MultiMix: A Multi-Task Deep Learning Approach for Travel Mode Identification with Few GPS Data

Xiaozhuang Song, Christos Markos, and James J.Q. Yu, *Member, IEEE*

***Abstract*— Understanding how people choose to travel is essential for intelligent transportation planning and related smart services. Recent advances in deep learning, coupled with the increasing market penetration of GPS devices, have paved the way for novel travel mode identification methods based on GPS data mining. While many have shown promising results, most methods have often relied heavily on the few available labeled data, leaving large amounts of unlabeled ones unused. To address this issue, we propose MultiMix, a semi-supervised multi-task learning framework for travel mode identification. Our framework trains a deep autoencoder using batches of labeled, unlabeled, and synthetic data by simultaneously optimizing three corresponding objective functions. We show that MultiMix outperforms several fully- and semi-supervised baselines, achieving a classification accuracy of 66.2% on Geolife using just 1% of labeled data, with accuracy reaching 84.8% when incorporating all available labels. We also verify the necessity of its components through an ablation study designed to provide insights into the proposed approach.**

I. Introduction

The objective of travel mode identification is to discover users' modes of travel by analyzing their mobility data. Such information is key for Intelligent Transportation Systems (ITSs) [1], [2], guiding the optimization of complex processes like transportation scheduling and smart service delivery [3]. Ultimately, successful travel mode identification could improve a number of significant issues faced by modern cities, including traffic accidents, congestion, and pollution. To do so, however, requires both fine-grained mobility data and powerful learning models [4].

In the earlier days of transportation research, mobility data were obtained through expensive and tedious methods such as questionnaires or telephone interviews [5]. However, these approaches depend on travelers' memory and motivation, often leading to unreliable data. Although later studies leveraged more accurate data produced by fixed-position sensors, they were still limited by narrow city coverage, as such sensors require significant installation and maintenance costs. Recently, with the integration of Global Positioning System (GPS) sensors into most smartphones, acquisition of user travel data has become ubiquitous and cost-effective [6], [7]. This has promoted research on various transportation-related issues [8].

This work is supported in part by the General Program of Guangdong Basic and Applied Basic Research Foundation No. 2019A1515011032 and in part by Guangdong Provincial Key Laboratory No. 2020B121201001. *(Corresponding author: James J.Q. Yu.)*

The authors are with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. Christos Markos is also with the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia.

GPS data have also driven a growing body of travel mode identification research. Typically, since raw GPS data are unsuitable for training machine learning models, preprocessing starts with extraction of motion-related features such as velocity and acceleration. These are then fed to machine learning models, e.g. support vector machines or decision trees [9]. More recently, several studies have also explored deep learning models including multi-layer perceptrons [10], as well as recurrent [11], [12] or Convolutional Neural Networks (CNNs) [13], often achieving better classification performance. However, to the best of our knowledge, most existing work on travel mode identification from GPS data has only used labeled data in fully-supervised training.

Although effective, training only with labeled data is not without shortcomings. First, acquisition of enough labeled data to train deep neural networks requires considerable time and human effort. Second, the process of labeling may contaminate datasets with false labels, thereby diluting the effectiveness of model training. Third, unlabeled data are just as likely to contain useful information as labeled data, hence ignoring them means not utilizing all available resources. Indeed, several semi-supervised training approaches in other fields have shown that further mining the information in unlabeled data and using them together with labeled ones can improve models' generalization capacity. Examples include adding noise to unlabeled data for *consistency regularization* [14], [15], generating pseudo-labels for unlabeled data and performing supervised training [16], [17], averaging model parameters during training [18], [19], or jointly learning from labeled and unlabeled data [20], [21]. In the area of travel mode identification, the authors in [22] were among the first to include unlabeled data for semi-supervised training of a convolutional autoencoder. As GPS sensors only capture timestamped positional information, having no knowledge of users' travel modes, it is crucial to develop travel mode identification methods that produce highly accurate results without being overly reliant on labeled GPS data.

In this work, we propose MultiMix, a multi-task deep learning approach to travel mode identification. MultiMix is trained using batches of the few available labeled data, mixed with the much larger amount of unlabeled ones. This particular form of data augmentation [21], combined with the three corresponding learning tasks, results in MultiMix significantly outperforming several fully- and semi-supervised approaches. Using only 1% labeled data, MultiMix attains 66.2% accuracy on the Geolife dataset [23], while inclusion of all available labels boosts accuracy to 84.8%. Finally, we conduct an ablation study to quantify the effect of the pro-

posed framework's components on classification accuracy.

The rest of this paper is organized as follows. Section II presents the techniques that were used for feature extraction, division of trajectories into fixed-size segments, and outlier removal. In Section III, we explain our method for synthetic data generation and analyze the components of MultiMix. Section IV presents the results of our experiments, as well as an ablation study measuring the contribution of individual components to the success of the proposed approach. Finally, conclusions and future work are presented in Section V.

II. Preliminaries

This section first introduces the methods used for extracting motion-related features from GPS data and removing anomalous instances. Then, it describes the process of trip segmentation, whereby GPS trajectories are divided into segments of the same length. This is a necessary step, as the deep convolutional architecture of MultiMix requires fixed-size input.

A. Feature Extraction and Outlier Removal

The raw GPS trajectory data used in this work are organized in sequences of triples containing timestamps, latitudes, and longitudes. We define GPS point $p_i = (lat_i, long_i, t_i)$, where lat_i and $long_i$ are its latitude and longitude in decimal degrees, as captured at date and time t_i. However, raw spatio-temporal information is not ideal for training machine learning models. Indeed, for such a model to generalize to any location, it would need to be trained on samples containing all possible latitudes and longitudes.

In the travel mode identification literature, it is thus common to extract motion-related features such as relative distance or velocity from the available GPS data. To obtain the former, we use Vincenty's formula [24], which calculates the distance between any two points on a sphere. It is also trivial to calculate the elapsed time from p_j to p_{j+1}, by subtracting their timestamps. In this work, we use relative distance, velocity, acceleration, and jerk as input features, inspired by [22]. For p_j, they are calculated as follows:

$$\Delta t_j = t_{j+1} - t_j, \tag{1}$$

$$\mathrm{RD}_j = \mathrm{Vincenty}(lat_j, long_j, lat_{j+1}, long_{j+1}), \tag{2}$$

$$V_j = \mathrm{RD}_j / \Delta t_j, \tag{3}$$

$$A_j = (V_{j+1} - V_j)/\Delta t_j, \tag{4}$$

$$J_j = (A_{j+1} - A_j)/\Delta t_j, \tag{5}$$

where Δt_j, RD_j, V_j, A_j, and J_j represent the elapsed time, relative distance, velocity, acceleration and jerk of p_j, respectively.

Often, signal interference can cause GPS sensors to transmit inaccurate information. This may occur as a result of proximity to tunnels, airplanes, or even high cloud density. Thus, before using the extracted motion features to train our model, it is necessary to first remove any abnormal instances in each segment [25]. For labeled data, we set velocity and acceleration thresholds for each travel mode, as per [22]. Any instances that do not satisfy these thresholds are removed. For unlabeled data, since we have no knowledge of their corresponding travel modes, we remove instances whose velocity or acceleration exceeds or falls short of 1.5 times the interquartile range between the first and third quartiles; this was also inspired by [22]. Finally, we discard any out-of-order GPS data points, that is, points whose timestamp exceeds that of the next point.

B. Trip Segmentation

As will be detailed in Section III-B, MultiMix uses a CNN-based architecture. By default, CNNs require fixed-size input; hence, we need to segment the arbitrary-length trajectory data into fixed-size blocks. A naive approach would be to extract a new segment every M instances. In doing so, however, there could be more than one travel mode in several of the obtained segments. In such cases, even if we considered the dominant travel mode to be the final one, we would be introducing unnecessary noise to the data. Therefore, before dividing the available GPS trajectories into segments of fixed length, we must first divide them by transportation mode.

Inspired by the seminal work of [26], [27], we first preprocess labeled trajectories by splitting them at the GPS points where the travel mode changes, according to the available labels. This process results in GPS segments of varying lengths, denoted as sequences $Seg =< p_1, ..., p_n >$, where $label(p_i) = label(Seg)$ for every $p_i \in Seg$. Then, to satisfy the fixed-size input requirement for CNNs, we further split the above segments into segments containing exactly M points each. For segments with length less than M, we artificially increase the number of points using a zero-padding strategy. Given that segments with very few points might not be informative enough to identify the corresponding travel mode, a minimum threshold is also set here; if the segment length before zero-padding is less than this minimum threshold, the segment is discarded. As in [22] this threshold is set to 20 points, while M is set to 248.

III. Proposed Methodology

MultiMix simultaneously optimizes the sum of three hyperparameter-controlled objective functions, which are used to incorporate labeled, unlabeled, and synthetic data during training. In this section, we first introduce the methods for generating synthetic data. Next, we analyze the three models that compose MultiMix and their respective loss functions, and conclude by explaining how they are combined for multi-task learning.

A. Synthetic Data Generation

The design choice of training on synthetic data is inspired by the work of [20], [21]. In [20], the authors proposed Mixup, a supervised method that achieved better performance than merely training on labeled data by learning from both labeled data and linear interpolations thereof. This practice was shown to improve model generalization, as well as sensitivity to adversarial samples and false labels [28]. In [21], the authors extended Mixup by proposing MixMatch, a

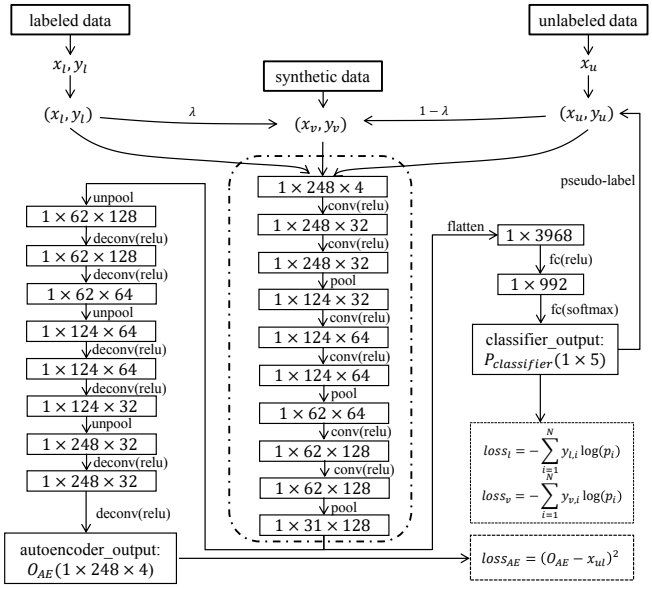

Fig. 1. The architecture of MultiMix.

semi-supervised method which produced synthetic data not only from labeled but also from unlabeled data. The inclusion of the latter in the process of synthetic data generation resulted in significant performance gains compared to the original approach [20].

In this work, we obtain synthetic data by simply mixing unlabeled and labeled data in each batch during training. Synthetic samples x_s and labels y_s are generated as follows:

$$x_s = \lambda x_u + (1-\lambda)x_l, \tag{6}$$

$$y_u = P_{\text{Classifier}}(y|x_u;\theta), \tag{7}$$

$$y_s = \lambda y_u + (1-\lambda)y_l, \tag{8}$$

where real-valued hyperparameter $\lambda \in [0,1]$, x_l are samples with corresponding labels y_l, and x_u are unlabeled samples. On every training iteration, pseudo-labels y_u are set to the class index of the the maximum softmax value in the model's output. This output is the model's predicted distribution $P_{\text{Classifier}}(y|x_u;\theta)$, where θ refers to the model's learned parameters, i.e. layer weights and biases.

B. Multi-Task Learning

1) Labeled Data Classifier: The architecture of the labeled data classifier is inspired by [13]. It consists of convolution, max pooling, and fully-connected layers. Every combination of two convolution layers and a pooling layer may be regarded as a module; by adopting this view, the main structure of the network consists of three such modules. The input layer dimensions are $(1, M, 4)$, while the filter size in each convolution layer is (1×3) with a unit stride. We set the number of filters in the first module to be 32, doubling with every subsequent module. Since we apply "same" convolutions, we use zero-padding to ensure that channel height and width are not changed by the convolution operation. In each module, a pooling layer is connected to two convolution layers. The filter size of the pooling layer is (1×2), and the stride is 2. The last module is connected to a fully connected layer, whose output passes through a dropout layer before finally being fed into a final softmax layer [29]. The latter generates a probability distribution over the classes, i.e. $P = \{p_1, ..., p_N\}$, where N is the number of travel modes; in this work, $N = 5$. The labeled data classifier uses the categorical cross entropy loss function:

$$\text{loss}_l = -\sum_{i=1}^{N} y_{l,i} log(p_i), \tag{9}$$

where $y_{l,i}$ is the binary ground-truth value corresponding to class i in the one-hot label y_l of Seg_i.

2) Synthetic Data Classifier: The synthetic data classifier uses the exact same structure as the labeled data classifier, and both are trained concurrently while sharing layer parameters. The synthetic data classifier also uses the categorical cross entropy loss function:

$$\text{loss}_v = -\sum_{i=1}^{N} y_{v,i} log(p_i), \tag{10}$$

where $y_{v,i}$ is the binary pseudo-ground-truth value corresponding to class i in the one-hot pseudo-label y_v of Seg_i.

3) Autoencoder: To integrate the unlabeled data, we perform unsupervised training of an autoencoder model. Autoencoders are neural networks typically trained to reconstruct their input under deliberate limitations that promote learning useful features [30]. They generally consist of two symmetrical parts: an encoder and a decoder. The encoder outputs a latent representation of the input, which the decoder uses to try and reconstruct the original input.

Autoencoders may use fully-connected, convolutional, or recurrent layers. In this work, we adopt a convolutional autoencoder architecture similar to [22]. The encoder consists of consecutive convolutional and max pooling layers, while the decoder instead has deconvolutional and upsampling layers. During training, the autoencoder is fed with unlabeled data x_u, while its loss function is defined as:

$$\text{loss}_{\text{AE}} = (o_{\text{AE}} - x_u)^2, \tag{11}$$

where o_{AE} refers to the autoencoder's output, i.e. the reconstructed input.

4) Unified Model: The proposed method trains a labeled data classifier, an unlabeled data classifier, and an autoencoder, all at the same time. Their model parameters are shared during training; in multi-task learning, this is referred to as *hard parameter sharing* [31]. Our multi-task learning method is mainly achieved by minimizing the combination of these three loss functions. Specifically, the total loss function can be expressed as the weighted sum of equations (9), (10), and (11):

$$\text{loss}_{\text{total}} = \alpha\,\text{loss}_l + \beta\,\text{loss}_v + \gamma\,\text{loss}_{\text{AE}}, \tag{12}$$

where hyperparameters α, β, and γ are used to balance the strength of loss_l, loss_v, and loss_{AE}, respectively.

TABLE I

ACCURACY FOR PERCENTAGES OF LABELED DATA

Method	1%	5%	10%	25%	50%	100%
Supervised-KNN	0.481	0.560	0.572	0.536	0.582	0.579
Supervised-SVM	0.531	0.570	0.585	0.613	0.647	0.654
Supervised-RNN	0.490	0.583	0.605	0.751	0.718	0.796
Supervised-DT	0.478	0.551	0.589	0.600	0.618	0.630
Supervised-CNN	0.571	0.612	0.701	0.756	0.796	0.824
Semi-Two-Steps	0.543	0.527	0.556	0.534	0.575	0.593
SECA	0.612	0.655	0.734	0.757	0.808	0.837
MultiMix	**0.662**	**0.698**	**0.742**	**0.787**	**0.821**	**0.848**

TABLE II

F1-SCORE FOR PERCENTAGES OF LABELED DATA

Method	1%	5%	10%	25%	50%	100%
Supervised-KNN	0.517	0.519	0.540	0.525	0.559	0.568
Supervised-SVM	0.443	0.483	0.519	0.569	0.620	0.626
Supervised-RNN	0.388	0.478	0.512	0.734	0.684	0.767
Supervised-DT	0.482	0.553	0.586	0.600	0.620	0.631
Supervised-CNN	0.467	0.612	0.679	0.733	0.786	0.820
Semi-Two-Steps	0.446	0.423	0.451	0.434	0.478	0.494
SECA	0.537	0.633	0.712	0.743	0.801	0.831
MultiMix	**0.625**	**0.650**	**0.726**	**0.775**	**0.815**	**0.832**

IV. EXPERIMENTS

In this section, we first introduce the details of our experimental setup. We then compare the performance of our model with that of established supervised baselines, as well as recent semi-supervised ones specifically proposed for travel mode identification. Finally, as MultiMix combines a variety of components, we conduct an ablation study to evaluate their impact on the selected performance metrics.

A. Implementation Details

We evaluate MultiMix on Microsoft's Geolife dataset [23], which has been commonly used in transportation research [3]. It contains 17,621 GPS trajectories, of which only a small fraction are labeled by travel mode. Furthermore, even though Geolife contains multiple classes, the majority do not have enough data to train a deep learning model. Therefore, we select five traffic modes, namely "walk", "bike", "bus", "drive" and "train" as the modes to be identified.

After preprocessing the data as described in Section II, we obtain 14,424 labeled segments and 135,573 unlabeled ones. Using 5-fold cross-validation, we select 80% and 20% of the labeled segments as the training and test sets, respectively. In addition, 10% of the training set is extracted and subsequently used as a validation set for hyperparameter tuning. All unlabeled segments are used during training.

Except for our model's final softmax layer, all others others use the Rectified Linear Unit (ReLU) function as their activation function. The dropout layer before the softmax one uses a dropout rate of 0.4 to reduce overfitting. When training, we empirically set hyperparameters α, β, γ, and λ to 1.0, 0.5, 1.0, and 0.8, respectively; these were found to perform best on a separate validation set. We minimize $\mathrm{loss}_{\mathrm{total}}$ using the default Adam optimizer, and train MultiMix for a maximum of 50 epochs; using the same validation set as above, we found that training usually converged in about 20 epochs. The experiments were conducted on a server with an Intel Xeon Silver 4210 CPU and eight NVIDIA RTX 2080Ti GPUs. The simulation was implemented using Python and the proposed network structure was modeled with the TensorFlow machine learning platform.

B. Baselines and Evaluation Metrics

We select seven methods used in [9], [13], [22], [32] as our baselines. These are five supervised methods, i.e. K-Nearest Neighbors (KNN) [9], Support Vector Machine (SVM) [9], Decision Tree (DT) [9], Recurrent Neural Network (RNN) with the long short-term memory module [32], CNN [13], and two semi-supervised methods, namely Semi-Two-Steps and SEmi-supervised Convolutional Autoencoder (SECA) [22]. Semi-Two-Steps is implemented using the same convolutional autoencoder proposed in [22]. First, it pretrains the autoencoder on labeled and unlabeled samples, then extracts the learned data representation from the encoder's final layer, and finally uses that representation for supervised autoencoder training by attaching a softmax layer.

We evaluate classification performance using the established accuracy and F1-score metrics. Accuracy is measured as the ratio of correctly identified samples to the total number of samples in the dataset:

$$\mathrm{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{13}$$

where TP, FP, TN, FN are the number of true positives, false positives, true negatives, and false negatives.

The F1-score, defined as a function of the precision and

TABLE III
MULTIMIX CONFUSION MATRIX

		Predicted Mode					Recall
		Walk	**Bike**	**Bus**	**Drive**	**Train**	
True Mode	**Walk**	1094	19	3	1	0	0.979
	Bike	55	454	20	7	1	0.845
	Bus	48	32	727	52	11	0.836
	Drive	23	12	153	320	26	0.599
	Train	17	8	27	33	466	0.846
Precision		0.884	0.865	0.782	0.775	0.925	

recall metrics, is calculated as follows:

$$\text{precision} = \frac{\text{TP}}{\text{TP}+\text{FP}}, \tag{14}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}, \tag{15}$$

$$\text{F1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{16}$$

C. Experimental Results

As shown in Tables I and II, MultiMix achieved higher accuracy than all evaluated baselines when using any percentage of labeled data. It is evident that all supervised methods performed poorly when trained on just 1% or even 5% of the labeled data. The only supervised model that significantly benefited from the inclusion of more labeled data appears to be the CNN.

Among the semi-supervised baselines, Semi-Two-Steps did not achieve competitive results compared to SECA. Furthermore, MultiMix outperformed SECA for all percentages of labeled data, with its advantage being particularly prominent when using just 1% of labeled data. In this case, both the accuracy and F1-score of MultiMix were higher than those of SECA at 66.2% and 62.5%, respectively. When using all labeled data, MultiMix still achieved the best results with an accuracy of 84.8% and F1-score of 83.2%.

For more insights into the per-class performance of MultiMix when trained on all labeled data, Table III shows the confusion matrix as obtained by evaluating our framework on the test set. We note that classes for which Geolife contains more labeled samples generally achieved higher precision and recall than those with fewer ones. This is part of the reason why MultiMix did better at identifying "walk" instances than "drive" ones. On the other hand, it is also due to pedestrians and cars having quite dissimilar motion patterns. It is possibly for the same reason that, although "bike" has about half the number of samples than "walk" does, MultiMix still demonstrated high precision and recall for the former. As also noted by [22], the relatively high precision and recall for buses and trains may be due to the fact that these travel modes tend to follow predefined routes. However, cars were most often mistaken for buses; this is not surprising, as both may have similar velocity and acceleration in congested traffic scenarios.

D. Ablation Study

The performance of MultiMix depends on multiple factors. First, it incorporates synthetic training samples generated using both labeled and unlabeled data. Second, it is trained by optimizing the sum of three losses. To evaluate the influence of the above on classification accuracy, we iteratively remove one of them while maintaining the rest and train MultiMix using 1%, 10%, 25%, and 100% of the available labeled data. These ablations are defined as follows:

- **A:** Generate synthetic data using only labeled data x_l, y_l.
- **B:** Generate synthetic data using only unlabeled data x_u.
- **C:** Remove the loss of the synthetic data classifier, loss_v, from the total loss.
- **D:** Remove the loss of the autoencoder, loss_{AE}, from the total loss.
- **E:** Remove the loss of the labeled data classifier, loss_l, from the total loss.

The results of our ablation study are shown in Table IV. It seems that generating the synthetic data only from either labeled or unlabeled ones caused a decrease in accuracy for all percentages of labeled data. Specifically, between ablations A and B, we note that the former resulted in the biggest reduction in accuracy. This could be due to the fact that the labeled data were fewer than the unlabeled ones to begin with, meaning that in ablation A, the synthetic data were generated from a distribution having fewer samples. This intuition is supported by the results of ablation B, demonstrating accuracy almost as high as that of MultiMix. Since our proposed approach achieved higher accuracy than ablations A and B, especially when using 1% labeled data, it is evident that synthetic data should be generated from all available data.

Next, we evaluated the impact of removing either of the proposed loss terms on the accuracy. According to the experimental results, this also caused varying degrees of accuracy reduction. It seems that MultiMix is most robust to the removal of loss_v and loss_{AE}, while being very sensitive to the elimination of loss_l. Indeed, we note that disregarding all labeled data in the training set (ablation E) resulted in the model essentially producing random guesses, with an accuracy roughly corresponding to the inverse number of classes. Therefore, all three proposed loss terms contribute significantly to the performance of the proposed framework, especially in the case of few labeled data.

V. CONCLUSIONS

In this work, we proposed a multi-task learning framework for deep semi-supervised travel mode identification from GPS data. To address the problem of classifying users' travel modes from few labeled trajectories, MultiMix was trained on mixed batches of labeled, unlabeled, and synthesized data by minimizing the weighted sum of three corresponding loss functions. Compared with previous approaches, both supervised and semi-supervised, MultiMix exhibited the best

TABLE IV
ACCURACY FOR ABLATIONS AND PERCENTAGES OF LABELED DATA

Ablation	1%	10%	25%	100%
A	0.642	0.739	0.758	0.839
B	0.654	0.752	0.781	0.841
C	0.613	0.732	0.755	0.838
D	0.592	0.713	0.741	0.825
E	0.183	0.201	0.192	0.205
MultiMix	**0.662**	**0.758**	**0.787**	**0.848**

overall performance; on Geolife, it achieved an accuracy of 66.2% while only using 1% of the available labeled data. In addition, when leveraging all labels, its accuracy reached 84.8%. We also conducted an ablation study, which showed that removing any of the proposed model components resulted in lower accuracy. In future work, we will explore new ideas in semi-supervised learning to improve MultiMix, and will assert its effectiveness on datasets from other research areas, i.e. in transfer learning.

REFERENCES

[1] L. Wu, B. Yang, and P. Jing, "Travel mode detection based on gps raw data collected by smartphones: a systematic review of the existing methodologies," *Information*, vol. 7, no. 4, p. 67, 2016.

[2] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[3] S. Dabiri and K. Heaslip, "Transport-domain applications of widely used data sources in the smart transportation: A survey," *arXiv preprint arXiv:1803.10902*, 2018.

[4] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *Proceedings of the 10th International Conference on Ubiquitous Computing*. ACM, Seoul, Korea, 2008, pp. 312–321.

[5] J. L. Wolf, "Using gps data loggers to replace travel diaries in the collection of travel data," Ph.D. dissertation, Citeseer, 2000.

[6] L. Gong, T. Morikawa, T. Yamamoto, and H. Sato, "Deriving personal trip data from gps data: A literature review on the existing methodologies," *Procedia-Social and Behavioral Sciences*, vol. 138, no. Supplement C, pp. 557–565, 2014.

[7] L. Montini, S. Prost, J. Schrammel, N. Rieser-Schüssler, and K. W. Axhausen, "Comparison of travel diaries generated from smartphone data and dedicated gps devices," *Transportation Research Procedia*, vol. 11, pp. 227–241, 2015.

[8] M. J. Duncan, H. M. Badland, and W. K. Mummery, "Applying gps to enhance understanding of transport-related physical activity," *Journal of Science and Medicine in Sport*, vol. 12, no. 5, pp. 549–556, 2009.

[9] Z. Xiao, Y. Wang, K. Fu, and F. Wu, "Identifying different transportation modes from trajectory data using tree-based ensemble classifiers," *ISPRS International Journal of Geo-Information*, vol. 6, no. 2, p. 57, 2017.

[10] Y. Endo, H. Toda, K. Nishida, and A. Kawanobe, "Deep feature extraction from trajectories for transportation mode estimation," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, Auckland, New Zealand, 2016, pp. 54–66.

[11] T. H. Vu, L. Dung, and J.-C. Wang, "Transportation mode detection on mobile devices using recurrent nets," in *Proceedings of the 24th ACM international conference on Multimedia*, Amsterdam, The Netherlands, 2016, pp. 392–396.

[12] J. V. Jeyakumar, E. S. Lee, Z. Xia, S. S. Sandha, N. Tausik, and M. Srivastava, "Deep convolutional bidirectional lstm based transportation mode recognition," in *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, New York, USA, 2018, pp. 1606–1615.

[13] S. Dabiri and K. Heaslip, "Inferring transportation modes from gps trajectories using a convolutional neural network," *Transportation research part C: emerging technologies*, vol. 86, pp. 360–371, 2018.

[14] M. Sajjadi, M. Javanmardi, and T. Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," in *Advances in Neural Information Processing Systems*, Barcelona, Spain, 2016, pp. 1163–1171.

[15] M. K. Takeru Miyato, Shin-ichi Maeda and S. Ishii, "Synthetic adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[16] P. Bachman, O. Alsharif, and D. Precup, "Learning with pseudo-ensembles," in *Advances in Neural Information Processing Systems*, Montreal, Canada, 2014, pp. 3365–3373.

[17] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on Challenges in Representation Learning, 30th International Conference on Machine Learning*, vol. 3, Atlanta, USA, 2013, p. 2.

[18] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semisupervised deep learning results," *In Advances in Neural Information Processing Systems*, pp. 1195–1204, Long Beach, California, USA, 2017.

[19] B. Athiwaratkun, M. Finzi, P. Izmailov, and A. G. Wilson, "There are many consistent explanations of unlabeled data: Why you should average," in *7th International Conference on Learning Representations*, New Orleans, Louisiana, USA, 2019.

[20] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, Vancouver, British Columbia, Canada, 2018.

[21] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Advances in Neural Information Processing Systems*, Vancouver, British Columbia, Canada, 2019, pp. 5050–5060.

[22] S. Dabiri, C.-T. Lu, K. Heaslip, and C. K. Reddy, "Semi-supervised deep learning approach for transportation mode identification using gps trajectory data," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[23] Y. Zheng, X. Xie, W.-Y. Ma, *et al.*, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data(base) Engineering Bulletin*, vol. 33, no. 2, pp. 32–39, 2010.

[24] T. Vincenty, "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations," *Survey Review*, vol. 23, no. 176, pp. 88–93, 1975.

[25] L. Shen and P. R. Stopher, "Review of gps travel survey and gps data-processing methods," *Transport Reviews*, vol. 34, no. 3, pp. 316–334, 2014.

[26] Y. Zheng, L. Liu, L. Wang, and X. Xie, "Learning transportation mode from raw gps data for geographic applications on the web," in *Proceedings of the 17th International Conference on World Wide Web*. ACM, Beijing, China, 2008, pp. 247–256.

[27] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, p. 107299, 2019.

[28] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang, "Unlabeled data improves adversarial robustness," in *Advances in Neural Information Processing Systems*, Vancouver, British Columbia, Canada, 2019, pp. 11190–11201.

[29] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[30] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[31] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[32] G. Asci and M. A. Guvensan, "A novel input set for lstm-based transport mode detection," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Kyoto, Japan, March 2019, pp. 107–112.