Transportmetrica B: Transport Dynamics

ISSN: 2168-0566 (Print) 2168-0582 (Online) Journal homepage: www.tandfonline.com/journals/ttrb20

Behavior recognition in non-motorized traffic using deep embedded contrastive clustering from GPS trajectories

Huanting Xu, Haipeng Zeng, Zhigang Wu, Wangyong Xing & Zhaocheng He

To cite this article: Huanting Xu, Haipeng Zeng, Zhigang Wu, Wangyong Xing & Zhaocheng He (2025) Behavior recognition in non-motorized traffic using deep embedded contrastive clustering from GPS trajectories, Transportmetrica B: Transport Dynamics, 13:1, 2556746, DOI: 10.1080/21680566.2025.2556746

To link to this article: https://doi.org/10.1080/21680566.2025.2556746

TRANSPORTMETRICA B: TRANSPORT DYNAMICS
2025, VOL. 13, NO. 1, 2556746
https://doi.org/10.1080/21680566.2025.2556746

Behavior recognition in non-motorized traffic using deep embedded contrastive clustering from GPS trajectories

Huanting Xu[a,b], Haipeng Zeng[a,b], Zhigang Wu[a,b], Wangyong Xing[d] and Zhaocheng He[a,b,c]

[a]School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, People's Republic of China; [b]Guangdong Provincial Key Laboratory of Intelligent Transportation Systems, Sun Yat-sen University, Guangzhou, People's Republic of China; [c]Pengcheng Laboratory, Shenzhen, People's Republic of China; [d]Institute of Intelligent Transportation, Guangdong Leatop Technology Investment Co., Ltd., Guangzhou, People's Republic of China

ABSTRACT

Recognizing the crossing behavior of non-motorized traffic at intersections is crucial for urban transportation safety and efficiency. However, traditional methods often overlook environmental context, while deep learning approaches require extensive labeled data. To address these limitations, we propose an unsupervised learning framework that infers waiting and passing behaviors by combining motion features with spatio-temporal semantic context. Our model employs a convolutional autoencoder with an integrated clustering layer, trained through a joint optimization of reconstruction, clustering, and K-neighborhood contrastive losses. Experimental results show our framework outperforms other unsupervised methods. Notably, incorporating semantic features improves clustering accuracy from 87.3% to 91.3%, validating the effectiveness of our approach in capturing complex behavior.

ARTICLE HISTORY
Received 30 September 2024
Accepted 9 July 2025

KEYWORDS
Behavior recognition; unsupervised cluster; deep learning; non-motorized transport

1. Introduction

In urban transportation systems, intersections are critical convergence points where traffic flows meet, and various traffic participants frequently interact. Non-motorized transport participants, including pedestrians and non-motorized vehicles, are more vulnerable to fatal and severe injuries in road crashes, compared to vehicle occupants. Recognizing their crossing behaviors is, therefore, essential for improving traffic efficiency and ensuring safety (Hasain and Ahmed 2024; Malenje et al. 2018; Xin et al. 2022; Zhang and Berger 2023). This paper aims to infer waiting and passing behaviors of non-motorized transport participants, which is particularly challenging due to factors such as the high density and overlapping nature of trajectories during peak hours, and the subtle variations in movement patterns that distinguish waiting from passing (Korbmacher, Dang, and Tordeux 2024).

End-to-end behavioral recognition through video and image is one of the most direct approaches for this task. These methods utilize continuous sequence of video frames as input and apply techniques such as action recognition and deep learning to identify specific behaviors of objects within the images (Batchuluun et al. 2017; Dang et al. 2024; Hu et al. 2023). Moreover, some approaches focus on group behavior recognition, but they primarily emphasize macroscopic features such as changes in density and movement direction within the group (Thilakarathne et al. 2024). However, these video-based methods often struggle in crowded environments due to occlusions, require clear action features for accurate recognition, and their deployment can be limited by camera coverage, infrastructure costs, varying environmental conditions (e.g. lighting, weather), and potential privacy concerns (Buch, Velastin, and Orwell 2011; Dupuis, Subirats, and Vasseur 2016). These limitations highlight the need for complementary approaches, particularly for large-scale analysis across diverse urban settings.

Consequently, another approach, leveraging the increasing availability and precision of Global Positioning System (GPS) data, focuses on analyzing trajectories (Bian et al. 2019). Indeed, trajectory data analysis

CONTACT Zhaocheng He hezhch@mail.sysu.edu.cn School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen 518107, People's Republic of China; Guangdong Provincial Key Laboratory of Intelligent Transportation Systems, Sun Yat-sen University, Guangzhou, People's Republic of China; Pengcheng Laboratory, Shenzhen 518107, People's Republic of China

has become a vibrant research area, providing foundational insights for a range of advanced applications. These include sophisticated behavioral modeling using techniques like inverse reinforcement learning (Liu, Alsaleh, and Sayed 2024), predictive modeling of human mobility with probabilistic models (Chen et al. 2024). Historically, micro-behavior analysis using GPS was challenging due to accuracy limitations and difficulties in data collection for non-motorized transport. While contemporary GPS data offers significant advantages in terms of broader spatial–temporal coverage, scalability for large populations, and relative insensitivity to visual obstructions compared to video methods, it inherently lacks the rich visual context and suffers from lower spatial precision, especially in dense urban environments (Zandbergen 2009; Zheng et al. 2008). Despite these challenges, GPS trajectory data, comprising ordered time-stamped latitude-longitude pairs reflecting movement characteristics, has become a primary input for behavior recognition. Traditional methods often utilize raw trajectory features for behavior classification, employing techniques like direct clustering or machine learning models such as decision trees (DT), support vector machines (SVM) (Stenneth et al. 2011; Sun and Ban 2013; Xiao, Juan, and Zhang 2015). However, these methods heavily rely on the manual feature construction effects and are susceptible to GPS measurement errors as well as traffic and environmental conditions (Zheng et al. 2008). In recent years, deep learning has been widely used in trajectory data mining and traffic behavior recognition (Ahmed et al. 2025; Chen et al. 2023; Ismaeel et al. 2023; Yi et al. 2024). Through automatic feature extraction, deep learning methods effectively overcome the dependence on manual feature construction of traditional methods, and improve the robustness and accuracy of models (Noor and Ige 2024). In the field of traffic pattern recognition, researchers have applied deep learning models to process large-scale trajectory data, solving problems such as low efficiency and poor accuracy. For example, the researchers proposed a semi-supervised learning method based on Long Short-Term Memory (LSTM) autoencoder, which can achieve high-precision detection of traffic patterns using minimal labeled data, which achieved good results (Sadeghian et al. 2024). In addition, other deep networks like recurrent neural network (RNN) (Simoncini et al. 2018) and convolutional neural network (CNN) (Wang et al. 2017), were also widely used for trajectory data mining. However, despite the progress of deep learning methods in traffic pattern recognition, their application still faces challenges. Model training relies on a large amount of labeled data and may be affected by data noise and environmental changes when dealing with complex traffic environments (Hu et al. 2022).

To address these challenges, unsupervised deep learning has emerged as a promising solution. Unlike supervised methods, unsupervised deep learning does not require labeled samples, instead embedding original features into complex, nonlinear spatio-temporal relationships. Autoencoders, in particular, are favored for their simplicity and strong performance (Graser et al. 2024). However, traditional autoencoders primarily focus on minimizing the reconstruction error between input and output data, which does not guarantee that embedding representation is conducive to clustering (Song et al. 2013). To improve this, much of the literature has optimized autoencoders for clustering tasks. For instance, some studies pre-train a autoencoder on the input data and then cluster the low-dimensional learning embeddings by optimizing the clustering loss. (Yang, Parikh, and Batra 2016) applied agglomerative clustering to the features learned by the autoencoder. While (Xie, Girshick, and Farhadi 2016) first pre-trains a denoising autoencoder and then replaces the decoder with a custom clustering layer. Recent approaches have achieved better results by jointly optimizing the clustering and the autoencoder reconstruction loss. (Guo et al. 2017) retained the decoder to reduce feature space distortion, while DeepCluster (Caron et al. 2019) iteratively groups the outputs of a neural network using K-Means (KM) and sets the resulting clustering assignments as training labels to update its parameters. These methods, which have shown success with image and text data, inspire new approaches for trajectory-based street-crossing behavior recognition.

More recently, contrastive clustering methods have achieved notable success in unsupervised image classification by leveraging individual similarity (Dang et al. 2021; Li et al. 2020). While typically used in image domains where significant differences between classes are required, this paper addresses a binary classification problem, where class differences are minimal. However, the idea of individual similarity can still improve clustering accuracy and generalization by incorporating relational information during clustering (Dang et al. 2021; Xu et al. 2022).

In this paper, we present Deep Embedded Contrastive Clustering (DECC) framework for waiting and passing behavior recognition using GPS trajectories. First, we pre-train a deep convolutional autoencoder (CAE)

on fixed-size trajectory segments. A clustering layer is then attached to the embedding layer of the CAE, with clustering centers maintained as trainable weights. To preserve the local structure of the sample space, we introduce a K-neighborhood contrastive loss. Finally, we fine-tune the clustering model by balancing reconstruction, clustering and contrastive losses, encouraging cluster-friendly input data representations. To further enhance model performance, we design semantic features for intersection scenes, in addition to the commonly used motion features. The contributions of our approach are as follows:

(1) We propose the DECC model for recognizing non-motorized transport's wait and pass behavior. This unsupervised framework is capable of training a deep network without the need for labeled data and demonstrates superior performance in behavior identification compared to baseline models.
(2) We design semantic features to describe the influence of traffic organization and signal control on behavior. By incorporating spatial and temporal semantic features, we diversify the input features, which effectively improves model accuracy.
(3) We introduce a K-neighborhood sample contrastive loss to facilitate model training. This loss function helps preserve the local structure of the embedded space and reduces the risk of overfitting.

The rest of the paper is organized as follows. After reviewing the related work in Section 2, we present our framework in detail in Section 3, including trajectory segmentation, input features computation, the structure of the model, and loss function design. In Section 4, we evaluate our proposed model architecture on a real trajectory dataset. Section 4 also compares our results with classical clustering algorithms and previous studies. Finally, we summarize the paper in Section 5.

2. Related works

2.1. Behavior recognition based on GPS trajectory

With the increasing popularity of GPS, studies leveraging GPS data, such as trajectory prediction and activity recognition (Sadeghian et al. 2025; Sadeghian, Håkansson, and Zhao 2021), have garnered significant attention in recent years. The recognition of wait-or-pass behaviors studied in this paper is a subset of activity recognition, both of which aim to distinguish different behaviors from trajectories.

To develop a model capable of extracting valuable knowledge from available GPS data, early research efforts heavily relied on the intuition of transforming raw trajectory data into a suitable form for training machine learning models. (Zheng et al. 2010) proposed a method for estimating transportation behavior using only GPS trajectories. They described a method to segment GPS trajectories by detecting change points in transportation behaviors based on speed and acceleration. A classifier was then used to estimate the transportation behavior from the features of the road segments.

Earlier, (Zheng et al. 2008) had first proposed fundamental features like distance traveled, speed, and acceleration. Building on this work, subsequent studies explored various approaches. For example, (Xiao, Juan, and Zhang 2015) utilized Bayesian networks, (Zheng et al. 2008) used decision trees (DT), (Stenneth et al. 2011) applied random forests, and (Sun and Ban 2013) employed support vector machines (SVM). Moreover, (Sadeghian et al. 2022) proposed a stepwise methodology for transport mode detection in GPS tracking data through the use of multiple supervised methods.

Compared to traditional manual features, deep neural networks can automatically extract effective features from raw images. In fact, supervised learning with deep features has been shown to perform well in recognition and prediction (Chen et al. 2023; Ismaeel et al. 2023; Yi et al. 2024). In their study, (Endo and Hiroyuki 2016) used a fully connected deep neural network (DNN) to automatically extract deep features and classify trajectories based on transportation behavior. The core idea of their work is to convert GPS trajectories into a 2D image structure that serves as an input to the DNN model. (Dabiri and Heaslip 2018) designed trajectory data as input for a convolutional network (CNN), utilizing CNN to predict travel patterns based solely on raw GPS trajectories. Based on a small amount of trajectory data, (Xin et al. 2022) used a gradient booster regression tree to investigate the wait-or-go (WOG) behavior of pedestrians. Most of these methods are supervised and rely on labeled data for training. Recently, there has been a growing interest in applying unsupervised deep clustering methods across various fields.

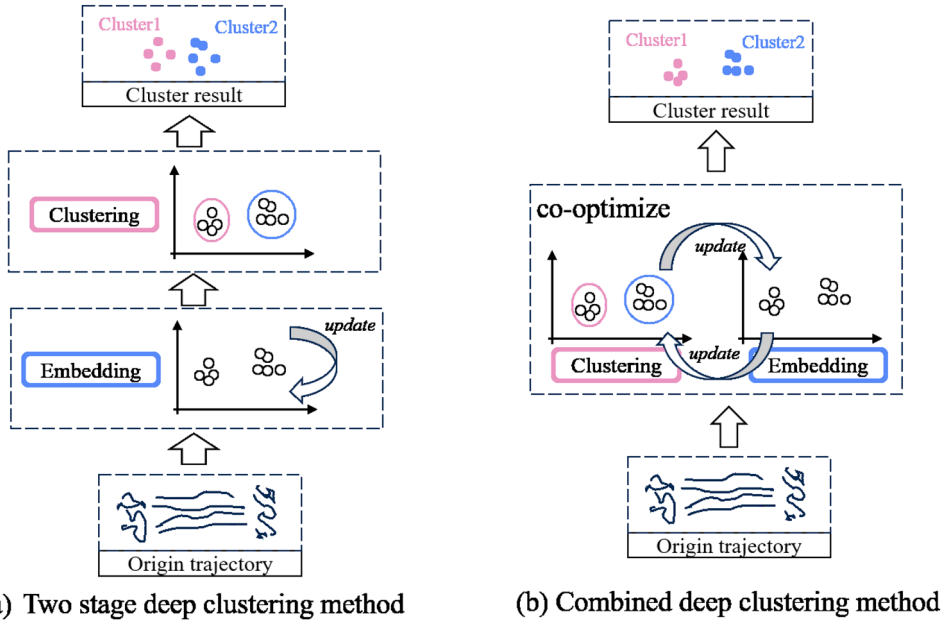

Figure 1. Mainstream unsupervised deep clustering methods.

2.2. Mainstream unsupervised deep clustering

Mainstream unsupervised deep clustering approaches are broadly classified into two categories:(i) two-stage method that apply clustering after learning the representations, and (ii) combined method that co-optimize feature learning and clustering.

As shown in Figure 1a, the two-stage methods directly leverage unsupervised deep learning frameworks to learn embedded feature representations prior to clustering. For example, (Tian et al. 2014) used an autoencoder to learn the low-dimensional features of the original graph, and then run the KM algorithm to obtain the clustering results. (Jiang et al. 2017) employed a variational autoencoder to generate embeddings of the input data, following with a clustering algorithm to group the data points based on the learned representations. (Wang et al. 2017) proposed a marginalized graph convolutional network that corrupts node content to learn graph feature representations, and then spectral clustering algorithm is used for graph clustering.

Combined methods attempt to explicitly model the classification error during the embedding learning, as shown in Figure 1b. Influential early works in this area include Deep Embedded Clustering (DEC) by (Xie, Girshick, and Farhadi 2016), which leverages deep neural networks to map data to a lower-dimensional feature space and uses a key clustering loss to enhance the model's ability to group similar samples. Another example is DeepCluster by (Caron et al. 2019), which iteratively clusters network-extracted features using K-Means and uses the resulting assignments as pseudo-labels for network training, pioneering the joint learning of features and clusterings. However, DEC discards the decoder after pre-training, causing the model to lose guidance from reconstruction losses in subsequent fine-tuning. This can lead to distortion of the feature space, which can impair clustering performance. To deal with this issue, (Guo et al. 2017) proposed Improved Deep Embedded Clustering (IDEC), which retains the autoencoder structure throughout training and improves results by explicitly preserving the local structure of the data through the joint optimization of a reconstruction loss and the clustering loss.

The combined method, particularly IDEC-like frameworks that integrate feature learning (e.g. via autoencoders) with a direct clustering objective, has shown considerable promise. For instance, (Markos and Yu 2020) successfully applied an IDEC-like structure for clustering GPS trajectory data into transportation modes. Our proposed DECC framework also adopts this combined approach, leveraging a Convolutional Autoencoder for representation learning and a KL divergence-based clustering loss, similar to IDEC. However, many such combined methods, including the aforementioned application by (Markos and Yu 2020) and the original IDEC, primarily consider overall differences between clusters (i.e. point-to-centroid relationships) while potentially neglecting the finer-grained variations and distributions of samples within clusters and the explicit

relationships between individual samples. This limitation underscores the need for alternative or supplementary mechanisms, such as contrastive learning, which aim to capture both inter-cluster distinctions and intra-cluster sample variability. Addressing this, a key innovation in our DECC approach is the integration of a K-neighborhood contrastive loss. This component, further detailed in Section 2.3 and Section 3.2.2, explicitly leverages instance-level relationships. It encourages the learned embeddings of an anchor sample to be closer to its K-nearest neighbors inferred to be from the same class and further from samples of different classes. This mechanism allows DECC to capture local structure more effectively and enhance both intra-cluster compactness and inter-cluster separability, which is particularly beneficial for distinguishing the often subtle differences in trajectory-based behaviors. The following subsection examines the advancements in contrastive learning and its potential for addressing these challenges in clustering.

2.3. Contrastive learning using in clustering

Contrastive Learning (CL) is an unsupervised learning method that learns general features of a data set by instructing the model to recognize similarities or differences between samples (Chen et al. 2020; He et al. 2020). The core idea of this method is to learn meaningful feature representations without labeling by comparing pairs of samples in a way that brings similar samples as close together as possible and pushes dissimilar samples as far away as possible. It was widely used in clustering, known as Contrastive Clustering (CC).

The most classical approach is Contrastive Clustering (Li et al. 2020), which started with labels as representations and performs contrastive clustering at the instance level and at the cluster level. Similarly, Double Contrastive Deep Clustering (DCDC) (Dang et al. 2021) constructed contrastive loss from both sample view and class view, which allowed learning more discriminative representations. This method relied on the similarity between the original features of the image to achieve automatic clustering, which required high data requirements and sufficiently large differences between different classes.

SCAN (Van Gansbeke et al. 2020) integrated semantically meaningful nearest neighbors as a prior into a learnable clustering method that classifies each image and its nearest neighbors together. Unlike SCAN, Nearest Neighbor Matching (NNM) (Dang et al. 2021) matched nearest neighbors from the perspective of local and global information, respectively. Nearest Neighbor Comparison Clustering (NNCC) (Xu et al. 2022) combined comparison learning with neighbor relation mining, which was alternately updated during the training process. The introduction of the K-neighborhood strategy enabled these methods to make better use of the information of the neighboring samples to aggregate similar samples, which effectively reduced the method's dependence on the underlying features.

3. Deep clustering model for behavior recognition

In this section, we begin by performing trajectory segmentation and feature extraction to obtain normalized model inputs with fixed length, each representing a single behavior. Next, we describe the CAE framework and its associated loss function. Finally, we outline the structure of the clustering layer within the framework, integrating clustering loss, contrastive loss, and reconstruction loss to develop a comprehensive clustering model. The basic architecture for crossing behavior identification is shown in Figure 2.

3.1. Trajectory segmentation and features extraction

3.1.1. Trajectory segmentation

The original trajectory represents the complete travel process of an object and may encompass multiple behaviors. To obtain consistent behavior within sub-trajectory segments, we first perform trajectory segmentation. While trip segmentation is crucial in the absence of labels, it is not the focus of this study. Therefore, we employ an existing method for segmentation. (Izakian, Saadi Mesgari, and Weibel 2020) receives multiple motion features as input and is able to efficiently deal with noisy trajectories, which is useful in several application scenarios. It is pertinent to note that our study focuses on the analysis of trajectories at intersections, where the recognition of behaviors is inherently challenging due to the inherent limitations of GPS accuracy and the complex interactions between objects. The discrepancies in features between the segmented sub-trajectories

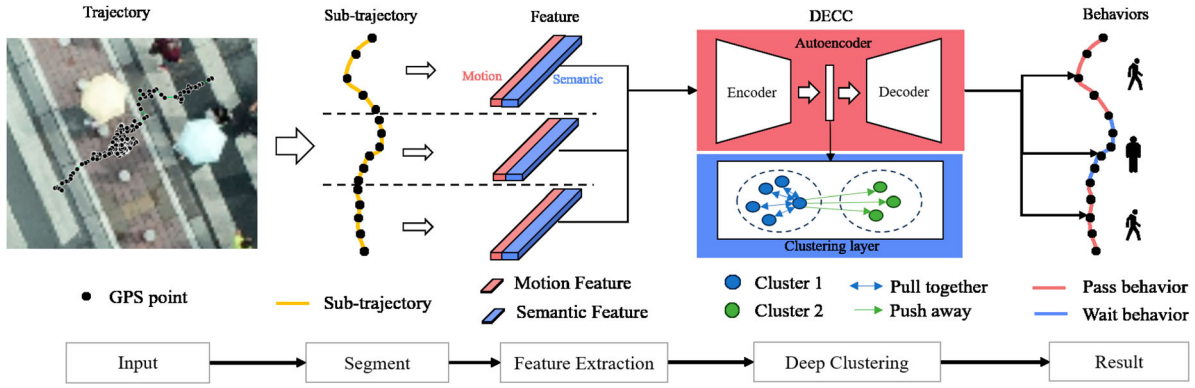

Figure 2. Basic architecture of our method.

are not readily apparent, necessitating the use of a deep clustering method to effectively mine and classify the underlying features.

The sub-trajectories resulting from segmentation vary significantly in both the number of points and length. However, our DECC requires the input with a shape of $(1, n_P, n_F)$, where n_P represents the number of points per segment and n_F denotes the number of features per point. In this paper, we set n_P to 64, which guarantees that the sub-trajectory is longer than 1 s, and adapts to the input size of the convolutional network (Dabiri and Heaslip 2018). Consequently, we divide every segment into 64-unit blocks without overlap, discarding any segments with fewer than 64 GPS points. This specific length, corresponding to a time span of 1.28 s at our data's sampling rate, was chosen based on a comprehensive sensitivity analysis detailed in Section 4.3.3. As shown in that section, this value provides an optimal balance between capturing sufficient behavioral context and avoiding the inclusion of mixed behaviors within a single segment.

3.1.2. Motion features

Latitude, longitude and time are the basic properties of GPS point, and used to characterize the motion features. However, using latitude and longitude pairs as features directly to train a clustering model can be problematic. Intuitively, the clustering model must be retrained whenever the user moves to a location that the model has not seen before (Markos and Yu 2020). Therefore, the advanced features should be designed to describe trajectories. Commonly used motion features in trajectory-based behavior recognition studies include velocity, acceleration, direction, and angle difference (Graser et al. 2024). However, the limited spatial extent of intersections and the accuracy of collected trajectories present challenges for reliable acceleration computation. Additionally, angle difference proves more effective than directional features at intersections with multiple travel directions. Given these considerations, this study adopts velocity and angle difference as the primary motion features.

We are concerned with waiting and passing behaviors, among which speed is a key feature for differentiation. To calculate speed, we use Vincenty's formula to determine the geographic distance between two subsequent GPS points, P_1 and P_2. Denoting the time difference between P_1 and P_2 as Δt (in seconds), the speed (in meters per second, m/s) is calculated as follows:

$$S_{p_1} = \frac{Vincenty(P_1, P_2)}{\Delta t} \tag{1}$$

Since people walking or bicycling have the ability to change direction frequently (Zheng et al. 2008), we use the angular difference of the trajectory points as one of the motion characteristics. The angle between the trajectory point direction and the reference direction (usually true north) is defined as Azimuth, and the angular difference is calculated as follows:

$$A_{p_1} = |Azimuth(P_2) - Azimuth(P_1)| \tag{2}$$

It is demonstrated that stationary individuals have higher localized densities compared to when they are in motion (Mullick et al. 2022). Therefore, we involve density as one of the motion features in the subsequent

analysis, calculated as follows:

$$N_{p_1} = \frac{Neighbour(P_1, R)}{Area(P_1, R)} \tag{3}$$

where R represents the statistical radius, $Neighbour(P, R)$ denotes the number of p neighboring GPS points exclusively from the same trajectory segment, and $Area(P, R)$ refers to the size of area within radius R of point P.

While motion features capture the dynamics of movement, the choice of input features critically influences model performance. Research in trajectory analysis has shown that supplementing motion data with relevant external features can lead to significant improvements (Wang et al. 2025). Therefore, to better capture the context of crossing behaviors, we introduce spatial and temporal semantic features.

3.1.3. Semantic features

The behavior of non-motorized transport at intersections is influenced by traffic organization and signal control plan. For instance, pedestrians and non-motorized vehicles are significantly more likely to wait at red lights than to pass. To better understand and reflect these patterns, we introduce spatial semantic features, which assess stopping behavior based on proximity to safety islands, and temporal semantic features, which evaluate stopping behavior in relation to the red phase of traffic signals. These features enhance our ability to recognition non-motorized transport behavior at intersections.

3.1.3.1. Spatial semantic feature. In this paper, spatial semantic feature is used to quantify the likelihood of waiting behavior for non-motorized transport at intersections, based on their distance to safety islands. Commonly, pedestrians and non-motorized vehicles display distinct behaviors at safety islands and crosswalks: safety islands function as designated waiting areas, while crosswalks are meant for passing through. Consequently, pedestrians and non-motorized vehicles are more likely to remain stationary at safety islands than at crosswalks.

To model this pattern, we employ the density function of the Generalized Gaussian Distribution (GGD), which encompasses a broader range of symmetric probability densities compared to the classical Gaussian model. This flexibility allows for more accurate representation of real data distributions. Specifically, compared to a standard Gaussian, the GGD's shape parameter provides greater flexibility to model different types of human behavior, such as the sharper, more concentrated stopping patterns observed near safety islands, which may not be perfectly captured by a standard Gaussian curve. The GGD function is defined as follows:

$$K_i(x) = e^{-\frac{(x-u_i)^2}{\sigma_i^2}} \tag{4}$$

where x denotes the projected coordinates of the trajectory point, and u_i denotes the coordinates of the center of the safety island i. The parameter σ_i influences the shape of the curve, with a smaller value of $|x - u_i|$ resulting in a larger value of K_i. This relationship is illustrated in Figure 3a, where $K_1(P_1) > K_1(P_2)$. The parameter σ_i is defined as $\sigma_i = \frac{L_i}{2}$, where L_i is the radius of the safety island i. A decrease in the radius L_i leads to a more concentrated stationing behavior, thereby increasing the value of K_i. As depicted in Figure 3a, it can be observed that $K_1(P_1) > K_2(P_3)$. The choice of $\frac{L_i}{2}$ ensures that K_i captures the majority of the stationing probability within the radius of the safety island L_i, thereby encompassing the most likely stationing locations.

3.1.3.2. Temporal semantic feature. Temporal semantic feature is used to quantify the likelihood of waiting behavior for non-motorized transport at intersections, based on their proximity to the red phase of traffic signals. Signal control creates specific red light intervals during which pedestrians and non-motorized vehicles must wait before crossing the street. With the exception of a few non-compliant individuals, the majority of the population follows the rule of waiting during the red light and crossing the street during the green light. To model this behavior, we utilize the Generalized Gaussian Distribution (GGD) density function, which is expressed as follows:

$$T_i(t) = e^{-\frac{(t-t_i)^2}{\sigma_i^2}} \tag{5}$$

where t represents the moment of the trajectory point, t_i denotes the moment corresponding to the center of the red light phase. The parameter σ_i influences the behavior of the function, with a smaller value of $|t - t_i|$

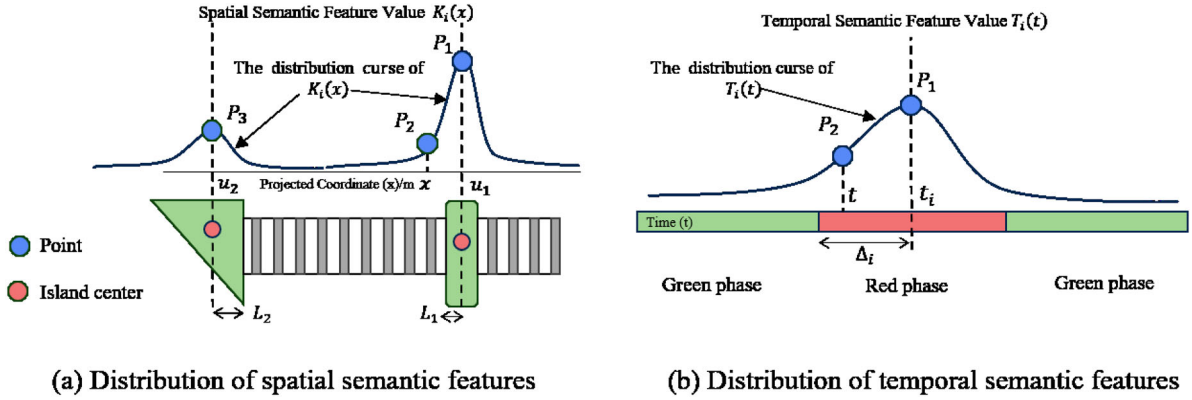

Figure 3. Illustration of spatial and temporal semantic features using GGD. (a) The GGD curve models the likelihood of waiting based on distance to a safety island center (u_i); a point P_1 closer to the center has a higher feature value than P_2. (b) The GGD curve models the likelihood of waiting based on time relative to the red light phase center (t_i); a point P_1 closer to the center of the red phase has a higher feature value.

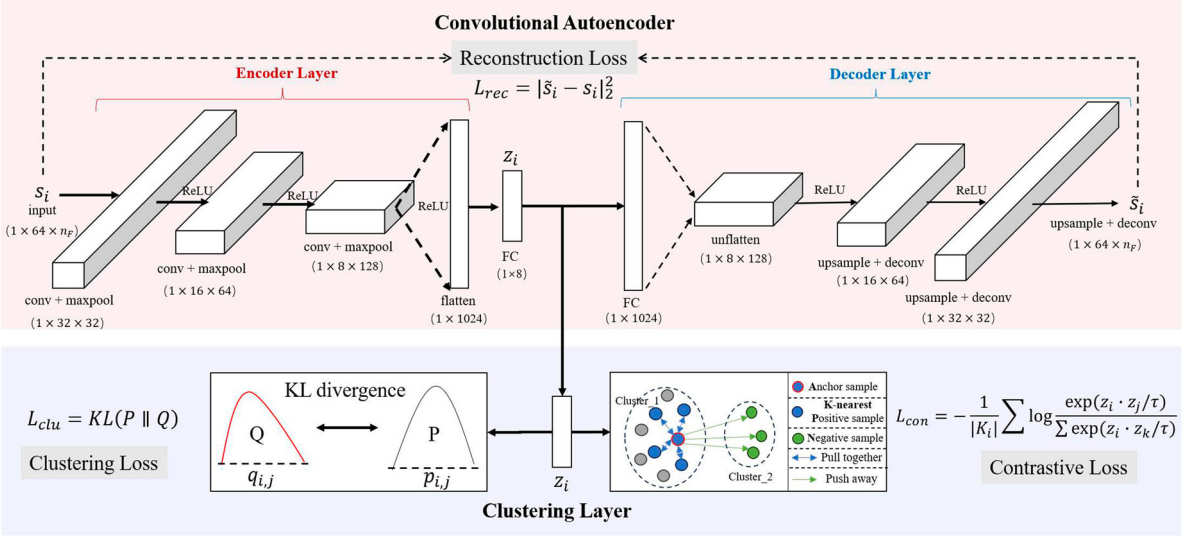

Figure 4. The overall architecture of our proposed unsupervised contrastive deep Learning framework, DECC.

resulting in a larger value of T_i. This relationship is illustrated in Figure 3b, where $T_1(P_1) > T_1(P_2)$. The σ_i is defined based on the duration of the red light phase, denoted as δ_i, and is set as $\sigma_i = \frac{\delta_i}{2}$.

Prior to being fed into the model, all input features are normalized to a range of [0, 1] to ensure stable training.

3.2. DECC architecture

Our approach, depicted in Figure 4, comprises two co-trained components: the CAE and clustering layer.

The CAE uses a series of convolutional and transposed convolutional operations to extract key information from the original features, which is known as feature embedding. Concurrently, the clustering layer attempts to train a classification model that guides the feature embedding process based on the classification effect of the features.

In the following sections, we will describe these two components in detail and present the proposed objective function.

3.2.1. Convolutional autoencoder

Convolutional Autoencoder (CAE) is a popular feature embedding framework. It processes data through a convolutional layer that is able to reconstruct the input to extract meaningful latent representations or embeddings that help the model better understand the data features. In our paper, the motion and semantic features from previous process are used as input of our CAE.

Specifically, our CAE is designed with a symmetric structure for both the encoder and the decoder, each comprising three convolutional layers and one fully connected layer. The encoder initially extracts significant features through three consecutive convolutional operations followed by max pooling. Subsequently, these features are mapped into a low-dimensional space using the fully connected layer. The decoder, on the other hand, performs inverse operations to restore the features through inverse convolution and upsampling. To enhance training efficiency and stability, a batch normalization layer is appended after each convolutional and inverse convolutional layer, except for the last one.

The filter sizes for all convolutional and maximum pooling layers are $(1 \times 3 \times C)$ and $(1 \times 2 \times C)$, respectively. For each layer, C indicates the number of channels in the input volume of the layer. Using smaller receptive fields reduces the number of parameters and mitigates the overfitting problem (Simonyan and Zisserman 2015). We multiply the number of channels of the model, which enables the network to learn a richer representation of the features. This setup has often been used in previous research (Dabiri and Heaslip 2018).

During training, we try to minimize the reconstruction loss L_{rec} of the CAE, quantified as the mean square error between the original input s_i and the reconstructed input $\hat{s}_i$:

$$L_{rec} = \frac{1}{n}\sum_{i=1}^{n}(s_i - \widehat{s_i})^2 \tag{6}$$

3.2.2. Clustering layer

The clustering layer of the CAE, as proposed by (Xie, Girshick, and Farhadi 2016), serves as a feature clustering mechanism attached to the embedding layer. The clustering layer obtains the feature representation from the CAE and then classifies the features. The DECC model's training, specifically for its clustering objective, is guided by a combination of loss functions designed to improve cluster quality. This primarily involves the clustering loss (L_{clu}), which is the KL divergence between predicted and target soft assignments, and is complemented by a contrastive loss (L_{con}) to enhance the separability of learned features. Together, these losses steer the model towards forming distinct clusters.

This layer comprises trainable weights $\{\mu_j\}_{j=1}^{C}$, where C represents the number of desired clusters and μ_j denotes the coordinates of the j-th center of mass. In this paper, we set C is set to 2 based on the number of behaviors we are analyzing.

3.2.2.1. Clustering loss. The original clustering layer employs a clustering loss to guide the training process, defined as the Kullback-Leibler (KL) divergence between two distributions: the distribution P and the distribution Q. The KL divergence measures the difference between these two distributions, effectively guiding the model to produce cluster assignments (via Q) that are increasingly aligned with the desired target distribution (P). Here, Q (defined in Equation [q]) represents the probability of assigning sample i to cluster j based on the similarity between its latent embedding z_i and the cluster centroid μ_j using the Student's t-distribution. These q_{ij} values are dynamically calculated for each sample in every training iteration based on the current state of its latent embedding z_i and the cluster centroids μ_j. The target distribution P (defined in Equation [p]) is computed from Q itself by squaring and normalizing, which helps to strengthen confident assignments and attenuate uncertain ones. This helps sharpen cluster assignments during training and the closer cluster assignments to desired target distribution, the better the model perform.

$$q_{ij} = \frac{(1+\parallel z_i - \mu_j\parallel^2)^{-1}}{\sum_j (1+\parallel z_i - \mu_j\parallel^2)^{-1}} \tag{7}$$

$$p_{ij} = \frac{q_{ij}^2/\sum_i q_{ij}}{\sum_j \left(q_{ij}^2/\sum_i q_{ij}\right)} \tag{8}$$

Thus, the clustering loss is defined as:

$$L_{clu} = KL(P \parallel Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{9}$$

Minimizing $KL(P \parallel Q)$ drives the learned distribution Q towards the target P. This encourages the model to produce more confident, less ambiguous soft assignments. In the context of our binary classification task (distinguishing 'wait' vs. 'pass' behaviors), this process is particularly beneficial. By iteratively sharpening the assignments, the loss function guides the autoencoder to learn a latent feature space (z_i) where the embeddings corresponding to the two distinct behaviors become more separable and cluster more tightly around their respective centroids (μ_{wait} and μ_{pass}). Even though the assignments remain soft, this increased separability in the latent space directly facilitates the differentiation between the two behavioral classes. This KL divergence-based loss is a well-established technique in deep unsupervised clustering, proposed by (Xie, Girshick, and Farhadi 2016) and refined by (Guo et al. 2017), demonstrating its effectiveness in learning discriminative features without direct supervision.

The initial clustering center μ_j is obtained by KM with pre-trained data embedding, and subsequent z_i are optimized together in turn. The original method by (Xie, Girshick, and Farhadi 2016) updates the target distribution P at the end of each epoch. In our approach, to achieve a more adaptive learning process where the target distribution P responds rapidly to changes in the learned features, we update P after each training iteration. This highly frequent update strategy ensures that P provides an immediate and dynamic target for the clustering loss L_{clu}, facilitating a tighter coupling between representation learning and cluster assignment refinement.

3.2.2.2. K-nearest neighbor contrastive loss. Previous works (Dang et al. 2021; Guo et al. 2017) pull instances belonging to the same class together while pushing samples from different classes. However, this often concentrates instances of a given class into a narrow region near the class center, reducing intra-class diversity, which is essential for capturing nuanced variations within a class. (Zhao et al. 2021) and (Li, Cheng, and Han 2024) found that larger intra-class diversity helps transfer knowledge to downstream tasks. To address this issue, we proposed a K-nearest neighbor contrastive loss, which preserves local sample similarity and relaxes the constraints by focusing only on the k-nearest neighbors of each sample. This approach avoids collapsing all class instances into a single point while maintaining meaningful local structure.

The key idea behind this loss is to balance the benefits of pulling similar samples closer while avoiding excessive concentration, thereby fostering a more dispersed intra-class distribution. By emphasizing only the nearest neighbors of the same class, the method accounts for local variations and ensures that meaningful relationships within the data are preserved. This approach reflects the natural structure of data distributions, where samples of the same class often form small and coherent clusters. We formulate the loss as follows.

$$L_{con} = -\frac{1}{|K_i|} \sum_{j=1}^{K_i} \log \frac{\exp(z_i \cdot z_j/\tau)}{\sum_{k=1}^{|A_{ij}|} \exp(z_i \cdot z_k/\tau)} \tag{10}$$

where K_i is the set of k nearest neighbors that have the same category as the i-th sample. A_{ij} denotes the concatenation of positive and negative samples that have different categories from the i-th sample.

3.2.2.3. Overall objective. In summary, our overall objective is:

$$L = L_{rec} + \gamma L_{clu} + \beta L_{con} \tag{11}$$

where the hyperparameters γ , β control the influence of clustering loss, constrastive loss on the total loss. In Section 4.3, we discuss the values of the hyperparameters.

4. Model evaluation with the observed dataset

This section first describes our simulation setup, including the configuration of model parameters, clustering hyperparameters, and the selection of evaluation metrics. It then presents the findings from two

case studies. The first case study compares the clustering performance of our proposed method against established baselines. The second case study examines the sensitivity of our methodology to variations in hyperparameters.

4.1. Simulation setup

4.1.1. Datasets

This dataset is collected using a DJI Mavic 2 ZOOM model drone, capturing road traffic in 2 K resolution at 50 frames per second. For target identification, we employed the YOLOv7 model, which facilitates the detection of objects. Subsequently, the Bot-SORT model is utilized for tracking, allowing us to extract individual labels and behavioral trajectories. The dataset consists of 3 million GPS points that cover the morning rush hour on the workday, from more than 8000 pedestrians and non-motorized vehicles. Each GPS point is annotated with ID for detected objects, their class labels (e.g. pedestrian, non-motorized vehicle), time, longitude and latitude. The other irrelevant or inaccurate fields have been discarded.

4.1.2. Comparing methods

To comprehensively evaluate the performance of our Deep Embedded Contrastive Clustering (DECC) algorithm, we have carefully selected a range of representative comparison methods. These methods encompass not only traditional clustering algorithms but also newest unsupervised deep clustering approaches that align with the research focus of our algorithm. The selected methods are categorized as follows:

(1) Traditional clustering on original features. This category includes widely used clustering algorithms that operate directly on the original feature space, such as K-means (KM) and Spectral Clustering (SC), implemented in Python's scikit-learn library.
(2) Two-stage clustering on embedded features. These methods involve transforming the original features into a more compact representation using an CAE, followed by clustering the embeddings with traditional methods like K-means or Spectral Clustering. This approach, referred to as CAE + KM and CAE + SC, helps assess the impact of feature embedding on clustering performance.
(3) Deep clustering with CAE and contrastive learning. This category includes deep clustering frameworks that combine feature learning and clustering. We compare Improved Deep Embedded Clustering (IDEC) (Guo et al. 2017), Contrastive Clustering (CC) (Li et al. 2020) and Transformer AutoEncoder for K-means Efficient clustering (TAKE) (Wu et al. 2024). IDEC integrates clustering loss into the autoencoder framework, while CC leverages contrastive learning to enhance feature separability. TAKE is a state-of-the-art method that combines a Transformer-based autoencoder with soft K-means to achieve unsupervised clustering. These methods provide a comparison to evaluate the effectiveness of the contrastive learning approach used in DECC.
(4) Newest adaptive clustering methods. Recent developments have introduced adaptive methods aimed at overcoming the limitations of traditional algorithms. Adaptive Marine Traffic Behavior Recognition Algorithm (AMTBRA) (Wei et al. 2024), an extension of DBSCAN, is included as a comparison to assess its effectiveness in handling large-scale, spatially complex data.

4.1.3. Evaluation metric

All clustering methods are evaluated by clustering accuracy (ACC), Normalized Mutual Information (NMI), Adjusted Rand Index (ARI) and Silhouette Coefficient (SIL) which are widely used in unsupervised learning scenario. Higher values of these metrics indicate better clustering performance.

4.1.4. Parameters setting

The CAE encoder consists of three convolutional layers with 32, 64, and 128 filters, followed by one fully connected multilayer perceptron (MLP). The convolutional layers utilize a kernel size of 1×3 with same padding. The MLP flattens the output of the convolutional layers into a vector and then applies a linear transformation to a hidden layer with dimension d. All internal layers, except for the input, output, and embedding layers, employ the ReLU activation function (Glorot, Bordes, and Bengio 2011). The CAE is pretrained for 200 epochs using the Adam optimizer with an initial learning rate of 0.01. After attaching the clustering layer, we retrain

the composite model using a reduced learning rate of 0.001. According to experiment, we determine $\gamma = 0.2$, $\beta = 0.1$ when clustering based on motion, semantics, or all features. Training stops when fewer than 0.1% of samples are reassigned to a different cluster during an iteration.

The proposed DECC model comprises approximately 195,000 trainable parameters. All experiments were conducted on a workstation equipped with an Intel Core i9-10900 K CPU and an NVIDIA GeForce RTX 3080 GPU with 32GB of RAM, using PyTorch 1.12.0. In terms of processing speed for the primary experiments, we observed that a data segment corresponding to approximately 3 million GPS points (after pre-processing and segmentation) could be processed by the DECC framework in roughly 5 minutes, indicating efficient batch processing capabilities.

4.2. Clustering evaluation

4.2.1. Comparison of clustering performance

To evaluate the effectiveness of integrating motion and semantic features, we assess clustering performance under three scenarios: (i) using motion features, (ii) using semantic features, and (iii) using a combination of both. Results, presented as mean % $\pm$ standard deviation over 30 independent runs (as detailed in Table 1), are compared against seven baseline methods outlined in Section 4.1. Statistical significance of the differences between DECC and key baselines (IDEC and TAKE) was evaluated using paired t-tests ($\alpha = 0.05$).

As shown in Table 1, our proposed DECC framework achieves the best performance across all scenarios, with an ACC of 91.3$\pm$1.0% and a silhouette score (SIL) of 84.3$\pm$0.4% when using the full feature set. This performance is statistically significantly higher ($p < 0.05$, based on paired t-tests) than that of traditional clustering methods (KM and SC), embedding-based methods (CAE + KM and CAE + SC), and the self-supervised approach TAKE (ACC: 82.9$\pm$2.4%, NMI: 39.7$\pm$0.7%, ARI: 43.3$\pm$3.4%) († in Table 1). Notably, DECC statistically significantly surpasses IDEC (ACC: 86.9$\pm$1.2%, NMI: 46.4$\pm$2.0%, ARI: 54.3$\pm$2.3%) ($*$ in Table 1) in ACC, NMI, and ARI when using all features. For instance, DECC's average ACC is 4.4% higher than IDEC's and 8.4% higher than TAKE's under this condition, with these differences being statistically significant. These results demonstrate the critical role of contrastive loss and semantic feature fusion in enhancing discriminative power. The superiority of DECC in terms of cluster quality is further validated by its SIL score (84.3$\pm$0.4%), which is statistically significantly higher than those of TAKE (71.5$\pm$4.0%) and IDEC (79.9$\pm$0.5%) ($*$ and † respectively), indicating more compact and well-separated clusters.

The advantage of DECC becomes particularly pronounced in complex scenarios. While TAKE leverages Transformer architectures to model global spatio-temporal dependencies, its performance declines sharply with semantic features alone (ACC = 59.0$\pm$2.1%, SIL = $-$0.4$\pm$7.0%), as it struggles to interpret contextual semantic indicators like traffic signal phases, which differ significantly from the dense spatio-temporal sequences Transformers typically model. In contrast, DECC maintains robust performance even with semantic features alone (ACC = 63.4$\pm$1.5%, SIL = 78.1$\pm$0.9%), owing to its adaptive fusion mechanism that weights semantic contexts based on spatial proximity and temporal alignment. This capability allows DECC to resolve ambiguities in crowded intersections – for instance, distinguishing pedestrians waiting near safety islands during red lights from those temporarily pausing in crosswalks.

Motion features alone yield strong results across most methods, reflecting their fundamental role in capturing dynamic patterns. With motion features, DECC achieves an ACC of 87.3$\pm$0.9% and a SIL of 73.7$\pm$1.2%. This performance is statistically significantly higher ($p < 0.05$ based on respective markers in Table 1) in terms of ACC, NMI, and ARI compared to both TAKE (ACC = 77.2$\pm$1.2%, NMI = 22.0$\pm$2.3%, ARI = 21.3$\pm$2.5%) and IDEC (ACC = 79.4$\pm$1.0%, NMI = 34.2$\pm$2.5%, ARI = 37.8$\pm$2.8%). This gap in accuracy metrics stems from DECC's contrastive loss, which enforces local similarity preservation among trajectory segments. For example, abrupt deceleration patterns (characteristic of waiting behaviors) are clustered more cohesively in DECC's embedding space, whereas TAKE's global attention mechanism occasionally conflates them with slow-moving passing trajectories.

The integration of motion and semantic features consistently improves performance. DECC's average ACC increases by 4.0% (from 87.3$\pm$0.9% to 91.3$\pm$1.0%) when combining both features, while TAKE shows a larger average gain of 5.7% (from 77.2$\pm$1.2% to 82.9$\pm$2.4%), though DECC's final combined performance remains significantly higher. This discrepancy in overall leading performance highlights DECC's unique ability to synergize heterogeneous features: semantic constraints (e.g. proximity to safety islands) refine cluster boundaries that

Table 1. Comparison of Clustering Performance (metrics in %). Results show mean % ± standard deviation over 30 runs. Statistical significance compared to DECC (our) was determined using paired t-test ($\alpha = 0.05$). $*$ indicates $p < 0.05$ compared to DECC. † indicates $p < 0.05$ compared to DECC.

	Motion Features				Semantic Features				All Features			
Method	ACC	NMI	ARI	SIL	ACC	NMI	ARI	SIL	ACC	NMI	ARI	SIL
KM	64.0±1.5	16.8±2.0	6.9±1.0	41.5±3.0	**64.7±1.8**	0.6±0.3	3.3±0.8	25.7±4.0	73.6±0.2	22.6±0.4	22.3±0.4	42.6±0.6
SC	67.4±1.6	3.0±0.8	0.9±0.5	55.1±2.5	63.3±2.0	1.0±0.5	9.0±1.5	48.8±3.0	68.2±1.4	11.0±1.0	13.0±1.2	65.9±2.0
CAE + KM	65.9±1.7	20.2±2.2	15.8±1.8	15.7±5.0	55.3±2.5	3.2±0.9	0.5±0.4	43.4±3.5	68.3±1.5	24.6±1.8	12.8±1.5	16.7±4.5
CAE + SC	59.6±2.0	12.0±1.5	6.2±1.1	14.0±5.5	58.1±2.2	0.1±0.1	0.5±0.3	−9.8±6.0	62.8±1.8	6.4±0.9	6.5±0.8	18.9±4.0
IDEC	79.4±1.0*	34.2±2.5*	37.8±2.8*	77.3±1.0	57.0±2.3	2.6±0.7	1.8±0.6	67.9±2.0	86.9±1.2*	46.4±2.0*	54.3±2.3*	79.9±0.5*
CC	48.4±2.5	2.3±0.7	1.2±0.6	43.2±3.5	30.2±3.0	1.1±0.5	0.8±0.4	7.8±6.5	54.7±2.0	2.3±0.6	2.1±0.5	37.2±3.0
TAKE	77.2±1.2†	22.0±2.3†	21.3±2.5†	**78.5±0.8**	59.0±2.1	0.3±0.2	0.1±0.1	−0.4±7.0	82.9±2.4†	39.7±0.7†	43.3±3.4†	71.5±4.0†
AMTBRA	69.1±1.5	28.0±2.0	17.3±1.9	56.9±2.2	46.6±2.8	**25.4±2.5**	**13.8±1.7**	48.2±3.2	71.3±1.3	28.6±1.7	19.8±1.6	61.3±2.5
DECC (our)	**87.3±0.9**	**49.6±1.8**	**49.9±2.0**	73.7±1.2	63.4±1.5	3.2±0.8	2.8±0.7	**78.1±0.9**	**91.3±1.0**	**57.1±1.6**	**67.3±1.7**	**84.3±0.4**

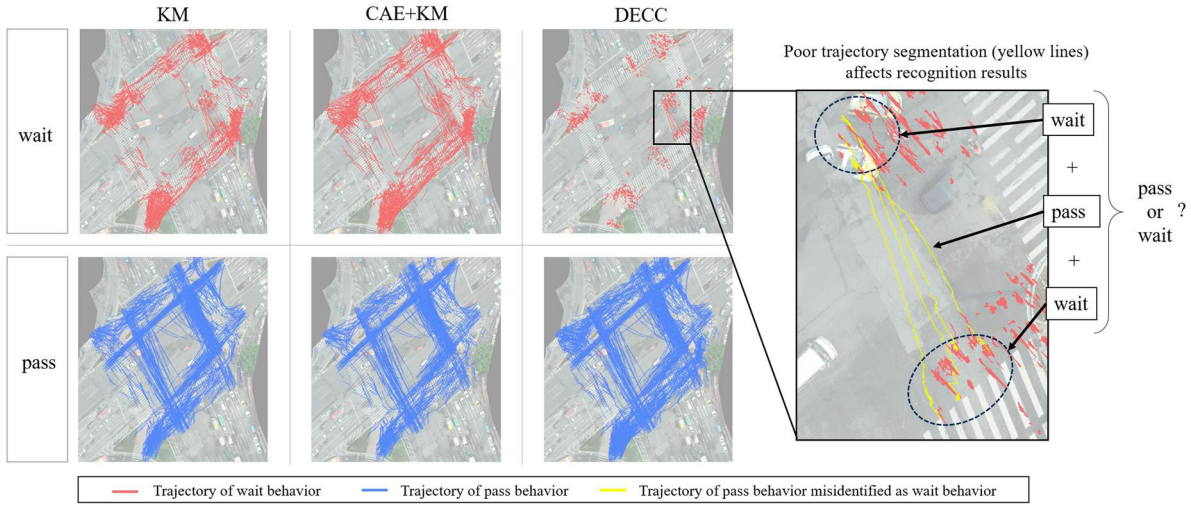

Figure 5. Visualization of clustering results for KM, CAE + KM, and DECC. Red lines indicate trajectories classified as 'wait' behavior, blue lines indicate 'pass' behavior, and yellow lines highlight 'pass' trajectories misclassified as 'wait' by DECC. The rightmost panel shows a zoomed-in view.

motion features alone cannot disambiguate. The SIL metric corroborates this observation, with DECC's average score rising from 73.7±1.2% to 84.3±0.4%, compared to TAKE's decrease from 78.5±0.8% to 71.5±4.0% when all features are combined.

Traditional methods (KM, SC) and recent approaches (CC, AMTBRA) exhibit limited adaptability. CC's reliance on sample-wise differences results in poor performance (ACC = 54.7±2.0%), as subtle distinctions between GPS trajectories are insufficient for effective contrastive learning. AMTBRA, designed for large-scale spatial data, underperforms (ACC = 71.3±1.3% with All Features) due to insufficient trajectory length in our dataset. In contrast, DECC's hybrid design – jointly optimizing reconstruction, clustering, and contrastive losses – enables robust performance across diverse feature configurations, achieving statistically significant state-of-the-art results in most key metrics even in noisy, short-trajectory scenarios.

4.2.2. Visualization of clustering

We visualize the results on a map to intuitively demonstrate the clustering effects of various methods, as illustrated in Figure 5. Most waiting behavior trajectories at intersections are expected to exhibit clustered geometries concentrated near safety islands, while passing behavior trajectories typically display linear geometries that span a larger area. These characteristics enable an evaluation of clustering effectiveness based on trajectory distribution and morphology. However, the visualization of KM and CAE + KM shows suboptimal performance, with numerous passing behaviors incorrectly classified as waiting behaviors near safety islands and along pedestrian crossings. Our method clearly distinguishes between the two types of behaviors, with waiting and passing trajectories aligning with established expectations. We observe that waiting behaviors are predominantly concentrated in the edge regions of the safety island, with many extending beyond its boundaries. In contrast, the visualizations for KM and CAE + KM are less encouraging, as they incorrectly classify many passing behaviors within the safety island and pedestrian crossing.

It is important to note that our method also encounters instances of misclassifying crossing behavior, as highlighted in the black box. These misclassifications are more prevalent among trajectories with geometric variations, such as the yellow trajectory, where segments are tighter at the start and end points but sparser in between. We believe this issue arises from trajectory segmentation that does not adequately separate segments for waiting and passing behaviors.

4.2.3. Performance analysis during training

Figure 6 illustrates the performance of our method during training, revealing several key observations. First, the overall loss and reconstruction loss are nearly identical and significantly larger than the clustering loss. This can be attributed to the high initial clustering accuracy achieved through pretraining, which provides a strong

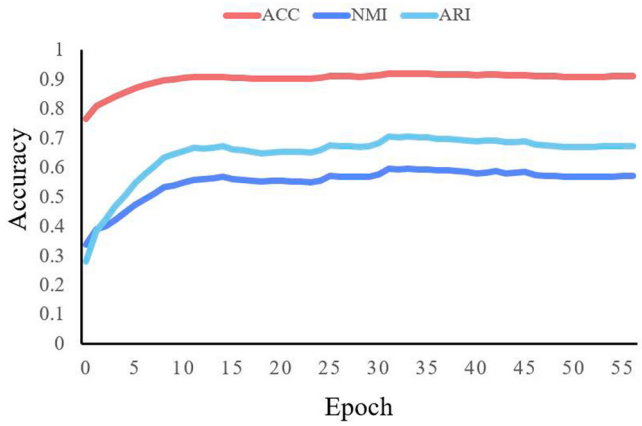

Figure 6. Losses and accuracies during training.

initialization, and the limited number of clusters, which reduces the complexity of the clustering task. A similar phenomenon has been reported in preview work (Xie, Girshick, and Farhadi 2016).

The rapid decline of contrastive loss in the initial phase is attributed to its local nature. By focusing on K-nearest neighbors (K = 20), the contrastive loss quickly identifies and reinforces local similarities between trajectory segments (e.g. waiting behaviors near safety islands). This enables the model to form preliminary clusters even before global alignment is achieved. In contrast, cluster loss decreases gradually because it optimizes the global alignment of cluster assignments. The high initial accuracy of K-means (73.6% in Figure 6(b) red line) provides a strong starting point, but further refinement requires iterative adjustments to the cluster centroids and the target distribution P. This phased optimization ensures that local structures are preserved while global clusters become more compact and separable. These observations align with the design philosophy of DECC, which leverage local contrastive learning to bootstrap clustering, followed by joint optimization to consolidate global behavior patterns.

Additionally, we note that the losses remain relatively stable, with a notable decrease in contrastive loss at the beginning of training, while model accuracy improves significantly. This indicates that even though the reconstructed features closely resemble the originals, our loss function effectively manipulates the embedding space to enhance the representation of intermediate embeddings, thereby improving clustering accuracy.

4.2.4. Visualization of the feature learning

We further prove our assumption about the role CAE acts by visualizing the embedded feature space during training. The t-SNE (van der Maaten and Hinton 2008) visualization of dataset is shown in Figure 7. In the initial (epoch 0) embedding space, although there is a clear clustering phenomenon, the two classes of samples are not well separated and the clustering spaces overlap. With training proceeding, different categories of samples gradually separated to form two regions independent of each other. From the final results, the distribution of these misclassified samples is also relatively concentrated, although there is still a mix of samples from different categories.

4.3. Hyperparameter sensitivity

In the process of model construction, the selection of hyperparameters has a significant impact on the performance. We carry out an in-depth study on the three key hyperparameters γ , β and K, and determine their optimal values through a series of experiments.

4.3.1. Sensitivity in loss function weight

To assess the proposed model's sensitivity to hyperparameters γ and β, we conduct experiment by sampling γ and β in range $[10^{-1}; 10^{2}]$. The Figure 8 shows the corresponding clustering accuracy results for different γ and β.

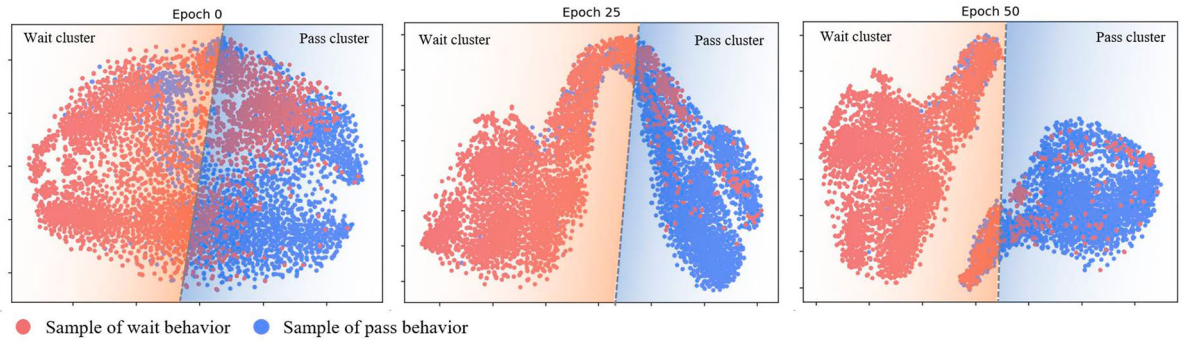

Figure 7. Visualization of features on dataset during training at Epoch 0, Epoch 25, and Epoch 50. Different colors mark different clusters. The t-SNE visualization was generated with a perplexity value of 30 to ensure reproducibility.

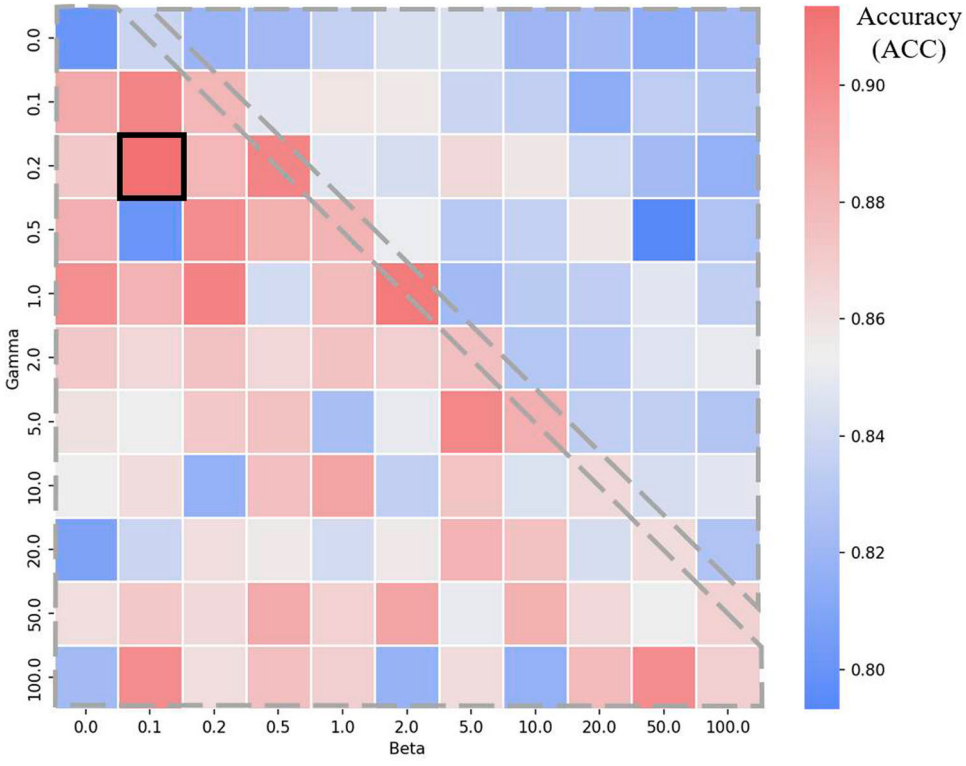

Figure 8. Sensitivity of accuracy to strength of clustering loss and contrastive loss.

The selection of γ and β was guided by their roles in balancing the clustering loss L_{clu}, contrastive loss L_{con}, and reconstruction loss L_{rec}. Specifically, γ controls the influence of the clustering loss, which is based on the Kullback-Leibler (KL) divergence between the target distribution P and the predicted distribution Q. A small γ (e.g. $\gamma < 0.1$) reduces the impact of clustering loss, leading to suboptimal cluster formation, while a large γ (e.g. $\gamma > 0.5$) may distort the latent feature space by overemphasizing clustering at the expense of reconstruction accuracy. Similarly, β controls the contribution of the contrastive loss, which preserves the local structure of the data by pulling similar samples closer and pushing dissimilar samples apart. However, experiments have shown that clustering based solely on contrastive loss is ineffective, so the value of β should be set lower to ensure that the contrastive loss complements the clustering loss without overwhelming it.

When both γ and β are set to 0, the model deteriorates into a purely convolutional autoencoder, resulting in the lowest accuracy. When $\gamma = 0$ and $\beta > 0$, the model accuracy increases but not significantly. Conversely, with $\beta = 0$ and $\gamma > 0$, the model accuracy is sensitive to the value of γ . γ and β are related hyperparameters, and when β is set to be greater than γ (upper right part of Figure 8), the clustering accuracy tends to be lower compared to when β is less than γ (lower left part of the Figure 8). Adjusting these coefficients can improve

Table 2. Clustering performance in different K Values (metrics in %).

K Value	1	2	5	10	20	50	100
ACC	85.1	85.7	87.2	86.6	91.3	85.5	85.8
NMI	43.1	44.7	46.7	48.7	57.1	43.2	44.0

Table 3. Clustering performance with different segment lengths (metrics in %).

Metric	32	64	128	256	512	1024
ACC	87.7	**91.3**	90.6	79.8	67.8	63.6
NMI	43.6	**57.1**	59.5	30.0	1.7	12.0

clustering accuracy, but the relationship is not linear. When the coefficients take on excessively large values, the clustering accuracy actually decreases.

Considering the results shown in Figure 8, we recommend using the combination of $\gamma = 0.2$ and $\beta = 0.1$ to achieve high performance from the model. This specific setting of the hyperparameters strikes a good balance, allowing the model to leverage the strengths of both the clustering loss (weighted by γ) and the contrastive loss (weighted by β) to produce accurate clustering results.

4.3.2. Sensitivity in K-neighboring values

We further investigate the influence of the K value on the accuracy of results in the K-neighborhood contrast design. The experiment was conducted using the optimal values of γ and β, with K values ranging from 1 to 100. Table 2 presents the corresponding clustering accuracy results. The parameter K determines the size of the neighborhood used in the contrastive loss, which plays a critical role in preserving the local structure of the data. A small K (e.g.$K < 10$) is susceptible to noise and fails to capture sufficient local structure, while a large K (e.g. $K > 50$) dilutes the discriminative power of the contrastive loss by including too many dissimilar samples. To strike a balance, we chose $K = 20$, which allows the model to effectively leverage local similarity information without excessive concentration of samples around cluster centers. This choice is supported by previous studies, such as SCAN (Van Gansbeke et al. 2020), where clustering performance was optimal when K was set to values around 20. As shown in Table 2, when $K = 20$, the model achieves the highest clustering accuracy (91.3%) and NMI (57.1%). This value effectively captures the local structure of the data while avoiding the pitfalls of overly small or large neighborhoods.

4.3.3. Sensitivity to input segment length

To assess the influence of the fixed-length segmentation on model performance, we conducted a sensitivity analysis by varying the input segment length. Trajectories were sampled at 50 frames per second (0.02 s per point). We evaluated segment lengths from 32 points (0.64 s) to 1024 points (20.48 s), keeping other DECC hyperparameters ($\gamma = 0.2, \beta = 0.1, K = 20$) optimal.

As shown in Table 3, the segment length of 64 points (1.28 s), utilized in our main experiments, achieved the highest ACC (91.3%) and NMI (57.1%). With shorter segments, such as 32 points (0.64 s), performance moderately decreased (ACC: 87.7%, NMI: 43.6%). This very short duration might lead to features being susceptible to transient data fluctuations or not capturing sufficient contextual information to robustly represent a distinct 'wait' or 'pass' behavior. Increasing the segment length to 128 points (2.56 s) maintained high performance (ACC: 90.6%, NMI: 59.5%), with NMI even slightly improving, suggesting this duration still effectively captures coherent behaviors. However, further increasing the segment length to 256 points (5.12 s) and beyond led to a notable decline in both ACC and NMI. For instance, at 256 points, ACC dropped to 79.8% and NMI to 30.0%; at 1024 points (20.48 s), ACC was 63.6% and NMI 12.0%. This degradation with these longer durations (over 5 s) is likely because such segments have a higher probability of encompassing multiple distinct behaviors (e.g. a short wait followed by a pass).

These results indicate that while performance is robust within a certain temporal range (e.g. approximately 1–3 s for this dataset), very short or excessively long segments can be detrimental. The chosen length of 64 points (1.28 s) is thus a reasonable and empirically supported selection for our primary analysis, effectively balancing contextual information and behavioral homogeneity.

5. Discussion and conclusion

This paper presents the Deep Embedded Contrastive Clustering (DECC) framework for recognizing the wait-or-pass behavior of non-motorized transport participants using GPS trajectory data. By integrating semantic features with motion features and employing a K-nearest neighbor contrastive loss, the proposed framework achieves state-of-the-art performance, with clustering accuracy reaching 91.3%. The results validate the importance of combining semantic information and motion characteristics to enhance behavior recognition at intersections.

Despite its strengths, the proposed method faces challenges related to trajectory segmentation. The use of fixed-length segmentation can sometimes produce sub-trajectories that do not align with the natural boundaries of behaviors. Addressing this limitation through adaptive segmentation methods or integrating segmentation into the learning process is a promising direction for future research. Furthermore, while the framework focuses on binary classification, its extension to multi-class behavior recognition would enhance its applicability in more complex traffic scenarios.

From a practical perspective, the DECC framework offers substantial potential for urban traffic management and safety applications. Real-time recognition of non-motorized transport behaviors can enable dynamic traffic signal adjustments, such as extending green light durations for crowded pedestrian crossings, to improve traffic flow and reduce congestion. The generated behavior data can also inform urban planning decisions, such as optimizing road layouts, enhancing infrastructure for non-motorized transport, and developing targeted safety policies. Moreover, the framework's unsupervised learning approach eliminates the reliance on extensive labeled datasets, making it adaptable to a wide range of urban environments.

From a practical perspective, the DECC framework offers substantial potential for urban traffic management and safety applications. The efficient recognition of non-motorized transport behaviors, supported by the model's computational performance (as discussed in Section 4.1 where model size and processing speed observations are noted), can enable dynamic traffic signal adjustments, such as extending green light durations for crowded pedestrian crossings, to improve traffic flow and reduce congestion. The generated behavior data can also inform urban planning decisions, such as optimizing road layouts, enhancing infrastructure for non-motorized transport, and developing targeted safety policies. Moreover, the framework's unsupervised learning approach eliminates the reliance on extensive labeled datasets, making it adaptable to a wide range of urban environments.

From a practical perspective, the DECC framework offers substantial potential for applications within intelligent transportation control systems and urban planning. The model's capability for rapid processing of trajectory data (as evidenced by its efficiency in handling GPS data in Section 4.1) facilitates the timely recognition of non-motorized transport behaviors. This timely behavioral information can, for instance, enable adaptive traffic signal adjustments on a periodic basis. For example, insights derived from the behavior patterns of one signal cycle or a short time window could be used to optimize signal timings for subsequent cycles, such as extending green light durations for crowded pedestrian crossings, thereby improving traffic flow and reducing congestion. The generated behavior data can also inform longer-term urban planning decisions, such as optimizing road layouts, enhancing infrastructure for non-motorized transport, and developing targeted safety policies. Moreover, the framework's unsupervised learning approach eliminates the reliance on extensive labeled datasets, making it adaptable to a wide range of urban environments.

While the DECC framework demonstrates strong performance in the studied intersection, its generalizability to diverse urban environments is an important consideration for future research. The current study focuses on validating the framework's core capabilities in a specific context, but future work will explore adaptive feature engineering and robustness improvements to enhance its applicability to varying intersection designs, signal controls, and traffic conditions. Additionally, we will investigate techniques to improve the robustness of motion and semantic features in environments with low GPS accuracy or high noise levels.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the Department of Science and Technology of Guangdong Province through the Guangdong Provincial Key-Area Research (2023YFB4301900) and Development Program project (2022B0101070002) titled 'Research on the Artificial Intelligence Application Innovation System and Technology Platform for Smart Highways'.

References

Ahmed, Shams Forruque, Md. Sakib Bin Alam, Maliha Kabir, Shaila Afrin, Sabiha Jannat Rafa, Aanushka Mehjabin, and Amir H. Gandomi. 2025. "Unveiling the Frontiers of Deep Learning: Innovations Shaping Diverse Domains." *Applied Intelligence* 55 (7): 1–55.

Batchuluun, Ganbayar, Jong Hyun Kim, Hyung Gil Hong, Jin Kyu Kang, and Kang Ryoung Park. 2017. "Fuzzy System Based Human Behavior Recognition by Combining Behavior Prediction and Recognition." *Expert Systems with Applications* 81:108–133. https://doi.org/10.1016/j.eswa.2017.03.052.

Bian, Jiang, Dayong Tian, Yuanyan Tang, and Dacheng Tao. 2019. "Trajectory Data Classification: A Review." *ACM Transactions on Intelligent Systems and Technology* 10 (4): 1–34. https://doi.org/10.1145/3330138.

Buch, Norbert, Sergio A. Velastin, and James Orwell. 2011. "A Review of Computer Vision Techniques for the Analysis of Urban Traffic." *IEEE Transactions on Intelligent Transportation Systems* 12 (3): 920–939. https://doi.org/10.1109/TITS.2011.2119372.

Caron, Mathilde, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2019. "Deep Clustering for Unsupervised Learning of Visual Features." https://arxiv.org/abs/1807.05520.

Chen, Hao, Xi Zhang, Wenyan Yang, and Yiwei Lin. 2023. "A Data-Driven Stacking Fusion Approach for Pedestrian Trajectory Prediction." *Transportmetrica B: Transport Dynamics* 11 (1): 548–571. https://doi.org/10.1080/21680566.2022.2103050.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. "A Simple Framework for Contrastive Learning of Visual Representations." https://arxiv.org/abs/2002.05709.

Chen, Wangxing, Haifeng Sang, Jinyu Wang, and Zishan Zhao. 2024. "IMGCN: Interpretable Masked Graph Convolution Network for Pedestrian Trajectory Prediction." *Transportmetrica B: Transport Dynamics* 12 (1): 2389896. https://doi.org/10.1080/21680566.2024.2389896.

Dabiri, Sina, and Kevin Heaslip. 2018. "Inferring Transportation Modes from GPS Trajectories Using a Convolutional Neural Network." *Transportation Research Part C: Emerging Technologies* 86:360–371. https://doi.org/10.1016/j.trc.2017.11.021.

Dang, Min, Gang Liu, Qijie Xu, Ke Li, Di Wang, and Lihuo He. 2024. "Multi-Object Behavior Recognition Based on Object Detection for Dense Crowds." *Expert Systems with Applications* 248:123397. https://doi.org/10.1016/j.eswa.2024.123397.

Dang, Zhiyuan, Cheng Deng, Xu Yang, and Heng Huang. 2021a. "Doubly Contrastive Deep Clustering." https://arxiv.org/abs/2103.05484.

Dang, Zhiyuan, Cheng Deng, Xu Yang, Kun Wei, and Heng Huang. 2021b. "Nearest Neighbor Matching for Deep Clustering." 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 13688–13697.

Dupuis, Yohan, Peggy Subirats, and Pascal Vasseur. 2016. "A Survey of Vision-Based Traffic Monitoring of Road Intersections." *IEEE Transactions on Intelligent Transportation Systems* 17 (April): 2681–2698. https://doi.org/10.1109/TITS.2016.2530146.

Endo, Yuki, and Toda Hiroyuki. 2016. Deep Feature Extraction from Trajectories for Transportation Mode Estimation.".

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. 2011. "Deep Sparse Rectifier Neural Networks." International Conference on Artificial Intelligence and Statistics. https://api.semanticscholar.org/CorpusID:2239473.

Graser, Anita, Anahid Jalali, Jasmin Lampert, Axel Weißenfeld, and Krzysztof Janowicz. 2024. "MobilityDL: A Review of Deep Learning from Trajectory Data." *GeoInformatica* 29 (1): 115–147. https://doi.org/10.1007/s10707-024-00518-8.

Guo, Xifeng, Long Gao, Xinwang Liu, and Jianping Yin. 2017. "Improved Deep Embedded Clustering with Local Structure Preservation." Proceedings of the 26th International Joint Conference on Artificial Intelligence, 1753–1759. IJCAI'17. Melbourne, Australia: AAAI Press.

Hasain, N. Mohamed, and Mokaddes Ali Ahmed. 2024. "Identification of Critical Conflicts and Its Implementation in Microsimulation for Effective Safety Assessment in Unsignalised Intersections." *Transportmetrica B: Transport Dynamics* 12 (1): 2350520. https://doi.org/10.1080/21680566.2024.2350520.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. "Momentum Contrast for Unsupervised Visual Representation Learning." https://arxiv.org/abs/1911.05722.

Hu, Danlei, Ziquan Fang, Hanxi Fang, Tianyi Li, Chunhui Shen, Lu Chen, and Yunjun Gao. 2022. "Estimator: An Effective and Scalable Framework for Transportation Mode Classification over Trajectories." https://arxiv.org/abs/2212.05502.

Hu, Kai, Junlan Jin, Fei Zheng, Liguo Weng, and Yiwu Ding. 2023. "Overview of Behavior Recognition Based on Deep Learning." *Artificial Intelligence Review* 56 (3): 1833–1865. https://doi.org/10.1007/s10462-022-10210-8.

Ismaeel, Ayad Ghany, Krishnadas Janardhanan, Manishankar Sankar, Yuvaraj Natarajan, Sarmad Nozad Mahmood, Sameer Alani, and Akram H. Shather 2023 "Traffic Pattern Classification in Smart Cities Using Deep Recurrent Neural Network." *Sustainability* 15 (19): 14522. https://doi.org/10.3390/su151914522.

Izakian, Zahedeh, M. Saadi Mesgari, and Robert Weibel. 2020. "A Feature Extraction Based Trajectory Segmentation Approach Based on Multiple Movement Parameters." *Engineering Applications of Artificial Intelligence* 88:103394. https://doi.org/10.1016/j.engappai.2019.103394.

Jiang, Zhuxi, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. "Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering." https://arxiv.org/abs/1611.05148.
Korbmacher, Raphael, Huu-Tu Dang, and Antoine Tordeux. 2024. "Predicting Pedestrian Trajectories at Different Densities: A Multi-Criteria Empirical Analysis." *Physica A: Statistical Mechanics and Its Applications* 634:129440. https://doi.org/10.1016/j.physa.2023.129440.
Li, Cong, Gong Cheng, and Junwei Han. 2024. "Boosting Knowledge Distillation via Intra-Class Logit Distribution Smoothing." *IEEE Transactions on Circuits and Systems for Video Technology* 34 (6): 4190–4201. https://doi.org/10.1109/TCSVT.2023.3327113.
Li, Yunfan, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2020. "Contrastive Clustering." https://arxiv.org/abs/2009.09687.
Liu, Yan, Rushdi Alsaleh, and Tarek Sayed. 2024. "Modelling Motorized and Non-Motorized Vehicle Conflicts Using Multiagent Inverse Reinforcement Learning Approach." *Transportmetrica B: Transport Dynamics* 12 (1): 2314762. https://doi.org/10.1080/21680566.2024.2314762.
Malenje, Jairus Odawa, Jing Zhao, Peng Li, and Yin Han. 2018. "An Extended Car-Following Model with the Consideration of the Illegal Pedestrian Crossing." *Physica A: Statistical Mechanics and Its Applications* 508:650–661. https://doi.org/10.1016/j.physa.2018.05.074.
Markos, Christos, and James J. Q. Yu. 2020. "Unsupervised Deep Learning for GPS-Based Transportation Mode Identification." 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 1–6.
Mullick, Pratik, Cécile Appert-Rolland, William H. Warren, and Julien Pettré. 2022. "Methods of Density Estimation for Pedestrians Moving in Small Groups Without a Spatial Boundary." https://arxiv.org/abs/2212.01338.
Noor, Mohd Halim Mohd, and Ayokunle Olalekan Ige. 2024. "A Survey on State-of-the-Art Deep Learning Applications and Challenges." https://arxiv.org/abs/2403.17561.
Sadeghian, Paria, Arman Golshan, Mia Xiaoyun Zhao, and Johan Håkansson. 2025. "A Deep Semi-Supervised Machine Learning Algorithm for Detecting Transportation Modes Based on Gps Tracking Data." *SSRN Electronic Journal* 52 (4): 1745–1765.
Sadeghian, Paria, Johan Håkansson, Mengjie Han, and Mia Xiaoyun Zhao. 2024b. "Testing Feasibility of Using a Hidden Markov Model on Predicting Human Mobility Based on GPS Tracking Data." *Transportmetrica B: Transport Dynamics* 12 (1): 2336037. https://doi.org/10.1080/21680566.2024.2336037.
Sadeghian, Paria, Johan Håkansson, and Xiaoyun Zhao. 2021. "Review and Evaluation of Methods in Transport Mode Detection Based on GPS Tracking Data." *Journal of Traffic and Transportation Engineering (English Edition)* 8 (4): 467–482. https://doi.org/10.1016/j.jtte.2021.04.004.
Sadeghian, Paria, Xiaoyun Zhao, Arman Golshan, and Johan Håkansson. 2022. "A Stepwise Methodology for Transport Mode Detection in GPS Tracking Data." *Travel Behaviour and Society* 26:159–167. https://doi.org/10.1016/j.tbs.2021.10.004.
Simoncini, Matteo, Leonardo Taccari, Francesco Sambo, Luca Bravi, Samuele Salti, and Alessandro Lori. 2018. "Vehicle Classification from Low-Frequency GPS Data with Recurrent Neural Networks." *Transportation Research Part C: Emerging Technologies* 91:176–191. https://doi.org/10.1016/j.trc.2018.03.024.
Simonyan, Karen, and Andrew Zisserman. 2015. "Very Deep Convolutional Networks for Large-Scale Image Recognition." https://arxiv.org/abs/1409.1556.
Song, Chunfeng, Feng Liu, Yongzhen Huang, Liang Wang, and Tieniu Tan. 2013. "Auto-Encoder Based Data Clustering." Proceedings, Part i, of the 18th Iberoamerican Congress on Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications – Volume 8258, 117–124. CIARP 2013. Berlin, Heidelberg: Springer-Verlag.
Stenneth, Leon, Ouri Wolfson, Philip S. Yu, and Bo Xu. 2011. "Transportation Mode Detection Using Mobile Phones and GIS Information." Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 54–63. *GIS '11*. New York, NY, USA: Association for Computing Machinery.
Sun, Zhanbo, and Xuegang Jeff Ban. 2013. "Vehicle Classification Using GPS Data." *Transportation Research Part C: Emerging Technologies* 37:102–117. https://doi.org/10.1016/j.trc.2013.09.015.
Thilakarathne, Haritha, Aiden Nibali, Zhen He, and Stuart Morgan. 2024. "Group Activity Recognition Using Unreliable Tracked Pose." https://arxiv.org/abs/2401.03262.
Tian, Fei, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. 2014. "Learning Deep Representations for Graph Clustering." *Proceedings of the AAAI Conference on Artificial Intelligence* 28:1. https://doi.org/10.1609/aaai.v28i1.8916.
van der Maaten, L. J. P., and G. E. Hinton. 2008. "Visualizing High-Dimensional Data Using t-SNE." *Journal of Machine Learning Research* 9 (nov): 2579–2605.
Van Gansbeke, Wouter, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. 2020. "SCAN: Learning to Classify Images Without Labels." https://arxiv.org/abs/2005.12320.
Wang, Bo, Jian Zhang, Haiyan Zhang, and Shanglu He. 2025. "AHFS-PTP: An Adaptive Hybrid Feature Selection Method for Pedestrian Trajectory Prediction." *Transportmetrica B: Transport Dynamics* 13 (1): 2490549. https://doi.org/10.1080/21680566.2025.2490549.
Wang, Chun, Shirui Pan, Guodong Long, Xingquan Zhu, and Jing Jiang. 2017b. "MGAE: Marginalized Graph Autoencoder for Graph Clustering." Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 889–898. CIKM '17. New York, NY, USA: Association for Computing Machinery.

Wang, Hao, GaoJun Liu, Jianyong Duan, and Lei Zhang. 2017a. "Detecting Transportation Modes Using Deep Neural Network." *IEICE Transactions on Information and Systems* E100.D (5): 1132–1135. https://doi.org/10.1587/transinf.2016EDL8252.

Wei, Zhaokun, Yaning Gao, Xiaoju Zhang, Xiaojun Li, and Zhifeng Han. 2024. "Adaptive Marine Traffic Behaviour Pattern Recognition Based on Multidimensional Dynamic Time Warping and DBSCAN Algorithm." *Expert Systems with Applications* 238:122229. https://doi.org/10.1016/j.eswa.2023.122229.

Wu, Wenhao, Weiwei Wang, Xixi Jia, and Xiangchu Feng. 2024. "Transformer Autoencoder for k-Means Efficient Clustering." *Engineering Applications of Artificial Intelligence* 133:108612. https://doi.org/10.1016/j.engappai.2024.108612.

Xiao, Guangnian, Zhicai Juan, and Chunqin Zhang. 2015. "Travel Mode Detection Based on GPS Track Data and Bayesian Networks." *Computers, Environment and Urban Systems* 54:14–22. https://doi.org/10.1016/j.compenvurbsys.2015.05.005.

Xie, Junyuan, Ross Girshick, and Ali Farhadi. 2016. "Unsupervised Deep Embedding for Clustering Analysis." https://arxiv.org/abs/1511.06335.

Xin, Xiuying, Ning Jia, Shuai Ling, and Zhengbing He. 2022. "Prediction of Pedestrians' Wait-or-Go Decision Using Trajectory Data Based on Gradient Boosting Decision Tree." *Transportmetrica B: Transport Dynamics* 10 (1): 693–717. https://doi.org/10.1080/21680566.2022.2027294.

Xu, Chaoyang, Renjie Lin, Jinyu Cai, and Shiping Wang. 2022. "Deep Image Clustering by Fusing Contrastive Learning and Neighbor Relation Mining." *Knowledge-Based Systems* 238:107967. https://doi.org/10.1016/j.knosys.2021.107967.

Yang, Jianwei, Devi Parikh, and Dhruv Batra. 2016. "Joint Unsupervised Learning of Deep Representations and Image Clusters." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5147–5156.

Yi, Ruolong, Mingyu Du, Weiguo Song, and Jun Zhang. 2024. "Fast Trajectory Extraction and Pedestrian Dynamics Analysis Using Deep Neural Network." *Physica A: Statistical Mechanics and Its Applications* 638:129611. https://doi.org/10.1016/j.physa.2024.129611.

Zandbergen, Paul A. 2009. "Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning." *Transactions in GIS* 13 (s1): 5–25. https://doi.org/10.1111/j.1467-9671.2009.01152.x.

Zhang, Chi, and Christian Berger. 2023. "Pedestrian Behavior Prediction Using Deep Learning Methods for Urban Scenarios: A Review." *IEEE Transactions on Intelligent Transportation Systems* 24 (10): 10279–10301. https://doi.org/10.1109/TITS.2023.3281393.

Zhao, Nanxuan, Zhirong Wu, Rynson W. H. Lau, and Stephen Lin. 2021. "What Makes Instance Discrimination Good for Transfer Learning?" https://arxiv.org/abs/2006.06606.

Zheng, Yu, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. 2010. "Understanding Transportation Modes Based on GPS Data for Web Applications." *ACM Transactions on the Web* 4 (1): 1–36. https://doi.org/10.1145/1658373.1658374.

Zheng, Yu, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. 2008. "Understanding Mobility Based on GPS Data".

Zheng, Yu, Like Liu, Longhao Wang, and Xing Xie. 2008. "Learning Transportation Mode from Raw Gps Data for Geographic Applications on the Web." Proceedings of the 17th International Conference on World Wide Web, 247–256. WWW'08. New York, NY, USA: Association for Computing Machinery.