Improving Transportation Mode Identification with Limited GPS Trajectories

Yuanshao Zhu[1,2], Christos Markos[1,3], James J.Q. Yu[1,2,*]
[1] *Department of Computer Science and Engineering, Southern University of Science and Technology*
[2] *Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation*
[3] *Faculty of Engineering and Information Technology, University of Technology Sydney*
yasozhu@gmail.com, christos.k.markos@gmail.com, yujq3@sustech.edu.cn

***Abstract*—The deployment of Global Positioning System (GPS) sensors in modern smartphones and wearable devices has enabled the acquisition of high-coverage urban trajectories. Extracting knowledge from such diverse spatiotemporal data is essential for optimizing intelligent transportation system operations. Yet a deeper understanding of users' mobility patterns also requires identifying their associated transportation modes. Combined with growing privacy concerns, the considerable effort involved in manual data annotation means that GPS trajectories are in reality not labeled by transportation mode. This poses a significant challenge for machine learning classifiers, which often perform best when trained on large amounts of labeled data. As such, this paper investigates a wide range of time series augmentation methods aiming to improve the real-world applicability of transportation mode identification. In our extensive experiments on Microsoft's Geolife dataset, both discrete wavelet transform and flip augmentations pushed the transportation mode identification accuracy of a convolutional neural network from 85.1% to 87.3% and 87.2%, respectively.**

I. Introduction

The ability to associate users' mobility patterns with their corresponding transportation modes is crucial for urban planning and transportation management [1]–[3]. Knowledge of the travel mode distribution along urban transportation networks can help develop more effective strategies towards optimizing infrastructure utilization, thereby alleviating significant issues such as traffic congestion [4]–[6]. It can also provide individuals with better route recommendations, conditioned on their desired travel mode and destination [7]. With Global Positioning System (GPS) sensors being installed in modern smartphones and other wearable devices, acquiring rich GPS trajectories for transportation mode identification has become easier than ever.

Most GPS-based transportation mode identification approaches have been in supervised learning settings. Because raw GPS trajectories are ill-suited for direct processing by machine learning models, the seminal work of [8] first computed pointwise motion features such as speed and acceleration from consecutive pairs of GPS points, before feeding them to a decision tree classifier. This motion feature extraction step has since become standard practice in the transportation mode identification literature. [2] combined the predictions of a random forest classifier with a rule-based method. [9] first trained a sparse autoencoder to extract latent representations of handcrafted motion features such as speed and acceleration, before feeding them to a Convolutional Neural Network (CNN) for the final classification. Inspired by computer vision applications, [10] treated GPS trajectories as image pixels by mapping GPS points to grid cells and adjusting pixel intensity according to location stay time. The authors then trained a CNN to extract high-level representations which were ultimately fed to a logistic regression classifier. [11] proposed a deep ensemble of CNNs, while [12], [13] leveraged a single CNN equipped with the attention mechanism. Others successfully used recurrent neural networks based on the Long Short-Term Memory (LSTM) module, due to their demonstrated effectiveness in modeling long-term temporal dependencies [14]–[16].

Despite the aforementioned advances in supervised GPS-based transportation mode identification methods, the relative lack of trajectories labeled by travel mode remains a limiting factor. In reality, GPS trajectories are typically unlabeled, since GPS sensors do not automatically capture travel mode information. Another reason is that trajectory annotation is both time-consuming and labor-intensive [17], with users often citing privacy concerns [12]. Consequently, how to improve the performance of transportation mode classifiers when few labeled trajectories are available is an open problem.

In this direction, some researchers have combined labeled and unlabeled data in semi-supervised learning [17]–[19], while others have strictly used unlabeled data in unsupervised learning [20]. Among the semi-supervised approaches, [17] jointly trained a convolutional autoencoder and a CNN by first balancing their losses and then gradually assigning more weight to the latter's supervised loss. [18] instead leveraged a semi-supervised LSTM ensemble trained on multiple views of the data, including frequency-domain and latent representations thereof that were learned end-to-end. [19] used the mixup augmentation technique [21] to train a convolutional autoencoder on mixed batches of labeled, unlabeled, and synthetic samples by simultaneously minimizing their associated objective functions. On the other hand, [20] proposed a fully unsupervised approach whereby a convolutional autoencoder was equipped with a custom clustering layer and trained

This work is supported by the Stable Support Plan Program of Shenzhen Natural Science Fund No. 20200925155105002 and by the General Program of Guangdong Basic and Applied Basic Research Foundation No. 2019A1515011032. James J.Q. Yu is the corresponding author.

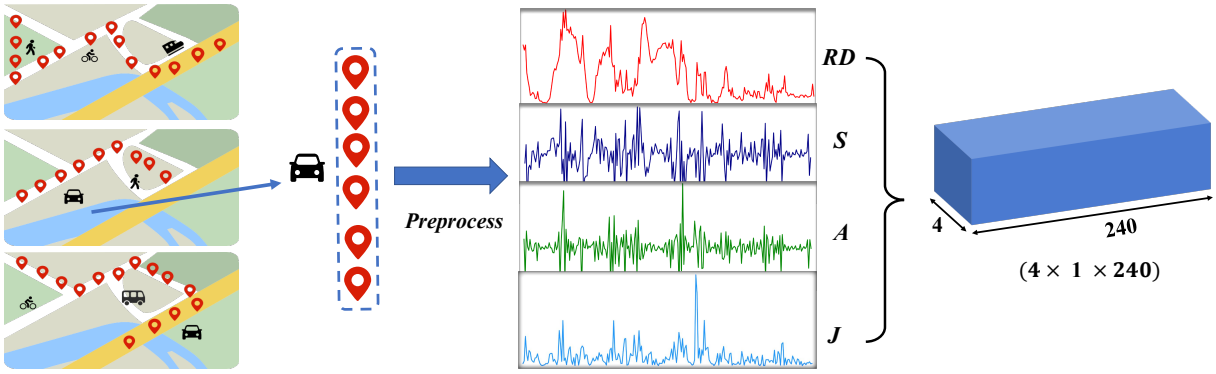

Fig. 1. Overview of the preprocessing framework for identifying transportation modes. Raw GPS trajectories are first segmented by transportation mode using the available labels. Then, pointwise motion features are computed for each segment and converted to a 4-channel tensor.

by jointly optimizing a weighted sum of reconstruction and clustering losses, thus encouraging clustering-friendly representations at the autoencoder's low-dimensional embedding layer.

To address the limitations caused by the scarce availability of labeled trajectories, we instead follow a different approach. Specifically, we explore a collection of time series augmentation methods[1] to assess their impact on the performance of supervised transportation mode classifiers. We provide an analysis of the underlying principles, effects on classification performance, as well as hyperparameter selection guidelines for each method. We conduct a series of comprehensive experiments on Microsoft's Geolife [8], [22] dataset, a real-world dataset of GPS trajectories, showing that both discrete wavelet transform and flip augmentations are effective methods towards improving transportation mode identification with limited data.

The remainder of this paper is organized as follows. Section II presents our preprocessing steps and formulates the problem of data-augmented supervised transportation mode identification. Section III introduces the time series augmentation techniques that are investigated towards enhancing GPS-based transportation mode identification with limited data. Section IV analyzes our experimental results and provides guidelines into hyperparameter selection for the above augmentation methods, while Section V concludes this paper.

II. Preliminaries

This section first presents how we preprocess GPS trajectories into multivariate time series of motion features, including relative distance, speed, acceleration, and jerk. It then formulates the problem of data-augmented, supervised transportation mode identification.

[1]While image augmentation techniques such as random rotations and horizontal/vertical shifts have been shown to boost classification accuracy in computer vision applications, they are not directly applicable to either raw GPS data or the multivariate time series of motion features that trajectories are typically converted to.

A. GPS Trajectory Preprocessing

We represent GPS trajectory T_i as a sequence $\{p_1,\ p_2,\ \ldots,\ p_{L_{T_i}}\}$ of length L_{T_i}. Within T_i, GPS points are denoted by $p_i = \langle lat_i,\ lng_i,\ t_i\rangle$, where lat_i, lng_i indicate the device's latitude and longitude in decimal degrees at time t_i. The relative distance RD_i between p_i and its successor p_{i+1} can be estimated in meters using the Vincenty formula [23], denoted as:

$$\mathrm{RD}_i = \mathrm{Vincenty}\left(lat_i,\ lng_i,\ lat_{i+1},\ lng_{i+1}\right). \quad (1)$$

Based on RD_i and its associated time interval $\Delta t_i = t_{i+1} - t_i$, we follow established literature [16], [17], [22] in calculating pointwise motion features of speed S_i, acceleration A_i and jerk J_i according to the following equations:

$$S_i = \frac{\mathrm{RD}_i}{\Delta t_i}, \quad 1 \le i \le n, \quad S_n = S_{n-1}, \quad (2)$$

$$A_i = \frac{S_{i+1} - S_i}{\Delta t_i}, \quad 1 \le i \le n, \quad A_n = 0, \quad (3)$$

$$J_i = \frac{A_{i+1} - A_i}{\Delta t_i}, \quad 1 \le i \le n, \quad J_n = 0. \quad (4)$$

After the above feature extraction steps, we eliminate any timesteps with velocity or acceleration outliers based on upper thresholds defined for each transportation mode in [17]. We finally apply min-max normalization to each of the four features and stack them into a 4-channel tensor $X_i = \{x_1,\ x_2,\ \ldots,\ x_{L_{T_i}}\}$, where $x_i = \{RD_i,\ S_i,\ A_i,\ J_i\}$. Since our experiments leverage both recurrent and non-recurrent neural network architectures, the latter requiring fixed-size input, we finally split each motion feature tensor X_i calculated for T_i into $\lceil L_{T_i}/N \rceil$ segments of length N. The last segment is padded with zeros if it has fewer than N timesteps; in this work, we empirically set $N = 240$. Please note that all data augmentation methods discussed and evaluated in this paper are based on motion feature tensors rather than raw GPS trajectories.

B. Problem Formulation

Given labeled dataset $\mathcal{D} = \{(X_i,\ y_i)\}_{i=1}^{n}$ preprocessed as per Section II-A and classifier $f_\omega(\cdot)$ parameterized by trainable parameters ω, we formalize transportation mode identification as a standard supervised classification problem, i.e., the problem of obtaining the optimal set of parameters ω such that the following loss is minimized:

$$\arg\min_{\omega} L(\omega) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(y_i, f_\omega(X_i)), \tag{5}$$

$$\ell_i = -[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)], \tag{6}$$

where $\hat{y}_i = f_\omega(X_i)$, y_i are the i-th predicted and ground-truth transportation modes, and $\ell(\cdot)$ is the categorical cross-entropy loss function.

Next, we define the general data augmentation function $Aug(\cdot)$ that produces synthesized sample X' when applied to X_i, denoted by $X' = Aug(X_i)$. Assuming that each sample is augmented exactly once, the above loss function can then be rewritten as:

$$\begin{aligned}\arg\min_{\omega} L(\omega) = \frac{1}{2n}\sum_{i=1}^{n} &\ell_i(y_i, f_\omega(X_i)) + \\ &\ell_i(y_i, f_\omega(Aug(X_i))).\end{aligned} \tag{7}$$

In this paper, we study a wide range of data augmentation techniques (see Section III) in place of $Aug(\cdot)$ with the purpose of evaluating their contribution towards improving transportation mode identification with limited GPS trajectories.

III. Methodology

This section details the time series augmentation techniques that we adopt towards improving the accuracy of transportation mode classifiers. These include data perturbation, flipping, mixup [21], mixing, and discrete wavelet transform.

A. Data Perturbation

Data perturbation refers to injecting each input motion feature tensor X_i with random noise. In practice, this is achieved via addition with a noise tensor Z of the same dimensionality. For simplicity, Z is sampled from a Gaussian distribution; specifically, each $z \in Z$ is sampled according to:

$$p(z; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right), \tag{8}$$

where μ, σ denote the mean and standard deviation of z, respectively. We determine the values for μ and σ by the mean and standard deviation of X_i, controlled by hyperparameter k as follows:

$$\begin{aligned}\mu &= k \cdot \text{mean}(X_i), \\ \sigma &= k \cdot \text{stddev}(X_i).\end{aligned} \tag{9}$$

For original sample X_i, the synthesized sample can thus be written as:

$$\begin{aligned}X' &= X_i + Z, \\ y' &= y_i.\end{aligned} \tag{10}$$

B. Data Flip

In computer vision applications, data augmentation is usually performed by randomly rotating, cropping, or flipping images. However, most of the above methods would destroy the motion features' temporal correlations and interdependencies. Considering that each input channel represents a different motion feature, we simply flip X_i along the temporal dimension for each channel. The flip operation can be expressed as:

$$\begin{aligned}X' &= \{x_n,\ x_{n-1},\ \ldots,\ x_1\}, \\ y' &= y_i.\end{aligned} \tag{11}$$

C. Mixup

Originally proposed for computer vision applications, Mixup [21] expands the training data by mixing pairs of images and their corresponding labels. The mixup method is a form of data augmentation which encourages the classifier $f_\omega(\cdot)$ to learn linear interpolations between pairs of training samples, generated as follows:

$$\begin{aligned}X' &= \lambda X_i + (1-\lambda) X_j, \\ y' &= \lambda y_i + (1-\lambda) y_j,\end{aligned} \tag{12}$$

where λ is sampled from a beta distribution $Beta(\alpha, \alpha)$ parameterized by $\alpha \in (0,\ \infty)$. In eq. (12), $(X_i,\ y_i)$ and $(X_j,\ y_j)$ are two randomly-selected samples from the original training data with one-hot encoded labels y_i and y_j. The mixing hyperparameter α controls the mixing strength between feature-target pairs; when $\alpha \approx 0$, X' is identical to X_i, i.e., no mixing is performed.

D. Data Mixing

The intuition behind data mixing comes from the fact that GPS trajectories with the same transportation mode would have similar trends in terms of motion features. To this end, we perform a weighted mix of k motion feature tensors having the same transportation mode and assign the resulting synthetic sample with the same label as the original ones. In theory, such a scheme allows for synthesizing an infinite number motion feature tensors. In this paper, we adopt two data mixing schemes, namely double-trajectory mixing and multi-trajectory weight decay mixing. For double-trajectory mixing, we randomly select two samples with identical transportation modes and mix them as follows:

$$\begin{aligned}X' &= w_1 X_1 + w_2 X_2, \quad w_1 + w_2 = 1, \\ y' &= y_i.\end{aligned} \tag{13}$$

For multi-trajectory weight decay mixing, we randomly select k trajectories with the same transportation mode and mix them using gradually smaller weights:

$$\begin{aligned}X' &= w_1 X_1 + w_2 X_2 + \ldots + w_k X_k, \\ y' &= y_i.\end{aligned} \tag{14}$$

where $\sum_{i=1}^{k} w_i = 1$ and $w_1 \geq w_2 \geq \ldots \geq w_k$. Please note that double-trajectory mixing is simply a special case of multi-trajectory weight decay mixing where $k = 2$.

E. Discrete Wavelet Transform

Given that motion feature variations can also be distinguished in the frequency domain [16], [18], we examine the effect of augmentation by Discrete Wavelet Transform (DWT) on the performance of transportation mode identification. Given time series $x(t)$, DWT results in a multi-resolution decomposition of the input signal [24] as follows:

$$\begin{aligned} x(t) &= \sum_b A_{M,b} 2^{-M/2} \varphi\left(\frac{t}{2^M} - b\right) \\ &\quad + \sum_a^M \sum_b d_{a,b}(x(t), \psi(t)) 2^{-a/2} \psi\left(\frac{t}{2^a} - b\right) \\ &= A_M(t) + \sum_a^M D_a(t), \end{aligned} \tag{15}$$

where $A_{M,b} = \langle x(t),\ \varphi_{M,b}(t) \rangle$ is the approximation coefficient at decomposition level M and $\varphi(t)$ is an auxiliary scaling function. In other words, $x(t)$ is decomposed into an approximation signal $A_M(t)$ and M detailed signals $D_a(t)$. When augmenting X_i, the synthesized sample X' is again associated with the same transportation mode label, i.e., $y' = y_i$.

IV. Experiments

This section first introduces the real-world dataset of GPS trajectories that we used for our experiments and describes our simulation setup. It finally presents our experimental results and provides hyperparameter tuning guidelines for the evaluated time series augmentation methods.

A. Dataset Description and Simulation Setup

1) ***Dataset****:* All data augmentation methods in Section III are evaluated on Microsoft's Geolife dataset [8], [22], which has been widely used in the transportation mode identification literature [10], [17], [20]. It contains GPS trajectories collected by 182 users over five years. Out of these users, 69 users have labeled parts of their trajectories by transportation mode. preprocess them as per Section II-A. Following the dataset authors' recommendation, we select main transportation modes for identification, namely *walking*, *biking*, *bus*, *driving* and *railway*. After preprocessing all GPS trajectories as per Section II-A, we obtain a total of $24,741$ labeled samples of length 240 (*walking*: 7315, *biking*: 3848, *bus*: 5964, *driving*: 4338 , *railway*: 3278). Following a stratified data split to maintain the transportation mode distribution, 85% of the above samples are used for training and validation, while the remaining 15% are used for testing. Please note that all data augmentation methods are only applied to the training set.

2) ***Simulation Setup****:* We first present our hyperparameter settings for the time series augmentation methods described in Section III. Perturbation is applied with $k = 0.02$, while Mixup [21] uses $\alpha = 0.5$. Data mixing expands each data class by 2000 samples,[2] where *mixing-1* and *mixing-2* denote the double-trajectory mixing and multi-trajectory weight decay mixing methods, respectively. For the former, we set $w_1,\ w_2 \sim Beta(0.5,\ 0.5)$, while the latter uses $k = 5$ (i.e., we mix five trajectories of the same transportation mode) with $w_1 = 0.9$, $w_2 = 0.04$, $w_3 = 0.02$, $w_4 = 0.02$, $w_5 = 0.02$.

[2]Even though we could generate as many samples per class as required to eliminate the training set class imbalance, this would lead to a different class distribution compared to the test set.

TABLE I
Accuracy Percentage (Mean ± Standard Deviation) for Different Data Augmentation Methods and Classifiers

Augmentation	MLP	CNN	LSTM
Baseline	70.8 ± 1.32	85.1 ± 0.31	76.3 ± 0.28
Perturbation	71.4 ± 0.51	85.9 ± 0.18	76.8 ± 0.23
Flip	71.8 ± 0.92	87.3 ± 0.23	77.4 ± 0.27
Mixup [21]	69.8 ± 0.88	84.2 ± 0.22	75.1 ± 0.31
Mixing-1	72.6 ± 0.72	86.0 ± 0.21	76.9 ± 0.20
Mixing-2	71.3 ± 0.56	85.5 ± 0.19	76.7 ± 0.23
DWT	80.1 ± 0.92	87.2 ± 0.13	78.5 ± 0.18

We evaluate the above time series augmentation methods on a MultiLayer Perceptron (MLP), a CNN, and an LSTM. (1) The MLP has three fully connected layers with $\{512,\ 128,\ 5\}$ neurons. (2) The CNN consists of three one-dimensional (1D) convolution layers with a kernel size of 3 and $\{32,\ 64,\ 128\}$ channels, respectively. Each convolution layer is followed by a max pooling operation with a pool size of 2. The convolution layers are followed by a flattening operation resulting in 3840 features, followed by a fully connected layer with 960 neurons. (3) The LSTM has three LSTM layers with $\{64,\ 64,\ 64\}$ units, respectively. The output of the last LSTM layer is flattened and fed to two fully connected layers with $\{256,\ 5\}$ neurons. For all three neural networks, all hidden layers are activated using the Rectified Linear Unit (ReLU) function, while the softmax activation function is used to predict the transportation mode at the output layer. Please note that we do not use regularization methods such as dropout or batch normalization; instead, we prevent our networks from overfitting by reducing their size (i.e., number of layers and hidden units) and therefore the number of trainable parameters. All models are trained for 200 epochs using the Adam optimizer with a learning rate of 0.001. We report the mean classification accuracy calculated over the last 20 training epochs.

Our experiments were developed using Python 3.7. All neural networks were built using PyTorch 1.7 and trained on a server with an Intel Xeon Silver 4210 CPU and an NVIDIA GeForce RTX 2080 Ti GPU with 11GB of GDDR6 memory.

B. Results

Our experimental results are shown in Table I. With the exception of Mixup [21], which performed worse than just using the original samples, all evaluated augmentation methods contributed to improving classification performance. Among them, discrete wavelet transform and flip augmentations achieved the best results for our CNN and LSTM models, pushing the former's accuracy from 85.1% to 87.2% and 87.3%, respectively. DWT was also by far the most effective augmentation method for our MLP, increasing its baseline accuracy of 70.8% to 80.1% . Both *mixing-1* and *mixing-2* resulted in modest im-

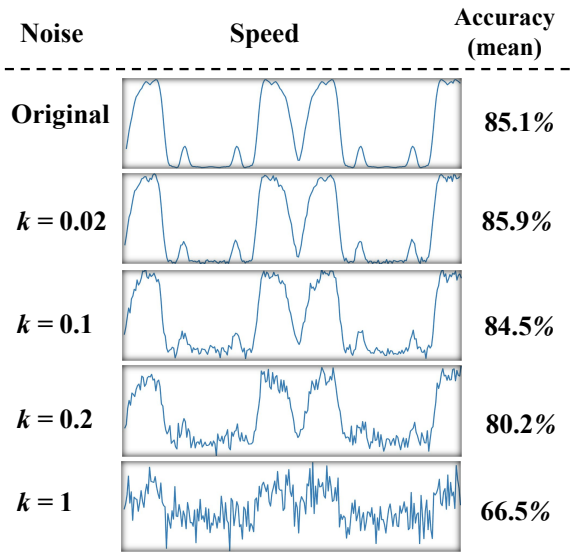

Fig. 2. Changes in the speed signal of a randomly-selected sample after adding noise to all training samples with $k \in \{0.02,\ 0.1,\ 0.2,\ 1\}$. The CNN's mean accuracy declines beyond a certain noise magnitude, indicating that the classifier fails to identify meaningful information within the augmented samples.

TABLE II
HYPERPARAMETER SENSITIVITY OF CNN ACCURACY (MEAN ± STANDARD DEVIATION) TO MIXUP

Augmentation	Accuracy
Baseline	85.1 ± 0.31
Mixup ($\alpha = 0.2$)	83.9 ± 0.21
Mixup ($\alpha = 0.5$)	84.2 ± 0.22
Mixup ($\alpha = 1$)	83.6 ± 0.27
Mixup ($\alpha = 10$)	82.8 ± 0.46

TABLE III
HYPERPARAMETER SENSITIVITY OF CNN ACCURACY TO DATA MIXING

Augmentation	Parameter Settings	Mean (%)
mixing-1	$w_1 = 0.5, \quad w_2 = 0.5$	84.3
	$w_1 = 0.8, \quad w_2 = 0.2$	85.6
	$w_1 = 0.95, \quad w_2 = 0.05$	85.8
	$w_1,\ w_2 \sim Beta(0.5,\ 0.5)$	86.0
mixing-2	$\{0.5, 0.3, 0.1, 0.05, 0.05\}$	82.6
	$\{0.7,\ 0.1,\ 0.1,\ 0.05,\ 0.05\}$	83.1
	$\{0.8,\ 0.05,\ 0.05,\ 0.05,\ 0.05\}$	85.3
	$\{0.9,\ 0.04,\ 0.02,\ 0.02,\ 0.02\}$	85.5

provement, with the former outperforming the latter. Moreover, perturbation attained nearly identical results to *mixing-1*. The above experimental results confirm the potential of time series augmentation in improving GPS-based transportation mode identification with limited data.

*1) **Data Perturbation**:* As described in Section III-A, the intuition behind data perturbation is that learning from noisy counterparts of the original data may help the classifier learn more general features. However, adding too much noise may result in unrealistic samples that are hard to learn meaningful representations from. Fig. 2 shows that perturbation indeed boosted classification accuracy from 85.1% to 85.9% for $k = 0.02$, which is the hyperparameter value used throughout our experiments. Values of $k > 0.1$, however, resulted in significant accuracy degradation.

*2) **Flip**:* By simply flipping X_i along the temporal dimension for each motion feature, our expectation is that the generated sample would still realistically correspond to the same transportation mode. Each feature would demonstrate the same minimum and maximum values, despite having different temporal dynamics. According to our results in Table I, flipping resulted in the highest classification accuracy for our CNN and LSTM but only modestly benefited our MLP. This is likely due to the latter not accounting for temporal dependencies.

*3) **Mixup**:* As per Section III-C, mixup [21] generates synthetic data via a linear combination of paired samples and their corresponding ground-truth labels. This practice aims to encourage the classifier to interpolate smoothly between samples and reduce the effect of adversarial ones. Our hyperparameter sensitivity tests, shown in Table II, demonstrate that mixup did not outperform the non-augmented transportation mode identification baseline. However, note that mixup assigns labels to the synthesized samples by simply blending their original ones. As such, we expect that it could perform better in semi-supervised training, where the effect of the generated labels on the learned representations would be attenuated. This is out of the scope of this paper and is left for future work.

*4) **Data Mixing**:* Although data mixing did not dramatically boost classification accuracy, we found that it resulted in higher training stability during our experiments. This may be due to how data mixing is performed, which is via timestep-wise addition of two or more samples of the same transportation mode. We hypothesize that this may help the classifier learn the main motion feature trends of each transportation mode while simultaneously becoming more robust to trajectory variations not observed in the original data.

We also analyzed the impact of different data mixing hyperparameter settings on classification accuracy; our experimental results are summarized in Table III. Although mixing the motion features of either two or five trajectories did increase model accuracy compared to the baseline, *mixing-2* did not result in significant improvement. This is not surprising, as mixing more sets of motion features will also incur an increase in uncertainty.

*5) **DWT**:* Here, we explore the effect of extracting features via different wavelet decomposition functions on classification accuracy. As shown in Table IV, using different wavelet decomposition functions did not significantly affect classification accuracy, with Daubechies wavelets achieving the best results. This suggests that DWT has the desirable property of not being particularly sensitive to the choice of wavelet function.

Recall that, according to eq. (15), $x(t)$ can be decomposed into approximate signal $A_M(t)$ and detailed signal $D_a(t)$. Having compared the influence of DWT on classification

TABLE IV
SENSITIVITY OF CNN ACCURACY TO DIFFERENT WAVELET DECOMPOSITION FUNCTIONS IN DWT

Wavelet	Mean (%) w/ $A_M(t)$	Mean (%) w/ $D_a(t)$
Daubechies	87.2	87.0
Symlets	87.0	87.1
Coiflets	86.8	86.9
Haar	87.1	87.0

accuracy when using either $A_M(t)$ or $D_a(t)$, our experimental results showed no significant prevalence of one over the other. This is consistent with recent work indicating that capturing motion feature *trends* rather than details may be more important when distinguishing among transportation modes [18].

V. CONCLUSION

In this paper, we investigated a range of data augmentation techniques to improve GPS-based transportation mode identification performance when limited labeled data are available. Since the literature typically performs transportation mode identification on time series of motion features extracted from GPS trajectories rather than the raw trajectories themselves, we followed the same procedure and investigated the impact of several time series augmentation techniques on classification accuracy. We also provided guidelines into tuning their hyperparameters to encourage their use in future transportation mode identification research. Through a set of comprehensive experiments on Microsoft's Geolife, an openly available real-world dataset of GPS trajectories, we demonstrated that the simple operation of flipping resulted in the highest accuracy of 87.3% for a convolutional neural network. In addition, extracting features in the frequency domain via DWT pushed classification accuracy from the baseline of 85.1% to 87.2%.

In future work, we will investigate the influence of time series augmentation methods on the transportation mode identification accuracy of more sophisticated neural network architectures, such as generative adversarial networks and Transformers.

REFERENCES

[1] F.-Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 630–638, 2010.

[2] B. Wang, L. Gao, and Z. Juan, "Travel mode detection using GPS data and socioeconomic attributes based on a random forest classifier," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1547–1558, 2017.

[3] M. Ashifuddin Mondal and Z. Rehena, "Intelligent traffic congestion classification system using artificial neural network," in *Companion Proceedings of The 2019 World Wide Web Conference*, ser. WWW '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 110–116.

[4] 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.

[5] G. Li, C.-J. Chen, S.-Y. Huang, A.-J. Chou, X. Gou, W.-C. Peng, and C.-W. Yi, "Public transportation mode detection from cellular data," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 2499–2502.

[6] E. Anagnostopoulou, B. Magoutas, E. Bothos, and G. Mentzas, "Persuasive technologies for sustainable smart cities: The case of urban mobility," in *Companion Proceedings of The 2019 World Wide Web Conference*, ser. WWW'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 73–82.

[7] A. C. Prelipcean, G. Gidofalvi, and Y. O. Susilo, "Transportation mode detection – an in-depth review of applicability and reliability," *Transport Reviews*, vol. 37, no. 4, pp. 442–464, 2017.

[8] 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*. New York, NY, USA: Association for Computing Machinery, 2008, pp. 247–256.

[9] H. Wang, G. Liu, J. Duan, and L. Zhang, "Detecting transportation modes using deep neural network," *IEICE TRANSACTIONS on Information and Systems*, vol. 100, no. 5, pp. 1132–1135, 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*. Cham, Switzerland: Springer International Publishing, 2016, pp. 54–66.

[11] 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.

[12] Y. Zhu, S. Zhang, Y. Liu, D. Niyato, and J. J. Q. Yu, "Robust federated learning approach for travel mode identification from non-iid GPS trajectories," in *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, 2020, pp. 585–592.

[13] Y. Zhu, Y. Liu, J. J. Q. Yu, and X. Yuan, "Semi-supervised federated learning for travel mode identification from GPS trajectories," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2021.

[14] H. Liu and I. Lee, "End-to-end trajectory transportation mode classification using bi-lstm recurrent neural network," in *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, Nanjing, China, 2017, pp. 1–5.

[15] 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*. Association for Computing Machinery, 2018, pp. 1606–1615.

[16] J. J. Q. Yu, "Travel mode identification with GPS trajectories using wavelet transform and deep learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1–11, 2021.

[17] S. Dabiri, C. 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*, vol. 32, no. 5, pp. 1010–1023, 2020.

[18] J. J. Q. Yu, "Semi-supervised deep ensemble learning for travel mode identification," *Transportation Research Part C: Emerging Technologies*, vol. 112, pp. 120–135, 2020.

[19] X. Song, C. Markos, and J. J. Q. Yu, "Multimix: A multi-task deep learning approach for travel mode identification with few gps data," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[20] C. Markos and J. J. Q. Yu, "Unsupervised deep learning for GPS-based transportation mode identification," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

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

[22] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, *Understanding Mobility Based on GPS Data*. New York, NY, USA: Association for Computing Machinery, 2008, p. 312–321.

[23] 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.

[24] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.