GMDNet: A Graph-Based Mixture Density Network for Estimating Packages' Multimodal Travel Time Distribution

Xiaowei Mao[1,2], Huaiyu Wan[1,3], Haomin Wen[1,2], Fan Wu[2], Jianbin Zheng[2], Yuting Qiang[2], Shengnan Guo[1,3], Lixia Wu[2], Haoyuan Hu[2], Youfang Lin[1,3]*

[1]School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China
[2]Artificial Intelligence Department, Cainiao Network, Hangzhou, China
[3]Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing, China
{maoxiaowei, hywan, wenhaomin, guoshn, yflin}@bjtu.edu.cn;
{wf118503, jabin.zjb, xiushui.qyt, wallace.wulx, haoyuan.huhy}@cainiao.com

Abstract

In the logistics network, accurately estimating packages' Travel Time Distribution (TTD) given the routes greatly benefits both consumers and platforms. Although recent works perform well in predicting an expected time or a time distribution in a road network, they could not be well applied to estimate TTD in logistics networks. Because TTD prediction in the logistics network requires modeling packages' multimodal TTD (MTTD, i.e., there can be more than one likely output with a given input) while leveraging the complex correlations in the logistics network. To this end, this work opens appealing research opportunities in studying MTTD learning conditioned on graph-structure data by investigating packages' travel time distribution in the logistics network. We propose a Graph-based Mixture Density Network, named GMDNet, which takes the benefits of both graph neural network and mixture density network for estimating MTTD conditioned on graph-structure data (i.e., the logistics network). Furthermore, we adopt the Expectation-Maximization (EM) framework in the training process to guarantee local convergence and thus obtain more stable results than gradient descent. Extensive experiments on two real-world datasets demonstrate the superiority of our proposed model.

Introduction

Millions of packages are transported through the logistics network every day in China. In logistics platforms, one of the most crucial tasks is to estimate the Travel Time Distribution (TTD) of a package given the route from its start node to the destination node. Accurately estimating the travel time distribution is of great value for both consumers and platforms. Notifying consumers about travel time distribution can help them schedule the delivery time and alleviate their waiting anxiety. For logistics platforms, estimating the travel time distribution can help the destination node in the logistics network make better scheduling plans in advance.

In a logistics network, packages are sent from a start node and follow a predetermined route consisting of several nodes and edges to a destination node. The travel time is the sum of the stay time in those nodes and the transfer time in those edges. Similar problems have been studied for many years in traffic road networks. However, packages' TTD in the logistics field still lacks effective methods due to the following challenges:

1) **Complex spatial correlations and influence factors in the logistics network**. Firstly, nodes in the logistics networks are naturally connected and correlated with others through package flows, as shown in the left part of Figure 1. A sudden increase of packages at upstream nodes will spread to downstream nodes, resulting in changes in stay time and transfer time in the process towards downstream nodes. Thus the nodes in the logistics networks are spatially correlated. Secondly, a package's stay time in a node is affected by multiple complex factors, such as the current number of packages in the node, e.g., the more packages in the node, the more time a package may wait to leave the node. Previous works like (Ramezani and Geroliminis 2012; Ma et al. 2017; Zhang et al. 2019) estimate TTD on traffic road networks based on the travel times of link pairs, which could not be well applied to handle graph-structure data and complex correlations in logistics networks.

2) **As shown in Figure 1, the packages' travel time distribution is multimodal,** which means there can be more than one likely output with a given route. In Figure 1, package 1 and package 2 are on the same route starting from A to D at 17:00. However, package 1 arrives at the destination D earlier than package 2. Because of the uncertainty in the transfer process, it is hard to determine which truck each package will be assigned to (even though they are on the same route). Some packages (in this case, package 1) may be luckily taken on an almost filled truck to the next node early, while other packages (in this case, package 2) that arrive at the same time have to wait for the next truck. Such characteristics generate typically multimodal travel time distribution for packages in the logistics network. Although extensive works towards the travel time estimation (TTE) in road networks take graph-structure data into account, such as (Fang et al. 2020), (Hong et al. 2020), (Jin et al. 2022), they treat TTE as a regression problem that predicts an average value, thus fail to depict the MTTD of package's travel time.

To address the aforementioned challenges simultane-

*Corresponding author: Youfang Lin

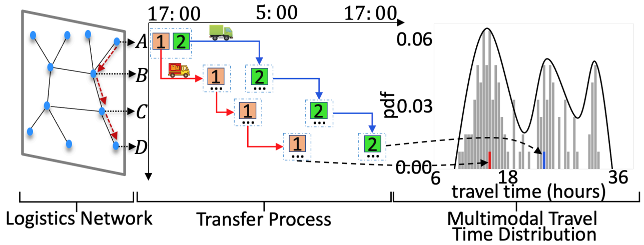

Figure 1: Packages' MTTD given a route in the logistics network (better viewed in color). The travel time distribution of packages sent out at 17:00 from node A to node D in the logistics network (left part) is multimodal (right part). Package 1 (orange) arrive earlier than package 2 (green) due to uncertainty in the transfer process of the route (middle part).

ously, we propose a graph-based mixture density network, named GMDNet, for predicting packages' multimodal travel time distribution (MTTD). Moreover, an Expectation-Maximization (EM) framework is adopted to guarantee local convergence in the training process. Unlike previous works, GMDNet can both handle graph-structure data and approximate the MTTD. Specifically, we design a graph-cooperated route encoding layer to obtain the embedding of the package' route via handling complex spatial correlations and influence factors in the logistics network. Then a mixture density decoding layer is proposed to fully leverage the route embedding to achieve accurate multimodal distribution estimation. Overall, the contribution of this paper is summarized as follows:

- By investigating the multimodal distribution of packages' travel time in the logistics network, this work opens appealing research opportunities in the study of multimodal travel time distribution learning conditioned on graph-structure data and extends the capabilities of graph-based TTE methods whose output is limited to an expected value or a unimodal distribution.
- A graph-based mixture density network, named GMDNet, is proposed for accurately predicting packages' MTTD. GMDNet takes the benefits of both graph neural networks and mixture density networks for estimating MTTD while leveraging complex spatial correlations in the logistics network. Moreover, the Expectation-Maximization (EM) framework is adopted in the training process to guarantee local convergence and thus achieve more stable results than gradient descent.
- Extensive experiments on two real-world logistics datasets demonstrate that GMDNet significantly outperforms other solutions.

Related Work

Travel Time Estimation (TTE). The road segment-based methods (Wang, Zheng, and Xue 2014) predict the travel time on each road segment and then sum up the predicted times of all road segments. Such methods are efficient but fail to incorporate the contextual information of the route, such as delays caused by the intersections in the routes. Route-based methods take a route as a whole and estimate the travel time directly to address the limitations of road segment-based methods. (Wang et al. 2018) and (Wang, Fu, and Ye 2018) employ an RNN to learn the travel time while using various features. To capture the spatial-temporal dependency in the road network, (Fang et al. 2020), (Jin et al. 2022), and (Derrow-Pinion et al. 2021) adopt models based on graph neural networks to achieve more accurate TTE.

Although these graph-based methods can achieve accurate TTE, estimating a single value in some scenarios, such as estimating MTTD in the logistics network, is insufficient. To this end, we extend the capabilities of graph-based TTE methods whose output is restricted to an expected value or a unimodal distribution.

Travel Time Distribution Estimation (TTD). To derive the uncertainty of the expected travel time. Many works estimate the TTD for a given route on road networks. (Hunter et al. 2013) utilizes Gaussian Markov Random Field to compute the travel time distribution of a path. (Wu et al. 2016) proposes a model to learn the mean value of travel time and models the relationship between the variance and mean value to derive the distribution. To relax the assumption that traffic conditions in the same time slot are temporally-invariant, DeepGTT(Li et al. 2019) develops a deep generative model to learn the travel time distribution by conditioning on the real-time traffic. However, it extracts the traffic condition representation based on grid structure data, which cannot well reflect the traffic network's actual topology. To address this limitation, (Song, Zhang, and James 2021) estimates the travel time in a distribution form with deep graph learning and a generative adversarial network.

The above-mentioned TTD methods did not explore estimating MTTD. However, travel time distribution in some real-world scenarios can be multimodal such as packages' TTD in the logistics network, leading to research on learning multimodal travel time distributions. (Ma et al. 2017) utilizes the Gaussian Mixture model and Markov chain model to estimate the MTTD of routes in the road network. (Zhang et al. 2019) estimates MTTD in the road network within the framework of generative adversarial networks. These approaches estimate MTTD by modeling the travel time of link pairs without considering the topology of the whole network. Thus, they are hard to handle complex spatial correlations and influence factors in the logistics network. To address this limitation, we propose a graph-based mixture density network for accurately predicting MTTD while leveraging complex spatial correlations in the logistics network.

Preliminaries

In this section, related definitions are provided, and the packages' MTTD estimation problem is formalized.

Definition 1. ***Logistics Network.*** The logistics network is intrinsically a directed graph, which is defined as $G = (\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{A}, \mathbf{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$, and each node corresponds to a logistics entity (e.g., store, transfer center). $\mathcal{E} = \{e_{ij} \mid v_i, v_j \in \mathcal{V}\}$ is the set of edges. $\mathbf{X} \in \mathbb{R}^{N \times d_v}$ and $\mathbf{E} \in \mathbb{R}^{N \times N \times d_e}$ are the node and edge features respectively, where d_v and d_e are the node feature dimension and edge feature dimension. $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the adjacent matrix.

Definition 2. ***Route.*** A route in the logistics network is denoted as a tuple $R = (\mathbf{r}, \mathbf{f})$. $\mathbf{r} = [e_1, \ldots, e_l]$ is the sequence of edges in route R, where $e_1, \ldots, e_l \in \mathcal{E}$. $\mathbf{f} \in \mathbb{R}^{d_f}$ are the route related features in $\mathbf{r}$, where d_f is the dimension of the features.

Problem Statement. Given the logistics network G_t and the route R_t of a package at the request time t, we aim to predict the package's travel time distribution $P(y_s|s)$, where $s = (G_t, R_t)$, and y_s is the actual travel time (i.e., the label) given s.

Proposed GMDNet Model

The main idea of our proposed model is to learn the packages' MTTD conditioned on the input route and the logistics network.

Overall Idea: MLE Hypothesis

To equip the model with multimodal output capabilities, we leverage the benefits of the mixture density network (BISHOP 1994) to learn the conditional distribution $P(y_s|s)$. Specifically, we combine K mixture components with mixture weights $\pi_s \in [0,1]^K$ that sum to 1, to produce the output distribution. And the mixture components/weights are estimated by solving maximum likelihood estimation (MLE). Given a hypotheses space $\mathcal{H}$, we seek the optimal MLE hypothesis h^* that can maximize the likelihood $\prod_{s\in\mathcal{D}} P(y_s|s)$, formally:

$$\begin{aligned} h^* = \underset{h\in\mathcal{H}}{\operatorname{argmax}}\, L(h|\mathcal{D}) &= \underset{h\in\mathcal{H}}{\operatorname{argmax}} \prod_{s\in\mathcal{D}} P(y_s|s) \\ &= \underset{h\in\mathcal{H}}{\operatorname{argmax}} \prod_{s\in\mathcal{D}} \textstyle\sum_{k=1}^{K} P(y_s|\pi_s^k, s) P(\pi_s^k|s). \end{aligned} \tag{1}$$

The latent variable π_s is introduced through marginalization whose k-th component is π_s^k. To model the distribution $P(y_s|s)$ in Equation 1, we first implement a graph-cooperated route encoding layer to obtain the route embedding taken s as input. Then the mixture weights $P(\pi_s^k|s)$ and the mixture components $P(y_s|\pi_s^k, s)(k = 1, ..., K)$ are produced by the mixture density decoding layer based on the route embedding. At last, $P(y_s|s)$ is produced by combining the mixture weights and components. We sketch the overall architecture in Figure 2.

Input Layer

At the request time t, an input contains the logistics network G_t and the route R_t. The feature construction for them is elaborated in this section.

Network Features. Let a_{ij} be the (i,j)-th entry of the adjacent matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$. If packages can be transferred from node i to node j ($i \neq j$), a_{ij} equals 1; if $i = j$, a_{ij} equals -1. Otherwise, a_{ij} equals 0.

Given a node $v_i \in \mathcal{V}$, the node feature vector $\mathbf{x}_i$ is formulated as: $\mathbf{x}_i = (x_i^{in}, x_i^{out}, weekday, hour)$. x_i^{in} is the number of incoming packages from all upstream nodes, and x_i^{out} is the number of packages sent out to all downstream nodes. $weekday$ and $hour$ are the day-of-week and hour-of-day, respectively.

Given an edge $(i,j) \in \mathcal{E}$ at time t, the edge feature vector is: $\mathbf{e}_{ij} = (e_{ij}^{in}, e_{ij}^{out}, e_{ij}^{stay}, e_{ij}^{trans}, a_{ij}, weekday, hour)$. e_{ij}^{in} / e_{ij}^{out} is the number of packages brought in/sent out from node i to node j, respectively. e_{ij}^{stay} is the average stay time in node i for packages from node i to node j. e_{ij}^{trans} is the average transfer time from node i to node j . And a_{ij} is the proximity between node i and j.

Route Features. The edge sequence in route R_t is denoted as $\mathbf{r}$. The route related features $\mathbf{f}$ in route R_t are formulated as: $\mathbf{f} = (f^{out}, f^{trans}, weekday, hour)$. f^{out} is the number of packages sent out from the start node to the destination node. f^{trans} is the average travel time from the start node to the destination node.

Among the features mentioned above, x_i^{in}, x_i^{out}, e_{ij}^{in}, e_{ij}^{out}, e_{ij}^{stay}, e_{ij}^{trans}, f^{out}, and f^{trans} are calculated in a given time window (one day in this paper) before t.

Graph-Cooperated Route Encoding Layer

We design a graph-cooperated route encoding layer that models the spatial dependency in the logistics network and integrates mutual information among edges in the route to generate a comprehensive representation of the route.

Spatial Dependency Modeling. Given the d_h-dimensional node and edge embeddings obtained by the node and edge features through linear transformations as input, the spatial dependency among nodes and edges in the logistics network is modeled through a graph neural network with L layers, each of which updates the node and edge embeddings by modeling their interactions.

Let $\mathbf{u}_i^l$ denote the embedding associated with node i, and $\mathbf{h}_{ij}^l$ denote the embedding associated with edge (i,j) at the l-th layer. In a logistics network, the package flows are directional and associated with both nodes and edges, so we jointly update the node embedding and edge embedding at layer $l+1$ by the following process:

$$\mathbf{u}_i^{l+1} = f(\mathbf{u}_i^l, \mathrm{Agg}\{\mathbf{u}_i^l, \mathbf{h}_{ij}^l : j \in \mathcal{N}_i\}), \tag{2}$$

$$\mathbf{h}_{ij}^{l+1} = g(\mathbf{h}_{ij}^l, \mathrm{Agg}\{\mathbf{h}_{ij}^l, \mathbf{u}_i^l, \mathbf{u}_j^l\}), \tag{3}$$

where $\mathcal{N}_i$ denotes the set of neighbors centered at node i, $\mathrm{Agg}(\cdot)$ is the aggregation function. The updating function f, g can be further specified by non-linear transformations:

$$\mathbf{u}_i^{l+1} = \mathbf{u}_i^l + \sigma_1(\mathrm{BN}(\mathbf{W}_1^l \mathbf{u}_i^l + \textstyle\sum_{j\in\mathcal{N}_i} \sigma_2(\mathbf{h}_{ij}^l) \odot \mathbf{W}_2^l \mathbf{u}_j^l)), \tag{4}$$

$$\mathbf{h}_{ij}^{l+1} = \mathbf{h}_{ij}^l + \sigma_1(\mathrm{BN}(\mathbf{W}_3^l \mathbf{h}_{ij}^l + \mathbf{W}_4^l \mathbf{u}_i^l + \mathbf{W}_5^l \mathbf{u}_j^l)), \tag{5}$$

where $\mathbf{W}_i^l \in \mathbb{R}^{d_h \times d_h}$ $(i = 1, \ldots, 5)$ are trainable parameters, σ_1 is ReLU activation function, and σ_2 is the sigmoid function. $\mathrm{BN}(\cdot)$ represents batch normalization. After the computation of the graph neural network with L layers, we get the output of spatial-correlation encoding: $\mathbf{U}_s$ and $\mathbf{H}_s$, which are the embeddings of nodes and edges, respectively.

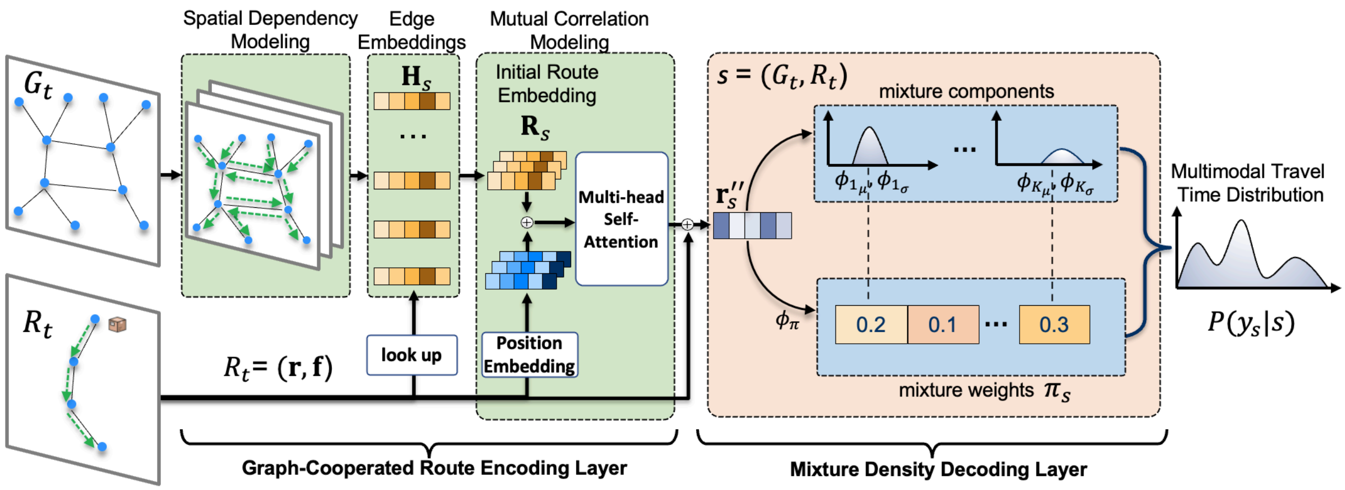

Figure 2: The architecture of GMDNet. Firstly, The graph-cooperated route encoding layer produces the route embedding $\mathbf{r}_s^{''}$ via capturing the spatial dependency in the logistics network and mutual correlations among edges in the route. Secondly, the mixture density output layer outputs the mixture weights π_s and the mixture components $P(y_s|\pi_s^k, s)(k = 1, ..., K)$ based on $\mathbf{r}_s^{''}$. Finally, $P(y_s|s)$ can be obtained by combing the mixture weights and mixture components.

Mutual Correlation Modeling. We generate a comprehensive embedding for the route by integrating the mutual correlations among edges in the route.

The initial route embedding (denoted by $\mathbf{R}_s \in \mathbb{R}^{l \times d_h}$) is obtained by stacking the edge embeddings (from $\mathbf{H}_s$) in that route. Secondly, we adopt the multi-head self-attention mechanism to integrate the mutual information among edges in the route and obtain the updated route embedding $\mathbf{r}_s^{'}$. The attention function is formulated as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}})\mathbf{V}, \quad (6)$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ (queries, keys and values) are constructed by the route embedding $\mathbf{R}_s$, and d is the dimension of $\mathbf{K}$. Here we adopt the multi-head self-attention to jointly attend to information from different representation subspaces. Formally,

$$\text{MHSelfAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \oplus(head_1, \ldots, head_h)\mathbf{W}^O, \quad (7)$$

$$head_j = \text{Attention}(\mathbf{Q}\mathbf{W}_j^Q, \mathbf{K}\mathbf{W}_j^K, \mathbf{V}\mathbf{W}_j^V), \quad (8)$$

where h is the number of attention heads. $\mathbf{W}_j^Q$, $\mathbf{W}_j^K$, and $\mathbf{W}_j^V$ are projection matrices applied on $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$; $\mathbf{W}^O$ is the final output projection matrix.

In the route encoding layer, we equip the initial route embedding $\mathbf{R}_s$ with fixed position embedding to incorporate the order bias into the model. We concatenate the edges' embeddings with the position embedding, which helps the model recognize the relative order among edges in a route.

Lastly, we reshape the updated embedding of the route $\mathbf{r}_s^{'}$ to $\mathbb{R}^{l * d_h}$, and concatenate it with route related feature $\mathbf{f} \in \mathbb{R}^{d_f}$ to obtain the final route embedding $\mathbf{r}_s^{''} \in \mathbb{R}^{l * d_h + d_f}$ based on the input $s = (R_t, G_t)$ at the request time t.

Mixture Density Decoding Layer

We develop a mixture density decoding layer to model the mixture weights π_s and the parameters of mixture components $P(y_s|\pi_s^k, s)$ $(k = 1, ..., K)$ based on the route embedding $\mathbf{r}_s^{''}$. The multimodal travel time distribution $P(y_s|s)$ can be obtained by combing the mixture weights and mixture components. More formally, the process of modeling $P(y_s|s)$ can be represented by the Bayesian network as shown in Figure 3.

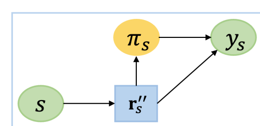

Figure 3: The process of modeling $P(y_s|s)$ can be represented as a Bayesian network where the round green (yellow) nodes are observed (unobserved) random variables, and the blue square represents deterministic route embedding $\mathbf{r}_s^{''}$.

We model the mixture weights $\pi_s \in [0, 1]^K$ as a categorical distribution with K possible states. And π_s satisfies $\sum_{k=1}^{K} P(\pi_s^k) = 1$. The posterior distribution $P(\pi_s|s)$ can be computed by the deterministic route embedding $\mathbf{r}_s^{''}$:

$$P(\pi_s|s) = \phi_\pi(\mathbf{r}_s^{''}) = \sigma(f_\pi(\mathbf{r}_s^{''})), \quad (9)$$

where ϕ_π is a nonlinear transformation composed of a linear model f_π and a softmax function σ.

As for the mixture components, we assume that the conditional distributions of the mixture components come from the family of Gaussian distributions. Because the combination of Gaussian distributions is proved to be capable of approximating any given density function to arbitrary accuracy (McLachlan and Basford 1988), moreover, Gaussian distribution is a common practice for representing a variable without prior knowledge. In this way, the conditional density function of the mixture components can be formulated as:

$$p(y_s|\pi_s^k, s) = \frac{1}{(2\pi)^{1/2}\sigma_k(s)}\exp\{-\frac{(y_s - \mu_k(s))^2}{2\sigma_k^2(s)}\}. \quad (10)$$

Then networks ϕ_{k_μ}, ϕ_{k_σ} are adopted to output the parameters of the conditional density function based on the route embedding:

$$\mu_k(s) = \phi_{k_\mu}(\mathbf{r}_s^{''}) = f_{k_\mu}(\mathbf{r}_s^{''}), \quad (11)$$

$$\sigma_k(s) = \phi_{k_\sigma}(\mathbf{r}_s^{''}) = \exp(f_{k_\sigma}(\mathbf{r}_s^{''})), \quad (12)$$

where f_{k_μ}, f_{k_σ} are linear models. Note that the route's embedding $\mathbf{r}_s^{''}$ is shared in the mixture density output layer transformations instead of requiring a new encoder for each mixture component (Masoudnia and Ebrahimpour 2014). This form of weight sharing reduces the number of parameters and helps the model encode more useful information into the route embedding.

Model Training via EM Framework

Training. Recall that the logarithm of the likelihood function in the MLE estimation is taken as:

$$\log L(h|\mathcal{D}) = \sum_{s\in\mathcal{D}} \log\textstyle\sum_{k=1}^{K} P(y_s|\pi_s^k, s)P(\pi_s^k|s). \quad (13)$$

To train the model (i.e., maximize the above log-likelihood), an intuitive way is to update the parameters via gradient descent. However, due to the presence of the latent variable in the log-likelihood function, it is more effective to solve the MLE estimation via Expectation-Maximization (EM) framework (Dempster, Laird, and Rubin 1977). Since compared with gradient descent, the EM framework can theoretically guarantee local convergence (Errica, Bacciu, and Micheli 2021; Bilmes et al. 1998). Thus the results are more stable, as we will also prove this in the experiment.

For the convenience of solving the MLE estimation based on the EM framework, we introduce the indicator variable $z_s^k \in \mathcal{Z}$ ($z_s^k = 1$ when the input s is in latent state k, where $k \in \{1, \ldots, K\}$). The introduction of z_s^k provides us with the opportunity to write the lower bound of the log-likelihood $\log L(h|\mathcal{D})$) by Jensen's inequality (Chandler 1987), which can be formulated as:

$$\log L(h|\mathcal{D}) \geq \mathbb{E}_{\mathcal{Z}|\mathcal{D}}[\log L(h|\mathcal{D})]. \quad (14)$$

Maximizing the lower bound can thereby indirectly maximize the log-likelihood. The lower bound can be further derived as:

$$\mathbb{E}_{\mathcal{Z}|\mathcal{D}}[\log L(h|\mathcal{D})] = \sum_{s\in\mathcal{D}}\sum_{k=1}^{K} E[z_s^k|\mathcal{D}]P(y_s|\pi_s^k, s)P(\pi_s^k|s). \quad (15)$$

Based on the lower bound of the log-likelihood, we can perform the E-step of the EM algorithm by computing the posterior probability of the indicator variables:

$$E[z_s^k|\mathcal{D}] = P(z_s^k = 1|s) = \frac{1}{Q}P(y_s|\pi_s^k, s)P(\pi_s^k|s), \quad (16)$$

where Q is the normalization term and derived by marginalization:

$$Q = \textstyle\sum_{k=1}^{K} P(y_s|\pi_s^k, s)P(\pi_s^k|s). \quad (17)$$

The M-step is implemented by gradient ascent to maximize Equation 15. This implementation is known as Generalized EM (GEM) (Dempster, Laird, and Rubin 1977). GEM guarantees local convergence if each optimization step improves Equation 15. If no prior information is introduced to the distribution of $P(\pi_s|s)$, the posterior probability mass may collapse onto a single state, thus resulting in the output of a unimodal distribution. To address this problem, we apply an optional Dirichlet regularizer λ with hyper-parameter $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$ on the distribution $P(\pi_s|s)$ to the original objective function. Note that $\boldsymbol{\alpha} = \mathbf{1}^K$ corresponds to a uniform prior, which means no regularization. Finally, the objective function to be maximized is formulated as follows.

$$\mathcal{L}_{\text{obj}} = \mathbb{E}_{\mathcal{Z}|\mathcal{D}}[\log L(h|\mathcal{D})] + \sum_{s\in\mathcal{D}} \log\lambda(\pi_s|\boldsymbol{\alpha}). \quad (18)$$

Note that the local convergence can also be guaranteed to maximize Equation 18 after adding the regularization, if Equation 15 increases at each step. To conclude, in the training process, we first implement the E-step by Equation 16 and then implement the M-step using gradient descent to maximize Equation 18. This process is iterated until convergence. Finally, we obtain the optimal MLE hypothesis h^*.

Prediction. The whole process of estimating a package's travel time is illustrated in Algorithm 1.

Algorithm 1: Prediction by GMDNet.

Input: The graph and route of a package $s = (G_t, R_t)$ at t.
Output: Travel time distribution $P(y_s|s)$.

```
// Graph-Cooperated Route Encoding Layer
for l = 1, ..., L do
    Update node embeddings by Equation 4;
    Update edge embeddings by Equation 5;
end for
Obtain r_s'' according to Equation 6 - Equation 8;
// Mixture Density Decoding Layer
Output mixture weights P(π_s|s) by Equation 9;
Output parameters of mixture components μ_k(s), σ_k(s)
    by Equation 11 and Equation 12;
P(y_s|s) = Σ_{k=1}^{K} P(π_s^k|s) × N(μ_k(s), σ_k(s));
return P(y_s|s);
```

Experiments

In order to evaluate the effectiveness of our proposed method, we carried out comparative experiments and component analysis on two real-world logistics datasets.

Dataset. The experiment is conducted on two real-world logistics datasets collected from two different regions by Cainiao[1], one of the largest logistics companies in China, handling millions of packages per day. Both datasets contain the travel information of packages in the logistics network from Feb. 06, 2022, to Mar. 08, 2022. Detailed statistics of the dataset is shown in Table 1.

Dataset	$\mathcal{D}_1$	$\mathcal{D}_2$
Average travel time (hours)	15.8	16.1
Average number of edges in a route	3	3
#training samples	358,822	492,831
#validation samples	59,529	77,572
#test samples	54,257	76,003

Table 1: Statistics of the Datasets.

Baselines. There are few related public achievements for directly comparison. So we choose both TTE and TTD methods as baselines for a comprehensive comparison:

- *Historical Average (HA)*. The prediction travel time is given by summing the average travel time of each edge in the route.
- *LightGBM (Ke et al. 2017)*. A popular traditional machine learning algorithm that predicts the expected travel time of the package.
- *Wide-Deep-Recurrent (Wang, Fu, and Ye 2018) (WDR)*. WDR is a deep neural network that can model different types of features by combing the wide, deep, and recurrent network components.
- *Mixture Density Network (BISHOP 1994) (MDN)*. MDN is proposed to approximate arbitrarily complex conditional target distribution, it can be used to model the multimodal distributions of travel time.
- *Kernel Density Estimation (Weglarczyk 2018) (KDE)*. KDE is a nonparametric method to estimate the probability density function of the travel time based on kernels as weights.
- *GCGTTE (Song, Zhang, and James 2021)*. GCGTTE is proposed to estimate the TTD with graph deep learning and generative adversarial network (GAN).
- *GMDNet-GD*. GMDNet trained via gradient descent.

Evaluation Metrics. We evaluate the performance of different models by the following metrics: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Log-likelihood ($\log\mathcal{L}$) (Shao 2003), and Continuous Ranked Probability Scores (CRPS) (Matheson and Winkler 1976). MAE and MAPE measure the degree to which the prediction deviates from the label. $\log\mathcal{L}$ and CRPS measure how good predictions are in matching observed outcomes. Note that we use the weighted mean of the output distribution to calculate MAE/MAPE for models that output a distribution. Larger $\log\mathcal{L}$ and smaller MAE/MAPE/CRPS means better performance in the experiment.

[1]https://www.cainiao.com/

Settings. We set the bandwidth of KDE with Gaussian kernel to 1. We implement all the deep models by Pytorch. The batch size of each epoch is 32, and the learning rate of the Adam optimizer is 0.001. For all deep models, hyperparameters are tuned by using the validation set, and test results are reported at the best validation epoch. To test the stability of the deep models, we ran over 10 times for each model and recorded each metric's mean and standard deviation. For deep baseline models, we searched parameters according to their paper and our prediction task. For the parameter search space of GMDNet, the number of hidden units is searched from $\{16, 32, 64, 128, 256\}$, the embedding dimension of categorical features is searched from $\{8, 16, 32\}$, the number of attention heads is searched from $\{2, 4, 8\}$, the number of GNN layers is searched from $\{1, 2, 3\}$, K is searched from $\{1, 2, 3, 4, 5\}$, and $\boldsymbol{\alpha}$ is searched from $\{\mathbf{1}^K, \mathbf{1.05}^K\}$. The code is available at https://github.com/maoxiaowei97/GMDNet.

Experimental Results. Table 2 shows the comparison of different approaches. HA, LightGBM, and WDR cannot output the travel time distribution, so we do not calculate $\log\mathcal{L}$ or CRPS for those methods. The results show that GMDNet outperforms other methods on both datasets.

HA is an intuitive but less effective approach. WDR jointly trains wide linear models, deep neural networks, and recurrent networks to model various input features. LightGBM is efficient for implementation and achieves reasonable results. However, the complex spatial correlations in the logistics network could not be well modeled by LightGBM or WDR, which can be essential for accurately estimating the travel time. Additionally, LightGBM and WDR can only output an expected value of the travel time, thus failing to handle the MTTD given an input route and graph.

MDN and KDE can output the MTTD based on the input route. However, MDN and KDE perform unsatisfactorily since they could not handle graph-structure data, thus failing to learn the spatial correlations in the logistics network and mutual correlations in the package's route.

GCGTTE fails to achieve ideal performance for estimating TTD in the logistics network. Although it uses a graph neural network to learn the spatial correlations among nodes in the logistics network, it fails to model the directional package flows on edges and mutual correlations in the packages' route. Moreover, it is difficult to control the synchronization of the two adversarial networks, so the training process can be unstable (Gonog and Zhou 2019).

Our GMDNet presents the most effective results. On the one hand, we design a graph-cooperated route encoding layer to model spatial dependency in the logistics network and mutual correlation in the route. On the other hand, the mixture density decoding layer enables outputting multimodal distribution, so the ground truth distribution can be better approximated while breaking the limitation of predicting an expected value or a unimodal distribution. Furthermore, the standard deviation is lower when training via the EM framework than gradient descent. Since the local convergence can be guaranteed to adopt the EM framework in the training process, more stable results can be obtained than gradient descent.

Method	$\mathcal{D}_1$				$\mathcal{D}_2$			
	MAE(h)	MAPE(%)	$\log\mathcal{L}$	CRPS	MAE(h)	MAPE(%)	$\log\mathcal{L}$	CRPS
HA	3.75	27.2	-	-	3.57	25.5	-	-
LightGBM	2.06	13.6	-	-	1.81	12.1	-	-
WDR	2.10 ± 0.07	13.9 ± 0.56	-	-	1.85 ± 0.06	12.6 ± 0.83	-	-
MDN	2.15 ± 0.03	14.2 ± 0.20	-2.19 ± 0.08	1.64 ± 0.01	1.90 ± 0.04	13.3 ± 0.32	-1.61 ± 0.06	1.37 ± 0.01
KDE	2.04	12.6	-2.02	1.61	1.83	11.7	-1.41	1.35
GCGTTE	2.56 ± 0.13	16.2 ± 0.76	-4.27 ± 0.98	1.95 ± 0.18	2.47 ± 0.14	15.7 ± 0.81	-4.11 ± 0.85	1.65 ± 0.14
GMDNet-GD	1.92 ± 0.08	11.5 ± 0.75	-0.81 ± 0.16	1.43 ± 0.04	1.66 ± 0.08	11.0 ± 0.70	-0.99 ± 0.12	1.23 ± 0.03
GMDNet	$\mathbf{1.89 \pm 0.03}$	$\mathbf{11.3 \pm 0.28}$	$\mathbf{-0.73 \pm 0.02}$	$\mathbf{1.39 \pm 0.01}$	$\mathbf{1.63 \pm 0.02}$	$\mathbf{10.8 \pm 0.37}$	$\mathbf{-0.94 \pm 0.04}$	$\mathbf{1.21 \pm 0.01}$

Table 2: Experiment Results.

Component Analysis. To further analyze the components of GMDNet, we design three variants of GMDNet and compare them on the $\mathcal{D}_1$ dataset. Figure 4 illustrates the results.

Firstly, we replace the graph-cooperated route encoding layer of GMDNet with a Multi-Layer Perceptron (wo-GR). The performance drops at all evaluation metrics. This demonstrates that effectively handling complex spatial dependency and mutual correlation in the logistics network is essential for accurately estimating packages' MTTD.

Secondly, removing the mutual correlation modeling block (wo-R) also brings a decrease in the model's performance. The results demonstrate that producing the route embedding by integrating the mutual information among edges in the route contributes to performance improvement.

Lastly, we set the number of mixture components of GMDNet to 1, resulting in a model (wo-M) without the ability to output multimodal TTD. We report MAE and MAPE using the weighted mean of the mixture components in GMDNet, so the difference between MAE and MAPE is not significant. However, the lower $-\log\mathcal{L}$ and CRPS indicate that modeling the MTTD can help better approximate the ground truth distribution.

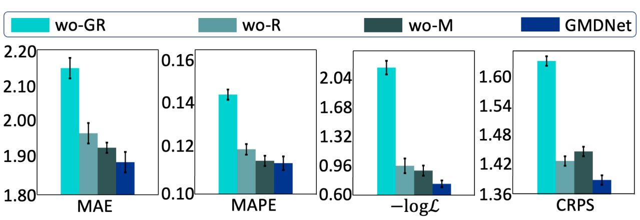

Figure 4: Ablation study.

Case Study. To analyze the performance of GMDNet more intuitively, we provide a case study shown in Figure 5. We plot the output distributions of wo-GR, wo-M, GCGTTE, KDE, GMDNet, and packages' actual travel time distribution for a given input route. Firstly, wo-GR and KDE could not well approximate the packages' travel time distribution since they are unaware of the graph-structure information and unable to model complex spatial dependencies in the logistics network. Secondly, wo-M models the spatial dependency and mutual correlation via a graph-cooperated route encoding layer. It produces a unimodal distribution that accounts for majorities of the actual distribution. However, wo-M cannot produce a multimodal distribution that well approximates the actual distribution. Thirdly, GCGTTE fails to approximate the ground truth ideally. On the one hand, it fails to model the spatial dependency by considering the directional package flow and mutual correlation among edges in the route. On the other hand, the training of GAN is difficult and often unstable (Gui et al. 2021). In contrast, GMDNet can learn the complex spatial correlations and influence factors in the logistics network and produce a multimodal distribution that can better approximate the ground truth MTTD.

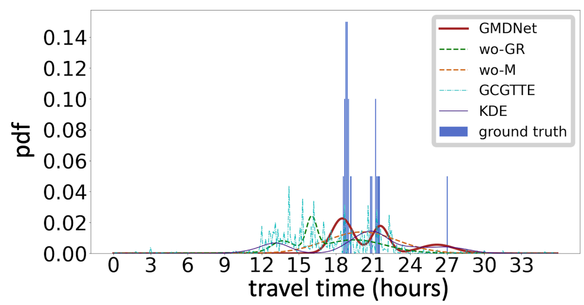

Figure 5: Case study. GMDNet can better approximate the ground truth MTTD.

Conclusion

This paper opens appealing research opportunities in the study of MTTD learning conditioned on graph-structure data, by investigating the package's MTTD in the logistics network. And a graph-based mixture density network, named GMDNet, is proposed for accurately predicting the package's travel time distribution. GMDNet is equipped with a graph-cooperated route encoding layer to model complex spatial dependency in the logistics network and mutual correlation among edges in the route. Then a mixture density decoding layer is leveraged to output multimodal distribution to extend the capabilities of graph-based TTE methods whose output is limited to an expected value or a unimodal distribution. Moreover, the EM framework is adopted in the training process to guarantee local convergence and thus achieve more stable results than gradient descent. Experiments conducted on two real-world datasets demonstrate the effectiveness of GMDNet.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62272033) and CCF-Alibaba Innovative Research Fund For Young Scholars.

References

Bilmes, J. A.; et al. 1998. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *International computer science institute*, 4(510): 126.

BISHOP, C. 1994. Mixture density networks. *Technical Report*.

Chandler, D. 1987. Introduction to modern statistical. *Mechanics. Oxford University Press, Oxford, UK*, 5: 449.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1): 1–22.

Derrow-Pinion, A.; She, J.; Wong, D.; Lange, O.; Hester, T.; Perez, L.; Nunkesser, M.; Lee, S.; Guo, X.; Wiltshire, B.; et al. 2021. Eta prediction with graph neural networks in google maps. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 3767–3776.

Errica, F.; Bacciu, D.; and Micheli, A. 2021. Graph mixture density networks. In *International Conference on Machine Learning*, 3025–3035. PMLR.

Fang, X.; Huang, J.; Wang, F.; Zeng, L.; Liang, H.; and Wang, H. 2020. Constgat: Contextual spatial-temporal graph attention network for travel time estimation at baidu maps. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2697–2705.

Gonog, L.; and Zhou, Y. 2019. A review: generative adversarial networks. In *2019 14th IEEE conference on industrial electronics and applications (ICIEA)*, 505–510. IEEE.

Gui, J.; Sun, Z.; Wen, Y.; Tao, D.; and Ye, J. 2021. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering*.

Hong, H.; Lin, Y.; Yang, X.; Li, Z.; Fu, K.; Wang, Z.; Qie, X.; and Ye, J. 2020. Heteta: heterogeneous information network embedding for estimating time of arrival. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2444–2454.

Hunter, T.; Hofleitner, A.; Reilly, J.; Krichene, W.; Thai, J.; Kouvelas, A.; Abbeel, P.; and Bayen, A. 2013. Arriving on time: estimating travel time distributions on large-scale road networks. *arXiv preprint arXiv:1302.6617*.

Jin, G.; Wang, M.; Zhang, J.; Sha, H.; and Huang, J. 2022. STGNN-TTE: Travel time estimation via spatial–temporal graph neural network. *Future Generation Computer Systems*, 126: 70–81.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Li, X.; Cong, G.; Sun, A.; and Cheng, Y. 2019. Learning travel time distributions with deep generative model. In *The World Wide Web Conference*, 1017–1027.

Ma, Z.; Koutsopoulos, H. N.; Ferreira, L.; and Mesbah, M. 2017. Estimation of trip travel time distribution using a generalized Markov chain approach. *Transportation Research Part C: Emerging Technologies*, 74: 1–21.

Masoudnia, S.; and Ebrahimpour, R. 2014. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2): 275–293.

Matheson, J. E.; and Winkler, R. L. 1976. Scoring rules for continuous probability distributions. *Management science*, 22(10): 1087–1096.

McLachlan, G. J.; and Basford, K. E. 1988. *Mixture models: Inference and applications to clustering*, volume 38. M. Dekker New York.

Ramezani, M.; and Geroliminis, N. 2012. On the estimation of arterial route travel time distribution with Markov chains. *Transportation Research Part B: Methodological*, 46(10): 1576–1590.

Shao, J. 2003. *Mathematical statistics*. Springer Science & Business Media.

Song, X.; Zhang, C.; and James, J. 2021. Learn Travel Time Distribution with Graph Deep Learning and Generative Adversarial Network. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 1385–1390. IEEE.

Wang, D.; Zhang, J.; Cao, W.; Li, J.; and Zheng, Y. 2018. When will you arrive? Estimating travel time based on deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Wang, Y.; Zheng, Y.; and Xue, Y. 2014. Travel time estimation of a path using sparse trajectories. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 25–34.

Wang, Z.; Fu, K.; and Ye, J. 2018. Learning to estimate the travel time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 858–866.

Weglarczyk, S. 2018. Kernel density estimation and its application. In *ITM Web of Conferences*, volume 23, 00037. EDP Sciences.

Wu, H.; Mao, J.; Sun, W.; Zheng, B.; Zhang, H.; Chen, Z.; and Wang, W. 2016. Probabilistic robust route recovery with spatio-temporal dynamics. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1915–1924.

Zhang, K.; Jia, N.; Zheng, L.; and Liu, Z. 2019. A novel generative adversarial network for estimation of trip travel time distribution with trajectory data. *Transportation Research Part C: Emerging Technologies*, 108: 223–244.