Transfer Learning Across Human Activities Using a Cascade Neural Network Architecture

Xin Du
Department of Electronics and Computer Science
University of Southampton
Southampton, UK
xd3y15@soton.ac.uk

Katayoun Farrahi
Department of Electronics and Computer Science
University of Southampton
Southampton, UK
K.Farrahi@soton.ac.uk

Mahesan Niranjan
Department of Electronics and Computer Science
University of Southampton
Southampton, UK
mn@ecs.soton.ac.uk

ABSTRACT

Cascade Learning (CL) [20] is a new adaptive approach to train deep neural networks. It is particularly suited to transfer learning, as learning is achieved in a layer-wise fashion, enabling the transfer of selected layers to optimize the quality of transferred features. In the domain of Human Activity Recognition (HAR), where the consideration of resource consumption is critical, CL is of particular interest as it has demonstrated the ability to achieve significant reductions in computational and memory costs with negligible performance loss. In this paper, we evaluate the use of CL and compare it to end to end (E2E) learning in various transfer learning experiments, all applied to HAR. We consider transfer learning across objectives, for example opening the door features transferred to opening the dishwasher. We additionally consider transfer across sensor locations on the body, as well as across datasets. Over all of our experiments, we find that CL achieves state of the art performance for transfer learning in comparison to previously published work, improving F_1 scores by over 15%. In comparison to E2E learning, CL performs similarly considering F_1 scores, with the additional advantage of requiring fewer parameters. Finally, the overall results considering HAR classification performance and memory requirements demonstrate that CL is a good approach for transfer learning.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Transfer learning**; • **Computer systems organization** → **Neural networks**.

KEYWORDS

Human Activity Recognition; Cascade Learning; Transfer Learning; Deep Learning

ACM Reference Format:
Xin Du, Katayoun Farrahi, and Mahesan Niranjan. 2019. Transfer Learning Across Human Activities Using a Cascade Neural Network Architecture. In *Proceedings of the 2019 International Symposium on Wearable Computers (ISWC '19), September 9–13, 2019, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3341163.3347730

1 INTRODUCTION

Recognizing human activities from sensor based measurements is a challenging and useful problem in machine learning with a wide range of potential applications, particularly related to personalized healthcare. Interest in this topic has grown significantly in recent years with increasing availability of cheap wearable sensors integrated into everyday devices such as smart phones. Remote monitoring of elderly in homes [16] and early diagnosis of complex diseases [21] are examples of the use of activity recognition.

Neural networks are state-of-the art techniques for solving pattern classification and regression problems. Recent surge in research activity on this topic, focusing specifically on deep architectures, has resulted in impressive advances in a number of topics such as computer vision [15], machine translation and audio processing [7]. While much of the impressive development seen in recent years is empirical, increasing the depth of networks is thought to enable the extraction of invariant features that help accurate inference. Specifically with respect to convolutional networks [33], it is possible to

ISWC '19, September 9–13, 2019, London, United Kingdom

ACM ISBN 978-1-4503-6870-4/19/09...$15.00
https://doi.org/10.1145/3341163.3347730

show early layers extracting recognizable low level image features, and later layers modelling more abstract representations.

While in much of neural network practice, the architecture of the network is either fixed or tuned as a hyper-parameter by cross validation, several authors have considered adaptive architectures. These are either constructive architectures, (*i.e.* start from a small network and grow in complexity) as in the case of Platt's Resource Allocation Network [10, 14, 25], or achieve architecture adaptation by pruning, by starting from a large network and gradually removing weights or nodes whose contribution to performance is minimal. Le Cun et al.'s Optimum Brain Damage [19] is an example of the latter. In this context, Fahlman et al.'s Cascade Correlation approach [10], where a multi-layer perceptron model is grown in architecture, is of specific interest in this paper. Deep CL [20] is an approach, inspired by Cascade Correlation, and is designed to train deep neural networks in a layer-wise fashion (discussed in more detail later), in which significant reductions in computational and memory costs were shown to be achieved at negligible loss of performance. Similar results are shown by Belilovsky et al. [1] on the ImageNet task. Layer-wise training of Restricted Boltzmann Machines (RBM) has been considered in [2].

Deep neural networks have also been applied to HAR by several authors [12, 30] taking advantage of their ability at carrying out feature extraction and classification simultaneously. As with a number of other problems, extracting relevant features by automatical training is seen as an advantage over the use of hand-crafted features as used in [3, 18, 24]. The most popular Deep Learning approaches applied in HAR include Multi-Layer Perceptrons (MLPs), Convolutional Neural Networks (CNNs) [23], Recurrent Neural Networks [9], and Long Short Term Memory networks (LSTMs) [11]. Although the above approaches show state of the art generalization performance, the computational complexity and memory requirements of CNNs is noted to be generally higher than the corresponding feedforward MLPs used [5, 13]. We note that HAR in real-world applications may require low computational cost solutions due to the necessity to integrate them on wearable devices. This is one motivation for the pursuit of the CL architecture pursued in this work.

Cascade Learning

As mentioned before, deep CL, inspired by Fahlman et al.'s cascade correlation approach [10] is a layer-wise training strategy for multi-layer networks. Empirically,

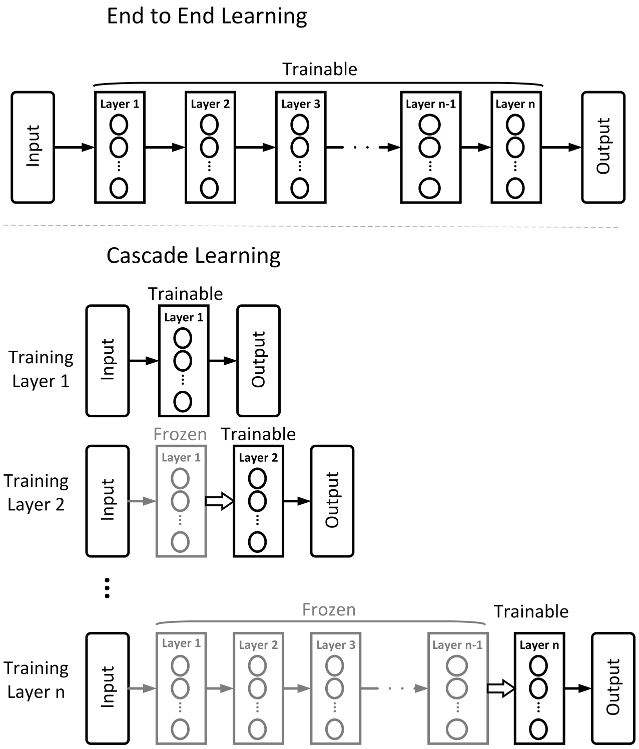


Figure 1: An overview of training MLPs using End to End (E2E) Learning and Cascade Learning (CL). In CL, layers are progressively added and weights of only the most recently added hidden layer and the output layer are trained by gradient descent. The previously trained layers are frozen. E2E learning is the standard learning approach where all of the layers in the network are trained simultaneously.

it is shown to trade some performance accuracy to significant gain in computation and memory requirements. The training process in CL is shown in Figure 1 and the idea is to add layers to the network and train only the weights of the most recently added hidden layer and the output layer, keeping the previous trained layers frozen. In several multi-class problems considered by Marquez et al. [20], the corresponding confusion matrix is shown to improve in a layer-wise fashion. We believe such a layer-wise strategy will have the effect of coarse-to-fine learning whereby early layers extract broad features of the problem domain due to their limited capacity and progressively later layers extract features fine-tuned to the specific problem. This is an appealing property for transfer learning as we demonstrate in this paper.

Transfer Learning on HAR

Transfer learning on HAR problems have been considered by several authors [4, 17, 22] and reviewed in Cook et al. [6]. Most of these approaches have considered traditional machine learning methods such as Hidden Markov Models [24, 28], k-Nearest neighbor and Support Vector Machines (SVM) [4]. More recently, Morales and Roggen [22] proposed transfer learning combined with deep learning networks on HAR, the closest in literature to the work we report in this paper. These authors use an eight-layer neural network of convolutional layers with 64 5×5 kernels in each layer and a final LSTM layer of 128 cells. Transfer learning is achieved by copying and freezing the first few layers of the source model and training the later layers from random initialization. With such a large network of a total of 986,257 parameters, GPU support is necessary for training the models. In addition to the need for heavy computation, the results reported for multi-class transfer learning classification were in the region of 50%, as measured by the F_1 score. The feed-forward cascade architecture we report in this paper, on the other hand, requires far fewer parameters (49,224) and could be run with CPU computing alone, achieving significantly higher transfer learning results of $76 - 80\%$, as measured by the F_1 score (see Results for Task 2). Note, throughout the paper, we use IMU sensors because which are better by being numerically integrated to obtain 3-D position/orientation of an object where linear accelerations and rotational velocities are measured directly, rather than obtaining accelerations and velocities by taking the time derivative of position data [26]. However, for comparing the results to [22] we use the exact same sensors and data for a fair comparison. Only the model architecture differs.

The contributions of this paper are as follows. First, we show that a cascade trained multi-layer architecture achieves competitive performance in comparison to E2E training in HAR. The networks we use are much simpler than the deep neural network architecture used by previous authors on the same benchmark problems. Secondly, we explore transfer learning of features extracted from a trained classifier to a related task and show that the cascade-trained architecture has the property of learning in a coarse-to-fine grained hierarchical fashion, and achieve state of the art performance at significantly low computational complexity. Finally, we carry out experiments in transfer learning across two different datasets and demonstrate results similar to what is achieved with tasks within a single dataset.

2 DATASETS

For empirical work reported in this paper, we used two benchmark datasets with different numbers of activity classes and data acquisition protocols. These are the Opportunity [8] and Skoda Mini checkpoint (Skoda) [29] datasets. The former consists of 18 activities performed by four subjects measured with inertial measurement units (IMU) and 3D accelerometers as on-body sensors. Figure 2 shows locations of sensor placement on the body. The 18 activities of daily living (ADL) in this dataset relate to behaviour in the kitchen such as opening and closing doors, and motions made during cleaning.

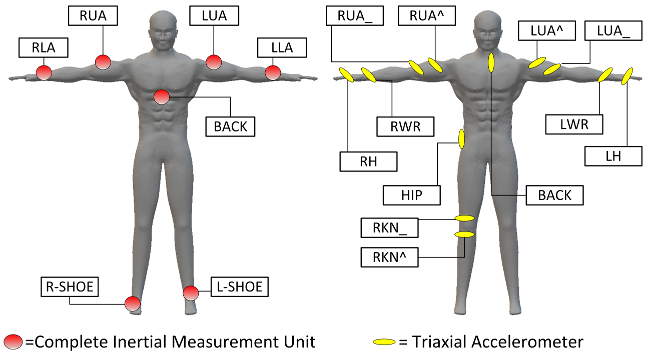


Figure 2: The on-body placement of sensors of the Opportunity dataset [27].

The Skoda dataset consists of 11 activities relating to quality control activities in a car production setting. Examples include opening and closing of an engine hood and closing of the vehicle's trunk. Sensors consist of 20 3D accelerometers placed on both arms whose locations are shown in Figure 3. This dataset was $98Hz$, which we down-sampled to $30Hz$ for consistency with Opportunity. In both cases, the three axes of each accelerometer are treated as separate channels.

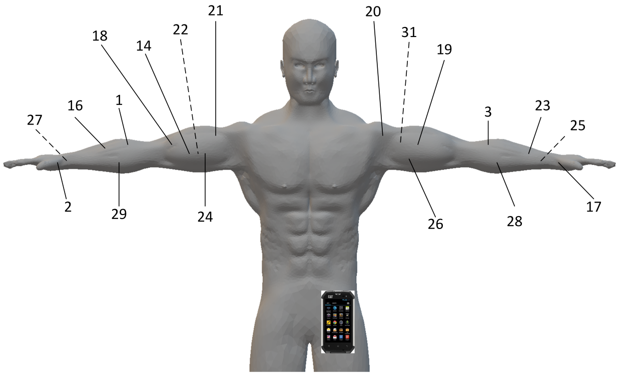


Figure 3: The on-body placement of sensors of the Skoda dataset [31].

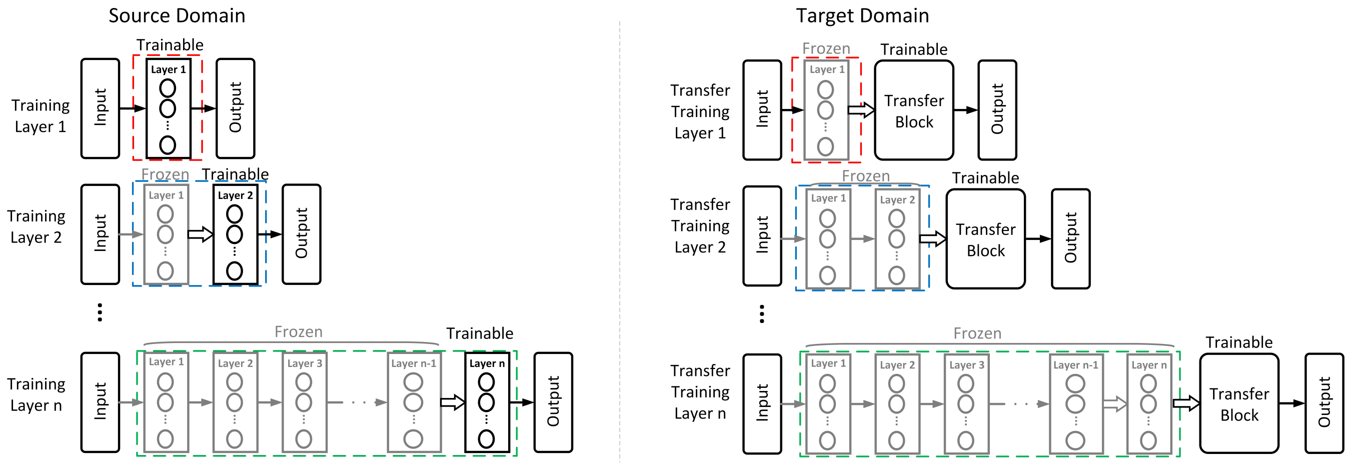


Figure 4: Transfer learning in a Cascade Learning setting.

3 EMPIRICAL WORK

Within the realm of HAR, we report results on three tasks in the setting of transfer learning with cascade architectures: (Task 1) the performance of a cascade architecture on multi-class activity recognition tasks to establish that cascade training achieves accuracies similar to E2E training; (Task 2) a comparison of transfer learning across tasks within a given dataset, comparing the transfer of features obtained from E2E and cascade trained models; and (Task 3) transfer of learned features across datasets, again comparing features learned by the two different training methods. In all experimental work reported here, we repeated the experiments with five runs to assess the uncertainty in results with 400 training epochs. Results are quoted as micro F_1 scores as well as weighted F_1 scores, where weighted averaging was used to account for imbalance across classes. The Adam optimizer was used in all experiments with a heuristically determined initial learning rate of 0.0001.

Task 1: Activity Classification with CL

The multi-layer networks we empirically used in this experiment for `Opportunity` had 5 layers with 25 units for each layer, and for `Skoda` had 128 input units and three hidden layers of 64, 64 and 32 units. The number of output units corresponded to the number of classes. We used hyperbolic tangent non-linearities in the hidden layer units and softmax for the output layer. For the `Opportunity` dataset, we used all the IMU sensor measurements and for `Skoda` we used the sensors placed on the right arm. On `Opportunity`, we train the model by using the data from ADL1, ADL2, ADL3 and drill session, and test the model using data from ADL4 and ADL5 (the same protocols as [22]). For Skoda, we randomly set 20% of the data for testing and the remainder for training. Both train and test data are normalized to $[0,1]$ as in [23].

Task 2: Transfer Learning within an Activity Dataset

Table 1: Task 2: Summary of source and target domains used for experiments reported on task 2.

Subtasks	Source domain	Target domains
Multi-class across users	Subjects 1,2 and 3	Subject 4
Binary class (open *vs.* close)	`Door1`	Door2
		Fridge
		Dishwasher
		Drawer1
		Drawer2
		Drawer3
Multi-class	14 class (Open/close 7 objectives)^1	4 class (Open/close Type one Type Two)^2

1: 7 objectives include: Door1, Door2, Fridge, Dishwasher, Drawer1, Drawer2 and Drawer3.

2: Type one and Type two are separated by the difference of hand movements. Type one: Doors and Fridge. Type Two: Dishwasher and Drawers.

- Multi-class Classification across Users

For comparison with [22], we use the same evaluation method to explore the performance. The source data consists of data from subjects 1, 2 and 3 where the train data are all data from subject 1 and ADL1, ADL2, ADL3 and drill session from subjects 2 and 3. The test data includes all the data from subjects 2 and 3 in ADL4 and ADL5. The target data is all the data from subject 4 where the train and test are divided in the same way as for the source domain (i.e. ADL4 and ADL5 are test data).

- Multiple Binary Classification Tasks

To compare the transfer learning abilities of cascade versus E2E trained models within a domain of different tasks, we used the `Opportunity` dataset which has open and close activities on different objects (fridge, doors, drawers and dishwasher). The underlying tasks have similarities in the required movement, but may have differences in required force and posture. We carried out two sets of experiments in which we trained source classifiers with one of the pairs of open/close (Door 1 and Dishwasher) and all other open/close pairs as target domains to transfer. We carried out experiments in which the transfer was done from features taken from each of the hidden layers and training a classifier layer in the subsequent target domains. Note, these tasks are summarized in detail in Table 1. The networks used in these binary classification tasks consisted of six layers with 25 hidden units in each. For the remaining experiments, the method of dividing the train and test data remains the same as previous work [22].

- Multi-class Classification Tasks

To test transfer learning in a more challenging multi-class setting with features extracted from cascade and E2E trained networks, we set up a 14 class problem with open and close on seven different objects (fridge, door 1, dishwasher *etc.*) as the source problem and a four-class problem as the target to transfer summarized in Table 1. For the target problem we identified the opening and closing of doors and fridge as an activity with similar hand movements (Type One) and the opening and closing of drawers and dishwasher as different from these (Type Two). This grouping was done to be consistent with the confusion matrices resulting in Task 1. Hence, the four target classes in Task 2 were: `Open Type One`, `Close Type One`, `Open Type Two` and `Close Type Two`. To clarify the source and target domains for all experiments done for Task 2, Table 1 shows the summary of all the source and target domains.

Task 3: Transfer Learning across Datasets

To study transfer learning across two different domains of HAR, we set up a problem using the `Skoda` and `Opportunity` datasets. Sensors from similar positions on the left arm of `Skoda` and the right arm of `Opportunity` datasets were chosen as 18 dimensional inputs with open and close actions. These were from the six accelerometers located at positions 1,2,16,21,27,29 in Figure 3 for the `Skoda` data and two IMUs, RLA and RUA shown in Figure 2 for the `Opportunity`. We transfer features from the `Skoda` to the `Opportunity` dataset as a binary classification problem.

Performance comparisons were made considering three ways of changing the transfer block (Figure 4): (i) training a new classifier from random initialization; (ii) training a new hidden layer and classification layer with features transferred from the source problem; and (iii) using the trained weights of the source model as initial conditions and training the entire network on the target problem. Note, of these (i) and (iii) could be seen as differing only in the initial conditions of gradient descent training. Hence, we show all the results from the method (ii) for transfer learning.

4 RESULTS

Task 1: Activity Classification with CL

Tables 2 and 3 show the multi-class classification performance of E2E versus cascade trained models on the `Opportunity` and `Skoda` problems respectively. We note that on both problems, both models achieve performances comparable to results quoted by previous authors on these problems (*e.g.* [32, 34]). Although our simple neural network architecture shows lower performance than the DeepConvLSTM model [23] for HAR on `Opportunity` for classification, without considering transfer learning, accuracy is not the focus of this paper *per se*. Instead, we are interested in cascade trained networks for transfer learning.

Table 2: Task 1: Classification Performance of Cascade and E2E Learning on the 18-Class `Opportunity` Dataset. LX means the X_{th} layer from the network. The red and bold text show the best performing case. For the remaining tables, the same colour coding and layer notation is used. Due to the imbalance among classes (including the null class with the majority), the micro F_1 score shows higher performance.

Model	Micro F_1 score (%)	Weighted F_1 score (%)
CL L0	86.52±0.31	84.16±0.36
CL L1	**86.88±0.30**	85.14±0.39
CL L2	86.78±0.23	85.36±0.40
CL L3	86.76±0.38	85.46±0.48
CL L4	86.72±0.31	**85.50±0.47**
E2E	86.32±0.67	85.08±0.55

Table 3: Task 1: Classification Performance of CL and E2E Learning on the test Skoda Dataset. In this table, we include two situations, including null class (11-class) and no null class (10-class). As the null class results in an imbalanced distribution, we compare the weighted and micro F_1 scores on 11-class task where weighted F_1 shows slightly worse performance. Weighted F_1 score counters the imbalance issues and gives more reasonable overall results.

Evaluation	CL L0	CL L1	CL L2	CL L3	E2E
Weighted F_1 score (%) (No Null Class)	80.20±1.3	85.06±1.4	85.82±1.8	**86.40±1.3**	85.66±1.3
Micro F_1 score (%) (No Null Class)	81.06±1.2	85.24±1.3	86.18±1.3	**86.44±1.3**	85.96±1.2
Weighted F_1 score (%)	71.34±2.1	77.70±1.6	79.08±1.5	**79.54±1.5**	78.36±1.0
Micro F_1 score (%)	72.70±1.8	78.40±1.5	79.66±1.4	**79.98±1.3**	79.14±1.0

Considering HAR classification, our simple architecture achieves 86.88 ± 0.30% in comparison to DeepConvLSTM which achieves 91.5% micro F_1 score. In terms of parameters, our architecture only requires 49224, whereas DeepConvLSTM requires $996800 + (128 * 18) + 18$. We also have significant savings in terms of training time, requiring 1 to 2 seconds per epoch with a CPU in contrast to DeepConvLSTM which requires approximate 3 seconds per epoch with GPU. Looking at the results from Tables 2 and 3, we further note that with CL, there is a progressive increase in performance as additional layers are included making the model deep, and the performance of CL is better than E2E learning for both datasets.

Table 4: Task 2: Transfer Learning Performance Across Users of CL and E2E Learning on the 18-Class Opportunity Dataset.

Model	Micro F_1 score (%)	Weighted F_1 score (%)
CL L0	**84.18±0.33**	**80.70±0.37**
CL L1	83.56±0.67	80.16±0.54
CL L2	82.52±0.52	79.06±0.53
CL L3	82.14±0.63	78.42±0.63
CL L4	81.76±0.62	77.70±0.41
CL L5	81.34±0.70	77.10±0.58
E2E L0	80.90±0.46	76.24±0.44
E2E L1	81.44±0.64	77.12±0.48
E2E L2	82.80±0.43	78.82±0.33
E2E L3	82.92±0.52	79.26±0.46
E2E L4	82.64±0.75	79.08±0.92
E2E L5	81.68±0.52	77.58±0.51

Task 2: Transfer Learning Within a Dataset

For the results of transfer learning in this and following subsections, a single hidden layer and a classification layer are added to the extracted features and trained using target domain data.

- Multi-class Classification across Users

Table 4 shows the performance of transfer learning across users within Opportunity for comparison with the same task in [22]. While the results displayed in Table 4 are using the IMU sensors for consistency in results presented in our paper, we also ran additional experiments using the same sensor set-up as [22] using 15 channels from accelerometers. When considering the exact same set-up as [22], we are also able to achieve a performance of 75 - 78% micro F_1 score, which is a significant improvement to the best results presented in [22], (OnO(100%) in figure 3) which is around 60% micro F_1 score.

Considering CL versus E2E learning, we notice the best performance of transfer learning for the cascade architecture is approximately 4% higher. We also learn that with deeper layers of CL, the transferability decreases, with worse performance as we transfer later layers of the network, while with the E2E network, the performance first increases with subsequent layers, but then decreases if we add transfer too many layers.

- Multiple Binary Classification Tasks

As an initial experiment to validate the difficulty of the transfer task, we considered training on Open Door 1 *vs.* Open Door 2 as a source problem and testing on Close Door 1 *vs.* Close Door 2 as target. The performance (test accuracy with balance between two classes around 48% *vs.* 52%) was 89 ± 0.25%. This suggested that recognising the doors is an easy problem. However, when training Open *vs.* Close on Door 1 and testing the classifier on Open *vs.* Close on Door 2, the performance (micro F_1 score) was 48.25 ± 3.2%, suggesting this to be more suitable problem that requires transfer learning. Further problems we selected were based on this and similar preliminary experiments.

Table 5: Task 2: Transfer learning performance within the `Opportunity` dataset using CL and E2E Learning. The performance of the binary classification is evaluated by weighted F_1 score (%).

Model	Source: Door 1	Target: Door 2	Fridge	Dish washer	Drawer 1	Drawer 2	Drawer 3
CL L0	73.04±1.4	**71.20±1.7**	**71.84±1.0**	**63.18±3.0**	**67.78±4.1**	**59.14±1.6**	**65.62±2.9**
CL L1	73.44±1.5	54.60±3.7	63.56±2.5	56.22±1.5	60.40±1.4	55.38±1.0	49.38±2.3
CL L2	73.42±1.6	49.56±6.3	61.26±0.8	58.28±1.8	56.26±2.6	56.40±2.5	51.40±1.4
CL L3	73.44±1.5	42.70±6.9	49.44±4.3	50.26±8.1	34.58±4.5	47.46±4.0	50.16±1.6
CL L4	73.42±1.5	41.32±6.8	45.34±4.5	45.46±7.5	32.38±3.5	46.76±3.5	48.34±2.8
CL L5	73.42±1.5	39.56±4.7	43.98±1.3	43.14±5.0	31.62±1.7	45.52±0.7	46.34±5.1
E2E L0	–	64.64±1.3	**76.50±1.4**	60.30±6.4	**69.34±2.7**	54.16±2.3	56.48±1.4
E2E L1	–	65.92±1.9	72.68±4.8	55.20±8.1	66.36±4.0	55.12±4.0	55.64±2.4
E2E L2	–	64.64±3.7	71.44±4.9	55.16±4.9	63.48±6.2	54.32±3.8	51.82±3.1
E2E L3	–	63.36±4.1	68.44±5.5	54.38±4.8	62.08±3.5	51.78±4.2	52.16±4.7
E2E L4	–	59.78±3.8	62.78±3.6	52.26±4.8	59.52±4.8	52.12±1.7	49.58±3.3
E2E L5	76.82±1.5	56.32±5.1	59.34±3.2	54.22±4.6	53.94±8.4	51.84±4.2	50.62±3.5

Table 5 shows several binary classification tasks of transfer learning with E2E and cascade trained networks with the source problems being opening and closing of door 1 and six other problems taken as targets. We show results of transferring features taken from various different layers of the networks trained on the source problems. We note from Table 5 that for cascade trained networks as feature extractors evaluated by weighted F_1 score, there is a consistent monotonic decline in performance with network depth where features are taken from, while this is only partly true for E2E trained networks. Best features for cascade trained models are from the first hidden layer. This confirms the motivating idea that there is a progressive specialization included by layer-wise training, coarse features learnt early on and more detailed ones specific to the task picked up in later layers. We also note that the cascade trained network is competitive in performance with the more widely used E2E trained models. We also notice the overall level of transfer learning from door 1 to similar objectives, door 2 and fridge, is much higher than to dissimilar objectives, dish washer and drawers. Besides, the performance on dissimilar objectives (*e.g.* Dish washer) has high variance (around 8%), which may affect the observations.

These results should be compared against a baseline of training on the source problem and testing on the target problem without any further training. When we tested this, the performance on the target class was in the region of 48% for all of the problems considered. This is sufficiently low to justify further training in the target domains as undertaken in this study.

- Transfer learning from 14 classes to 4 classes

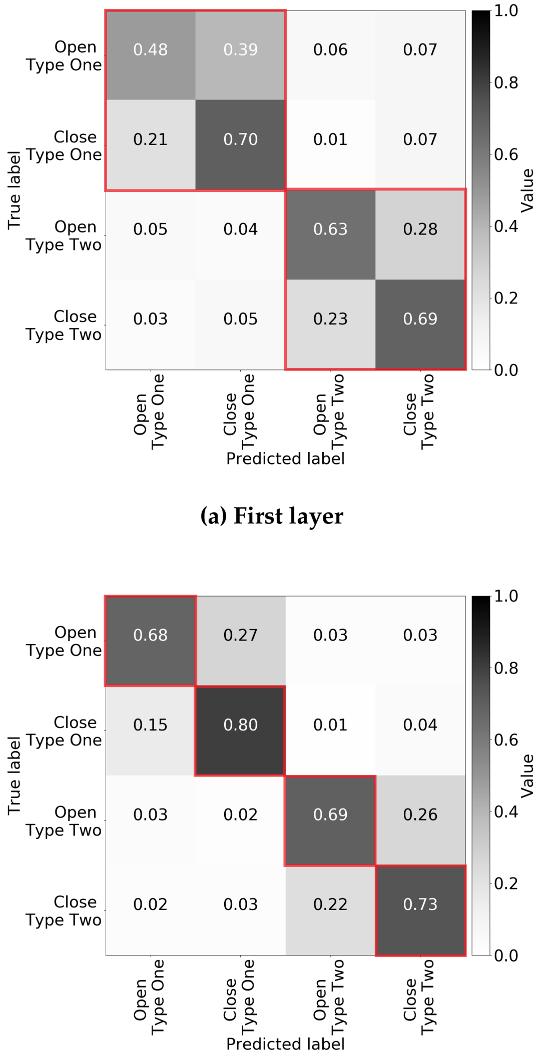
Results of transfer learning from cascade trained networks on this multi-class task are shown as confusion matrices in Figure 5. Note in this task, we have introduced a hierarchy into the classifier outputs, going from a group of coarse tasks (`Type One` *vs.* `Type Two`) and then finer tasks within the groups (Open *vs.* Close). Hence, these results show a pattern different from what was observed in Table 5 in that transfer from the final layer gives better performance than transfer from the first layer. This is a consequence of demanding the transfer of features to a fine-grained classification problem and this transfer confirms the coarse-to-fine nature of features learned by CL.

Task 3: Transfer Learning across Datasets

A set of transfer learning experimental results across datasets (training on a task in the `Skoda` data and transferring to `Opportunity`) are shown in Figure 6, with competitive performance from the computationally simpler cascade trained model and showing monotonic decrease in performance with deeper layers. Here again we observe the same patterns of performance noted with the results of Table 5.

5 DISCUSSION

The results shown in the previous section suggest a monotonic decline in transferability from features taken from early layers to later ones from the cascade network. However, the later layers are necessary for increased recognition accuracy of the source domain classification. This observation confirms the view we wish to advance that cascade training packs feature information in a

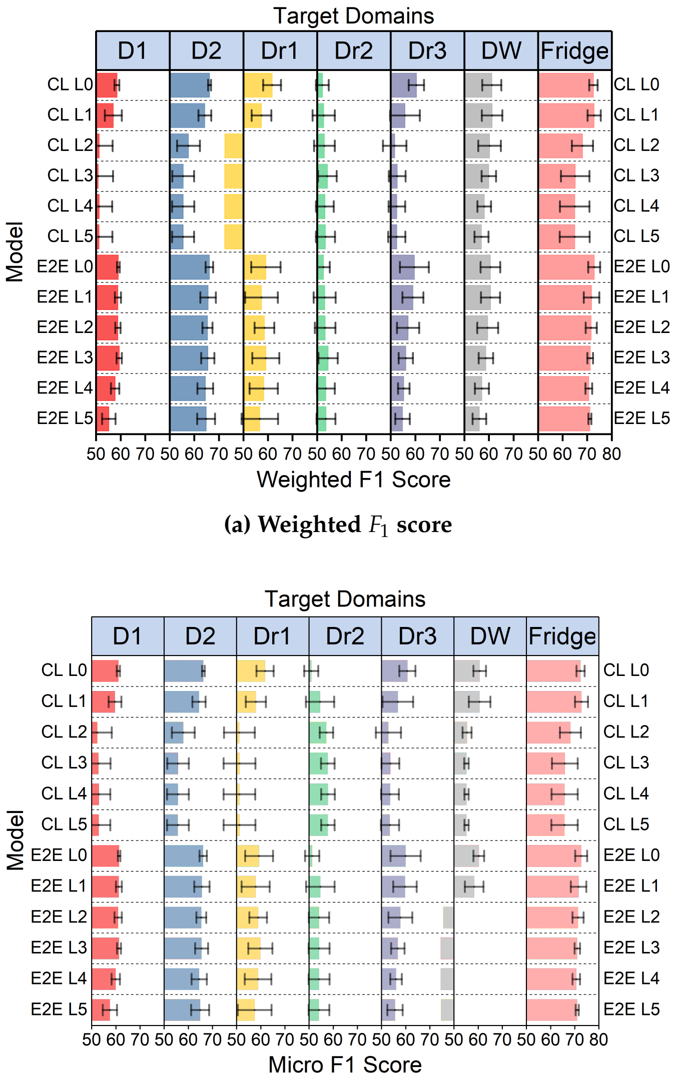


(a) First layer

(b) Last layer

Figure 5: Confusion matrices of transfer learning in the `Opportunity` **dataset, learning from a 14− class problem down to a four-class problem with (a) transfer from layer zero and (b) transfer from layer five.**

specific way, coarse information in early layers and finer details related to the source task in later layers.



(a) Weighted F_1 score

(b) Micro F_1 score

Figure 6: Transfer learning performance from `Skoda` **to** `Opportunity` **based on cascade and E2E learning. Corresponding to task 3, the transfer learning is evaluated by (a) weighted F_1 score and (b) micro F_1 score. For simplification, D1, D2, DW, Dr1, Dr2, Dr3 are the abbreviation of Door 1, Door 2, Dish Washer, Drawer 1, Drawer 2 and Drawer 3, respectively, which is followed in this work unless otherwise specified.**

6 CONCLUSIONS

In this paper, we have explored the usefulness of a particular approach to learning in layered networks: deep CL. Layer-wise training restricts how information relating to the target may be packed in the network, which is inherently different from the flexibility enjoyed by E2E training of the same architecture. Despite this difference, we find that cascade training achieves state of the art performance on the two HAR classification problems, not too different from E2E training, and with significantly fewer parameters and training time than previous applications of deep neural networks. The extraction of relevant features, in a hierarchical manner, across layers in CL is demonstrated by the fact that features taken from different layers to transfer across tasks show monotonically decreasing performance when we move from the first to the final layer. Coarse features transferred from the first hidden layer give the best performance with cascade trained networks as source networks and, most importantly, these are as good as, and sometimes better than, features transferred from any layer of E2E trained networks. When we construct a task that demands the transfer of finer features, we are able to show that better features are obtained from the final layer, further reinforcing this point.

REFERENCES

[1] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. 2018. Greedy Layerwise Learning Can Scale to ImageNet. *arXiv preprint arXiv:1812.11446* (2018).

[2] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2006. Greedy Layer-wise Training of Deep Networks. In *Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS'06)*. MIT Press, 153–160.

[3] Andreas Bulling, Ulf Blanke, and Bernt Schiele. 2014. A Tutorial on Human Activity Recognition Using Body-worn Inertial Sensors. *ACM Computing Surveys (CSUR)* 46, 3 (2014).

[4] Alberto Calatroni, Daniel Roggen, and Gerhard Tröster. 2011. Automatic Transfer of Activity Recognition Capabilities Between Body-worn Motion Sensors: Training Newcomers to Recognize Locomotion. In *8th International Conference on Networked Sensing Systems (INSS'11)*.

[5] Kumar Chellapilla, Sidd Puri, and Patrice Simard. 2006. High Performance Convolutional Neural Networks for Document Processing. In *10th International Workshop on Frontiers in Handwriting Recognition*. Suvisoft.

[6] Diane Cook, Kyle D Feuz, and Narayanan C Krishnan. 2013. Transfer Learning for Activity Recognition: A Survey. *Knowledge and Information Systems* 36, 3 (2013), 537–556.

[7] Sander Dieleman and Benjamin Schrauwen. 2014. End-to-end Learning for Music Audio. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6964–6968.

[8] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[9] Marcus Edel and Enrico Köppe. 2016. Binarized-BLSTM-RNN Based Human Activity Recognition. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 1–7.

[10] Scott E. Fahlman and Christian Lebiere. 1990. The Cascade-correlation Learning Architecture. In *Advances in Neural Information Processing Systems*. 524–532.

[11] Yu Guan and Thomas Plötz. 2017. Ensembles of Deep LSTM Learners for Activity Recognition Using Wearables. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (2017), 1–28.

[12] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. 2016. Deep, Convolutional, and Recurrent Models for Human Activity Recognition Using Wearables. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 1533–1540.

[13] Kaiming He and Jian Sun. 2015. Convolutional Neural Networks at Constrained Time Cost. In *Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition*. 5353–5360.

[14] Visakan Kadirkamanathan and Mahesan Niranjan. 1992. Application of an Architecturally Dynamic Network for Speech Pattern Classification. *Proceedings of the Institute of Acoustics* 14 (1992), 343–350.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*. 1097–1105.

[16] Chung-Hsien Kuo, Fang-Ghun Huang, Keng-Liang Wang, Ming-Yih Lee, and Huai-Wen Chen. 2004. Development of Internet based Remote Health and Activity Monitoring Systems for the Elders. *Journal of Medical and Biological Engineering* 24, 1 (2004), 57–66.

[17] Marc Kurz, Gerold Hölzl, Alois Ferscha, Alberto Calatroni, Daniel Roggen, and Gerhard Tröster. 2011. Real-time Transfer and Evaluation of Activity Recognition Capabilities in an Opportunistic System. *Machine Learning* 1, 7 (2011), 8–14.

[18] Oscar D Lara and Miguel A Labrador. 2013. A Survey on Human Activity Recognition using Wearable Sensors. *IEEE Communications Surveys & Tutorials* 15, 3 (2013), 1192–1209.

[19] Yann LeCun, John S Denker, and A Sara. 1990. Optimal Brain Damage. In *Advances in Neural Information Processing Systems 2*. Morgan-Kaufmann, 598–605.

[20] Enrique S Marquez, Jonathon S Hare, and Mahesan Niranjan. 2018. Deep Cascade Learning. *IEEE Transactions on Neural Networks and Learning Systems* 99 (2018), 1–11.

[21] Antony Milne, Mihalis Nicolaou, and Katayoun Farrahi. 2017. Discovering the Typing Behaviour of Parkinson's Patients Using Topic Models. In *International Conference on Social Informatics*. Springer, 89–97.

[22] Francisco Javier Ordóñez Morales and Daniel Roggen. 2016. Deep Convolutional Feature Transfer Across Mobile Activity Recognition Domains, Sensor Modalities and Locations. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*. ACM, 92–99.

[23] Francisco Ordóñez and Daniel Roggen. 2016. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* 16, 1 (2016), 1–25.

[24] Francisco Javier Ordóñez, Gwenn Englebienne, Paula De Toledo, Tim Van Kasteren, Araceli Sanchis, and Ben Kröse. 2014. In-home Activity Recognition: Bayesian Inference for Hidden Markov Models. *IEEE Pervasive Computing* 13, 3 (2014), 67–75.

[25] John Platt. 1991. A Resource-Allocating Network for Function Interpolation. *Neural Computation* 3, 2 (1991), 213–225.

[26] Reza Sharif Razavian, Sara Greenberg, and John McPhee. 2019. Biomechanics Imaging and Analysis. *Encyclopedia of Biomedical Engineering* (2019), 488–500.

[27] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, et al. 2010. Collecting Complex Activity Datasets in Highly Rich Networked Sensor Environments. In *2010 Seventh International Conference on Networked Sensing Systems (INSS)*. IEEE, 233–240.

[28] Daniel Roggen, Luis Ponce Cuspinera, Guilherme Pombo, Falah Ali, and Long-Van Nguyen-Dinh. 2015. Limited-memory Warping LCSS for Real-time Low-power Pattern Recognition in Wireless Nodes. In *European Conference on Wireless Sensor Networks*. Springer, 151–167.

[29] Daniel Roggen and Piero Zappi. 2015. Human Activity/Context Recognition Datasets. http://har-dataset.org/doku.php?id=wiki:dataset

[30] Charissa Ann Ronao and Sung-Bae Cho. 2016. Human Activity Recognition with Smartphone Sensors using Deep Learning Neural Networks. *Expert Systems with Applications* 59 (2016), 235–244.

[31] Thomas Stiefmeier, Daniel Roggen, and Gerhard Troster. 2007. Fusion of String-matched Templates for Continuous Activity Recognition. In *11th IEEE International Symposium on Wearable Computers*. IEEE, 41–44.

[32] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. 2015. Deep Convolutional Neural Networks on Multichannel Time series for Human Activity Recognition. In *24th International Joint Conference on Artificial Intelligence*. 3995–4001.

[33] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How Transferable are Features in Deep Neural Networks?. In *Advances in Neural Information Processing Systems*. 3320–3328.

[34] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. 2014. Convolutional Neural Networks for Human Activity Recognition using Mobile Sensors. In *6th International Conference on Mobile Computing, Applications and Services*. IEEE, 197–205.