5.1.4 HAD-AW

The HAD-AW dataset (Mohammed et al., 2018) consists of 4,344 varying-length samples collected from 16 subjects. The data was obtained from an Apple SmartWatch series I at a sampling rate of 50Hz, similar to the Gomaa-1 dataset. It contains 32 widely-varying activities.

5.1.5 Realistic Sensor Displacement

The Realistic Sensor Displacement (REALDISP) dataset (Baños et al., 2012), (Banos et al., 2014) consists of samples collected from 17 subjects. Xsens IMU units were used to capture the data at a sampling rate of 50Hz. It consists of samples of 33 different activities recorded while the data capture device was displaced in one of three ways, which was done to capture real-world placement scenarios.

Originally consisting of 4000+ varying-length samples, 1,397 of these were found to be usable after preprocessing and the subsequent elimination of indeterminate/unlabelled samples. Therefore, we consider only these samples for our evaluations.

Table 1: Details of Datasets Considered.

Name	Samples	Activities
Gomaa-1	603	14
HAPT	1214	12
Daily_Sports	9120	19
HAD-AW	4344	32
REALDISP	1397	33

The details of these datasets are summarized in Table 1.

5.2 Experimental Evaluations

To determine the efficacy of our technique, we train on each of the described datasets, and then perform transfer learning (as described in section 4) on each of the other four datasets. Therefore we report the results from both self-testing (i.e., train and test on same dataset) scenarios, which serve as performance baselines, and transfer learning (i.e., train CNN on some dataset, fine-tune classifier on some other dataset) scenarios. We vary the parameters of the spatial pyramid pooling layer (which changes the fixed-length feature vector size) to investigate their effect on the performance of our method. In the transfer learning scenarios, since we use a feedforward network as classifier, we also tested different common mini-batch sizes to find a suitable value. A mini-batch size of 128 was found to be best and was used for the evaluations of all the pooling parameters considered. We use learning-rate scheduling and fix the number of training epochs in both self-testing and transfer learning scenarios at 35 epochs.

The training and testing cycles are repeated fifteen (15) times, with different samples of data be-

Table 2: Classification Accuracy using 2-1 Pooling.

Training Dataset	Testing Dataset				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	94.76 ± 1.67	94.95 ± 1.30	91.69 ± 0.88	81.59 ± 1.22	70.66 ± 1.68
HAPT	93.86 ± 2.09	95.76 ± 0.93	91.04 ± 0.54	82.04 ± 1.45	67.82 ± 2.17
D_Sports	95.18 ± 1.82	94.60 ± 1.05	94.99 ± 0.47	82.32 ± 0.92	67.65 ± 2.35
HAD-AW	93.33 ± 2.22	95.43 ± 1.45	92.29 ± 0.53	87.29 ± 1.25	68.66 ± 2.55
REALDISP	93.55 ± 1.04	95.59 ± 0.96	92.75 ± 0.52	82.27 ± 1.01	74.78 ± 2.24

Table 3: Classification Accuracy using 4-2-1 Pooling.

Training Dataset	Testing Dataset				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	96.99 ± 1.28	94.53 ± 1.55	92.19 ± 0.64	82.66 ± 1.01	71.94 ± 2.64
HAPT	95.09 ± 1.90	96.00 ± 1.18	91.06 ± 0.76	83.10 ± 1.12	68.85 ± 1.77
D_Sports	95.98 ± 1.48	95.24 ± 1.09	95.31 ± 0.36	83.20 ± 1.15	71.14 ± 3.70
HAD-AW	95.93 ± 1.55	95.61 ± 1.18	92.38 ± 0.50	88.36 ± 1.18	71.63 ± 2.15
REALDISP	95.58 ± 1.18	94.82 ± 1.55	92.59 ± 0.41	83.60 ± 1.02	76.60 ± 1.88

Table 4: Classification Accuracy using 8-4-2-1 Pooling.

Training Dataset	Testing Dataset				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	95.58 ± 1.76	94.86 ± 1.13	92.04 ± 0.85	82.97 ± 1.16	72.85 ± 1.75
HAPT	94.83 ± 1.75	96.05 ± 1.30	90.40 ± 0.79	83.14 ± 1.22	69.52 ± 2.68
D_Sports	95.09 ± 1.79	95.15 ± 1.11	95.09 ± 0.33	83.22 ± 1.39	73.92 ± 2.25
HAD-AW	96.86 ± 1.20	95.10 ± 1.21	91.89 ± 0.77	88.65 ± 1.24	73.27 ± 2.52
REALDISP	95.62 ± 0.63	95.41 ± 1.18	92.35 ± 0.67	84.36 ± 1.05	77.40 ± 2.28

Table 5: Training Times (in Seconds) for Self Testing and Transfer Learning Scenarios (4-2-1 Pooling).

Training Dataset	Testing Dataset				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	426.10 ± 5.95	16.44 ± 0.50	62.30 ± 1.37	49.70 ± 1.18	27.07 ± 0.95
HAPT	9.91 ± 0.29	513.33 ± 4.50	62.02 ± 0.94	49.44 ± 0.90	27.02 ± 0.65
D_Sports	9.98 ± 0.40	16.30 ± 0.40	1503.26 ± 13.38	50.20 ± 1.35	27.08 ± 0.75
HAD-AW	9.92 ± 0.35	16.32 ± 0.54	63.12 ± 1.63	1952.15 ± 16.19	27.16 ± 0.63
REALDISP	9.98 ± 0.31	16.33 ± 0.45	62.57 ± 1.42	49.84 ± 1.26	1413.45 ± 16.13

ing used for training and testing each time. 75% of the data is used for training in both self-testing and transfer learning scenarios, with the remaining 25% being used for testing. We also carry out some timing evaluations in order to investigate the computational speed of our proposed method relative to training from scratch. Specifically we measure the training time in the self-testing scenarios and the sum of the feature extraction and training times in the transfer learning scenarios, with the latter measurements reflecting the typical use case. The timings were also obtained over fifteen trials. The machine used to obtain these timings utilized an 8-core Intel Xeon Platinum 8168 CPU with 16GB of RAM. During the training cycles, CPU-only training was used to give a fair estimate of performance and eliminate any extraneous advantages due to the use of graphics processing units (GPUs). We thereafter report the mean and standard deviation of the obtained times measured in seconds.

The results obtained and their subsequent discussion are provided in the following section.

6 RESULTS AND DISCUSSION

The results obtained from our experimental evaluations are shown in Tables 2-4. The diagonal elements (highlighted in gray) indicate the self-testing scenarios (training and testing over the same dataset) and can therefore be considered to be the baseline performance estimates for each dataset. The off-diagonal, column-wise elements show the transfer learning re-

sults, i.e. the results obtained from testing on the column-specified dataset after training on the row-specified dataset. The classification accuracies obtained are shown together with their standard deviations in the tables (i.e $accuracy \pm std.deviation$). As stated previously, these values were obtained from averaging the results over fifteen experiments/trials.

In general, it can be seen that the results obtained from transfer learning are averagely within 5% of the results obtained from self-testing scenarios. The standard deviation of the results are also generally low, remaining below 3% in all the scenarios considered. This indicates the stability/robustness of the proposed approach, and implies that transfer learning across the datasets is effective and comparable to training from scratch. It can subsequently be inferred that the feature extraction portion of the network is capable of extracting highly discriminative features from the parent dataset, and that these features are common to activity data in general, regardless of the particular type of activity or the device used in collecting the data. This is also observed to hold true regardless of the size of the parent dataset - the Gomaa-1, HAPT and REALDISP datasets are comparatively small compared to the Daily_Sports and HAD-AW datasets. However, the performance of our proposed method is maintained even when these small datasets are used as the parent datasets.

Additionally, it can be observed that the HAD-AW and REALDISP datasets consistently have the highest transfer learning results. This can be attributed to the fact that they consist of a large and diverse range of activities, causing the convolutional filters to be relatively more discriminative than when trained on other datasets. The HAPT dataset, having the smallest number of activities, accordingly shows the poorest transfer learning results in all scenarios. This makes a case for the use of class-diverse datasets as source datasets for transfer learning. We examine this notion further in section 7.

It can also be observed that the spatial pyramid pooling size has an almost negligible impact on the performance of the method. In particular, it has a very slight effect on the self-testing results. However, the performance obtained in the transfer learning scenarios shows noticeable degradation when the smallest pooling size (2-1, yielding 2+1 features per feature map) is used. This can be expected since the dimensionality of the feature vector is the smallest compared to the other sizes. The virtually identical performance of the 8-4-2-1 (yielding 8+4+2+1=15 features per feature map) and 4-2-1 (yielding 4+2+1=7 features per feature map) pooling levels suggests that activity recognition signals display medium to long-term trends. That is, the additional local summarization offered by the 8-4-2-1 pooling level compared to the 4-2-1 pooling level does not offer information that is (very) beneficial to the discrimination of activities. This makes a case for the use of the 4-2-1 pooling level as a suitable pooling setting in the proposed scheme.

The average time required for self-testing and transfer learning scenarios (when using 4-2-1 Pooling) is shown in Table 5. We do not evaluate the timings for all the tested pooling parameters because we consider the resulting differences to be negligible. As can be observed, the self-training scenarios take relatively long times, due to the multitude of operations inherent in training the convolutional layers. The Daily_Sports dataset also shows a lower training time in the self-training scenarios than the HAD-AW and REALDISP datasets (which are smaller). This can be attributed to the fact even though the Daily_Sports dataset has more samples, the samples in the HAD-AW and REALDISP datasets are likely much longer (i.e have more timesteps) than the samples in the Daily_Sports dataset. As a result, the striding of the convolutional filters for such samples takes significantly longer than for the samples in the Daily_Sports dataset. The transfer learning scenarios take significantly smaller amounts of time, yielding a 24.17x speedup compared to the self-testing scenario in the worst case (i.e on the Daily Sports dataset) and a 52.31x speedup in the best case (i.e on the REALDISP dataset). It can also be observed that the timings for the transfer-learning scenarios are fairly steady, regardless of the dataset which was used to (pre)train the model. This implies that the transfer learning times are purely dependent on the size of the fine-tuning/testing dataset. With the use of GPUs, the transfer learning times may possibly be reduced even further due to the speed benefits obtainable from the use of such devices.

7 ADDITIONAL CONSIDERATIONS

In this section we briefly describe relevant additional experiments carried out and the results obtained thereof.

Table 6: Classification Accuracy - Feature Extraction Portion + 1 Fully-Connected Layer and 4-2-1 Pooling.

Training Dataset	**Testing Dataset**				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	96.37 ± 1.08	93.00 ± 1.34	90.09 ± 0.77	81.31 ± 0.93	67.75 ± 2.04
HAPT	92.75 ± 1.73	96.31 ± 1.03	88.60 ± 0.80	78.45 ± 1.01	59.21 ± 3.38
D_Sports	91.87 ± 1.54	92.85 ± 1.24	95.04 ± 0.54	78.87 ± 1.43	62.03 ± 1.91
HAD-AW	95.05 ± 1.25	94.60 ± 0.86	91.72 ± 0.55	88.34 ± 0.49	69.65 ± 1.58
REALDISP	93.90 ± 2.23	93.50 ± 1.41	90.91 ± 0.55	81.63 ± 1.17	76.01 ± 2.86

Table 7: Classification Accuracy - Feature Extraction Portion + 2 Fully-Connected Layers and 4-2-1 Pooling.

Training Dataset	**Testing Dataset**				
	Gomaa-1	HAPT	D_Sports	HAD-AW	REALDISP
Gomaa-1	96.15 ± 1.24	88.72 ± 2.15	78.61 ± 1.14	60.50 ± 1.46	48.01 ± 2.49
HAPT	64.94 ± 4.57	96.18 ± 1.11	72.58 ± 1.35	47.12 ± 2.45	36.99 ± 3.98
D_Sports	57.26 ± 4.47	75.85 ± 2.53	95.08 ± 0.47	53.09 ± 2.28	38.99 ± 3.08
HAD-AW	85.96 ± 1.81	87.74 ± 2.08	85.00 ± 1.10	88.31 ± 0.75	54.80 ± 3.37
REALDISP	72.62 ± 4.78	82.17 ± 3.54	80.69 ± 1.40	60.61 ± 1.61	75.00 ± 2.27

Table 8: Comparative Results between (Chikhaoui et al., 2018) and our proposed technique.

Experiment	**Baseline F-Metric**	**Obtained F-Metric**
MobiAct $\rightarrow$ RealWorld (Chest)	0.496	0.903
MobiAct $\rightarrow$ RealWorld (Head)	0.796	0.832
MobiAct $\rightarrow$ RealWorld (Forearm)	0.796	0.809
RealWorld (Chest) $\rightarrow$ MobiAct	0.568	0.747
RealWorld (Head) $\rightarrow$ MobiAct	0.658	0.751
RealWorld (Forearm) $\rightarrow$ MobiAct	0.658	0.740

7.1 Effect of Layers and Dataset Diversity on Transfer Learning Performance

In order to justify the use of only the convolutional filters as the feature extraction portion, we carry out two additional experiments. In the first of these, we include one of the fully-connected layers in the feature extraction portion, then train only the two remaining fully-connected layers. In the second, we include two of the fully-connected layers in the feature extraction portion, then train only the last fully-connected layer i.e the classifier layer. In both experiments, we fix the pooling sizes at 4-2-1. We aimed to determine the effect of including the fully-connected layers in the feature extraction portion i.e considering higher-level features instead of lower-level ones. Additionally, we also aimed to provide some insight into the behavior of the discriminative powers of the convolutional filters when trained on different datasets.

The results obtained from the first and second experiments are shown in Tables 6 and 7 respectively. As expected, a degradation in performance is observed in both schemes, with the higher degradation being shown in the second experiment i.e training only the last layer. This can be explained by the fact that the features existing at successively-deeper layers are higher-level (i.e more dataset-specific) and therefore transfer poorly to other datasets. This makes a case for the use of only the convolutional filters for transfer learning as they learn lower-level (i.e more generic) features.

It is also pertinent to note that in both experiments, the transfer learning performance when trained on the HAD-AW and REALDISP datasets remains the best relative to the other datasets. This is more apparent in Table 6, where both datasets clearly show superior performance to the others in the scenarios considered. As described earlier, these datasets contain a large and heterogenous number of activities, which causes the features (i.e both low and high-level) learnt at successive layers to be significantly more discriminatory than features learnt from other, less-diverse datasets.

Although the Daily_Sports dataset contains 19 activities and 9000+ samples, it is generally outperformed by the much-smaller (14 activities, 603 samples) Gomaa-1 dataset. This occurs because the Gomaa-1 dataset, in spite of its smaller size and ac-

tivity count, contains a wider variety of activities, while the Daily_Sports dataset has much less variety in its constituent activities. As a result, it can also be observed from the tables that the Gomaa-1 dataset comes after the HAD-AW and REALDISP datasets in transfer learning performance. The HAPT dataset contains the fewest activities with a low variety/heterogeneity. As such, it shows the highest degradation in the resulting transfer learning performance of all the considered datasets, as can be observed from Tables 6 and 7. This is in consonance with the behavior observed and discussed in Section 6 previously.

We believe that the results obtained from these experiments further highlight the benefit of training such transfer learning models on activity-diverse datasets.

7.2 Comparison to Similar Transfer Learning Schemes

In order to clearly illustrate the benefit of *VersaTL*, we also compare it to (Chikhaoui et al., 2018), which is the most similar method to it. Due to constraints on space and challenges with one of the considered datasets, we consider six evaluations from their work which are concerned with transfer-learning performance between devices placed at three different locations on the body. We segment the two datasets concerned (MobiAct (Chatzaki et al., 2016) and RealWorld HAR (Sztyler and Stuckenschmidt, 2016)) as described in (Chikhaoui et al., 2018) and derive the average F-metric obtained from the scenarios they investigated. The results of these experiments are reported on a normalized scale and shown in Table 8.

The baseline results (i.e from (Chikhaoui et al., 2018)) are shown in the "Baseline F-Metric" column. It can be seen that the results obtained from our method (displayed in the "Obtained F-Metric" column) shows an improvement of up to 40% in the F-metric. This illustrates the superiority of *VersaTL* relative to the scheme proposed in (Chikhaoui et al., 2018).

8 CONCLUSION

In this work we propose *VersaTL*, a transfer learning scheme for activity recognition data using convolutional neural networks (CNNs). We design a small CNN and include a spatial pyramid pooling layer to allow for inputs of any type (i.e fixed or varying-length). We then train the network (initially for classification) using standard methods. Subsequently, we use the convolutional filters and the pooling layer as a feature extractor. The feature extractor is then reused for other datasets (different than the one used to train the network initially) and dataset-specific feedforward neural networks are then trained to classify the generated feature vectors.

The results obtained from our evaluations indicate that *VersaTL* yields comparable performance (within 5%) to CNN-based classifiers trained from scratch, while requiring a fraction of the training time. This highlights the utility of our transfer learning technique in this domain. We believe that this could pave the way for the widespread use of pretrained models in activity recognition, similar to the way pretrained models are used in computer vision, e.g., AlexNet, VGGNet, etc.

In the future we intend to investigate the generalizability of *VersaTL* to other types of timeseries data which share the same number of axes/channels. The restriction on the number of axes/channels is required due to the convolutional filter-based frontend of our method. Going further we may also investigate specific training of the feature extraction portion of the network with a view to optimizing the learnt features for discriminability.

REFERENCES

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

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Baños, O., Damas, M., Pomares, H., Rojas, I., Tóth, M. A., and Amft, O. (2012). A benchmark dataset to evaluate sensor displacement in activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 1026–1035. ACM.

Banos, O., Toth, M. A., Damas, M., Pomares, H., and Rojas, I. (2014). Dealing with the effects of sensor displacement in wearable activity recognition. *Sensors*, 14(6):9995–10023.

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 17–36.

Chatzaki, C., Pediaditis, M., Vavoulas, G., and Tsiknakis, M. (2016). Human daily activity and fall recognition using a smartphone's acceleration sensor. In *International Conference on Information and Communication Technologies for Ageing Well and e-Health*, pages 100–118. Springer.

Chikhaoui, B., Gouineau, F., and Sotir, M. (2018). A cnn based transfer learning model for automatic activity recognition from accelerometer sensors. In

International Conference on Machine Learning and Data Mining in Pattern Recognition, pages 302–315. Springer.

Gomaa, W., Elbasiony, R., and Ashry, S. (2017). Adl classification based on autocorrelation function of inertial signals. In *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*, pages 833–837. IEEE.

Harley, A. W. (2015). An interactive node-link visualization of convolutional neural networks. In *International Symposium on Visual Computing*, pages 867–877. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

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

Hu, D. H., Zheng, V. W., and Yang, Q. (2011). Cross-domain activity recognition via transfer learning. *Pervasive and Mobile Computing*, 7(3):344–358.

Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732.

Khan, M. A. A. H. and Roy, N. (2017). Transact: Transfer learning enabled activity recognition. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 545–550.

Khan, M. A. A. H. and Roy, N. (2018). Untran: Recognizing unseen activities with unlabeled data using transfer learning. In *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 37–47.

Mohammed, S. A., Elbasiony, R., and Gomaa, W. (2018). An lstm-based descriptor for human activities recognition using IMU sensors. In *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2018 - Volume 1, Porto, Portugal, July 29-31, 2018.*, pages 504–511.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724.

Pan, S. J., Yang, Q., et al. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Reyes-Ortiz, J.-L., Oneto, L., Samà, A., Parra, X., and Anguita, D. (2016). Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767.

Salehinejad, H., Baarbe, J., Sankar, S., Barfett, J., Colak, E., and Valaee, S. (2018). Recent advances in recurrent neural networks. *CoRR*, abs/1801.01078.

Sztyler, T. and Stuckenschmidt, H. (2016). On-body localization of wearable devices: An investigation of position-aware activity recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE Computer Society. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7456521.

Yin, W., Kann, K., Yu, M., and Schütze, H. (2017). Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*.

Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.