TAML: A Traffic-aware Multi-task Learning Model for Estimating Travel Time

JIAJIE XU, Cyberspace Institute of Advanced Technology, Guangzhou University and School of Computer Science and Technology, Soochow University, China
SAIJUN XU, School of Computer Science and Technology, Soochow University, China
RUI ZHOU and CHENGFEI LIU, Swinburne University of Technology, Australia
AN LIU, School of Computer Science and Technology, Soochow University and State Key Laboratory of Software Architecture, Neusoft Corporation, China
LEI ZHAO, School of Computer Science and Technology, Soochow University, China

Travel time estimation has been recognized as an important research topic that can find broad applications. Existing approaches aim to explore mobility patterns via trajectory embedding for travel time estimation. Though state-of-the-art methods utilize estimated traffic condition (by explicit features such as average traffic speed) for auxiliary supervision of travel time estimation, they fail to model their mutual influence and result in inaccuracy accordingly. To this end, in this article, we propose an improved traffic-aware model, called TAML, which adopts a multi-task learning network to integrate a travel time estimator and a traffic estimator in a shared space and improves the accuracy of estimation by enhanced representation of traffic condition, such that more meaningful implicit features are fully captured. In TAML, multi-task learning is further applied for travel time estimation in multi-granularities (including road segment, sub-path, and entire path). The multiple loss functions are combined by considering the homoscedastic uncertainty of each task. Extensive experiments on two real trajectory datasets demonstrate the effectiveness of our proposed methods.

CCS Concepts: • **Applied computing** → **Transportation;**

Additional Key Words and Phrases: Travel time estimation, deep learning, trajectory data mining

ACM Reference format:
Jiajie Xu, Saijun Xu, Rui Zhou, Chengfei Liu, An Liu, and Lei Zhao. 2021. TAML: A Traffic-aware Multi-task Learning Model for Estimating Travel Time. *ACM Trans. Intell. Syst. Technol.* 12, 6, Article 75 (December 2021), 14 pages.
https://doi.org/10.1145/3466686

This work was supported by the National Natural Science Foundation of China under grant nos. 61872258, 61802273, Major project of natural science research in Universities of Jiangsu Province under grant no. 20KJA520005.
Authors' addresses: J. Xu, Cyberspace Institute of Advanced Technology, Guangzhou University and School of Computer Science and Technology, Soochow University, Guangzhou, Suzhou, Guangdong, Jiangsu, China; email: xujj@suda.edu.cn; S. Xu and L. Zhao, School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu, China; emails: sjxu@stu.suda.edu.cn, zhaol@suda.edu.cn; R. Zhou and C. Liu, Swinburne University of Technology, Australia; emails: rzhou@swin.edu.au, cliu@swin.edu.au; A. Liu, School of Computer Science and Technology, Soochow University and State Key Laboratory of Software Architecture, Neusoft Corporation, Suzhou, Shenyang, Jiangsu, Liaoning, China; email: anliu@suda.edu.cn.

2157-6904/2021/12-ART75 $15.00
https://doi.org/10.1145/3466686

1 INTRODUCTION

With the advancement of location acquisition technologies, large amounts of trajectory data have been collected every day, which provide us great opportunities to understand spatial dynamics of city crowd and the mobility of moving objects by conducting trajectory data mining. There are lots of trajectory data mining tasks, such as destination prediction [27], travel time estimation [28, 29], traffic prediction [22, 31, 36], crowd flow prediction [21], and so on. As a trajectory data mining task, travel time estimation is considered to be essential for many location-based applications, such as departure time suggestion [3], trip planning [10, 25, 26], vehicle dispatching [33], and so on. However, it is difficult to achieve an accurate estimation due to dynamic traffic conditions, different weather factors, and diverse driving behaviors.

In recent years, the majority of the research [11, 20, 24] applies trajectory embedding for travel time estimation and obtains promising results. In Reference [20], mobility patterns are learned from trajectories in 2D geographical space first, and then used to estimate the travel time. Since the movement of vehicles is restricted by the road network, topology-aware models [11, 24] improve the representation of spatio-temporal prior knowledge by leveraging the influence of road network. While in reality, the traffic condition has a direct impact on the travel time, and the traffic-aware models [24, 34] aim to take traffic-related contextual information into account. The WDR model [24] further utilizes a traffic estimator to estimate the time-dependent traffic condition of the departure time, which is fed into the deep neural network afterwards for more accurate estimation.

Although the traffic-aware models [24, 34] mentioned above achieve satisfactory results and become state-of-the-art, they still have limitations. *First*, when incorporating traffic condition as contextual information, they treat traffic estimator as an independent component in training, but they neglect the effect of trajectory travel time for supervising the estimation of traffic condition, i.e., derived information of trajectories. Since vehicle travel time and its related traffic condition have mutual influence on each other, it calls for an improved model that can integrate traffic estimator and travel time estimator seamlessly for joint learning. *Second*, these models utilize explicit traffic features only, e.g., traffic speed and volume, while explicit features are insufficient to reflect the traffic-related context in the real world, due to traffic conditions fluctuating dynamically over time and exhibiting periodicity patterns, as indicated by References [13, 35]. Thus, the dynamic fluctuation and periodicity, which should be captured by implicit traffic features, are often overlooked by existing methods. Therefore, enhanced representation of traffic contexts are sought after for more accurate travel time estimation.

To this end, in this article, we propose a novel **Traffic-Aware Multi-task Learning travel time estimation model (TAML)**, which achieves more accurate estimation of travel time via the enhanced representation of traffic-related contexts. It contains a travel time estimator and a traffic estimator. It takes advantage of multi-task learning to seamlessly combine them in a shared space. Specifically, the latent states of traffic estimator, rather than estimated values of explicit features, are shared with the travel time estimator for co-training. Besides, in the travel time estimator, it also adopts multi-task learning to estimate the travel time of the entire path and each road segment of the path simultaneously. This ensures the travel time estimator to estimate the travel time more accurately due to the superior understanding of the complex effect of implicit traffic features. Furthermore, the mutual influence of traffic conditions on travel time can be fully modelled by multi-task supervision. In TAML, we finally combine multiple loss functions by considering the homoscedastic uncertainty of each task. To sum up, the contributions of the article are summarized as follows:

- We propose a novel traffic-aware model called TAML, which takes advantage of multi-task learning to seamlessly integrate a travel time estimator and a traffic estimator, which are

trained together in a shared space. TAML achieves accurate estimation via the enhanced representation of the traffic-related context, which is described by more meaningful implicit features that can estimate future traffic condition and vehicle travel time.
- We further adopt multi-task learning to estimate the travel time for the entire path, and for each road segment of the path simultaneously by a multi-task loss function. The multiple loss functions reasonably combined by considering homoscedastic uncertainty of each task.
- We conduct extensive experiments on real datasets. The results demonstrate the advantages of our approach compared with several baseline methods.

2 RELATED WORK

Existing approaches of estimating travel time on the path could be classified into two categories. Segment-based approaches [4, 30] estimate the travel time on each road segment but ignore the correlations between the road segments. Path-based approaches [12, 16, 23] resolve those issues in segment-based approaches by extracting and aggregating sub-paths from historical trajectories, but suffer from data sparsity for the lack of sub-paths in historical trajectories.

As mentioned above, neither segment-based nor path-based approaches could achieve satisfactory performance in travel time estimation. Meanwhile, sufficiently large datasets and enough computational power enable the success of deep learning these years [18]. In Reference [20], it first applies trajectory embedding by utilizing geo-convolution in 2D geographical space to find trajectories roughly following the target path for aggregation and then estimates the travel time. Since the movement of vehicles is restricted by the road network, topology-aware models [11, 24] are proposed to improve the representation of spatio-temporal prior knowledge by leveraging the influence of road network. In Reference [11], it learns representations of road network and uses the spatio-temporal knowledge for travel time estimation.

Furthermore, considering the impact of underlying traffic information of the path on travel time, traffic-aware models [24, 34] come into being. DeepTravel [34] transforms a query path into a sequence of geographical grid cells. However, it takes a coarse granularity of trajectories along underlying traffic in each grid, thus leading to inaccurate results, since the average speed of a grid is unanimously imposed on all road segments of various actual speeds. The WDR model [24] incorporates traffic information obtained by a traffic estimator during the feature extraction, then an ensemble regression model is used to estimate travel time. However, it uses a traffic estimator as an independent component in training and neglects the mutual influence between the estimation of travel time and its related traffic condition, resulting in sub-optimal results accordingly. In this article, we design a multi-task learning network to integrate the travel time estimator and traffic estimator in a shared space and achieve accurate estimation of travel time by the enhanced representation of related traffic contexts.

3 PROBLEM FORMULATION

This section introduces some preliminaries and gives a formal statement of the problem studied in this article.

Definition 1 (Road Network). A road network is described as a directed graph $G(V; E)$ where vertex $v \in V$ is an intersection or terminal point of road segments, and directed edge $e \in E$ is a road segment with an ID $e.eid$, a length value $e.l$, a starting point $e.start$, and an ending point $e.end$. A path P is linearly connected road segments $P = (e_1, e_2, \ldots, e_n)$ on the road network where $e_i \in E$, $1 \leq i \leq n$.

Definition 2 (Trajectory). A raw GPS trajectory is a sequence of time-stamped GPS locations (latitude, longitude). We use map-matching [32] to match raw trajectory to a path on the road

network G. Each map-matched trajectory is represented as $x^{(i)} = (P^{(i)}, T^{(i)}, \tau^{(i)})$, where $P^{(i)} = (e_1^{(i)}, e_2^{(i)}, \ldots, e_n^{(i)})$ denotes the path it is map-matched on; $T^{(i)} = (t_1^{(i)}, t_2^{(i)}, \ldots, t_n^{(i)})$ represents time that the vehicle travels on each road segment; $\tau^{(i)}$ is departure time.

The historical trajectory dataset is accordingly represented by a set of trajectories $X = \{x^{(i)} | i = 1, 2, 3, \ldots, N\}$.

Problem Statement (Travel Time Estimation). Given a trajectory dataset X and a query $q = (P, \tau)$ with its query path P and a departure time τ, our goal is to estimate the travel time through P using X.

4 MODEL ARCHITECTURE

In this section, we introduce the architecture of our multi-task learning model TAML, as shown in Figure 1, which consists of a representation layer (Section 4.1), a traffic estimation layer for enhanced traffic representation (Section 4.2), travel time estimation layer for accurate travel time estimation (Section 4.3), and a multi-task learning layer to capture their mutual influence (Section 4.4).

4.1 Representation Layer

In this layer, we extract the enhanced representation of the traffic-related context from the given travel path. Specifically, given a query q with its path P and a departure time τ, we incorporate useful information that could affect travel time and transform them into learnable features, which include three categories: road network, the external features, and traffic-related contexts.

Representation of road network. Vehicle moving patterns are restricted by underlying road network and it is important to consider the road network structure in TTE. We utilize graph embedding techniques [2, 15] to capture road network topology and similarities between neighboring road segments, as implemented in MURAT [11]. We first construct the underlying road network as an adjacency matrix A, which denotes the connectivity among road segments, A_{ij} is 0 if road segment e_i and e_j are not directly connected, else is 1. Then matrix A could be fed into the graph embedding technique as the input to calculate a representation that related the surrounding road segment. The graph embedding method not only captures the topology structure information in the graph and shares similar representations among related road segments, but also effectively reduces the input dimension and thus it is more computationally efficient. In this case, a representation R_i of the road segment e_i can be generated.

Representation of external features. There are external factors that affect the travel time, including holidays, weekday or weekends, peak or non-peak hours, and weather conditions. We extract these factors and encode them into a real space $\mathbb{R}^{1\times A}$ by using the embedding mechanism [14], which not only reduces the dimension of categorical values but also represents categories in the transformed space. Then, we concatenate these embedding vectors of external features to obtain low-dimensional vectors E_τ.

Representation of traffic-related contexts. The traffic condition of the traffic network has a direct impact on the travel time. Existing solution [24] utilizes an estimator to derive the time-dependent traffic condition at the departure time described by explicit features (e.g., traffic volume and speed) and uses them as contextual information for travel time estimation. However, they are unable to fully reflect traffic condition in the real world. Instead of using explicit features, we regard latent states of traffic estimation layer (i.e., traffic estimator in Section 4.2) as the enhanced representation of traffic context w.r.t. road segment e_i at departure time τ, denoted as $C_{i,\tau}$, which is eventually co-trained by two estimators, as shown in Figure 1.

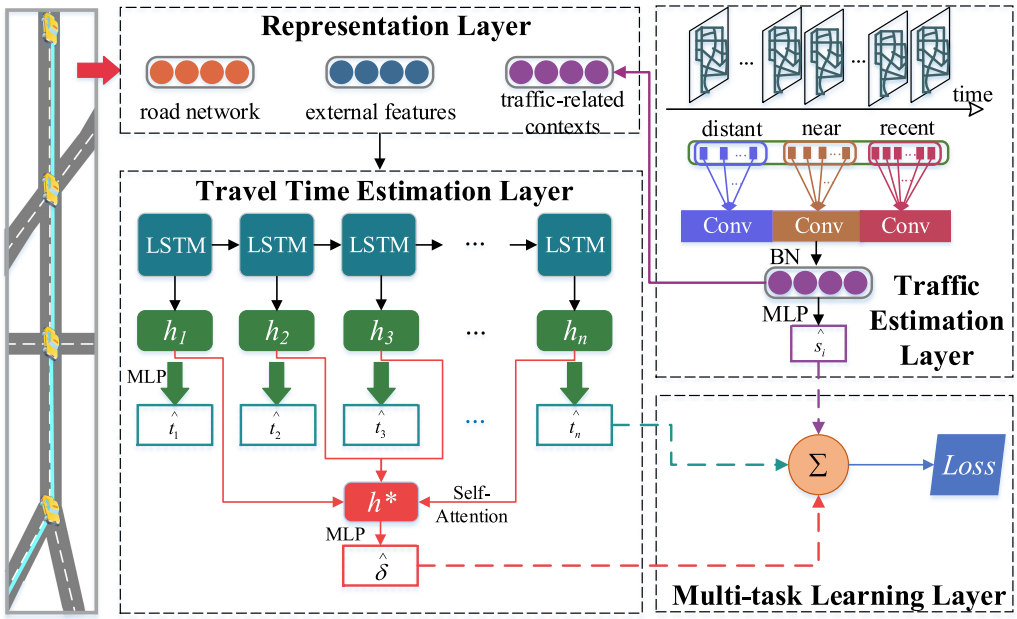


Fig. 1. The architecture of TAML.

To estimate the travel time of each road segment e_i, we combine all above features by a concatenation $V_{i,\tau}$=[R_i, E_τ, $C_{i,\tau}$], which provides the information of the road segment, related external features, and traffic-related contexts w.r.t. the departure time.

4.2 Traffic Estimation Layer

In this layer, let M record historical traffic flow information before the current time t_1, where $M_{i,t}$ denotes the average traffic speed of road segment e_i at the tth time slot (e.g., five minutes), which is derived from historical trajectories and is processed in Reference [13]. We aim to estimate the underlying traffic condition M_{i,t_τ} of road segment e_i during the t_τth time slot (that departure time τ belongs to) using M.

Traffic conditions have time periodicity pattern and we extract traffic speeds from M in three fragments of the time axis, reflecting recent time, near history, and distant history. First, we extract recent traffic speeds of e_i at the time slots of [t_τ-12, t_τ-1], which can be denoted as recent traffic features r_{i,t_τ}=[$M_{i,t_\tau-1}$, $M_{i,t_\tau-2}$, . . . , $M_{i,t_\tau-12}$] to represent temporal closeness pattern, since traffic speeds fluctuate with time, i.e., the traffic conditions of the previous time period affect that of the next time moment; second, we extract near traffic speeds of e_i at the t_τth time slot of the day in the previous six days, which are denoted as near traffic features n_{i,t_τ}=[$M_{i,t_\tau-288}$, $M_{i,t_\tau-288\times2}$, . . . , $M_{i,t_\tau-288\times6}$], since traffic speeds have time periodicity pattern, for example, traffic conditions during morning rush hours may be similar on consecutive weekdays, repeating every 24 hours; furthermore, we extract distant traffic speeds of e_i both at the t_τth time slot of the day and in the same weekday of the previous three weeks, which are denoted as distant traffic features d_{i,t_τ}=[$M_{i,t_\tau-2016}$, $M_{i,t_\tau-2016\times2}$, $M_{i,t_\tau-2016\times3}$].

Afterwards, based on the above recent, near and distant traffic features, we aim to derive useful latent features for traffic speed estimation. We first use three convolution operations to convolve recent, near, and distant traffic features in the recent, near, and distant term, respectively, and then

concatenate the latent states of recent, near, and distant traffic conditions generated by convolution operation as the enhanced representation of traffic context. To this end, convolutional operation [9] is applied on r_{i,t_τ}, n_{i,t_τ}, and d_{i,t_τ}, respectively, We use k convolutional filters to convolve them, respectively, which can be formulated as,

$$\mathcal{Z}_{i,r}^{t_\tau} = f\left(W_{conv}^{(r)} * r_{i,t_\tau} + b_r\right), \tag{1}$$

$$\mathcal{Z}_{i,n}^{t_\tau} = f\left(W_{conv}^{(n)} * n_{i,t_\tau} + b_n\right), \tag{2}$$

$$\mathcal{Z}_{i,d}^{t_\tau} = f\left(W_{conv}^{(d)} * d_{i,t_\tau} + b_d\right), \tag{3}$$

where $*$ represents the convolutional operation, $W_{conv}^{(r)}$, $W_{conv}^{(n)}$, $W_{conv}^{(d)}$ are the parameter weights, b_r, b_n, b_d are the biases. f is an activation function; here, we use rectifier linear unit $ReLU(x) = max(0, x)$ [9]. Batch Normalization [6] is employed for faster training speed after convolutional operations.

Finally, the traffic speed of e_i at the departure time slot is estimated on top of the latent features, which are concatenated to a vector F_{i,t_τ}=[$\mathcal{Z}_{i,r}^{t_\tau}$, $\mathcal{Z}_{i,n}^{t_\tau}$, $\mathcal{Z}_{i,d}^{t_\tau}$]. F_{i,t_τ} is the latent state of traffic estimation layer, which can be viewed as the enhanced representation of traffic context $C_{i,\tau}$, to better reflect traffic condition compared with using explicit features. Then, we incorporate $C_{i,\tau}$ with road network representation R_i and external features E_τ that affect the traffic condition, which are computed in Section 4.1 as inputs of **multi-layer perceptron (MLP)** [5] to estimate the final traffic speed of e_i at the departure time slot.

4.3 Travel Time Estimation Layer

This layer takes the information of the representation layer as input and aims to accurately estimate vehicle travel time using the learned spatial-temporal knowledge.

We take advantage of sequential models to perform the travel time estimation of the query path. Inspired by Reference [24], the road network consists of many road segments, and these road segments can be viewed as building blocks of paths. The road segments along each path have clear sequential structure. This motivates us to introduce **long short-term memory network (LSTM)** [17], which has been widely used in sequence modelling to capture the temporal dependencies in the extracted feature vector sequence. The feature vectors $V_{j,\tau}$ of the jth road segment e_j are fed into LSTM units in the chronological order. An LSTM unit consists of a memory cell c_j, an input gate i_j, a forget gate f_j, and an output gate o_j at the jth feature vectors. The output h_j of the LSTM unit is

$$h_j = o_j \circ tanh(c_j), \tag{4}$$

where $\circ$ denotes the element-wise product, and o_j is an output gate that modulates the amount of memory content exposure, which is computed by,

$$o_j = \sigma(W_o V_{j,\tau} + U_o h_{j-1} + A_o c_j), \tag{5}$$

where $\sigma(.)$ is an element-wise logistic sigmoid function and A_o is a diagonal matrix. The content of the memory cell c_j is the weighted sum of the new content $\widetilde{c}_j$ and the previous memory content c_{j-1}, that is,

$$c_j = f_j \circ c_{j-1} + i_j \circ \widetilde{c}_j. \tag{6}$$

$\widetilde{c}_j$ is calculated in the following:

$$\widetilde{c}_j = tanh(W_c V_{j,\tau} + U_c h_{j-1}), \tag{7}$$

where W_c, U_c are weight matrices. The input and forget gates control how much new content should be memorized and how much old content should be forgotten. These gates are computed from the previous hidden states and the current input:

$$i_j = \sigma(W_i V_{j,\tau} + U_i h_{j-1} + A_i c_{j-1}), \tag{8}$$

$$f_j = \sigma(W_f V_{j,\tau} + U_f h_{j-1} + A_f c_{j-1}), \tag{9}$$

where W_i, W_f, U_i, U_f are weight matrices and A_i, A_f are diagonal matrices. After feeding $V_{1,\tau}$, $V_{2,\tau}$,...,$V_{n,\tau}$ to LSTM, we get the hidden units sequence H=$\{h_1, h_2, \ldots, h_n\}$, which contains time dependencies for the following travel time estimation. To fully leverage the supervised labels in historical trajectories and achieve more accurate results, we adopt multi-task learning to estimate the travel time not only for entire path P but also for each road segment e_j of the path simultaneously. After feeding hidden units sequence H to MLP, we can get the final estimated time $\hat{t}_j$ on e_j. However, H cannot be directly used to estimate travel time of the query path, for the length of hidden units sequence is variable, thus, we need to encode H into a representation that can highly summarize the contextual information. Since self-attention technique [19] is good at capturing global dependencies that exist in the sequence, we use it to calculate h^* as the comprehensive representation of H, which considers the correlations between any two road segments in the path. The calculation process of self-attention could be first written as,

$$\begin{aligned} Q &= W_Q^T [h_1, h_2, \ldots, h_n] + b_Q, \\ K &= W_K^T [h_1, h_2, \ldots, h_n] + b_K, \\ V &= W_V^T [h_1, h_2, \ldots, h_n] + b_V, \end{aligned} \tag{10}$$

where Q, K, V denote a query vector, a key vector, and a value vector, and these vectors are created by three linear functions. The weights and biases of three linear functions are trained during the training process. Then, we compute the weight of each value by calculating a compatibility function of the query with the corresponding key. We compute the weights of outputs a:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right). \tag{11}$$

The d_k denotes the dimension of keys that leads to having more stable gradients. Here, Softmax function normalizes the weights so they are all positive and add up to 1. And $Attention(Q, K, V)$ is the weights matrix. Finally, we combine the attention matrix and values in the model, which is the following:

$$h^* = Attention(Q, K, V)V. \tag{12}$$

Here, h^* is the comprehensive representation of H. Finally, taking h^* as the inputs of MLP to output the final estimated time $\hat{\delta}$ of the path.

4.4 Multi-task Learning Layer

The multi-task learning layer combines a travel time estimator and a traffic estimator in a shared space (i.e., the enhanced representation of the traffic-related context) to compute more effective representations and parameters for more accurate results in both time estimator and traffic estimator. Both trajectory data and traffic speed historical observations are used as labels to supervise the travel time estimator and the traffic estimator.

Since the model includes a traffic estimator and a travel time estimator, we discuss how their losses should be defined first, and then combined afterwards. We use the **mean absolute percentage error (MAPE)** as the loss function in the training phase, because MAPE is a measure of

prediction accuracy of a forecasting method in statistics, which calculates error in terms of percentage and therefore it is not relative to the size of the numbers in the data itself. In this way, Loss is calculated by normalizing all errors on a common scale (of hundred). In addition, we use multiple metrics to evaluate the estimation accuracy in the testing phase. For the traffic estimator, we define the loss function of traffic speed estimation as

$$L_s = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{s_i - \hat{s}_i}{s_i}\right|, \tag{13}$$

where $\hat{s}_i$ is the estimated traffic speed of e_i at the departure time slot, and s_i is the ground truth, the actual traffic speed that is computed from historical trajectory data. Similarly, the loss function of TTE estimation of the path is

$$L_p = \left|\frac{\delta - \hat{\delta}}{\delta}\right|, \tag{14}$$

where $\hat{\delta}$ is the estimated travel time of the entire path and δ is the actual time. For the travel time estimator, we also adopt multi-task learning to not only estimate the travel time for the entire path, but also for each road segment of the path simultaneously. In this way, we can not only utilize trajectory temporal labels more sufficiently, but also model the path more accurately, since the intersections between two consecutive road segments are fully represented. Specifically, for each road segment of the query path, we use the following loss function for the travel time estimation in road segment granularity:

$$L_r = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{t_i - \hat{t}_i}{t_i}\right|, \tag{15}$$

where t_i is travel time on passing road segment and $\hat{t}_i$ is the estimated value.

Multiple tasks can be learned simultaneously by using a naive weighting of losses, where the loss weights are manually tuned. However, searching for an optimal weight is prohibitively expensive, and it is difficult to resolve with manual tuning. To deal with it, we introduce an approach in Reference [7] to weigh multiple loss functions by using task-dependent homoscedastic uncertainty. The relative weightings are dependent on the magnitude of the task's noise and are automatically optimized during the training without complex searching process. Specifically, the multi-task loss function is

$$L = \frac{1}{2\sigma_1^2}L_s + \frac{1}{2\sigma_2^2}L_p + \frac{1}{2\sigma_3^2}L_r + \log(\sigma_1\sigma_2\sigma_3), \tag{16}$$

where σ_1, σ_2, and σ_3 are the observation noise parameters whose initial values are set to a random value between 0–1; these noise parameters are learnable in the training phase. We minimize L as the training goal to accurately estimate the traffic speed and the travel time of both the path and each road segment simultaneously. In particular, during the testing phase, we eliminate both the travel time estimation of the road segment and the part of traffic estimation layer that uses MLP to estimate traffic speed, and we only estimate travel time $\hat{\delta}$ of the path.

5 EXPERIMENT

In this section, we conduct experiments on two real-world datasets to evaluate the performance of TAML against several travel time estimation baseline models under different parameter settings.

5.1 Datasets

- **Beijing:** Beijing Dataset consists of 3,459,534 trajectories of 11,369 taxis from April 24th, 2016, to May 31st, 2016, in Beijing, China. The shortest trajectory contains 9 GPS records (1.3 km), and the longest trajectory contains 125 GPS records (38.5 km).
- **Chengdu:** Chengdu Dataset consists of 2,717,391 trajectories of 10,162 taxis in June 2015 in Chengdu, China. The shortest trajectory contains 11 GPS records (1.4 km), and the longest trajectory contains 138 records (45.1 km).

We further collect weather conditions including wind speed, temperature, and weather types (e.g., Sunny, Cloudy, Rainy) in Beijing and Chengdu. For Beijing dataset, we select the data from April 24th to May 24th for model training, and the remaining seven days for testing; for Chengdu dataset, we use the data from June 1st to June 23th as the training set and the last seven days as the testing set.

5.2 Implementation Details

The architecture of our model TAML is shown in Figure 1; parameters of the model and experimental settings are described as follows:

- In the representation layer, we use unsupervised pre-training graph embedding technique DeepWalk [15] for the representations of road segment with the dimension 40. The day of the week, holidays, and weather type in the external features are embedded to $\mathbb{R}^{16}$, $\mathbb{R}^{8}$, $\mathbb{R}^{3}$, respectively.
- In the traffic estimation layer, the number of convolutional filters is 16 and the kernel size of convolutional filters are 12, 6, 3, respectively, to convolve recent, near, and distant traffic features.
- In the travel time estimation layer, we use two MLPs to predict travel time of each road segment and travel time of the entire path, respectively. The MLP that we use contains two hidden layers with ReLU activation and the size of each layer is 64 and 32, respectively.
- In the multi-task learning layer, the initial observation noise parameters σ_1, σ_2, and σ_3 are set to a random value between 0–1 and are updated in the training phase.

We set batch size as 128 and use Adam optimizer [8] as the optimization function. The initial learning rate is set to 0.001 and is reduced to $\frac{1}{5}$ of the previous learning rate when losses no longer drop every two epochs. The experiment is implemented by PyTorch. We train and evaluate the model on the server with one NVIDIA GTX1080 GPU and 24 CPU cores.

5.3 Baselines

We compare our model with the following baseline methods of travel time estimation:

- **GBDT: Gradient boosting decision tree (GBDT)** [1] is an ensemble machine learning model of decision trees. The input of GBDT includes the departure time, a travel path, historical average traffic speeds of each passing road segment at the time slot that departure time belongs to, and external factors (weather, holidays).
- **MlpTTE:** A three-layer perceptron with ReLU activation is used to estimate the travel time. The input of MlpTTE is the same as GBDT. The size of hidden layers in MlpTTE is fixed as 128.
- **WDR:** WDR [24] is a traffic-aware model that incorporates estimated traffic volume and speed information. An ensemble model consisting of wide linear models, deep neural models, and recurrent neural networks is used to estimate the travel time.

Table 1. Performance Comparison of Evaluated Approaches in Metrics MAE, RMSE, and MAPE

Dataset	*Beijing*			*Chengdu*		
Metrics	MAE(sec)	RMSE(sec)	MAPE(%)	MAE(sec)	RMSE(sec)	MAPE(%)
MlpTTE	218.31	389.32	22.99	198.69	365.68	23.14
GBDT	257.63	402.38	28.97	268.01	381.45	29.28
DeepTravel	187.72	370.53	18.99	171.65	339.24	19.91
WDR	189.96	375.97	19.33	185.41	347.93	20.04
DeepTTE	158.87	351.50	17.07	160.45	323.23	18.41
TAML(Ours)	**137.04**	**320.98**	**15.38**	**143.94**	**288.52**	**16.63**

- **DeepTravel:** DeepTravel [34] is a traffic-aware model that extracts spatial and temporal embeddings, driving state features and traffic features in a granularity of spatial grids and then uses Bi-LSTMs to estimate the travel time.
- **DeepTTE:** DeepTTE [20] is a model based on deep learning that transforms the raw GPS sequence to feature maps to capture local spatial correlations, then uses LSTM to learn the temporal dependencies and estimate the travel time.

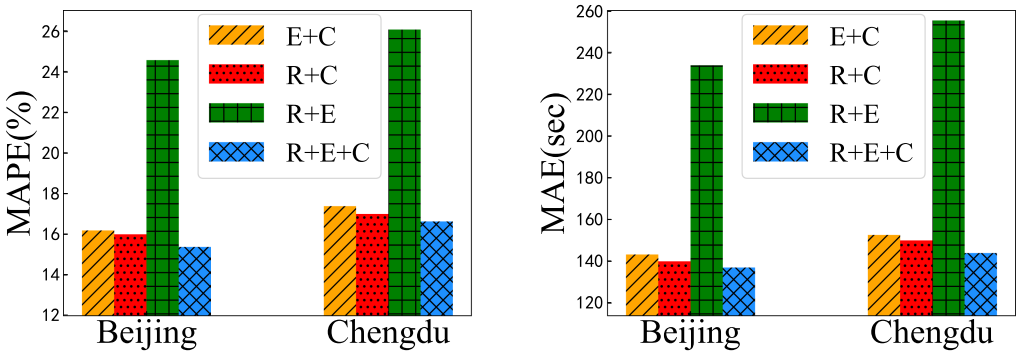
5.4 Performance Comparison

The parameters of all the comparison methods are tuned to be optimal. The comparison results are reported in Table 1. We use three metrics to evaluate the performance of the methods, including the **mean absolute percentage error (MAPE)**, the **rooted mean squared error (RMSE)**, and the **mean absolute error (MAE)**. MAPE is used to measure the relative errors between the average test value and the real value on the test set. It has intuitive interpretation in terms of relative error and is used in model evaluation. The MAE is used to measure the average absolute error between the predicted value and the real value on the experimental dataset. RMSE is used to measure the deviation between the observed value and the true value. RMSE is more sensitive to outliers.

Clearly, our proposed model TAML outperforms all the baseline models on two datasets in terms of all evaluation metrics. Several observations can be made from the results: (1) The GBDT has the worst performance, demonstrating the effectiveness of deep learning for travel time estimation; (2) DeepTTE, WDR, and DeepTravel outperform MlpTTE, indicating that the method of trajectory embedding can help to discover meaningful spatio-temporal prior knowledge for travel time estimation; (3) Traffic-aware models (DeepTravel and WDR) have similar performance with DeepTTE, which indicates that trajectory embedding contains some traffic information implicitly, and it helps little when directly incorporating explicit traffic features; (4) Our TAML model outperforms all baseline models due to the enhanced representation of traffic-related contexts, which is derived by the multi-task learning of travel time estimator and traffic estimator in a shared space.

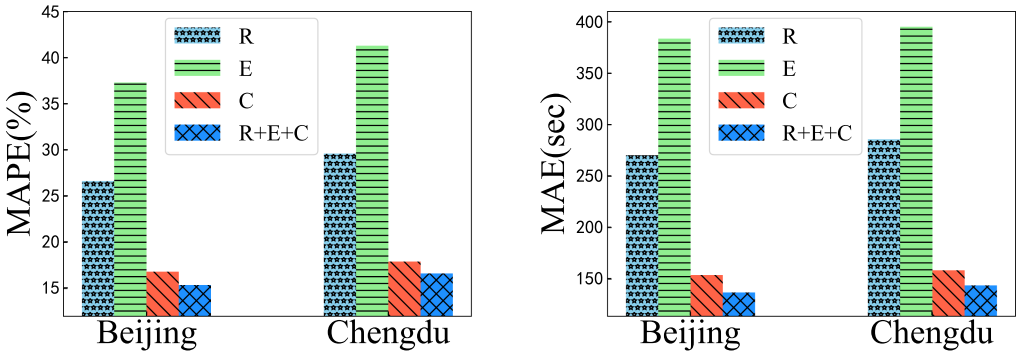
5.5 Model Analysis

In this section, we aim to evaluate the influence of three kinds of information extracted in the representation layer on the model estimation performance. Here, we denote the road network topology as R, the traffic-related context as C, the external factors as E. We construct the following models that are derived from TAML to study the influence of road network topology, traffic-related context, and external factors (weathers, holidays) in travel time estimation: (1) E+C, which removes the representation of road network is used to evaluate the effectiveness of infusing road network topology structure in travel time estimation; (2) R+C, which removes representation of external features, is used to evaluate the effectiveness of incorporating external features. (3) R+E, which removes representation of traffic-related contexts and accordingly modifies multi-task learning by



(a) MAPE (b) MAE

Fig. 2. Performance comparisons of different features.



(a) MAPE (b) MAE

Fig. 3. Performance comparisons of intuitive case study.

removing traffic speed estimator task, is used to evaluate the effectiveness of adding traffic-related context. (4) R+E+C, which is TAML model that contains all these representations above.

The MAPE and MAE performance on two datasets are shown in Figures 2(a) and 2(b). The RMSE performance is similar to the performance in MAPE and MAE. We can observe that: (1) R + E + C performs better than E + C, which shows the effectiveness of infusing road network topology structure in travel time estimation, since the vehicle moving patterns are restricted by underlying road network. (2) R + E + C outperforms R + C, which shows external features including holidays; weather conditions have an impact on the travel time estimation. (3) R + E + C significantly outperforms R + E, which not only proves the effectiveness of incorporating traffic-related context that contains implicit underlying traffic, but also proves the effectiveness of the addition of the traffic estimator for auxiliary supervision of travel time estimation.

Furthermore, in terms of intuitive case study, we conduct experiments with simplified variants derived from TAML that use road network topology (R), traffic-related context (C), and external factors (E) individually in the representation layer. The MAPE and MAE performance on two datasets are shown in Figures 3(a) and 3(b). From the comparison results, the traffic-related context has the greatest influence on travel time, then is the road network topology, and finally is the external factors.

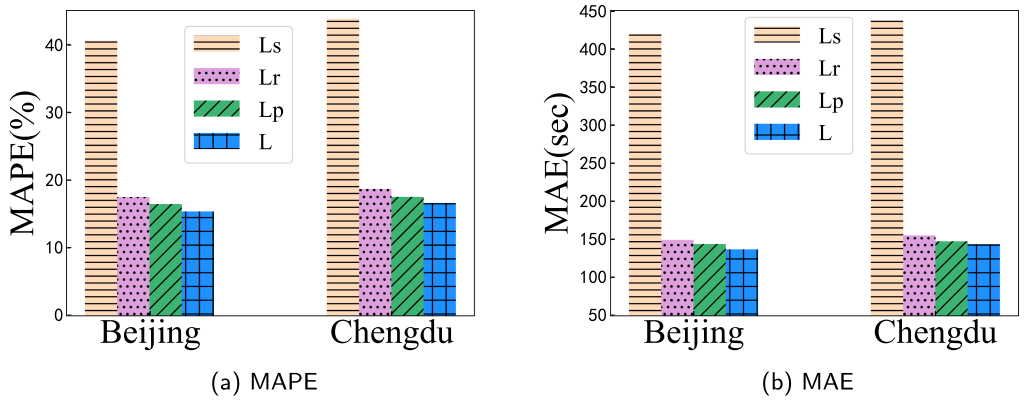


Fig. 4. Performance comparisons of using different loss functions.

Table 2. Comparisons of Multi-task Loss Combination Strategy and Explicit Traffic-related Features

Dataset	*Beijing*			*Chengdu*		
Metrics	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Approx.optimal	144.38	331.69	16.11%	152.23	303.86	17.18%
uncertainty	**137.04**	**320.98**	**15.38%**	**143.94**	**288.52**	**16.63%**
TAML-S	140.89	326.25	15.95%	147.16	296.83	17.09%

Moreover, to study the impact of using different loss functions on model performance, we train TAML with reduced loss functions that use the loss function L_s, L_p, and L_r individually. The comparison results are shown in Figure 4. From the results, we can observe that, first, using L_p and L_r as loss function can train more optimal parameters, because it aims to minimize the error between predicted travel time and actual time; second, using L_s as loss function has the worst effect on model training, because the goal of this loss function is about traffic speed estimation instead of travel time estimation; third, using the joint loss function performs better than using these loss functions individually, which shows that the combination of these loss functions is more effective in improving the performance of the model.

In addition, to evaluate the effectiveness of implicit traffic-related features, we conducted the ablation experiment, i.e., comparing the TAML model with a simplified variant, namely, TAML-S, which utilizes explicit traffic-related features (which are the traffic speeds and can be derived by a traffic estimator). The performance results are in Table 2. From the comparison results, the TAML model with implicit traffic-related features performs better than the one with explicit features. It indicates that explicit traffic-related features are insufficient to reflect the traffic-related context in the real world.

Finally, to show the effectiveness of adopting homoscedastic uncertainty weights of multi-task loss compared with using a naive weighted loss, we modify the multi-task loss with a uniform weighting summation of losses of tasks, and these weightings are set to the approximately optimal value through multiple parameter search experiments. The performance comparison results are shown in Table 2. From Table 2, we can observe that the model using task-dependent homoscedastic uncertainty performs better compared to the model using an approximately optimal value through multiple searching in all three evaluation metrics on both Beijing and Chengdu

datasets, which demonstrates the effectiveness of adopting homoscedastic uncertainty weights of losses. The reason is that the task weights are dynamic and are learned automatically from the data in the training phase.

6 CONCLUSION

In this article, we study the problem of travel time estimation. Existing approaches mainly estimate the travel time by learning mobility patterns from large-scale trajectories. However, they fail to model the mutual influence between traffic condition and vehicle travel time, resulting in inaccurate estimations. We propose a traffic-aware travel time estimation model, namely, TAML, which takes advantage of multi-task learning to integrate a travel time estimator and a traffic estimator in a shared space and achieves more accurate estimation by incorporating the enhanced traffic-related representation. We further adopt multi-task learning to estimate the travel time of the entire path and each road segment of the path. In addition, we weigh multiple loss functions by considering homoscedastic uncertainty of each task. Extensive experiments on two real trajectory datasets demonstrate the effectiveness of our proposed methods.

REFERENCES

[1] Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *Ann. statist.* 29, 5 (2001), 1189–1232.

[2] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD'16*. 855–864. DOI: https://doi.org/10.1145/2939672.2939754

[3] Ziyuan Gu and Meead Saberi. 2019. A simulation-based optimization framework for urban congestion pricing considering travelers' departure time rescheduling. In *ITSC'19*. 2557–2562. DOI: https://doi.org/10.1109/ITSC.2019.8916910

[4] Aude Hofleitner, Ryan Herring, Pieter Abbeel, and Alexandre M. Bayen. 2012. Learning the dynamics of arterial traffic from probe data using a dynamic Bayesian network. *Trans. Intell. Transport. Syst.* 13, 4 (2012), 1679–1693. DOI: https://doi.org/10.1109/TITS.2012.2200474

[5] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Netw.* 2, 5 (1989), 359–366. DOI: https://doi.org/10.1016/0893-6080(89)90020-8

[6] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML'15*. 448–456.

[7] Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR'18*. 7482–7491. DOI: https://doi.org/10.1109/CVPR.2018.00781

[8] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR'15*.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS'12*. 1106–1114.

[10] Feifei Li, Dihan Cheng, Marios Hadjieleftheriou, George Kollios, and Shang-Hua Teng. 2005. On trip planning queries in spatial databases. In *SSTD'05*. 273–290. DOI: https://doi.org/10.1007/11535331_16

[11] Yaguang Li, Kun Fu, Zheng Wang, Cyrus Shahabi, Jieping Ye, and Yan Liu. 2018. Multi-task representation learning for travel time estimation. In *KDD'18*. 1695–1704. DOI: https://doi.org/10.1145/3219819.3220033

[12] Wuman Luo, Haoyu Tan, Lei Chen, and Lionel M. Ni. 2013. Finding time period-based most frequent path in big trajectory data. In *SIGMOD'18*. 713–724. DOI: https://doi.org/10.1145/2463676.2465287

[13] Zhongjian Lv, Jiajie Xu, Kai Zheng, Hongzhi Yin, Pengpeng Zhao, and Xiaofang Zhou. 2018. LC-RNN: A deep learning model for traffic speed prediction. In *IJCAI'18*. 3470–3476. DOI: https://doi.org/10.24963/ijcai.2018/482

[14] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS'13*. 3111–3119.

[15] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *KDD'14*. 701–710. DOI: https://doi.org/10.1145/2623330.2623732

[16] Mahmood Rahmani, Erik Jenelius, and Haris N. Koutsopoulos. 2013. Route travel time estimation using low-frequency floating car data. In *ITSC'13*. 2292–2297. DOI: https://doi.org/10.1109/ITSC.2013.6728569

[17] Hasim Sak, Andrew W. Senior, and Françoise Beaufays. 2014. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH'14*. 338–342.

[18] Gregor Ulm, Simon Smith, Adrian Nilsson, Emil Gustavsson, and Mats Jirstrand. 2021. OODIDA: On-Board/Off-Board distributed real-time data analytics for connected vehicles. *Data Sci. Eng.* 6, 1 (2021), 102–117. DOI: https://doi.org/10.1007/s41019-021-00152-6

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS'17*. 5998–6008.
[20] Dong Wang, Junbo Zhang, Wei Cao, Jian Li, and Yu Zheng. 2018. When will you arrive? Estimating travel time based on deep neural networks. In *AAAI'18*. 2500–2507.
[21] Senzhang Wang, Jiannong Cao, Hao Chen, Hao Peng, and Zhiqiu Huang. 2020. SeqST-GAN: Seq2Seq generative adversarial nets for multi-step urban crowd flow prediction. *ACM Trans. Spatial Algor. Syst.* 6, 4 (2020), 22:1–22:24.
[22] Senzhang Wang, Xiaoming Zhang, Fengxiang Li, Philip S. Yu, and Zhiqiu Huang. 2019. Efficient traffic estimation with multi-sourced data by parallel coupled hidden markov model. *IEEE Trans. Intell. Transp. Syst.* 20, 8 (2019), 3010–3023.
[23] Yilun Wang, Yu Zheng, and Yexiang Xue. 2014. Travel time estimation of a path using sparse trajectories. In *KDD'14*. 25–34. DOI: https://doi.org/10.1145/2623330.2623656
[24] Zheng Wang, Kun Fu, and Jieping Ye. 2018. Learning to estimate the travel time. In *KDD'18*. 858–866. DOI: https://doi.org/10.1145/3219819.3219900
[25] Jiajie Xu, Jing Chen, Rui Zhou, Junhua Fang, and Chengfei Liu. 2019. On workflow aware location-based service composition for personal trip planning. *Fut. Gen. Comput. Syst.* 98 (2019), 274–285.
[26] Jiajie Xu, Yunjun Gao, Chengfei Liu, Lei Zhao, and Zhiming Ding. 2015. Efficient route search on hierarchical dynamic road networks. *Distrib. Parallel Datab.* 33, 2 (2015), 227–252.
[27] Jiajie Xu, Jing Zhao, Rui Zhou, Chengfei Liu, Pengpeng Zhao, and Lei Zhao. 2021. Predicting destinations by a deep learning based approach. *IEEE Trans. Knowl. Data Eng.* 33, 2 (2021), 651–666.
[28] Saijun Xu, Jiajie Xu, Rui Zhou, Chengfei Liu, Zhixu Li, and An Liu. 2020. TADNM: A transportation-mode aware deep neural model for travel time estimation. In *DASFAA'20 (Lecture Notes in Computer Science*, Vol. 12112*)*, Yunmook Nah, Bin Cui, Sang-Won Lee, Jeffrey Xu Yu, Yang-Sae Moon, and Steven Euijong Whang (Eds.). Springer, 468–484.
[29] Saijun Xu, Ruoqian Zhang, Wanjun Cheng, and Jiajie Xu. 2020. MTLM: A multi-task learning model for travel time estimation. *GeoInformatica* (2020), 1–17. DOI: https://doi.org/10.1007/s10707-020-00422-x
[30] Bin Yang, Chenjuan Guo, and Christian S. Jensen. 2013. Travel cost inference from sparse, spatio-temporally correlated time series using markov models. *PVLDB* 6, 9 (2013), 769–780. DOI: https://doi.org/10.14778/2536360.2536375
[31] Haitao Yuan and Guoliang Li. 2021. A survey of traffic prediction: From spatio-temporal data to intelligent transportation. *Data Sci. Eng.* 6, 1 (2021), 63–85. DOI: https://doi.org/10.1007/s41019-020-00151-z
[32] Jing Yuan, Yu Zheng, Chengyang Zhang, Xing Xie, and Guangzhong Sun. 2010. An interactive-voting based map matching algorithm. In *MDM'10*. 43–52. DOI: https://doi.org/10.1109/MDM.2010.14
[33] Nicholas Jing Yuan, Yu Zheng, Liuhang Zhang, and Xing Xie. 2013. T-Finder: A recommender system for finding passengers and vacant taxis. *IEEE Trans. Knowl. Data Eng.* 25, 10 (2013), 2390–2403. DOI: https://doi.org/10.1109/TKDE.2012.153
[34] Hanyuan Zhang, Hao Wu, Weiwei Sun, and Baihua Zheng. 2018. DeepTravel: A neural network based travel time estimation model with auxiliary supervision. In *IJCAI'18*. 3655–3661. DOI: https://doi.org/10.24963/ijcai.2018/508
[35] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI'17*. 1655–1661.
[36] Yuxuan Zhang, Senzhang Wang, Bing Chen, Jiannong Cao, and Zhiqiu Huang. 2021. TrafficGAN: Network-Scale deep traffic prediction with generative adversarial nets. *IEEE Trans. Intell. Transport. Syst.* 22, 1 (2021), 219–230.

Received October 2020; revised April 2021; accepted May 2021