Personalised Human Activity Recognition Using Matching Networks

Sadiq Sani[1(✉)], Nirmalie Wiratunga[1], Stewart Massie[1], and Kay Cooper[2]

[1] School of Computing Science and Digital Media, Robert Gordon University, Aberdeen, Scotland AB10 7GJ, UK
[2] School of Health Sciences, Robert Gordon University, Aberdeen, Scotland AB10 7GJ, UK
{s.sani,n.wiratunga,s.massie,k.cooper}@rgu.ac.uk

Abstract. Human Activity Recognition (HAR) is typically modelled as a classification task where sensor data associated with activity labels are used to train a classifier to recognise future occurrences of these activities. An important consideration when training HAR models is whether to use training data from a general population (subject-independent), or personalised training data from the target user (subject-dependent). Previous evaluations have shown personalised training to be more accurate because of the ability of resulting models to better capture individual users' activity patterns. From a practical perspective however, collecting sufficient training data from end users may not be feasible. This has made using subject-independent training far more common in real-world HAR systems. In this paper, we introduce a novel approach to personalised HAR using a neural network architecture called a matching network. Matching networks perform nearest-neighbour classification by reusing the class label of the most similar instances in a provided support set, which makes them very relevant to case-based reasoning. A key advantage of matching networks is that they use metric learning to produce feature embeddings or representations that maximise classification accuracy, given a chosen similarity metric. Evaluations show our approach to substantially out perform general subject-independent models by at least 6% macro-averaged F1 score.

1 Introduction

Human Activity Recognition (HAR) is the computational discovery of human activity from sensor data and is increasingly being adopted in health, security, entertainment and defense applications [10]. An example of the application of HAR in healthcare is SelfBACK[1], a system designed to improve self-management of low back pain (LBP) by monitoring users' physical activity levels in order to provide advice and guidance on how best to adhere to recommended

[1] This work was fully sponsored by the collaborative project SelfBACK under contract with the European Commission (# 689043) in the Horizon2020 framework. Details of this project are available at: http://www.selfback.eu.

M. T. Cox et al. (Eds.): ICCBR 2018, LNAI 11156, pp. 339–353, 2018.
https://doi.org/10.1007/978-3-030-01081-2_23

physical activity guidelines [2]. Guidelines for LBP recommend that patients should not be sedentary for long periods of time and should maintain moderate levels of physical activity. The SelfBACK system uses a wrist-worn sensor to continuously recognise user activities in real time. This allows the system to compare the user's activity profile to the recommended guidelines for physical activity and produce feedback to inform the user on how well they are adhering to these guidelines. Other information in the user's activity profile include the durations of activities and, for walking, the counts of steps taken, as well as intensity e.g. slow, normal or fast. The categorisation of walking into slow, normal and fast allows us to better match the activity intensity (i.e. low, moderate or high) recommended in the guidelines. HAR is typically modelled as a classification task where sensor data associated with activity labels are used to train a classifier to predict future occurrences of those activities.

An important consideration for HAR is classifier training, where training examples can either be acquired from a general population (subject-independent), or from the target user of the system (subject-dependent). Previous works have shown using subject-dependent data to result in superior performance [5,7,19,21]. The relatively poorer performance of subject-independent models can be attributed to variations in activity patterns, gait or posture between different individuals [12]. However, training a classifier exclusively with user provided data is not practical in a real-world configuration as this places significant burden on the user to provide sufficient amounts of training data required to build a personalised model.

In this paper, we introduce an approach to personalised HAR using matching networks. Matching Networks are a type of neural network architecture introduced for the task of one-shot learning [22] which is a scenario where an algorithm is trained to recognise a new class from just a few examples of that class. Given a (typically small) support set of labelled examples, matching networks are able to classify an unlabelled example by reusing the class labels of the most similar examples in the support set. To apply matching networks for personalised HAR, we require the user to provide a small number of examples for each type of activity. Note that this is no different to the calibration approach which is commonly employed in gesture control devices and is already in use in the Nike + iPod fitness device [12]. The examples provided by the user are treated as the support set used by the matching network to classify future occurrences of the user's activities. In this way, the matching network generates a personalised classifier that is better able to recognise the individual user's activity pattern.

An advantage of matching networks is that they use metric learning in order to produce feature embeddings or representations that maximise nearest neighbour classification accuracy.

At the same time, because classification is only conditioned on the support set, matching networks behave like non-parametric models and can reason with any set of examples that are provided at runtime, without the need for retraining the network. This makes our system able to continuously adapt to changes in the user's context easily which is an important goal of CBR.

The rest of this paper is organised as follows: in Sect. 2, we discuss important related work on personalised HAR and highlight the importance of CBR and k-nearest neighbour in particular for personalisation. Section 3 presents technical details of the steps for training a HAR classifier. In Sect. 4, we formally introduce matching networks and in Sect. 5 we present how we apply this to the task of personalised HAR. A description of our dataset is presented in Sect. 6, evaluations are presented in Sect. 7 and conclusions follow in Sect. 8.

2 Related Work

The standard approach to classifier training for HAR involves using subject-independent examples to create a general classification model. However, comparative evaluation with personalised models, trained using subject-dependent examples, show this to produce more accurate predictions [5,7,21]. In [21], a general model and a personalised model both trained using a C4.5 decision tree classifier are compared. The general model produced an accuracy of 56.3% while the personalised model produced an accuracy of 94.6%, an increase of 39.3%. Similarly, [5,7] reported increases of 19.0% and 9.7% between personalised and general models respectively which are trained using the same classification algorithm. A more recent improvement on standard subject-dependent training which uses online multi-task (OMT) learning is presented in [20]. Here, individual users are treated as separate tasks where each task only contains the respective user's data. Personalised classifiers for each task are then trained jointly which allows the models to influence one-another, thereby improving accuracy. Evaluation shows OMT to perform better than personalised models trained independently. A common disadvantage of all subject-dependent approaches is that they require access to significant amounts of good quality end-user data for training. Such approaches have limited practical use for real-world applications because of the burden they place on users to provide sufficient training data.

An alternative solution is to bootstrap a general model with a small set of examples acquired from the user through semi-supervised learning approaches. Different types of semi-supervised learning approaches have been explored for personalised HAR e.g. self-learning, co-learning and active learning, which bootstrap a general model with examples acquired from the user [12]. Both self-learning and co-learning attempt to infer accurate activity labels for unlabelled examples without querying the user. This way, both approaches manage to avoid placing any labelling burden on the user. In contrast, active learning selectively chooses the most useful examples to present to the user for labelling using techniques such as uncertainty sampling which consistently outperform random sampling [16]. Evaluations show semi-supervised approaches mainly produce improvements in situations where baseline classification accuracy is low but no improvements were observed in situations where baseline accuracy was already very high [12]. In addition, semi-supervised approaches require retraining of the classifier at runtime every time new data needs to be incorporated into the model, which can be very expensive, especially on mobile devices.

Case-based reasoning (CBR) offers a convenient solution to the problem of model retraining at runtime. The k-nearest neighbour (kNN) retrieval approach at the core of CBR does not learn a model, which makes it able to easily assimilate new examples at runtime. However, performance of kNN largely depends on the choice of similarity metric, and manually defining good similarity metrics for specific problems is generally difficult [4].

Metric learning is an approach that is used to automatically learn a similarity metric from data in a way that better captures the important relationships between examples in that data [23]. An important point to note about metric learning is that learning a similarity metric from data is equivalent to transforming the data to a new representation and computing the similarity in this new space using any standard metric e.g. Euclidean [4]. For a comprehensive review of metric learning, we refer the reader to [4,9]. A more recent sub-field of metric learning called deep metric learning uses deep learning algorithms to learn this feature transformation, thereby taking advantage of the ability of deep learning algorithms to extract higher-level, abstract feature representations. Matching networks are an example in this category that are able to incorporate any deep learning architecture e.g. convolutional neural networks [11] or recurrent neural networks [6].

Given the novelty of deep metric learning, very few applications of this are available in case-based reasoning. A very recent work that uses deep metric learning in a Case-based reasoning system for adaptable clickbait detection is [14], where a word2vec model [15] is used in combination with a deep convolutional neural network to learn similarity between clickbait articles. Another CBR system for image-based Web page classification which uses Siamese convolutional neural networks is presented in [13]. Siamese neural networks learn a similarity metric by minimising a contrastive loss which penalises dissimilar example pairs being placed close in the representation space, and rewards similar pairs being placed close together [8]. In this work, we focus on matching networks in particular which have the ability to both learn appropriate feature transformations using metric learning, and at the same time perform nearest neighbour classification using neural attention mechanism [3].

3 Human Activity Recognition

The computational task of HAR consists of three main steps: windowing, feature extraction and classifier training as illustrated in Fig. 1. Windowing is the process of partitioning continuous sensor data into discrete instances of length l, where l is typically specified in seconds. Figure 2 illustrates how windowing is applied to a tri-axial accelerometer data stream with channels: a, b and c. Windows can be overlapped especially at train time in order have better coverage of the data, which also increases the number of examples available for training. We do not overlap windows at test time in order to simulate real-time streaming data.

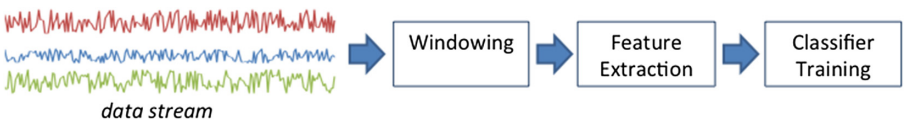

Fig. 1. Steps of human activity recognition.

The length of windows is an important consideration where very short window lengths typically produce less accurate performance, while longer windows produce latency at runtime due to the fact that several seconds worth of data need to be collected before before making a prediction [18]. In this work, we choose a window length of five seconds which provides a good balance between accuracy and latency. A tri-axial accelerometer partitioned in this way produces a window w_i is comprised of real-valued vectors $\boldsymbol{a_i}$, $\boldsymbol{b_i}$ and $\boldsymbol{c_i}$, such that $\boldsymbol{a_i} = (a_{i1}, \ldots, a_{il})$.

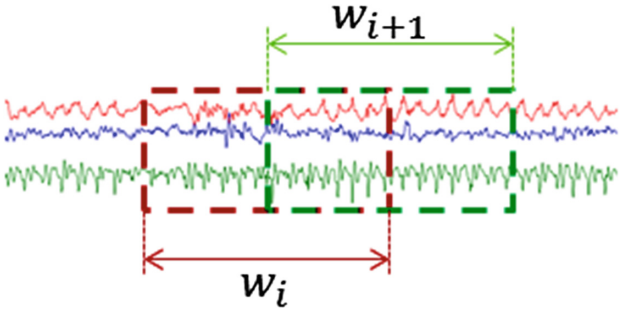

Fig. 2. Illustration of accelerometer data windowing.

Once windows have been partitioned, suitable features need to extracted from each window w_i in order to generate examples x_i used for classifier training. Many different feature extraction approaches have been applied for HAR. These include hand-crafted time and frequency domain features, coefficients of frequency domain transformations, as well as more recent deep learning approaches [17]. One feature extraction approach we have previously found to be both inexpensive to compute and very effective, is Discrete Cosine Transform (DCT) [17]. DCT is applied to each axis ($\boldsymbol{a_i}, \boldsymbol{b_i}, \boldsymbol{c_i}$) of a given window w_i to produce vectors of coefficients $\boldsymbol{v_a}$, $\boldsymbol{v_b}$ and $\boldsymbol{v_c}$ respectively that describe the sinusoidal wave forms that constitute the original signal. In addition, we also include the DCT coefficients of the magnitude vector $\boldsymbol{m}$ where each entry m_j in $\boldsymbol{m}$ is computed using the Euclidean norm of corresponding entries in a_j, b_j and c_j as defined in Eq. 1.

$$m_j = \sqrt{a_j^2 + b_j^2 + c_j^2} \tag{1}$$

DCT produces an ordered vector of coefficients such that the most significant information is concentrated at the lower indices of the vectors This means that the vector of coeffcients can be truncated to the first n indices without loss of information, making DCT ideal for compression. In this work, we truncate vectors to a length of $n = 60$. The truncated coefficient vectors $\boldsymbol{v_a}$, $\boldsymbol{v_b}$, $\boldsymbol{v_c}$ and $\boldsymbol{v_m}$ are concatenated together to form a single example representation x_i of length 240.

4 Matching Networks

The aim of matching networks is to learn a model that maps an unlabelled example $\hat{x}$ to a class label $\hat{y}$ using a small support set S of labelled examples. To provide a formal definition of matching networks, we define a set of class labels L and a set of examples X. We also define a support set S as shown in Eq. 2,

$$S = \{(x, y)|x \in X, y \in Y \subset L\} \tag{2}$$

i.e. S consists of a subset of classes Y with m examples in each class. Hence, the cardinality of S is $|S| = m \times |Y|$. A matching networks learns a classifier C_s which, given a test instance $\hat{x}$, provides a probability distribution over class labels $y \in Y$ i.e. $P(y|\hat{x}, S)$. Accordingly, the class label $\hat{y}$ of $\hat{x}$ is predicted as the class with the highest probability i.e.

$$\hat{y} = argmax_y P(y|\hat{x}, S) \tag{3}$$

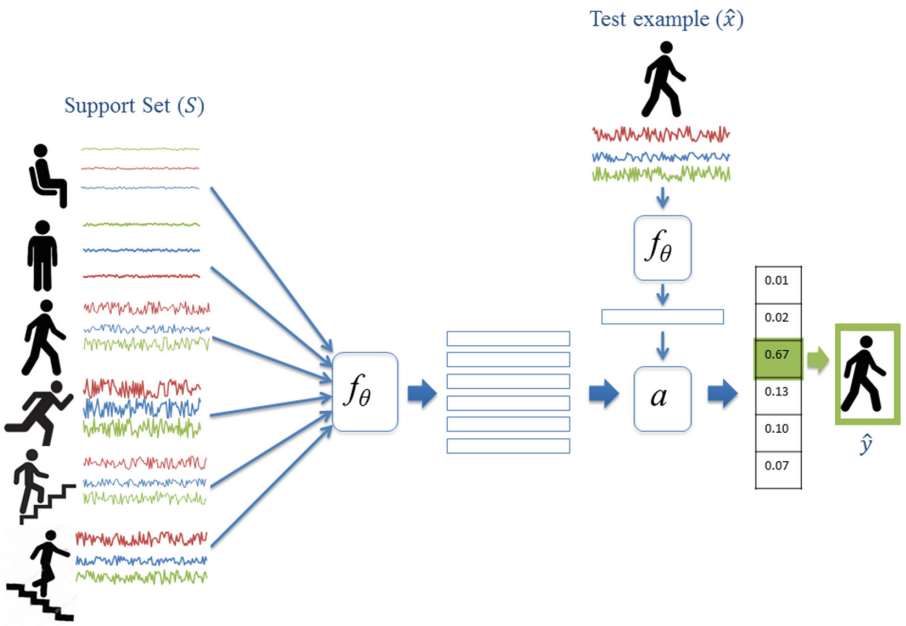

Fig. 3. Illustration of matching network for HAR.

Next we address the question of how to calculate $P(y|\hat{x}, S)$. This can be done using an attention function $a()$ which computes the probabilities in three operations. Firstly, we define an embedding function f_θ, which is a neural network that maps a given input to an embedded representation as shown in Eq. 4 (Fig. 3).

$$f_\theta(x) = x' \tag{4}$$

The embedding function f_θ is the embedding part of the matching network and it's goal is to produce representations that maximise similarity between examples belonging to the same class. Thus, we define a similarity metric $sim(\hat{x}', x'_i)$ which returns the similarity between the embedded representations of our unlabelled example $\hat{x}$ and any example $x_i \in S$. Here, any standard similarity metric e.g. Euclidean, dot product or cosine can be used. An example of sim using cosine similarity is shown in Eq. 5.

$$sim(\hat{x'}, x'_i) = \frac{\sum \hat{x}'_j x'_{i,j}}{\sqrt{\hat{x}'^2_j}\sqrt{x'^2_{i,j}}} \tag{5}$$

The last operation of the attention function is to convert the similarity values returned by sim into probabilities. This can be done using the softmax function as shown in Eq. 7.

$$a(\hat{x'}, x'_i) = e^{sim(\hat{x}', x'_i)} / \sum^{|s|} e^{sim(\hat{x'}, x'_i)} \tag{6}$$

Using a one-hot encoding vector $\boldsymbol{y}$ to represent any class label y, we can estimate a class probability for $\hat{x'}$ as follows:

$$\hat{\boldsymbol{y}} = \sum^{|S|} a(\hat{x'}, x'_i) * \boldsymbol{y} \tag{7}$$

Since $\boldsymbol{y_i}$ has a value of 1 at only the position corresponding to its class (with the rest being zero); the multiplication with a can be viewed as providing a similarity weighted estimate for each candidate class and thereby forming an estimated class distribution.

We can now use this estimated $\hat{\boldsymbol{y}}$ class probability and the actual $\boldsymbol{y}$ class probability (i.e. the one-hot vector) to derive the training loss using a function such as the categorical cross-entropy as shown in Eq. 9.

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_j^{|L|} y_j \; log(\hat{y_j}) \tag{8}$$

$$\mathcal{L}_{train} = \frac{\sum_i^{|N|} \mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)}{N} \tag{9}$$

Accordingly the entire matching network can be trained end-to-end using gradient descent.

5 Personalised HAR Using Matching Networks

In this section, we formally describe how we apply matching networks for personalised HAR. Recall that for personalised HAR, our aim is to obtain a network that can classify a particular user's activity using a small set of examples provided by the same user. Therefore, training such a network requires us to define a set of users U where each user $u_j \in U$ is comprised of a set of labelled examples as follows:

$$u_j = \{(x, y)|x \in X, y \in L\} \tag{10}$$

Next we define a set of training instances T_j for each user u_j as follows:

$$T_j = \{(S_j, B_j)\}^l \tag{11}$$

i.e., T_j is made up of user-specific support and target set pairs S_j and B_j respectively, where $S_j = \{(x, y)|x \in u_i, y \in L\}^k$ and $B_j = \{(x, y)|x \in u_j, x \notin S_j\}$. Note that the set of labels in S_j is always equivalent to L because we are interested in learning a classifier over the entire set of activity labels. Accordingly, S_j contains m examples for each class $y \in L$ and the cardinality of S_j is $k = m \times |L|$. Both S_j and B_j are sampled at random from u_j l times to create T_j. Each B_j is used with it's respective S_j by classifying each instance in B_j using S_j and computing loss using categorical cross entropy. This process is illustrated in Fig. 4. The network is trained using stochastic gradient descent and back propagation.

$$T_j = \{(S_j, B_j)\}^l \tag{12}$$

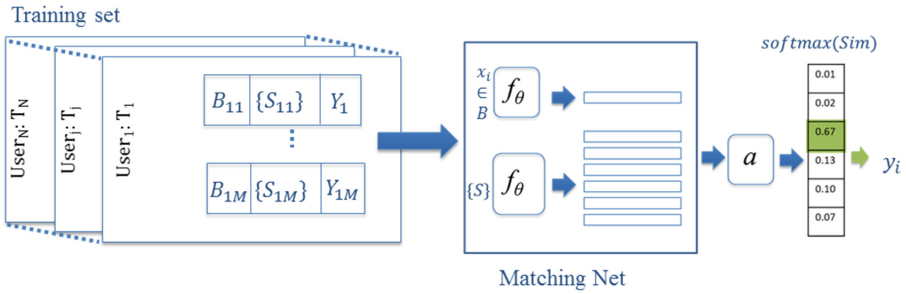

Fig. 4. Training matching network for personalised HAR

The embedding function used in this work is a neural network with one fully-connected layer with 1200 units. Before examples are input into the embedding network, they are passed through Discrete Cosine Transform (DCT) feature extraction. The fully connected layer is followed by a Batch Normalisation layer which reduces covariate shift and has been shown to result in faster training and better accuracy. An illustration of the configuration of the embedding network is presented in Fig. 5.

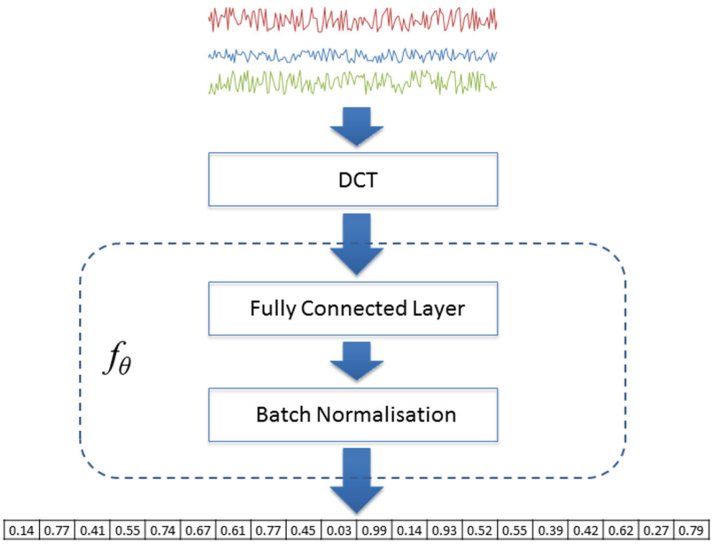

Fig. 5. Details of embedding network

6 Dataset

A group of 50 volunteer participants was used for data collection. The age range of participants is 18–54 years and the gender distribution is 52% Female and 48% Male. Data collection concentrated on the activities provided in Table 1.

Table 1. Description of activity classes.

Activity	Description
Lying	Lying down relatively still on a plinth
Sitting	Sitting still with hands on desk or thighs
Standing	Standing relatively still
Walking Slow	Walking at slow pace
Walking normal	Walking at normal pace
Walking fast	Walking at fast pace
Up stairs	Walking up 4–6 flights of stairs
Down stairs	Walking down 4–6 a flights of stairs
Jogging	Jogging on a treadmill at moderate speed

The set of activities in Table 1 was chosen because it represents the range of normal daily activities typically performed by most people. Three different walking speeds (slow, normal and fast) were included in order to have an accurate

estimate of the intensity of the activities performed by the user. Identifying intensity of activity is important because guidelines for health and well-being include recommendations for encouraging both moderate and vigorous physical activity [1]. We expect the distinction between different walking speeds to be particularly challenging for subject-independent models because one person's slow walking speed might be closer to another person's normal walking speed

Data was collected using the Axivity Ax3 tri-axial accelerometer[2] at a sampling rate of 100 Hz. Accelerometers were mounted on the right-hand wrists of the participants using specially designed wristbands provided by Axivity. Activities are roughly evenly distributed between classes as participants were asked to do each activity for the same period of time (3 min). The exceptions are Up stairs and Down stairs, where the amount of time needed to reach the top (or bottom) of the stairs was just over 2 min on average. This data is publicly available on Github[3].

Recall that in order to apply the matching network, we require the user to provide a small sample of data for each activity class which will be used to create the support set. To simulate this with our dataset, we hold out the first 30 s of each test user's data for creating the support set. This leaves approximately 150 s of data per activity which are used for testing, except for "Up Stairs" and "Down Stairs" classes which have about 90 s of test data each.

7 Evaluation

Evaluations are conducted using a hold-out methodology where 8 users were randomly selected for testing and the remaining users' data were used for training. A time window of 5 s is used for signal segmentation and performance is reported using macro-averaged F1 score, a measure of accuracy that considers both precision (the fraction of examples predicted as class c_i that correctly belong to c_i) and recall (the fraction of examples truly belonging to class c_i that are predicted as c_i) for each class. Discrete Cosine Transforms with features are used for data representation.

Our evaluation is composed of two parts. Firstly we explore the performance of our matching network against a number of baseline approaches. Accordingly we compare the following algorithms:

- kNN: Nearest-neighbour classifier trained on the entire training set
- SVM: Support Vector Machines trained on the entire training set
- MLP: A Feed-forward neural network trained on the entire training
- MNet: Our personalised matching network approach

Note that MLP is equivalent to our embedding network with one hidden layer, batch-normalisation and softmax classification layer. The comparison with MLP is meant to provide evidence for the effectiveness of the personalisation approach

[2] http://axivity.com/product/ax3.
[3] https://github.com/selfback/activity-recognition/tree/master/activity_data.

of MNet beyond it's use of the embedding network. Note also that increasing the number of hidden layers beyond one for both MNet and MLP did not produce any improvement in performance. For MNet, we use $n = 6$ examples per class. These parameter values are presented in Table 2.

Table 2. Parameter settings.

Parameter	kNN	SVM	MLP	MNet
Similarity metric/Kernel	Cosine	Gaussian	-	Cosine
Neighbours	10	-	-	6
Hidden layers	-	-	1	1
Hidden units	-	-	120	120
Training epochs	-	-	10	20
Batch size	-	-	64	64
Loss function	-	-	Cross entropy	Cross entropy
Optimiser	-	-	Adam	Adam

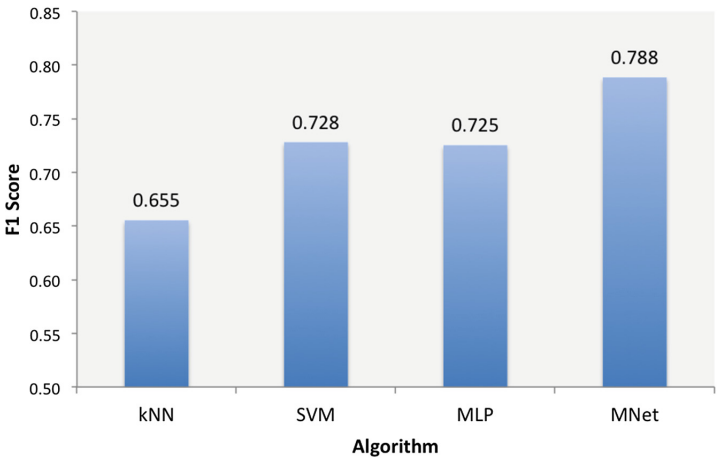

Fig. 6. Evaluation of MNet against popular classifiers.

It can be observed from Fig. 6 that MNet produces the best result; whilst SVM and MLP have comparative performance but kNN comes in last. The poor performance of kNN compared to SVM and MLP is consistent with our previous evaluations [17]. MNet out performs both SVM and MLP by more than 6% which shows the effectiveness of our matching network approach at exploiting personal data for activity recognition.

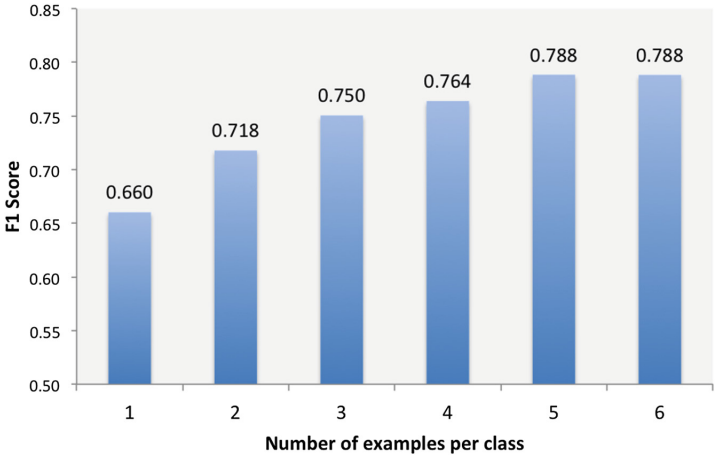

Fig. 7. Results of MNet with different number of samples per class.

The second part of our evaluation explores the influence of the number of examples per class n on classification performance. Recall that the amount "user-provided" data available to us are 30 s per activity. Considering our window length of 5 s, this provides us a maximum of 6 examples per class. Hence, we explore sizes of n from 1 to 6. Results are presented in Fig. 7. It can be observed that results of MNet improve with increase in size of n. However no improvement is observed between $n = 5$ and $n = 6$ which perhaps suggests not much improvement will be gained with continued increase in size of n. Evaluating sizes of n greater than 6 is not feasible with our experiment design and limited data, however, this can be explored further in future work.

A reasonable argument that can be made is that MNet has the added advantage of using end-user supplied data. Therefore, we present a comparison with versions of kNN, SVM and MLP (named kNN+, SVM+ and MLP+ respectively) which are trained on all user provided samples in addition to the entire training set. Results are presented in Fig. 8.

As can be observed, addition of the small number of user samples does not improve performance in kNN+, SVM+ and MLP+. In all three cases, results are approximately the same as those of training on the training set only presented in Fig. 6. An obvious explanation for the lack of improvement is the small size of the user provided data in which case, it can be expected that larger amounts of user data may lead to improved performance. However, the point to note is that the same size of data is sufficient to produce marked improvement in the performance of MNet.

A final point we explored in our evaluation is the significance of creating personalised support sets using the same user's data when training the matching network. In other words, for personalised HAR, do train support sets need to be personalised or can we get similar performance from non-personalised support

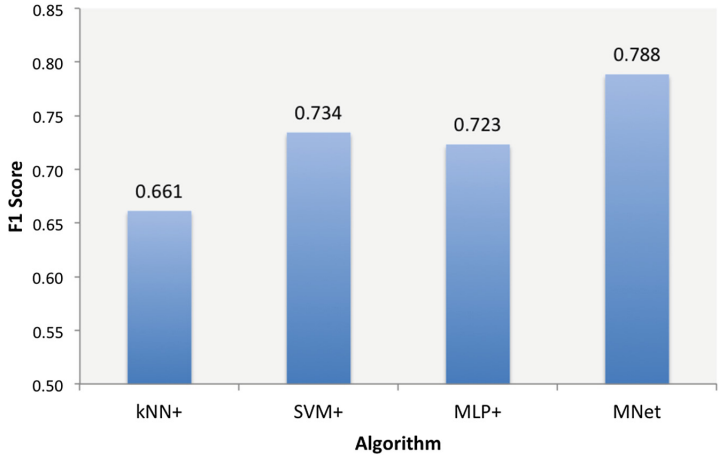

Fig. 8. Evaluation number of examples per class.

sets. The F1 Score using non-personalised support is 0.661 compared to 0.788 with personalised support set. This highlights the importance of matching train conditions to test condition as we have proposed in our methodology.

8 Conclusion

In this paper, we presented a novel approach for personalised HAR using matching networks. Matching networks adopt principles from both metric learning and attention in neural networks to perform effective k-nearest neighbour classification using a small support set of examples. We demonstrated how this support set can be constructed from a small set of labelled examples provided by the user at runtime, which allows the matching network to effectively build a personalised classifier for the user. Evaluation shows our approach to outperform a generals model by at least 6% of F1 score.

There are two main advantages to the approach we presented in this paper. Firstly, our approach is able to achieve high accuracy using only a small set of of user provided examples (30 s in this work) which makes it more practical for real-world applications compared to subject-dependent training which requires the end user to provide large amounts (possible hours) of labelled training data. Secondly, our approach does not require retraining the model at runtime when new data becomes available which makes the approach very adaptable.

The ability of matching networks to learn similarity metrics for particular domains as well as their ability to adapt at runtime make them very relevant for case-based reasoning applications. We hope that this work will inspire further work on adoption of these and similar approaches for application in CBR.

References

1. Abel, M., Hannon, J., Mullineaux, D., Beighle, A.: Determination of step rate thresholds corresponding to physical activity intensity classifications in adults. J. Phys. Act. Health **8**(1), 45–51 (2011)
2. Bach, K., Szczepanski, T., Aamodt, A., Gundersen, O.E., Mork, P.J.: Case representation and similarity assessment in the SELFBACK decision support system. In: Goel, A., Díaz-Agudo, M.B., Roth-Berghofer, T. (eds.) ICCBR 2016. LNCS (LNAI), vol. 9969, pp. 32–46. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47096-2_3
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
4. Bellet, A., Habrard, A., Sebban, M.: A survey on metric learning for feature vectors and structured data. arXiv preprint arXiv:1306.6709 (2013)
5. Berchtold, M., Budde, M., Gordon, D., Schmidtke, H.R., Beigl, M.: ActiServ: activity recognition service for mobile phones. In: Proceedings of International Symposium on Wearable Computers, pp. 1–8 (2010)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
7. Jatoba, L.C., Grossmann, U., Kunze, C., Ottenbacher, J., Stork, W.: Context-aware mobile health monitoring: Evaluation of different pattern recognition methods for classification of physical activity. In: Proceedings of 30th Annual International eConference of the IEEE Engineering in Medicine and Biology Society, pp. 5250–5253 (2008)
8. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: Proceedings of International Conference on Machine Learning Deep Learning Workshop, vol. 2 (2015)
9. Kulis, B.: Metric learning: a survey. Found. Trends Mach. Learn. **5**(4), 287–364 (2013)
10. Lara, O.D., Labrador, M.A.: A survey on human activity recognition using wearable sensors. IEEE Commun. Surv. Tutor. **15**(3), 1192–1209 (2013)
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
12. Longstaff, B., Reddy, S., Estrin, D.: Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In: Proceedings of 4th International Conference on Pervasive Computing Technologies for Healthcare, pp. 1–7 (2010)
13. López-Sánchez, D., Corchado, J.M., Arrieta, A.G.: A CBR system for image-based webpage classification: case representation with convolutional neural networks. In: Proceedings of Florida AI Research Society Conference (2017)
14. López-Sánchez, D., Herrero, J.R., Arrieta, A.G., Corchado, J.M.: Hybridizing metric learning and case-based reasoning for adaptable clickbait detection. Appl. Intell., 1–16 (2017)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)
16. Miu, T., Missier, P., Plötz, T.: Bootstrapping personalised human activity recognition models using online active learning. In: Proceedings of IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, pp. 1138–1147. IEEE (2015)

17. Sani, S., Massie, S., Wiratunga, N., Cooper, K.: Learning deep and shallow features for human activity recognition. In: Li, G., Ge, Y., Zhang, Z., Jin, Z., Blumenstein, M. (eds.) KSEM 2017. LNCS (LNAI), vol. 10412, pp. 469–482. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63558-3_40
18. Sani, S., Wiratunga, N., Massie, S., Cooper, K.: SELFBACK-activity recognition for self-management of low back pain. In: Bramer, M., Petridis, M. (eds.) Research and Development in Intelligent Systems XXXIII: Incorporating Applications and Innovations in Intelligent Systems XXIV, pp. 281–294. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47175-4_21
19. Sani, S., Wiratunga, N., Massie, S., Cooper, K.: kNN sampling for personalised human activity recognition. In: Aha, D.W., Lieber, J. (eds.) ICCBR 2017. LNCS (LNAI), vol. 10339, pp. 330–344. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61030-6_23
20. Sun, X., Kashima, H., Ueda, N.: Large-scale personalized human activity recognition using online multitask learning. IEEE Trans. Knowl. Data Eng. **25**(11), 2551–2563 (2013)
21. Tapia, E.M., Intille, S.S., Haskell, W., Larson, K., Wright, J., King, A., Friedman, R.: Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In: Proceedings of 11th IEEE International Symposium on Wearable Computers, pp. 37–40. IEEE (2007)
22. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D.: Matching networks for one shot learning. In: Proceedings of Advances in Neural Information Processing Systems, pp. 3630–3638 (2016)
23. Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y.: Distance metric learning with application to clustering with side-information. In: Proceedings of Advances in neural information processing systems, pp. 521–528 (2003)