MAML in Data Shift of Wearable-based Human Activity Recognition

ZIYUE XIN, Mcgill University, Canada
KAIYUAN LI, McGill university, Canada

Due to users' highly variant data distribution(data shift), predicting physical activity via wearable-based human activity recognition(HAR) devices is a challenging task. In this paper, we intend to improve activity prediction by presenting a meta-learning-based HAR system. This system identifies 11 different physical activities (eating, writing, standing, etc). The system is composed of two main modules, a data cluster grouping the individual data according to the similarity of characteristics and the model-agnostic meta-learning(MAML) algorithm obtaining a meta-model by training from a variety of learning tasks. This paper focuses on the generalization performance of pre-trained meta-models on target clusters/people. The paper also evaluates effects of different sources of data and different methods of task construction. Ultimately, our best model can achieve 80% of accuracy in testing sets, by success of solving data shift among users

CCS Concepts: • **Computing methodologies** → *Transfer learning*; **Supervised learning by classification**.

Additional Key Words and Phrases: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset, Model-agnostic meta-learning, Physical recognition, Data Shift

ACM Reference Format:
Ziyue Xin and Kaiyuan Li. 2018. MAML in Data Shift of Wearable-based Human Activity Recognition. In . ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In recent years, Human Activity Recognition (HAR) has become a popular research area due to its wide range of promising applications ranging from home care monitoring to obesity prevention, to personalized athletics training. With the increasing use of wearable devices like smartphones and smartwatches, HAR via wearable sensors(accelerometer, gyroscope, etc.) is becoming more relevant and accessible. These multi-sensory wearable devices are frequently in direct contact with the user's body, making them ideal for monitoring users' activity data, as well as measuring their bodies' acceleration and orientation. Therefore, wearable devices are able to ubiquitously identify the physical activity of a person or a group of people based on data and measurements. Till now, Deep learning methods such as CNN and RNN have shown great power and even achieved state-of-the-art recognition results by learning users' characteristics from the observed data. However, because the data distribution of one user is different from others'(data shift), HAR via wearable devices is still a highly dynamic and challenging task. In order to overcome the challenge of data shift, one straightforward way is to train every user's data separately and get the corresponding model. However, with limited time and resources, it's impractical and inefficient in reality to obtain data large enough from every user to train each model, but the models trained with a small amount of data can only get a poor accuracy on tests, which is obviously unsatisfiable.

To address this issue in a practical way, we use model-agnostic meta-learning algorithm(MAML)[6], which firstly

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2018 Association for Computing Machinery.
Manuscript submitted to ACM

obtains a generalized meta-model trained with data from multiple users/groups and then fine-tune our meta-model with a small amount of data from target user/group to get the corresponding model.
In this project, to evaluate the performance of MAML algorithm on the data shift problem, we use *WISDM Smartphone and Smartwatch Activity and Biometrics Dataset* to conduct all the experiments. Our goal is to train a model that can quickly adapt to a new task using only a few data and training iterations. First, we evaluate the performance of implementing the MAML algorithm and obtain the result that the MAML algorithm could effectively solve data shifts among users. Also, we find out the best-outputted model by the experiment of comparison between data from phone and those from watch, and furthermore, we present an idea that clustering by K-means could reduce the model complexity in fine-tuning at a modest cost.

2 DATASET

2.1 Dataset Description

The dataset we used is from "WISDM Smart and Smartwatch Activity and Biometrics Dataset", which includes the raw data collected from 51 objects(users), each of whom are asked to perform 3 minutes, recorded by the accelerometer and gyroscope on both smartphone and smartwatch. After the raw data was collected, the dataset is built by extracting 95 features from every 10's raw data according to dataset descriptions

2.2 Preprocessing

When we do the data preprocessing, we found the following problems:

- For every record of data features, the standard deviation is not matched to variance ($std \neq \sqrt{variance}$), so we decided to remove any data features involved with the standard deviation and variance.
- The datasets collected by phone for users from 1610 to 1618 are all identical, so we decided to only keep the dataset of user 1610.

Also, to maximize the data's utility, we only keep the data of 11 labels(Activities). In this case, we could have 70 records of data for every label in each user. Therefore, the dataset size we used is $70 \times 11 \times 44 = 33880$

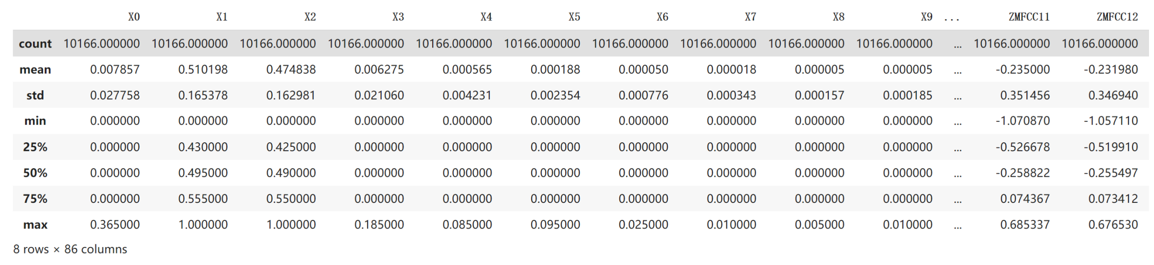
3 METHODOLOGY

3.1 Cluster

According to the dataset preprocessing, we have 44 users' data to be used, which may result in time-consuming training of a distinct fine-tuned model for every user. In order to reduce the model training complexity, we consider employing the algorithm of K-means to cluster users with similar data distributions.

3.1.1 data characteristics. For every data feature in every user, we use 8 data characteristics to define the distribution of each feature. As Figure 1 shows, data characteristics contain the mean, standard deviation, and percentile from 25% to 75%. Hence, In the later clustering implementation, for every user, we have 86(the number of data features) $\times$ 8 data characteristics.

3.1.2 K-means. In the clustering implementation, we employ the algorithm of K-means[9]. The aim of K-means is to minimize the sum of distances between each point and the centroid of its cluster.First, the algorithm would randomly initialize the positions of centroids. Second, it would go through the users one by one, measuring the distance between every user and these centroids, and then group the users with the closest centroid. Third, once it has assigned all of the



	X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	...	ZMFCC11	ZMFCC12
count	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	10166.000000	...	10166.000000	10166.000000
mean	0.007857	0.510198	0.474838	0.006275	0.000565	0.000188	0.000050	0.000018	0.000005	0.000005	...	-0.235000	-0.231980
std	0.027758	0.165378	0.162981	0.021060	0.004231	0.002354	0.000776	0.000343	0.000157	0.000185	...	0.351456	0.346940
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	-1.070870	-1.057110
25%	0.000000	0.430000	0.425000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	-0.526678	-0.519910
50%	0.000000	0.495000	0.490000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	-0.258822	-0.255497
75%	0.000000	0.555000	0.550000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.074367	0.073412
max	0.365000	1.000000	1.000000	0.185000	0.085000	0.095000	0.025000	0.010000	0.005000	0.010000	...	0.685337	0.676530

8 rows × 86 columns

Fig. 1. Data Characteristics for every User

users to their closest cluster centroids, the next step is to compute the centroids of newly formed clusters. The algorithm would repeat the step 2 and 3 until centroids of newly formed clusters do not change. Particularly, we use Constrained K-means[3], which just adds constraints on cluster sizes. For simplicity, we divide the 44 users into 11 clusters, with each cluster containing 4 users. In this case, we only need to train one fine-tuned model for every four users in average.

3.2 meta-learning

According to the characteristic of dataset, we aim to use MAML for Few-Shot Supervised learning. In this section, we discuss the detailed implementation of MAML algorithm.

3.2.1 Base Model. Since the MAML algorithm is compatible with any model trained with gradient descent and over the years deep-learning algorithm have always perform well in classification tasks, we choose to use deep residual net[10] with 18 layers setting to be our base model. The artitechture of model is shown in Figure 2

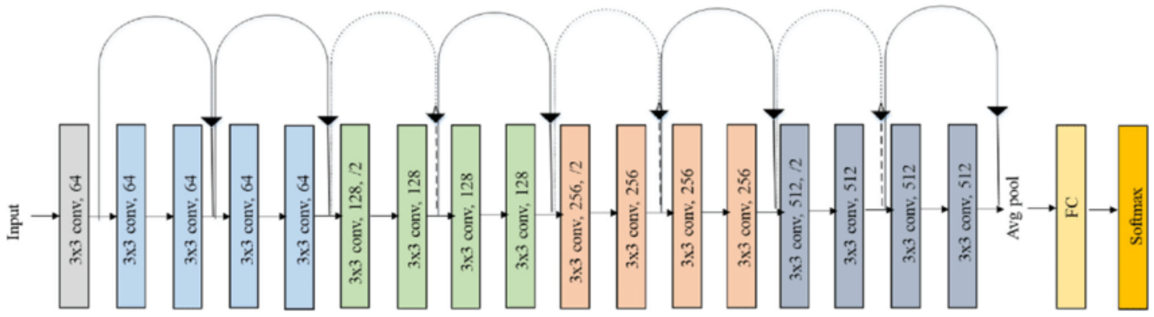


Fig. 2. Diagram of ResNet-18 (https://www.researchgate.net/figure/Original-ResNet-18-Architecture_fig1_336642248).

3.2.2 MAML. In MAML, the goal of the trained meta-model is to adapt the new task from a small amount of new data. In simple words, the meta-model is trained to be able to "learn quickly" on a large number of different tasks $\mathcal{T}$ in meta-dataset $\mathcal{D}$. In effect, the MAML aim to train the model's initial weight by gradient descent such that, when altered in the direction of the gradient of any task sampling from $\mathcal{D}$, small change of initial weight will greatly improve on that loss, as shown in Figure 3

Algorithm 1 MAML

```
Require: Distribution of Meta-dataset: $\mathcal{D}$
Require: Meta learning rate: $\alpha$; Number of meta update: $C$
Require: Learning rate: $\beta$; Number of iteration: $c$
 1: Initialize meta-model $\mathcal{M}_\theta$ with weight $\theta$
 2: for $k \leq C$ do
 3:     Randomly sample number of batches size of train-tasks $\mathcal{T}_i \backsim \mathcal{D}_{train}$
 4:     for all $\mathcal{T}_i$ do
 5:         $\theta' \leftarrow \theta$
 6:         for $j \leq c$ do
 7:             Compute $\nabla_{\theta'}\mathcal{L}_{\mathcal{T}_i}(\mathcal{M}_{\theta'})$
 8:             Inner update: $\theta' = \theta' - \beta \cdot \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\mathcal{M}_\theta)$
 9:         end for
10:         Model for each task $\mathcal{T}_i$ is $\mathcal{M}_{\theta'}$
11:         Sampling $\mathcal{S}_i = \mathbf{X}^j, \mathbf{y}^j$ from $\mathcal{T}_i$
12:     end for
13:     Meta Update with each $\mathcal{S}_i$: $\theta = \theta - \alpha \cdot \nabla_\theta \Sigma_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}(\mathcal{M}_{\theta'})$
14: end for
15: Return $\mathcal{M}_\theta$
```

In this project, we are solving the discrete classification problem, so we use cross-entropy loss which can be optimized by gradient based learning algorithm, and the loss for each task $\mathcal{T}_i$ is defined in Definition 1

$$\mathcal{L}_{\mathcal{T}_i}(\mathcal{M}_\phi) = \sum_{\mathbf{X}^j, \mathbf{y}^j \backsim \mathcal{T}_i} \mathbf{y}^j log(\mathcal{M}_\phi(\mathbf{X}^j)) + (1 - \mathbf{y}^j) log(1 - \mathcal{M}_\phi(\mathbf{X}^j)) \tag{1}$$

Concretely, we consider a ResNet-18 model represented by $\mathcal{M}_\theta$ with initial weight θ. Then, we create C copies of $\mathcal{M}'_\theta$ and we randomly sample C tasks. For each task $\mathcal{T}_i$, we create the copy of $\mathcal{M}_\theta$ and thus compute the gradient of loss(Equation (1)) with respect to support set in $\mathcal{T}_i$. For example, under setting of N-way k-shot, the size of support set is $N \cdot k$, thus the loss is calculated using $N \cdot k$ data points. After we compute the gradient, weight θ is gradient updated c times to be θ' with learning rate β. Overall, the purpose of step 4-11 is as follow:

$$\min_\theta \Sigma_{\mathcal{T}_i \backsim \mathcal{D}_{train}} \mathcal{L}_{\mathcal{T}_i}(\mathcal{M}_{\theta'_i})$$

Therefore, we only update the wight of copied model $\mathcal{M}_{\theta'_i}$ but not the original model so far. The weight of $\mathcal{M}_\theta$ is updated by gradient decent with learning rate α. Unlike the previous process of gradient decent, the loss of each tasks is calculated with respect to data in $\mathcal{S}_i$ in order to prevent overfitting. The full algorithm is outlined in Algorithm 1

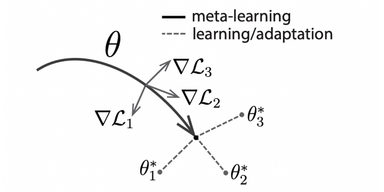


Fig. 3. MAML diagram (https://paperswithcode.com/method/maml).

4 EXPERIMENTS

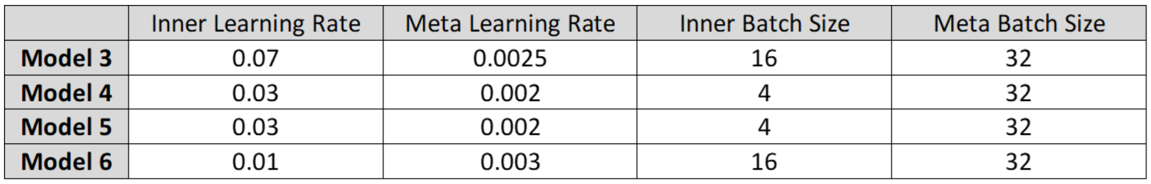
This section describes experiments conducted in this project.

4.1 Model

In this subsection, we describes all the trained models trained in different settings and thus observe the performance of the MAML algorithm, and outputting the best-performance model.

- **Model 1: Base Model(All Data - Resnet18)**
  First, we want to obtain a base model, which is a deep residual net with 18 layers trained by 0.75% of all datasets and tested by the rest data
- **Model 2: Base Model(User - Resnet18)**
  Second, we want to obtain a base model that is trained with only half of the particular user's data and tested by the rest of this user's data.
- **Model 3: Model(User - MAML)**
  Then, we try to get a Meta Model trained with the MAML algorithm, which would be later fine-tuned by target users. It's noteworthy that each task in the model, from training to testing, is merely the data of each user.
- **Model 4: Model(Phone, User - MAML)**
  After that, we want to figure out whether the source of data(phone or watch) would affect the performance of the model. Thus, this model is the same as the third model, except only trained and tested by the data collected by sensors on phone.
- **Model 5: Model(Watch, User - MAML)**
  This model is the same as the third model, except only trained and tested by the data collected by sensors on watch.
- **Model 6: Model(Watch, Cluster - MAML)**
  Then, we want to further reduce the training complexity, so we get a model, which is the same as the fifth model, except each task here is the data of each cluster that is a group of several users.

The hyperparameters in models with the MAML's algorithm are recorded in Figure 4



	Inner Learning Rate	Meta Learning Rate	Inner Batch Size	Meta Batch Size
Model 3	0.07	0.0025	16	32
Model 4	0.03	0.002	4	32
Model 5	0.03	0.002	4	32
Model 6	0.01	0.003	16	32

Fig. 4. Model's Hyperparameters

4.2 Results

In this section, we use the results concluded from above experiments to analyze how the sensor type or the data influences the algorithm.

4.2.1 MAML's Performance. This section mainly discusses the performance of the MAML algorithm in two aspects. In the first experiments, we aim to show the problem that the model is under-fitted if it is trained separately for each user.

20 Users/Tasks	*Average Loss*	*Average Accuracy*
Finetune	1.656	74.1%
Without Finetune	4.812	18%

Table 1. Table to compare meta-model and finetuned model

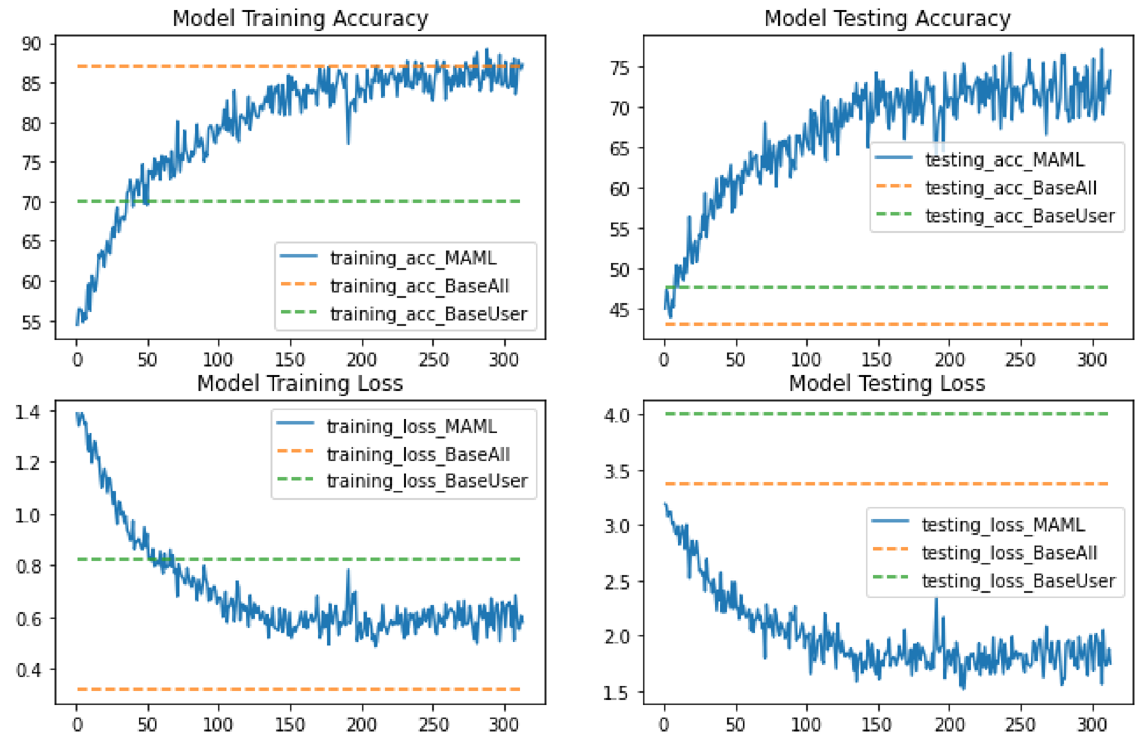


Fig. 5. MAML's performance

We can visualize that the problem exists in Figure 5. The average training loss is 0.8, the testing loss is 4.0, and the poor predicting accuracy also induces that the model is under-fitted in this case. To solve this problem, we utilized the MAML algorithm, which obtains one meta-model using all data in this work, and the setup is: we consider each user as one task. In simple words, the fine-tuned model will adopt toward the distribution of one user/people. From the blue line in Figure 5, we can see that the result given by MAML(Model 3) is significantly better than the base model trained by each user: the testing accuracy(%) difference is 25%(72% − 47%).

The result led us to consider another question: whether the MAML algorithm help solves the data shift problem or not. We compare the result obtained from Model 1 and Model 3. The experimental result shows no significant difference in the training accuracy and loss for the base model and meta-model obtained from the MAML algorithm. However, accuracy and loss are worse than the MAML algorithm when we conduct predicting tasks on testing set/tasks, which is from other users' data. Furthermore, we can see the "learning ability" of the meta-model from Table 1: after five times updates, the training accuracy of the fine-tuned model achieves 76.9% while the training accuracy of the meta-model is only 18%.

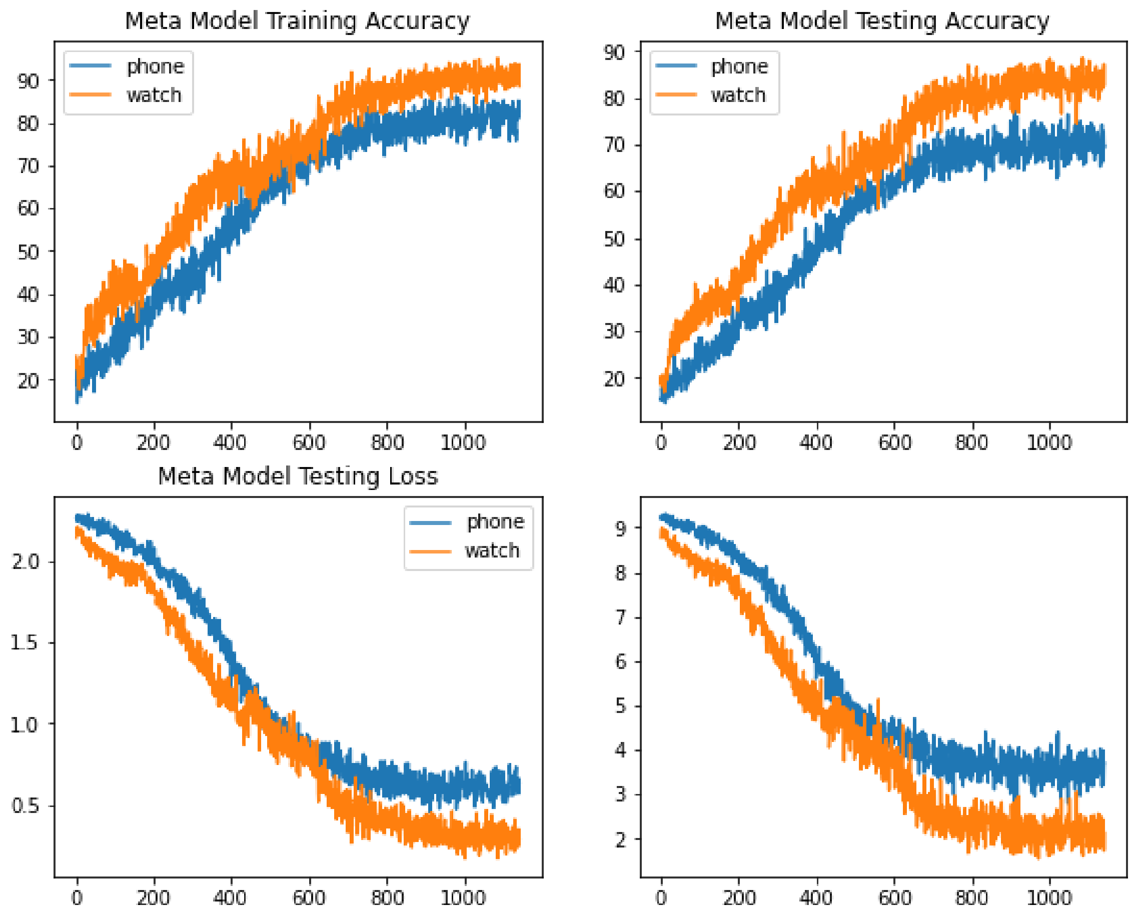


Fig. 6. Phone-Watch Comparison

4.2.2 Phone VS Watch. After the experiments above, we want to figure out whether the source of data can affect the ultimate performance of MAML algorithm in our project. Hence, we compare Mode l4 and Model 5, and find out there exist differences between their performance. Figure 6 shows that Model 5 can achieve a lower loss and better accuracy than Model 4 for both training and testing sets. If Model 5 is applied, we can finally get 80% accuracy on average on testing sets, which is the highest among all the models. Therefore, it can be concluded from such variation that the data collected by sensors on watch are more suitable for activity classification using MAML algorithm than those collected by sensors on phone.

4.2.3 Cluster's Performance. After that, we want to reduce the model training complexity further and only choose to use the data collected by sensors on watch this time because of the result of the last experiment, so we have the

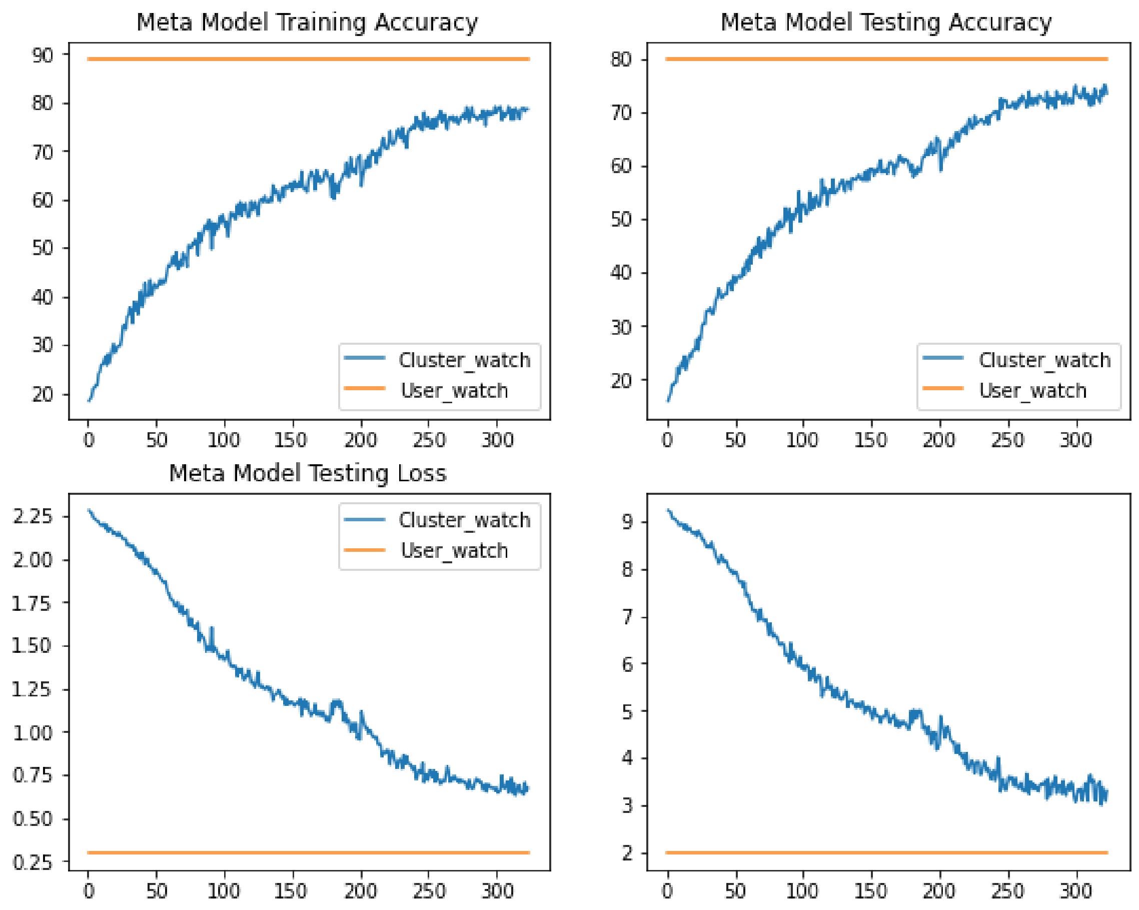


Fig. 7. The performance of Model6

Model 6, trained with the data from each task in each cluster. The Figure 7 shows that ultimately this model achieves an accuracy of 75% in testing sets, which is only 5% lower than the Model 5. Nevertheless, it is noteworthy that since we group every four users' data into one group, Model 6 only needs to fine-tune a distinct model for every four testing users , while our models before needed to fine-tune a distinct model for every user. Hence, Model6 saves four times model complexity in fine-tuning at the modest cost of slightly lower accuracy, which is inspiring.

5 RELATED WORK

In this section, we only discuss the other meta-learning method and other ways to solve data shift problem.

5.1 Historical Context of Meta-learning

Meta-learning first appears in the literature in 1987[13] which refers to learning algorithms that learn from other algorithms, and it observes the performance of the algorithm on a wide range of learning tasks. Therefore, in contrast to the traditional machine learning algorithm that aims to find the intrinsic pattern of data, task similarity influences the evaluation of the meta-learning algorithm. One way to define task similarity is by calculating the KL-divergence[11] between the distribution of two tasks. J' Schmidhuber proposed to learn the model itself using *self-referential* algorithms. Then Bengio propose the meta-learning rule[2]. Training meta-learning based on gradient descent technique and backpropagation were first presented by Schmidhuber in 1991, and more extensions are followed in [7], [6], [8].

5.2 Method for Data Shift

Transfer learning[12] uses experience from a source task to model's performance on the target task, like layer-freezing and finetuning[14]. Domain adaptation[5] is a variant of transfer learning that tries to mitigate the data shift problem by adapting the trained model with unlabeled data from the target set. One of the state-of-art domain-adaptation techniques that solve the data shift problem in wifi-based localization problem is proposed in [4].

6 CONCLUSION

As stated above, we employ the MAML algorithm in HAR via wearable devices. From the Result section, we did some experiments, and show that the implementation can both resolve the under-fitting in the model trained separately with each user's data and overcome the data shift in the model trained with multiple users' data together. Moreover, we also figure out that the data collected by sensors on watch is more fitted to the implementation of the MAML algorithm, and develop an idea that utilization of Clustering by K-means can significantly reduce the model complexity in fine-tuning for MAML implementation.

6.1 Future Work

It has been shown that our best model can achieve 80% percent of accuracy in testing sets, which we search to improve further. The possible reason for constraining the model's performance is that the data of each participant is still not large enough, which can not give full support to the model. In this case, we would find ways to augment data, including Variational Autoencoders (VAEs)[1]. If each participant's data is more adequate than those now, it can be expected that the model of the MAML algorithm can perform better.

REFERENCES

[1] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE* 2, 1 (2015), 1–18.

[2] Yoshua Bengio, Samy Bengio, and Jocelyn Cloutier. 1990. *Learning a synaptic learning rule.* Citeseer.

[3] Paul S Bradley, Kristin P Bennett, and Ayhan Demiriz. 2000. Constrained k-means clustering. *Microsoft Research, Redmond* 20, 0 (2000), 0.

[4] Xi Chen, Hang Li, Chenyi Zhou, Xue Liu, Di Wu, and Gregory Dudek. 2020. Fido: Ubiquitous fine-grained wifi-based localization for unlabelled users via domain adaptation. In *Proceedings of The Web Conference 2020.* 23–33.

[5] Gabriela Csurka et al. 2017. *Domain adaptation in computer vision applications.* Springer.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. https://doi.org/10.48550/ARXIV.1703.03400

[7] Chelsea Finn and Sergey Levine. 2017. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. *arXiv preprint arXiv:1710.11622* (2017).

[8] Sebastian Flennerhag, Andrei A Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell. 2019. Meta-learning with warped gradient descent. *arXiv preprint arXiv:1909.00025* (2019).

[9] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)* 28, 1 (1979), 100–108.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. https://doi.org/10.48550/ARXIV.1512.03385

[11] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[12] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

[13] Jürgen Schmidhuber and AI Blog. 2020. Metalearning Machines Learn to Learn (1987-). *URL https://people. idsia. ch/juergen/metalearning. html. https://people. idsia. ch/juergen/metalearning. html* (2020).

[14] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? (2014). https://doi.org/10.48550/ARXIV.1411.1792