VAMBC: A Variational Approach for Mobility Behavior Clustering

Mingxuan Yue ✉[1], Yao-Yi Chiang[1], and Cyrus Shahabi[1]

University of Southern California, Los Angeles, U.S.
{mingxuay, yaoyic, shahabi}@usc.edu

Abstract. Many domains including policymaking, urban design, and geospatial intelligence benefit from understanding people's mobility behaviors (e.g., work commute, shopping), which can be achieved by clustering massive trajectories using the geo-context around the visiting locations (e.g., sequence of vectors, each describing the geographic environment near a visited location). However, existing clustering approaches on sequential data are not effective for clustering these context sequences based on the contexts' transition patterns. They either rely on traditional pre-defined similarities for specific application requirements or utilize a two-phase autoencoder-based deep learning process, which is not robust to training variations. Thus, we propose a variational approach named VAMBC for clustering context sequences that simultaneously learns the self-supervision and cluster assignments in a single phase to infer moving behaviors from context transitions in trajectories. Our experiments show that VAMBC significantly outperforms the state-of-the-art approaches in robustness and accuracy of clustering mobility behaviors in trajectories.

1 Introduction

Mobility behavior (of a trajectory) refers to the travel activity that describes a user's movements regardless of the spatial and temporal coverage of the movements. For example, the work-to-home commute is one such mobility behavior that is a sequence of visiting locations between one's home and working location. Understanding mobility behaviors has enormous values ranging from location recommendations, geo-advertisements to urban planning, public health, transportation policies and economic studies [7, 17, 36]. For example, in target advertising, a gas station owner can infer that the major mobility behavior of trajectories passing through their gas station is work commute, thus deciding to display financial ads on weekdays. For public health during a pandemic, policymakers could make a better decision on what types of businesses should be closed to reduce mobility more effectively in different neighborhoods. For instance, they may decide to close restaurants/pubs in a neighborhood where the majority of mobility behavior is for entertainment but not in a neighborhood where the main mobility behavior is for work commute. The former closure impacts entertainment but the latter will impact work. For economic studies, by labeling work commutes, we can estimate the amount of jobs, or the amount of working people

in the neighborhoods of a city. For transportation studies, when repairing (or constructing a new) bridge or a freeway segment, policymakers could better determine the dates and times for construction by considering the major mobility behavior passing through the area (e.g., weekend closure vs. weekdays).

Inferring the mobility behavior directly from a trajectory is difficult since the raw coordinates do not provide useful information about the surrounding environment of the visited locations. Instead, one promising approach is to first generate a "context sequence" for each trajectory from nearby geographic entities, e.g., Points-Of-Interest (POIs), and then infer mobility behaviors by clustering context sequences from large numbers of trajectories based on the context transitions (e.g., [35]). Here, a context sequence is an ordered list of real-value feature vectors, each describing the "context" of a visited location (e.g., sports, shopping, or dining venues) in a trajectory. Clustering context sequences based on their transition patterns (similar dependencies across different dimensions and positions in the sequences) is challenging. For example, a transition in the context sequence can be: rest (a place surrounded by many residential POIs)→ shopping (a place surrounded by many restaurants, theaters, and malls) → dating (a place surrounded by many scenic POIs). Such transitions are usually driven by the trajectory data and vary from one dataset to another. However, traditional time series clustering approaches are usually based on pre-defined similarity and alignments between sub-sequences and shapes [27, 10, 26]. Other typical sequence clustering approaches only handle discrete variables (e.g., [38, 31, 18]). Autoencoder-based clustering approaches using Recurrent Neural Networks (RNN) can convert sequences of real-value feature vectors into a fixed-length vector for clustering the dynamics in the sequences using a two-phase training process [35, 23]). The first phase learns an initial representation by self-supervision (i.e., reconstruction), and the second phase improves the representation and clustering performance by optimizing a customized clustering objective. However, the first phase's self-supervision objective highly depends on the initial parameters and could lead to a poor initial feature representation (i.e., irrelevant to the clustering objective), which cannot be refined in the second phase to improve the clustering performance. Consequently, clustering accuracy cannot be guaranteed across training variations [25].

This paper presents a novel Variational Approach for Mobility Behavior Clustering (VAMBC) that can robustly handle sequences of context vectors in a single training phase. VAMBC assumes the pre-existence of clusters in the latent space and jointly learns the hidden representation and cluster formation in an end-to-end process. Though variational clustering approaches are recently well developed for image data [20, 11, 29, 13], directly applying them to variable-length sequential data requires an RNN decoder, which is sensitive to small changes in the latent space [4], resulting in poor clustering accuracy and robustness. The main problem originates from having minimum involvement of cluster assignments when constructing the latent space from the input sequences. Hence, the model would generate poor clustering results like having only one or a few large clusters, leaving many clusters empty (called the "empty cluster" problem in the

rest of the paper) or several similar clusters (called "trivial solution"). To address these challenges, VAMBC explicitly constructs two representations: one captures the unique information of a context sequence, and the other one captures the shared information within a cluster. We call the former the *individualized latent embedding* and the latter the *cluster latent embedding*. VAMBC makes the cluster latent embedding available to the cluster members during reconstruction and explicitly uses the embedding in constructing the latent space so that the final latent embedding is aware of the cluster assignments. VAMBC also encourages the cluster assignment to be flexible at early stages and become well separated as the model is trained adequately. Therefore, the model has sufficient involvement of the cluster membership in creating the embeddings and can avoid producing poor clustering results. We compare our approach with many baseline approaches, and the experimental results on real-world data show that VAMBC achieves a better clustering accuracy and robustness than all baselines.

The remainder of the paper is organized as follows. In Section 2, we review the related work. After introducing preliminaries in Section 3, we propose our model VAMBC in Section 4. Section 5 describes our experiments on real-world data and discussion of the ablation analysis followed by our conclusion in Section 6.

2 Related Work

Classical time series clustering approaches are widely developed for sequences indexed by regular time intervals, such as electrocardiogram (ECG) data. Researchers proposed various distance/similarity measurements (e.g., maximize the alignment) for time series and developed the corresponding clustering techniques (e.g., [27, 10, 26]). However, these pre-defined similarities designed for matching or aligning time series do not apply for context sequences on clustering the transition and dependencies. Researchers also study clustering other types of sequences like RNA and protein sequences [38], short text sequences [34, 31], event and action sequences [28, 18]. In these papers, the authors proposed various approaches using topic models, (Hidden) Markov models, graph transformations, etc. for specific problems. However, most of these approaches apply only to sequences of discrete variables and are limited to domain-specific objectives.

Recently deep-learning-based clustering approaches are widely investigated. One major group of these studies are based on various autoencoders and clustering neural network layers [32, 16, 23, 35]. A few approaches in this group are designed for sequential data, such as [23, 35], by using RNN layers for sequence modeling. However, these autoencoder-based approaches usually employ a two-phase training. The two-phase training performance highly relies on the result of its first phase (self-supervision), which does not always align with the clustering objective thus is not robust to training variations.

On the other hand, recent variational approaches can jointly learn the self-supervision and clustering structure from the beginning using various variational assumptions about the hidden space. For example, Dupont [13] proposed to use both a continuous latent space and a discrete latent space and concatenate the

layers for reconstruction. Dilokthanakul et al. [11], Jiang et al. [20] and Rui et al. [29] proposed variational autoencoder models with a Gaussian Mixture assumption in the latent space for capturing the manifolds. However, most of these approaches studied image data, and it would bring more challenges mentioned in Section 1 when applying those to variable-length feature sequences that we study. In contrast, our proposed VAMBC approach explicitly solves the problems and produces robust and accurate clustering on the context sequences.

3 Preprocessing and Problem Definition

In this paper, we consider the mobility behavior clustering problem proposed in [35]. The goal is to cluster trajectories into groups that have similar mobility behaviors. Since raw trajectory data have various temporal and spatial scales and do not have any contextual information, we prepare our input data by applying the preprocessing steps in [35].

Here we briefly summarize the preprocessing procedure that transforms the raw trajectories into **context sequences**. Figure 1 shows the procedure of the preprocessing. First, the Stay Point Detection (SPD) step will extract important stay points (large colored points) from a raw trajectory (small grey points). Then the POIs (small colored points) surrounding each stay point will be grouped, counted, and transformed into context vectors (stacked squares at the bottom). The goal is to cluster these sequences of context vectors into different groups that have a similar transition of visit patterns. Formally, in the remainder of

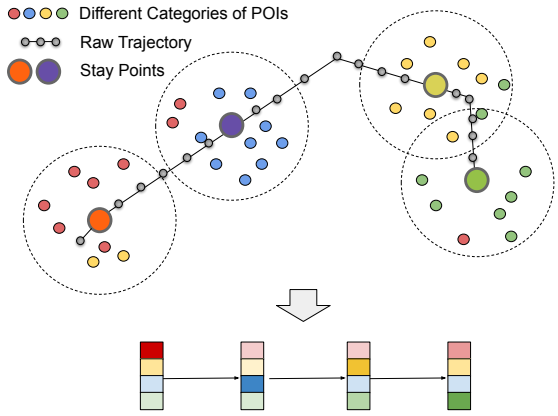


Fig. 1: Preprocess the raw trajectories

this paper, we study the following problem: Given a set of context sequences $X = \{x_i\}$ where $x_i = \{\vec{x}_{i,1}, \vec{x}_{i,2}, \ldots, \vec{x}_{i,L_i}\}$ is a variant-length sequence (i.e., L_i is the length of sequence x_i and is not fixed) of POI context vectors represented by $\vec{x}_{i,l} \in \mathbb{R}^D$ (where D is the number of POI types). Each element of

the vector represents the likelihood of visiting the corresponding POI type (e.g., residence). The goal is to cluster the set of sequences X into K (a predefined hyper-parameter) groups, s.t., within each group the sequences are of similar context transitions.

4 The VAMBC Model

This section introduces our model VAMBC. Specifically, we first introduce our novel idea of decomposing the hidden variables to improve the involvement of cluster hidden variable y. Then we explain the derived training objectives with well-designed layers of the model and discuss their roles. Finally, we describe the network structure of VAMBC followed by a discussion comparing the mechanisms of VAMBC and other variational models for clustering.

4.1 Decomposing Hidden Variables

The goal of the variational clustering model is maximizing the likelihood of the training data $\{x_i\}_{i=1}^N$ while learning a parametric latent embedding z_i representing the hidden information of each input sequence x_i and a latent variable y_i describing the cluster membership (i.e., the mobility behavior). In the following paragraphs, we hide the subscript i for simplicity.

To increase the involvement of cluster assignments (i.e., y) in constructing the latent embedding z, we decompose the hidden variable z into two independent parts: cluster latent information (z^c) and individualized latent information (z^b). Intuitively, each input can be represented using its cluster information (the cluster center) plus its individual (bias) information (the relative position to the cluster center). The cluster latent information is modeled as variable z^c that fully depends on the discrete variable y and, s.t., z^c is a learnable deterministic mapping from y, $z^c = f_c(y)$. Since the cluster latent information summarizes the latent characteristics of a cluster, we explicitly let z^c be the center of the cluster in the latent space that could be shared across x within individual clusters. In other words, for a given cluster k, $\mathbb{E}(Z_k)$ is the expectation of $Z_k = \{z_i | x_i \in \text{cluster } k\}$, s.t., $\mathbb{E}(Z_k) = z^c_{(y=k)} = f_c(y_k)$. We model the individualized latent representation as a continuous variable z^b that describes the bias to the cluster center, s.t., the distribution of z^b centers at 0 ($\mathbb{E}(z^b) = 0$). The overall latent representation z is modeled as the summation of the cluster latent representation and the individualized latent representation, i.e., $z = z^c + z^b$. In this way, the hidden space z still preserves the Gaussian-Mixture structure but can also be decomposed into two embeddings which can be supervised separately. Specifically, the generative process can be described in Equation (1).

$$
\begin{aligned}
& y \sim Cat(1/K), z^c = f(y; W) \\
& z^b \sim \mathcal{N}(0, I) \\
& z = z^c + z^b \\
& x \sim \mathcal{N}(\mu_x(g(z;\theta)), \sigma_x^2(g(z;\theta))I)
\end{aligned}
\tag{1}
$$

In Equation (1), y is a discrete variable that follows a categorical prior (denoted by $Cat(\cdot)$) and K is the predefined number of clusters. f is a deterministic function (implemented by a neural layer parameterized by W) of y that maps each cluster to a vector z^c. z^b is a continuous variable following a Gaussian prior $\mathcal{N}(0, I)$ and represents the individualized embedding. $g(z;\theta)$ denotes a neural network that decodes z to the input space parameterized by θ. $\mu_x(g(z;\theta))$ denotes the mean of the Gaussian likelihood distribution of x condition on z. We set $\sigma_x(g(z;\theta)) = 1$ which reduces the log likelihood to mean squared error following the common practice of VAE. We use $q(y, z|x)$ to approximate the posterior of $p(x, y, z)$. The problem can be reduced to maximizing the log-Evidence Lower Bound (ELBO). We refer the reader to [12, 21] for a detailed explanation of the derivation of ELBO. Thus the objective is minimizing the negative ELBO written as in Equation (2).

$$-\mathcal{L}_{ELBO} = -\mathbb{E}_{y,z\sim q(y,z|x)} \log \frac{p(x, y, z)}{q(y, z|x)} \tag{2}$$

According to the proposed generative process, we can substitute $p(z) = p(z^b)p(z^c)$, $p(x, y, z) = p(y)p(z^b)p(z^c|y)p(x|y, z)$, $q(y, z|x) = q(y|x)q(z^b|x)q(z^c|y)$ into the ELBO. By ignoring $q(z^c|y)$ because it is deterministic, we can break down Equation (2) and rewrite the (negative) ELBO as in Equation (3).

$$\begin{aligned} &- \mathcal{L}_{ELBO} \\ &= -\mathbb{E}_{y,z\sim q(y,z|x)}(\log \frac{p(y)}{q(y|x)} + \log \frac{p(z^b)}{q(z^b|x)} + \log p(x|y, z)) \\ &= D_{KL}(q(y|x)||p(y)) + D_{KL}(q(z^b|x)||p(z^b)) \\ &- \mathbb{E}_{y\sim q(y|x), z^b\sim q(z^b|x), z^c=f(y;W)} \log p(x|y, z^b, z^c) \end{aligned} \tag{3}$$

4.2 Training Objectives and Neural Layers

Now we expand all the Right-Hand-Side (RHS) terms in Equation (3) and formulate the objectives.

The first RHS term $D_{KL}(q(y|x)||p(y))$ describes the KL distance between the posterior estimate $q(y|x)$ and its prior $p(y)$. Since the prior $p(y) \sim Cat(1/K)$ is a categorical distribution, we can expand this term as below.

$$\begin{aligned} D_{KL}(q(y|x)||p(y)) &= \sum_y q(y|x) \log(q(y|x) \cdot K) \\ &= -\text{ Entropy}(y) + log(K), y \sim q(y|x) \end{aligned}$$

The first RHS term turns out to be the negative entropy of y, where $y \sim q(y|x)$ (the constant $log(K)$ could be omitted). Intuitively, a small negative entropy indicates high randomness in $y \sim q(y|x)$, and a large negative entropy value indicates less randomness in $y \sim q(y|x)$. Minimizing the negative entropy could prevent the prediction of cluster probability $q(y|x)$ from being "overly-confident",

i.e., always assigning the input to one cluster aggressively (output 0.99 as the probability of assignment). Therefore, the negative entropy term can prevent the model from having a result of empty clusters.

The posterior probability $q(y|x)$ is estimated using a Softmax activation after the encoder network. To enable sharing of cluster information, we need to sample the discrete variable y from $q(y|x)$, for which we employ the **Gumbel-Softmax layer**. Gumbel-Softmax can sample a pseudo one-hot vector (a real-value vector that is very similar to a one-hot vector) and propagates the parameter gradients backward to the previous Softmax layer [19]. Thus the output will be almost discrete (e.g., (1,0,0,0)) and multiple input sequences could share the same choice of y and hence the same cluster embedding z^c (e.g., $z_i^c = z_j^c = f_c(y_k)$ if x_i and x_j are both assigned to cluster k). In addition, Gumbel-Softmax is different from an Argmax operation on $q(y|x)$, which always chooses the cluster with the largest probability. Specifically, Gumbel-Softmax introduces randomness when sampling y according to the probability $q(y|x)$, so an input x assigned to cluster k_1 could "jump" to a different cluster k_2 in the next round if the likelihood of k_2 is similar to k_1. This avoids x being always assigned to the same cluster that is initially predicted.

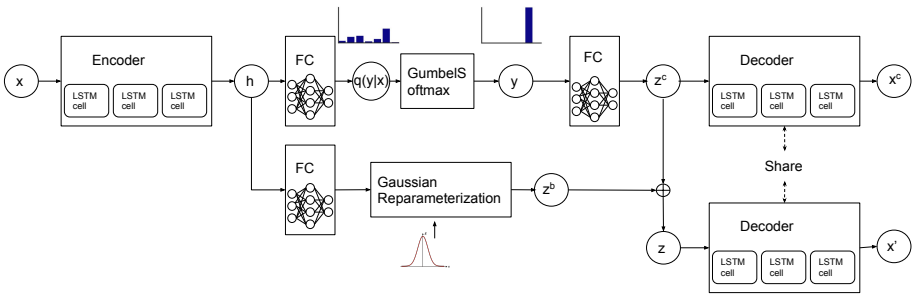


Fig. 2: VAMBC network structure.

The second RHS term $D_{KL}(q(z^b|x)||p(z^b))$ measures the Kullback–Leibler (KL) distance between the posterior $q(z^b|x)$ of the individualized latent embedding z^b and its prior $p(z^b) = N(0, I)$. Here we assume $q(z^b|x)$ is a Gaussian distribution with a learnable mean $\bar{z}_b$ and a constant variance following the constant-variance VAE (CV-VAE) which sacrifices a little capacity for robustness and mitigate the sensitivity in the decoder following [15, 2]. Therefore, the KL term can be rewritten into the following form:

$$D_{KL}(q(z^b|x)||p(z^b)) = ||\bar{z}^b||_2^2 + constant \tag{4}$$

The last RHS term $-\mathbb{E}_{y\sim q(y|x), z^b\sim q(z^b|x), z^c=f(y;W)} \log p(x|y, z^b, z^c)$, is the negative log likelihood of the observation x. Following the common practice of VAE, we can rewrite it to the mean square error (MSE) between the input

data x and the reconstruction x'.

$$\begin{aligned} &- \mathbb{E}_{y\sim q(y|x), z^b\sim q(z^b|x), z^c=f(y;W)} \log p(x|y, z^b, z^c) \\ &= MSE(x, x') \end{aligned}$$

In addition to the terms in $\mathcal{L}_{ELBO}$, we introduce a center loss regularizer $L_{center} = ||x - x^c||_2^2$. Here x^c is a sequence of the same dimension with input x (after padding), and is decoded from the center embedding z^c. The center loss could prevent the model from overly relying on the individualized embedding. Specifically, it generates a center sequence x^c from the cluster embedding z^c and minimizes the distance between x^c with all x assigned to this cluster. Therefore, the cluster embedding z^c is learned to be expressive to generate a sequence x^c similar to the sequences in the cluster. And $\mathcal{L}_{center}$ also improves the compactness and discrimination of clusters in the embedding space.

Finally, we can write the loss function of VAMBC in Equation (5).

$$\begin{aligned} \mathcal{L} =& \text{MSE}(x, x') + ||z^b||_2^2 - entropy(y) + ||x - x^c||_2^2 \\ =& \mathcal{L}_{recon} + \mathcal{L}_{KL} + \mathcal{L}_{NE} + \mathcal{L}_{center} \end{aligned} \tag{5}$$

4.3 Network Design

Based on the loss function, we design our network structure by modeling the probabilities $q(y|x), q(z^b|x), p(x|y, z^b, z^c)$ with neural layers. Figure 2 shows our network structure. On the left side, we use (stacked) LSTM (Long Short-Term Memory, an advanced RNN) layers and Fully Connected (FC) layers with the softmax and nonlinear activation to model the posterior $q(y|x), q(z^b|x)$. Then in the middle of the network, the discrete variable y is sampled through a Gumbel-Softmax layer and mapped to the cluster embedding z^c. The continuous individualized embedding z^b is sampled via the Guassian reparameterization [21]. After obtaining z^c and z^b, the embedding z of input x is computed by adding z^c and z^b in the addition layer denoted by $\bigoplus$. Finally, on the right side of the network, LSTM layers (decoder) are used to decode the hidden embedding z into a reconstructed sequence, x'. The shared decoder also generates the cluster sequence x^c from z_c for computing the center loss.

As explained in Section 4.2, the network is supervised by the RHS terms to balance the self-supervision and clustering structure. The network also employs the Gumbel-Softmax and center loss to prevent early/local convergence and preserve the involvement of cluster memberships.

4.4 Relationship to VAE and Gaussian-Mixture VAE

Here we briefly describe the basic concepts of the Variational AutoEncoder (VAE), its extension to the clustering scenario and the differences between VAMBC and the existing MoG-based VAEs. The goal of VAE is to learn a parametric latent embedding z and a generative model to maximize the marginal likelihood of the training data $\{x_i\}_{i=1}^N$. Its objective in Equation (6) is derived

by approximating the intractable posterior $p_\theta(z|x)$ with $q_\phi(z|x)$ and maximizing the ELBO. In general, the ELBO includes two terms, one for the reconstruction and the other for regularizing the hidden space to a Gaussian prior.

$$\mathcal{L}_{\text{VAE}} = \mathbb{E}_{\hat{p}(x)}[\mathbb{E}_{q_\phi(z|x)}[-log(p_\theta(x|z)] + D_{\text{KL}}(q_\phi(z|x)||p(z))] \tag{6}$$

For clustering purposes, the VAE can be extended to learn the clustering priors [3] driven by the data. The Gaussian prior in the hidden layer of VAE, $p(z) = \mathcal{N}(0, 1)$, in this case, is replaced with the Mixtures of Gaussians (MoG) such as in [20, 11, 29]. The generative process could be described in Equation (7). Here y is a discrete hidden variable representing the cluster assignments, and z is a continuous hidden variable that x is mapped to/conditional on. K is the predefined number of clusters and $Cat(.)$ is the categorical distribution.

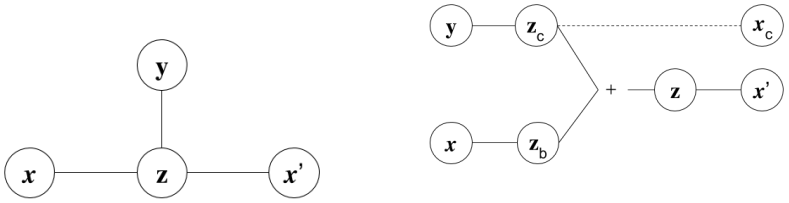
$$\begin{aligned} &y \sim Cat(1/K), z \sim \mathcal{N}(\mu_y, \sigma_y^2 I) \\ &x \sim \mathcal{N}(\mu_x(z), \sigma_x^2(z) I) \end{aligned} \tag{7}$$

The objective in Equation (6) is then rewritten as :

$$\begin{aligned} \mathcal{L}_{\text{VAE}_{GM}} =& \mathbb{E}_{\hat{p}(x)}[\mathbb{E}_{q_\phi(z,y|x)}[-log(p_\theta(x|z)] \\ &+ D_{\text{KL}}(q_\phi(z, y|x)||p(z, y))] \end{aligned} \tag{8}$$

However when we use RNN to encode and decode the variant-length context sequences (e.g, for learning the transition patterns of context vectors), the model above would fall into local optimum and produce undesired clustering results. Specifically, when applying the variational models to sensitive decoders, such as RNNs, for sequence modeling, the model might initially learn to ignore the hidden variable z or y and go after the low hanging fruit, producing a decoder that is easy to optimize [4]. Ignoring y could collapse the joint probability $q_\phi(z, y|x)$ to $q_\phi(z, y = 1|x)$ by assigning all data to one cluster $y = 1$ in the extreme case. This would cause the problem of empty clusters. It is also possible to learn a trivial parameterization for $p(z, c)$ to collapse to $p(z) = N(0, 1)$ by generating the same Gaussian components in the MoG, i.e., $\mu_{y=1} = \mu_{y=2} = \ldots \mu_{y=K}$. Therefore the model will be reduced to a general VAE without the clustering ability. The reason causing the ignorance or the little supervision over y is that the model requires z to include the information that can reconstruct x and to decide the cluster assignment y from their conditional relationship. Especially when the decoder is a sensitive RNN structure, there will not be enough capacity for z to provide enough supervision on y. Instead, z will focus more on the reconstruction thus makes the model inaccurate in clustering the context sequences.

On the contrary, VAMBC can avoid these problems by separating z into two embeddings z_c and z_b, emphasizing on clustering and reconstruction, respectively. As shown in Figure 3, we replace the conditional dependency between z and y with a joint relationship and add self-supervision on y with a center loss with x_c. For the proposed modeling of the latent variables z_c, z_b and y, we carefully derived the objectives and delicately designed networks to fulfill the assumptions and prevent practical problems.



(a) General Gaussian Mixture VAE (b) VAMBC

Fig. 3: Compare the graphic notations of a GMVAE and VAMBC

5 Experiments

In this section, we quantitatively evaluate the clustering performance of VAMBC by comparing it with the state-of-the-art approaches in Section 5.2. We also analyze variants of VAMBC to understand the role of each component in Section 5.3.

5.1 Environment and Experiment Settings

Datasets. Following [35], we utilize the GeoLife dataset [37] and DMCL dataset [8] produced by real human trajectories for the evaluation of our proposed approach. The POI information of the two datasets are from OpenStreetMap (OSM) [14] and the PKU Open Research Data [6], respectively. The experiments are evaluated based on the labeled moving behavior samples of the GeoLife and DMCL datasets reported in [35]. For GeoLife, six labels were provided as the ground-truth classes: "campus activities", "hangouts", "dining activities", "healthcare activities", "working commutes", "studying commutes". Four clusters were labeled in DMCL dataset: "studying commutes", "residential activities", "campus activities", "hangouts".

Baseline approaches. We compare our approaches with the state-of-the-art clustering approaches from four categories:

- For classical time series clustering approaches, we include **KM-DTW** (KMeans with Dynamic Time Warping distance) [27], **KM-GAK** (KMeans with Global Assignment Kernel) [10], **k-Shape** [26] and **DB-LCSS** (DBSCAN with Longest Common Sub-Sequence distance) [24].
- For discrete sequence clustering approaches, we include **SGT** [28] and **MHMM** (Mixed Hidden Marcov Model) [30]. Since they work for discrete sequences, we transform the context sequences into discrete sequences by mapping the real-value vectors to discrete categories via pre-clustering all context vectors;
- For AutoEncoder based deep clustering approaches, we include **DTC** [23], **DETECT** [35], and adapted **IDEC*** [16] and **DCN*** [32] by replacing the encoder and decoder with LSTM layers to work with context sequences.
- For variational deep clustering approaches, we adapted **GMVAE*** [11, 29], **VaDE*** [20] and **JointVAE*** [13] in the same way mentioned above. Here

we also add KL Annealing [4] to GMVAE* to solve its KL vanishing problem. The approaches adapted from image researches are marked with "*" after their names.

Environment and parameters. We implemented our approaches on a computing node with a 36 core Intel i9 Extreme processor, 128 GB RAM and 2 RTX2080Ti / Titan RTX GPUs. We implemented the KM-DBA, KM-GAK and kShape using tslearn[1]. DBSCAN clustering uses ScikitLearn[2] with LCSS distance[3]. We set the common sequence threshold as 0.15 for LCSS, and $\epsilon = 0.03$ and $minPts = 18$ as the neighborhood thresholds in DBSCAN. The proposed model VAMBC was built using Keras [9] with Tensorflow [1]. The discrete sequence clustering approach Mixed-HMM was implemented using the R package seqHMM.[4] The adapted baselines were revised for context sequences based on their public code on Github.[5] [6] [7] Both VAMBC and adapted baselines are using LSTM layers with 128 units in the encoder and decoder. The dimension of hidden variable z was set to 64.

5.2 Quantitative Analysis

Evaluation metrics. We use three clustering metrics that are widely used in the clustering community [23, 32, 16, 20]: Normalized Mutual Information (NMI) [5], Adjusted Rand Index (ARI) [33], and Clustering Accuracy (ACC) [5]. These metrics have different emphasis on evaluating the clustering quality; therefore, we believe a side-by-side comparison could indicate the overall clustering performances of the models. All of the three metrics reach 1 if the clustering result is fully consistent with ground truth. NMI and ACC have minimums of 0, and ARI has -1 as the minimum for the worst clustering result.

Evaluation results. We conducted the experiments ten times following the practice in [20, 16] and reported the clustering performance of the best/worst run and the average metrics with standard deviations (the numbers after $\pm$) (Table 1). DB-LCSS always produces the same results, so we did not include its standard deviation in the table. We believe the average performance is important because in real-world use cases one would not be able to tell which run is better without knowing the ground truth. We compare the performance of VAMBC with the baselines in Table 1. We observe that VAMBC outperforms all the baselines on both the worst run and the average metrics in both datasets. In addition, the standard deviation of the metrics produced by VAMBC is extremely low, i.e., 1/3-1/8, as compared with the baselines for GeoLife. This result indicates VAMBC is robust and can produce accurate results regardless

[1] https://github.com/rtavenar/tslearn
[2] https://scikit-learn.org/stable
[3] https://github.com/maikol-solis/trajectory_distance
[4] https://cran.r-project.org/web/packages/seqHMM/index.html
[5] https://github.com/XifengGuo/IDEC
[6] https://github.com/sarsbug/DCN_keras
[7] https://github.com/slim1017/VaDE

of repeating training. It is interesting to see the adapted DCN got the highest best NMI in GeoLife. But its variance across different experiments is high, and the average NMI is not as good which means it does not guarantee it produces a good result every time. This is because its first-phase training varies from one time to another and does not always produce a good initial representation for clustering.

5.3 Ablation Study

In this section, we demonstrate how different components of VAMBC work under the hood through ablation experiments. Specifically, we create three ablated variants of VAMBC by removing one component. We compare these variants by looking at different measurements during their training processes. In Figure 4, we plot the curves of these measurements versus the training epoch. Figure 4a shows the curves of accuracy. Figure 4b shows the curves of negative entropy and Figure 4c shows the curves of reconstruction errors. We plot the first 300 epochs and discard the rest of the curves, which already converge.

Removal of negative entropy. The negative entropy term in the loss penalizes the model if the clustering prediction is overly confident. In Figure 4a, we can observe that the model without using negative entropy (green line) stays at a low accuracy after some fluctuations and climbs up but fails to converge to high accuracy. The low accuracy period is because this variant model aggressively assigns all data to two or three clusters and does not split more clusters. This can also be observed in Figure 4b that during the same period (from epoch 100 to epoch 150), the negative entropy increases sharply. Although the accuracy increases again after this period (possibly start to split some clusters due to the randomness in the Gumbel-softmax component), it cannot fully escape from the over-confidence problem. In contrast, in Figure 4b, the curve of VAMBC also increases but in a relatively restrained pace. This means the VAMBC becomes confident gradually about the cluster assignment because of the restrain from negative entropy. This way, using negative entropy in the model prevents the model from being dominated by one or a few clusters. We can also observe this phenomenon in Figure 4c. The reconstruction loss of VAMBC drops relatively slowly during the first 100 epochs to let the model working on the cluster embedding and assignments. Therefore, we observe the VAMBC would eventually converge to a much higher accuracy than the ablated variants because VAMBC can escape from the sub-optimums.

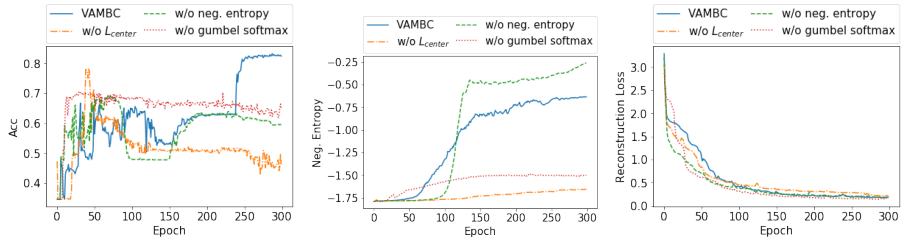
Removal of Gumbel-softmax. The Gumbel-softmax layer enables some randomness in the discrete variable. Such randomness enables an input to "jump" to a similar cluster if the model is not confident enough about their assignments. Without the Gumbel-softmax layer, the model would stay in a sub-optimal assignment and prevent other losses from directing the model to learn a better representation. As we can see in Figure 4a, the curve (red line) without Gumbel-Softmax quickly goes up and stay at a certain accuracy until convergence.

Removal of center loss. The center loss regularizer is very important in preventing the model from ignoring the discrete variable and cluster latent embed-

Table 1: Clustering performance comparison

Dataset	Method	NMI (aver)	NMI (best)	NMI (worst)	ARI (aver)	ARI (best)	ARI (worst)	ACC (aver)	ACC (best)	ACC (worst)
GeoLife	KM-DTW	0.610±0.021	0.645	0.579	0.635±0.019	0.656	0.617	0.742±0.031	0.763	0.655
	KM-GAK	0.591±0.057	0.657	0.507	0.505±0.076	0.573	0.392	0.737±0.033	0.770	0.688
	K-Shape	0.229±0.033	0.272	0.174	0.220±0.046	0.271	0.102	0.522±0.015	0.551	0.495
	DB-LCSS	0.547	0.547	0.547	0.412	0.412	0.412	0.697	0.697	0.697
	SGT	0.419±0.024	0.454	0.371	0.216±0.036	0.277	0.149	0.628±0.029	0.694	0.579
	MHMM	0.530±0.047	0.611	0.486	0.403±0.057	0.495	0.344	0.627±0.017	0.649	0.607
	IDEC*	0.605±0.035	0.673	0.572	0.465±0.097	0.664	0.404	0.67±0.08	0.819	0.596
	DCN*	0.646±0.051	**0.725**	0.594	0.635±0.065	0.693	0.503	0.782±0.061	0.840	0.624
	DTC	0.500±0.027	0.550	0.474	0.483±0.028	0.512	0.451	0.682±0.032	0.737	0.655
	DETECT	0.644±0.037	0.691	0.589	0.646±0.044	0.688	0.582	0.8±0.013	0.822	0.780
	GMVAE*	0.447±0.083	0.598	0.364	0.353±0.074	0.480	0.274	0.530±0.052	0.617	0.479
	VaDE*	0.631±0.053	0.669	0.502	0.603±0.078	0.658	0.440	0.783±0.037	0.822	0.720
	JointVAE*	0.459±0.056	0.556	0.408	0.227±0.123	0.442	0.161	0.519±0.062	0.597	0.473
	VAMBC	**0.697±0.015**	0.699	**0.692**	**0.7±0.019**	**0.719**	**0.682**	**0.825±0.01**	**0.842**	**0.810**
DMCL	KM-DTW	0.366±0.023	0.415	0.355	0.211±0.008	0.229	0.208	0.582±0.009	0.600	0.578
	KM-GAK	0.323±0.019	0.345	0.277	0.161±0.04	0.270	0.120	0.579±0.056	0.733	0.556
	K-Shape	0.409±0.055	0.531	0.344	0.241±0.06	0.396	0.183	0.616±0.08	**0.811**	0.522
	DB-LCSS	0.365	0.365	0.365	0.158	0.158	0.158	0.511	0.511	0.511
	SGT	0.458±0.012	0.466	0.440	0.256±0.009	0.262	0.242	0.763±0.005	0.766	0.755
	HMM	0.326±0.055	0.392	0.208	0.126±0.096	0.339	0.011	0.648±0.064	0.756	0.567
	IDEC*	0.442±0.012	0.448	0.409	0.333±0.006	0.338	0.318	0.776±0.005	0.778	0.767
	DCN*	0.447±0.02	0.479	0.413	0.343±0.014	0.375	0.328	0.781±0.011	0.800	0.767
	DTC	0.427±0.081	0.487	0.222	0.304±0.101	0.368	0.083	0.733±0.089	0.800	0.522
	DETECT	0.486±0.022	0.527	0.448	0.378±0.047	0.398	0.247	0.779±0.063	0.800	0.600
	GMVAE*	0.319±0.063	0.476	0.251	0.127±0.056	0.256	0.082	0.566±0.026	0.622	0.544
	VaDE*	0.456±0.02	0.493	0.446	0.341±0.007	0.355	0.338	0.778±0	0.778	0.778
	JointVAE*	0.12±0.123	0.263	0.000	0.044±0.048	0.104	0.000	0.524±0.016	0.544	0.511
	VAMBC	**0.512±0.02**	**0.527**	**0.475**	**0.384**±0.013	**0.398**	**0.351**	**0.799±0.004**	0.800	**0.789**

ding. As we can see in Figure 4a, the curve (orange line) without the center loss quickly drops to a low accuracy after a spike. This indicates that the model will soon rely less on the cluster embedding to minimize the reconstruction, which is not ideal. In Figure 4b, its negative entropy stays low, which also indicates that the model is reluctant to differentiate clusters and chooses to rely on the individualized embedding only.

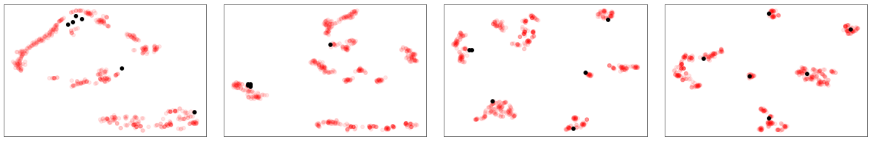


(a) Accuracy v.s. training epochs (b) Negative Entropy v.s. training epochs (c) Recon. Loss v.s. training epochs

Fig. 4: Changes of metrics by variants over their training epochs

5.4 The Training Progress of VAMBC

To understand the change of the latent embedding and the cluster embedding in VAMBC, we visualize the learned embeddings at different epochs in Figure 5 using t-SNE [22]. The red points denote the latent embeddings of the context sequences, and the black points represent the cluster embeddings z^c for each cluster. At the initial stages (epoch =1 and 60), the model learns giant clusters (where many nearby black points locate) that can roughly reconstruct the data. Subsequently, the negative entropy and the reparameterization by Gumbel-Softmax encourage the model to split more clusters and finally (epoch = 400) the clusters are well separated and the cluster embeddings are well-distributed at the centers of each cluster.



(a) epoch = 1 (b) epoch = 60 (c) epoch = 200 (d) epoch = 400

Fig. 5: Visualization of the training progress

6 Conclusion

In this paper, we proposed a novel deep learning framework VAMBC that can accurately and robustly cluster the context sequences of ordered real-value feature vectors based on their transition patterns to infer mobility behaviors. The framework explicitly decomposes the cluster latent representation and individualized latent representation via two reparameterization layers. Such decomposition and the finely designed network enable the model to learn the self-supervision and cluster structure jointly without collapsing to trivial solutions. The results evaluated on real-world data show that the proposed approach is more robust and accurate than a variety of baseline approaches.

References

1. Abadi, M., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. In: OSDI (2016)
2. Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end optimized image compression. In: ICLR 2017
3. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. TPAMI **35**(8), 1798–1828 (2013)
4. Bowman, S., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., Bengio, S.: Generating sentences from a continuous space. In: SIGNLL. pp. 10–21 (2016)
5. Cai, D., He, X., Han, J.: Locally consistent concept factorization for document clustering. IEEE TKDE **23**(6), 902–913 (2010)
6. Center, S.I.: Map poi (point of interest) data (2017), peking University Open Research Data Platform
7. Chang, B., Park, Y., Park, D., Kim, S., Kang, J.: Content-aware hierarchical point-of-interest embedding model for successive poi recommendation. In: IJCAI. pp. 3301–3307 (2018)
8. of Illinois at Chicago, D.U.: Real trajectory data (2006), `https://www.cs.uic.edu/~boxu/mp2p/gps_data.html`
9. Chollet, F., et al.: Keras. https://github.com/fchollet/keras (2015)
10. Cuturi, M.: Fast global alignment kernels. In: ICML. pp. 929–936 (2011)
11. Dilokthanakul, N., Mediano, P.A., Garnelo, M., Lee, M.C., Salimbeni, H., Arulkumaran, K., Shanahan, M.: Deep unsupervised clustering with gaussian mixture variational autoencoders. arXiv preprint arXiv:1611.02648 (2016)
12. Doersch, C.: Tutorial on variational autoencoders. arXiv preprint arXiv:1606.05908 (2016)
13. Dupont, E.: Learning disentangled joint continuous and discrete representations. In: NIPS. pp. 710–720 (2018)
14. Foundation, O.: Openstreetmap data (2018), `http://download.geofabrik.de/north-america.html`
15. Ghosh, P., Sajjadi, M.S., Vergari, A., Black, M., Schölkopf, B.: From variational to deterministic autoencoders. arXiv preprint arXiv:1903.12436 (2019)
16. Guo, X., Gao, L., Liu, X., Yin, J.: Improved deep embedded clustering with local structure preservation. In: IJCAI (2017)
17. He, J., Li, X., Liao, L., Song, D., Cheung, W.K.: Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In: AAAI (2016)

18. Helske, S., Helske, J.: Mixture hidden markov models for sequence data: The seqhmm package in r. arXiv preprint: 1704.00543 (2017)
19. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
20. Jiang, Z., Zheng, Y., Tan, H., Tang, B., Zhou, H.: Variational deep embedding: an unsupervised and generative approach to clustering. In: IJCAI (2017)
21. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
22. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. JMLR (2008)
23. Madiraju, N.S., Sadat, S.M., Fisher, D., Karimabadi, H.: Deep temporal clustering: Fully unsupervised learning of time-domain features. arXiv preprint arXiv:1802.01059 (2018)
24. Morris, B., Trivedi, M.: Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In: CVPR (2009)
25. Mrabah, N., Bouguessa, M., Ksantini, R.: Adversarial deep embedded clustering: on a better trade-off between feature randomness and feature drift. arXiv preprint arXiv:1909.11832 (2019)
26. Paparrizos, J., Gravano, L.: k-shape: Efficient and accurate clustering of time series. In: SIGMOD. pp. 1855–1870 (2015)
27. Petitjean, F., Ketterlin, A., Gançarski, P.: A global averaging method for dynamic time warping, with applications to clustering. Pattern Recognition (2011)
28. Ranjan, C., Ebrahimi, S., Paynabar, K.: Sequence graph transform (sgt): A feature extraction function for sequence data mining (extended version). arXiv preprint arXiv:1608.03533 (2016)
29. Shu, R., Brofos, J., Langlotz, C.: A note on deep variational models for unsupervised clustering (2017)
30. Smyth, P.: Clustering sequences with hidden markov models. In: NIPS. pp. 648–654 (1997)
31. Xu, J., Xu, B., Wang, P., Zheng, S., Tian, G., Zhao, J.: Self-taught convolutional neural networks for short text clustering. Neural Networks **88**, 22–31 (2017)
32. Yang, B., Fu, X., Sidiropoulos, N.D., Hong, M.: Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In: ICML. pp. 3861–3870. JMLR (2017)
33. Yeung, K.Y., Ruzzo, W.L.: Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data. Bioinformatics **17**(9), 763–774 (2001)
34. Yin, J., Wang, J.: A dirichlet multinomial mixture model-based approach for short text clustering. In: SIGKDD. pp. 233–242 (2014)
35. Yue, M., Li, Y., Yang, H., Ahuja, R., Chiang, Y.Y., Shahabi, C.: Detect: Deep trajectory clustering for mobility-behavior analysis. In: IEEE Big Data. pp. 988–997. IEEE (2019)
36. Zheng, Y.: Trajectory data mining: an overview. ACM TIST (2015)
37. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from gps trajectories. In: WWW (2009)
38. Zou, Q., Lin, G., Jiang, X., Liu, X., Zeng, X.: Sequence clustering in bioinformatics: an empirical study. Briefings in bioinformatics **21**(1), 1–10 (2020)