软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*,2019,30(3):770−783 [doi: 10.13328/j.cnki.jos.005683]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

# 时空依赖的城市道路旅行时间预测*

施 晋, 毛嘉莉, 金澈清

(华东师范大学 数据科学与工程学院,上海 200062)

通讯作者: 毛嘉莉, E-mail: jlmao@dase.ecnu.edu.cn

**摘 要**: 城市道路的旅行时间预测,对于路径规划以及交通管理至关重要.尽管旅行时间预测会受路段依赖、时空相关性以及其他因素的影响,但现有的方法并未考虑如何结合外部因素进行建模,因而可能会有引入错误信息、路段建模时忽略上下游路段间的依赖关系等问题,导致预测精度较差.鉴于此,提出了两阶段的旅行时间预测框架:首先,使用 Skip-Gram 模型对轨迹数据地图匹配后的路段序列进行编码,将其映射为低维向量,通过该编码方式避免引入错误信息的同时保留了路段间的上下游依赖信息.随后,基于路段编码模式整合天气、日期等外部因素,设计了基于深度神经网络的城市道路旅行时间预测模型.基于真实出租车轨迹数据集的对比实验结果表明,所提方法比对比算法具有更高的预测精度.

**关键词**: 旅行时间预测;路段编码;长短期记忆网络;时空依赖

**中图法分类号**: TP311

中文引用格式: 施晋,毛嘉莉,金澈清.时空依赖的城市道路旅行时间预测.软件学报,2019,30(3):770−783. http://www.jos.org.cn/1000-9825/5683.htm

英文引用格式: Shi J, Mao JL, Jin CQ. Travel time prediction for urban road based on spatial-temporal dependency. Ruan Jian Xue Bao/Journal of Software, 2019,30(3):770−783 (in Chinese). http://www.jos.org.cn/1000-9825/5683.htm

## Travel Time Prediction for Urban Road Based on Spatial-temporal Dependency

SHI Jin, MAO Jia-Li, JIN Che-Qing

(School of Data Science and Engineering, East China Normal University, Shanghai 200062, China)

**Abstract**: Travel time prediction is critical for route planning and traffic monitoring. Due to complex relationships among road segments, spatial-temporal dependency, and other factors, it is challenging to perform modeling upon trajectory dataset. Without incorporating external factors into modeling, existing methods may import incorrect information and ignore road segment dependence, which results in poor prediction accuracy. A two-phase travel time prediction framework is proposed to solve the mentioned issues. During the first stage, trajectory data are mapped to a sequence of segments to generate a low-dimensional vector, which avoids introducing incorrect information while preserving the road segment dependence. During the second phase, after integrating road segment encoding and external factors such as weather and date, a travel time prediction model based on deep neural network is designed. The detailed experimental results on a real-world taxi trajectory dataset show that the proposed method is more accurate than existing methods.

**Key words**: travel time predition; road segment encoding; long short term memory network; spatial-temporal dependency

随着机动车保有量的激增,城市交通拥堵的状况加剧恶化,从而导致出行效率低下、资源浪费、空气污染等一系列问题,城市治堵已刻不容缓.合理引导出行路线可以在一定程度上缓解这一问题.随着定位技术的发展

* 基金项目: 国家自然科学基金(61702423, 61532021, U1501252, 61402180); 国家重点研发计划(2016YFB1000905)

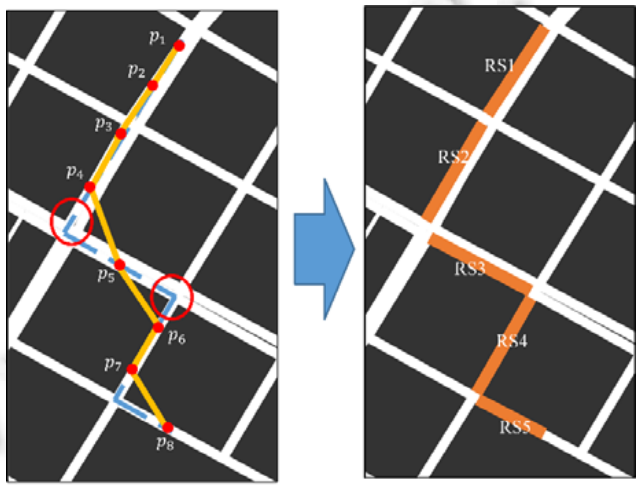
Foundation item: National Natural Science Foundation of China (61702423, 61532021, U1501252, 61402180); National Key Research and Development Program of China (2016YFB1000905)

本文由智能数据管理与分析技术专刊特约编辑樊文飞教授、王国仁教授、王朝坤副教授推荐.

收稿时间: 2018-07-17; 修改时间: 2018-09-20; 采用时间: 2018-11-01

与导航软件的普及,越来越多的司机依赖导航软件来规划出行路线.旅行时间预测,即预测行驶路线的实际通行时间,可以帮助司机制定行程,避开拥堵路段,在交通管理、拼车、车辆派单等应用中具有重要意义.

定位技术的普及产生了移动对象的海量轨迹数据,例如出租车、公交车等车辆的行驶轨迹.最直接的旅行时间预测方法是通过匹配相似历史轨迹数据来预测给定路线的旅行时间.此算法又可分为两类:一类是直接通过历史轨迹预测整段轨迹,但是易受到轨迹偏态分布性的影响,当综合天气、节假日等因素时,部分待预测轨迹可能无法找到可匹配的近似轨迹;另一类是将轨迹编码为路网上连续的子路段,通过分别预测单个路段的旅行时间来估算给定路线的时间开销,如图 1(b)所示.将轨迹映射到路段后,如果每个路段有足够多的轨迹,可以有效缓解轨迹稀疏的问题.但是对各路段单独预测会忽略了路段间上下游的信息,因此可能在一定程度上放大误差.




(a) 轨迹点编码 (b) 路段编码

Fig.1

图 1

车辆的旅行时间容易受到多重因素影响,包括路段依赖、时空相关性及其他外部因素.

1) 路段依赖主要由交通信号控制引起.例如,当行驶路线中存在绿波带控制的路段时,旅行时间会大幅减小;此外,在路口右转的等待时间比直行等待时间短;
2) 时空相关性体现在城市的拥堵路段,工作日早晚高峰的拥堵路段区域不同;节假日期间的拥堵路段区域与工作日的也不同;
3) 其他外部因素包括天气状况、司机的驾驶习惯、实时路况信息等.

由于上述因素之间相互作用关系复杂,因此旅行时间预测一直具有挑战性.鉴于轨迹数据是不定长度的时间序列,传统方法难以有效整合上述因素与轨迹数据.神经网络模型是一种有效的方法.长短期记忆网络(long short term memory network,简称 LSTM network)在每个时间点共享参数,适于处理任意长度的时间序列.Wang 等人[1]使用 LSTM 提取轨迹的时间依赖,并结合外部因素建模预测旅行时间.但是该方法依然存在不足:如图 1(a)所示,该算法将原始轨迹点均匀采样成等距离的轨迹点作为输入,可能会丢失部分路口的转弯信息.例如,$p_4 \rightarrow p_5 \rightarrow p_6$ 将被视作一条道路,从而在建模过程中引入错误的特征.

鉴于此,本文提出了基于路段编码优化的深度旅行时间预测框架(road-segment encoding deep travel time estimation framework,简称 REDTTE).REDTTE 框架包括两个阶段.

(1) 针对使用原始轨迹(以经纬度坐标表示)作为输入而引起误差的问题,拟通过对路段建模并使用路段向量映射(road segment vectorization,简称 RSV)模型将路段映射到低维向量中,保留路段间上下游依赖关系;

(2) 针对不定长序列数据难以处理以及外部特征较难引入的问题,结合神经网络模型能够处理不定长序列并且能够捕捉轨迹时空特征的优势,设计了长短期记忆网络和卷积神经网络(convolutional neural network,简称 CNN)的混合模型对(LSTM-CNN-LSTM,简称 LCL)预测旅行时间.

本文的贡献主要有以下几点.

(1) 受到词嵌入(word embedding)技术的启发,设计了 RSV 模型,融合神经网络语言模型的处理思路至路段编码,将路段映射到低维向量保留路段间的依赖关系.映射过程中,使用 Skip-Gram 模型对路段序列进行学习,确保具有上下游关系的路段向量间距离较近,无上下游关系的路段向量距离较远.本文首次通过将深度表征学习引入改进路段的编码方式,并将其应用于旅行时间的预测;

(2) 基于路段向量的编码模式,设计了 LCL 模型用于预测旅行时间.该模型通过输入组件将路段向量与其他外部特征(天气、节假日)进行整合,通过 LSTM 与 CNN 的混合神经网络分别提取时间、空间特征,在数据允许的情况下,还能将路况信息、道路限速信息、车辆型号等信息引入模型,提升预测精度;

(3) 结合 RSV 与 LCL 模型,提出了旅行时间预测框架 REDTTE.通过使用出租车的轨迹数据训练 REDTTE 框架,验证了该框架的有效性.此外,基于真实数据集合的对比实验结果表明,本文所提方法比对比算法的预测精度更高.

本文第 1 节介绍基于深度学习理论的轨迹序列处理技术及旅行时间预测的相关工作.第 2 节形式化定义相关概念,并介绍 REDTTE 的总体框架.第 3 节详细介绍框架中两个模型.第 4 节介绍本文所提方法在出租车轨迹数据集上的对比实验结果.第 5 节总结全文并指出未来的工作思路.

## 1 相关工作

本节回顾基于深度学习理论的轨迹序列处理方法以及现有的旅行时间预测方法.

### 1.1 基于深度学习的轨迹序列处理技术

深度学习已广泛应用在图像处理、自然语言处理等多个领域.近年来,不少研究工作利用深度学习理论处理轨迹数据,其中最为普遍的是卷积神经网络与循环神经网络(recurrent neural network,简称 RNN).

CNN 通过卷积、池化等操作,可以提取网格结构数据中的空间特征[2].部分工作将轨迹数据转换为网格数据使用 CNN 进行处理,有效提升了路段行驶速度与人流量预测的精度[3,4].RNN 是专门处理时间序列数据的神经网络模型[5].Dong 使用 RNN 构建了一个自编码器用于提取轨迹序列中的时间依赖特征,并分析司机的驾驶行为,在司机数量估算和轨迹分类上取得了较高的准确率[6].然而,传统的 RNN 模型仅由一个隐层记录历史信息,当输入序列过长时会产生梯度消失以及梯度爆炸问题[7],导致难以处理长时间的依赖关系.LSTM 模型通过引入记忆单元存储相关的历史数据,有效缓解了难以捕捉长期依赖的问题[8].Song 使用多层 LSTM 网络模型提取轨迹数据上的时间依赖关系,用于预测城市的出行模式[9].

### 1.2 旅行时间预测

早期的旅行时间预测工作主要基于线圈传感器所采集的数据,记录车辆通过一个路段的行驶时间进行建模,来预测车辆的旅行时间[10−12].然而,由于线圈传感器仅仅部署在城市的部分路段上,这类算法难以预测整个城市路网车辆的旅行时间.现有的工作大多使用浮动车的历史轨迹对旅行时间进行预测,根据轨迹数据的处理方式不同,可以划分为基于路段的旅行时间预测与基于路径的旅行时间预测.

- 基于路段的旅行时间预测:将轨迹切分成路段,可以缓解轨迹稀疏带来的影响.Jenelius 针对低采样率的轨迹数据使用概率建模,并用极大似然估计进行参数估计.考虑到外部因素的影响,模型将旅行时间视为一个多变量的正态分布[13].Yang 通过时空相关的隐马尔可夫模型对路段的时空依赖关系进行建模,对路段的旅行时间分布进行预测[14].该类算法未考虑结合多源的外部特征(如天气、日期等)来提高算法的预测精度;
- 基于路径的旅行时间预测:Rahmani 根据浮动车的历史轨迹数据提出了一种非参数的方法,通过对历

史轨迹的时间加权作为旅行时间估计结果[15].然而,直接基于历史轨迹时间估计旅行时间会面临轨迹稀疏问题.当待预测的行驶路线缺少与其对应的历史轨迹时,该方法难以给出一个精确解.部分研究工作使用 $k$ 近邻搜索和加权平均的思想,通过查找相似的轨迹来减轻轨迹稀疏带来的影响[16,17].Wang 等人使用张量对映射到路段的轨迹数据进行建模,同时,利用张量分解重建技术引入路段间上下文关系的特征[18].

基于树的模型可以整合多源的外部特征,Lam 使用梯度提升决策树(gradient boosting decision tree,简称 GBDT)、随机森林(random forest,简称 RF)等多种树模型预测旅行时间[19].Wang 利用树模型整合外部特征,结合时空相关的概率图模型预测旅行时间[20].这两个方法提高了预测精度,但因轨迹序列的长度不固定,传统的机器学习模型难以处理这类数据.同时,轨迹数据是时空相关数据,这些模型不能有效提取数据中的时空依赖特征.

Wang 等人提出了 DeepTTE 模型,结合 CNN 和 LSTM 提取轨迹数据的时空特征,并且整合 Attention 机制预测旅行时间[1].然而,由于该方法直接使用原始轨迹数据进行建模,在训练过程中无法学习到路网的相关特征,容易引入错误特征.可以通过将轨迹映射到路段的方式引入路网特征,然而路段如果采用序号编码无法反映路段间的上下游关系,需要对其进一步编码.这与自然语言处理中的语言模型的处理思路类似.早期的自然语言处理中,大多采用 one-hot 编码对单词进行表示.由于这一方法存在的巨大缺陷,Bengio 提出使用三层神经网络对语言模型进行建模[21],Collobert 使用神经网络语言模型生成词向量,并在词性标注、短语识别、语义角色标注等任务中表现良好[22].为了提升神经网络语言模型在大型语料库中的效果和性能,Mikolov 提出了 Skip-Gram 模型[23],并推动了词嵌入技术在这一领域的发展.由于基于负采样的 Skip-Gram 模型能够快速训练,并且在大规模的数据集中具有较好的编码效果,本文考虑将其引入用于路段编码的表示优化.在此基础上,采用 LCL 模型通过 LSTM 模型处理路段向量这一时间序列数据,并提取路段间的深层依赖关系,相比 DeepTTE 需先通过 CNN 提取轨迹点的转弯行驶等信息,本文的方法不会放大轨迹点的定位误差,因而有效提升了旅行时间预测精度.

## 2 概念定义与整体框架

### 2.1 概念定义

轨迹数据通过采集车辆行驶过程中的 GPS 数据获得.

**定义 1(轨迹)**. 一条轨迹 $Tra$ 是带有时间属性的位置点序列,表示为 $Tra=\{p_1,p_2,\ldots,p_m\}$,其中,每个位置点包含经、纬度信息以及记录该位置点的时间戳,$p_i=(lng_i,lat_i,time_i)$.

由于轨迹数据以一定的时间间隔采集,且 GPS 定位技术本身也存在误差,仅凭车辆的轨迹数据难以还原车辆真实的行驶情况.在对轨迹数据的预处理阶段,常结合路网数据对原始轨迹进行地图匹配,将车辆轨迹映射到路网中的真实路段.

**定义 2(路网)**. 一个城市的路网以有向图表示,即 $G=(V,R)$,其中:$V$ 为顶点集合,表示道路交叉口;$R=\{r\}$为连接顶点之间路段的集合,其中,每个路段 $r$ 包含路段标识符信息 $r.id$ 以及路段长度 $r.dis$.

将轨迹映射到路段后,轨迹点依次经过的路段序列可用来描述整条轨迹的信息.

**定义 3(路段序列)**. 一个路段序列 $RS$ 是通过地图匹配将历史轨迹序列 $Tra$ 映射到一个有序路段序列 $RS=\{r_1,r_2,\ldots,r_n\}$,其中,在一个路段上的连续轨迹点被映射到同一个路段 $r$.

如果使用路段序列对轨迹进行编码,各路段由其对应的 $id$ 标识,此编码丢失了路段之间的上下游关系,同时,也难以将其作为模型的输入进行训练.所以,我们使用路段向量序列对输入轨迹进行编码.

**定义 4(路段向量序列)**. 路段向量集合是通过一定的映射方式 $f$ 将路网中路段集合 $\{r\}$ 映射到一个向量集合 $\{rv\}$;而路段向量序列 $RV$ 则是由路段序列 $RS$ 对各路段使用 $f$ 映射获得的向量序列,表示为 $RV=\{\boldsymbol{rv}_1,\boldsymbol{rv}_2,\ldots,\boldsymbol{rv}_n\}$.

将路段向量序列与其他外部特征使用模型进行融合作为输入特征,并以该段轨迹的旅行时间作为标签,旅行时间预测问题可以转换为一个监督学习的问题.

**定义 5(旅行时间预测)**. 给定一段轨迹 $Tra=\{p_1,p_2,\ldots,p_m\}$,对这段轨迹的旅行时间 $T(T=p_m time-p_1 time)$进行预测,同时确保预测值 $T$ 与真实值间的误差最小化.此外,轨迹数据的时间戳仅用于结果的验证,在预测过程中会

将其去除.

### 2.2 总体框架

REDTTE 框架包括 3 部分:轨迹预处理、模型训练、在线预测,如图 2 所示.

在轨迹预处理阶段,通过地图匹配,将历史轨迹数据映射为路段序列,该序列将作为 RSV 模型训练的输入;

在模型训练阶段,主要针对 RSV 以及 LCL 模型进行两阶段训练.第 1 阶段的训练过程中,将轨迹序列输入到 RSV 模型进行训练,使得路网中每一个道路 *id* 映射到一个低维向量.RSV 模型训练完毕后,对路段序列所有路段使用该模型进行映射;第 2 阶段训练 LCL 模型的 3 个组件:首先,通过特征嵌入方式将路段向量与天气、节假日等外部特征相结合,作为模型的输入;输入组件对输入特征进行整合后,将输入的特征序列通过深度特征提取组件的LSTM和CNN模型获取时空相关特征;最后,由LSTM和均匀池化组成的预测组件预测旅行时间进行;根据预测结果与真实值的误差,LCL 模型的训练可通过反向传播调整模型参数;

在线预测阶段,使用训练好的模型参数进行预测.可以直接输入预测轨迹,也可以使用路段序列作为输入.除轨迹数据以外,预测阶段同样需要输入外部特征.天气和节假日等外部特征可以通过天气预报等信息提前获得.REDTTE 框架输入的序列数据本质上是路段数据,因此,REDTTE 框架也可以结合路况信息作为外部特征来提升预测精度.

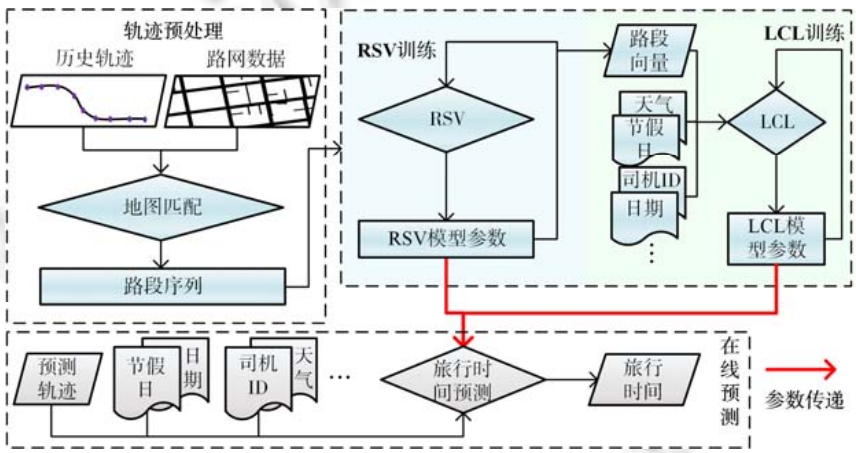



Fig.2 REDTTE's global architecture

图 2 REDTTE 整体框架图

## 3 模型概述

### 3.1 路段向量映射

将历史轨迹通过地图匹配方式转换为路段序列 $RS=\{r_1,r_2,\ldots,r_n\}$,其中每个路段有对应的路段序号 *r.id* 以及长度 *r.dis*.由于仅靠路段序号无法辨识路段之间的上下游依赖关系,该编码方式在一定程度上丢失了重要轨迹的信息.同时,若直接将序号作为模型的输入,模型可能会认为序号相近的路段具有上下游关系,而这与事实不符.因此,需要一种方法将路段映射到低维的向量中.

one-hot 编码是一种简单的映射方法.如后文图 4 所示,输入一个 $L$ 维的向量,$L$ 为类别的数量,其中大部分项值为 0,只有与序号对应的项值为 1,表示当前路段.但这一编码方式无法描述相邻路段关联性.另一种编码方式使用每个路段中心点的经、纬度坐标来表示一个路段,但是该编码无法识别双向车道的上下游路段.

词嵌入技术将词汇映射到低维向量,语义相近的词所对应的向量的距离较近,这与本任务的目标较为贴切,因此我们在 RSV 模型中引入基于负采样 Skip-Gram 模型对路段进行映射,其主要思想是:对于一个输入路段,

预测其前后 $T$ 个路段.这实质上是一个多标签(multi-label)问题,可以通过负采样生成负样本的方式转变成回归问题.

首先,对路段序列使用滑动窗口模型生成训练样本,如图 3 所示.对一个路段序列使用滑动窗口技术,当窗口指针指向 $r_4$ 向量,窗口大小为 2 时生成 4 个正样本.

随后构造中心路段的伪邻居路段作为负样本,负样本的生成根据每个路段出现的频次加权采样生成,出现频率越高的路段越容易被采样生成负样本.路段被采样到的概率如下:

$$P(r_i)=\frac{freq(r_i)+1}{\sum_{i=1}^{L} freq(r_i)+L} \tag{1}$$

其中,$freq(r_i)$函数用于统计路段出现的频次,这里引入了拉普拉斯修正防止部分路段因缺失数据而不被采样.

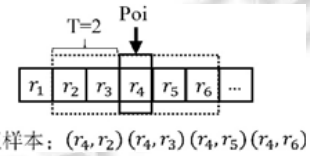



Fig.3 Example of sliding window model

图 3 滑动窗口模型示意图

样本打分模型应能根据映射到的隐层信息,对正负样本给予正确的判断,即判断出正确的具有上下游关系的路段,因此需要模型对输入的样本对进行打分.

图 4 为样本打分模型,主要分为输入层、隐层和输出层.输入层为中心路段和邻近路段(或伪邻近路段)的 one-hot 编码,分别为 $r^c$ 和 $r^n$.隐层 $\boldsymbol{h}$ 通过输入层和矩阵 $\boldsymbol{W}_{LH}$ 计算获得,即 $\boldsymbol{h}=f(r)=\sigma(\boldsymbol{W}_{LH}\boldsymbol{x}+\boldsymbol{b})$,其中:$\boldsymbol{b}$ 为该层的偏置向量;$\sigma$为激活函数,通常使用 *sigmoid* 函数$\sigma(x)=1/(1+e^{-x})$.中心路段和邻近路段分别采用不同的参数映射到隐层,即 $\boldsymbol{W}_{LH}^{c},\boldsymbol{b}^{c}$ 和 $\boldsymbol{W}_{LH}^{n},\boldsymbol{b}^{n}$ .通过矩阵将输入映射到隐层向量后,对两个隐层向量进行按元素相乘操作,并通过激活函数限制其大小,最后对向量进行求和,得到一个分数 $y_{score}$.模型训练完成后,中心路段的隐层为所求的低维向量,而输入层到隐层的映射 $f$ 就是所求的映射.

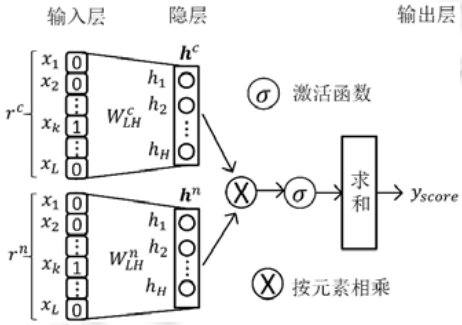



Fig.4 Scoring model

图 4 样本打分模型

RSV 模型同时对输入的正负样本进行打分,并控制其比例为 1:$k$.每次迭代,对输入的一个正样本对和 $k$ 个负样本对同时使用打分模型进行打分.打分模型的输入为$(r_i,r_j)$,模型需要最大化正样本分数的同时最小化负样本的分数,目标函数为

$$\max L=\prod_{r_i\in\{r\}}\prod_{r_j\in(Nei(r_i)\cup Neg(r_i))} P(r_j\mid r_i) \tag{2}$$

其中,$Nei(r_i)$表示 $r_i$ 的相邻路段,$Neg(r_i)$表示对 $r_i$ 负采样生成的所有路段.条件概率表示为

$$P(r_j \mid r_i)=\begin{cases}\sigma(f(r_i)^T g(r_j)), & r_j \in Nei(r_i)\\ 1-\sigma(f(r_i)^T g(r_j)), & r_j \in Neg(r_i)\end{cases} \tag{3}$$

$f$ 为将道路 $id$ 映射到路段向量的函数,$g$ 为辅助函数,两函数映射后的向量维数相同,$\sigma(x)$为 $sigmoid$ 函数.可以看出,目标函数存在大量的连乘.为便于计算,这里对 $L$ 取对数,目标函数变为

$$\max L=\sum_{r\in\{r\}}\sum_{r_j\in(Nei(r_i)\cup Neg(r_i))}\log(P(r_j \mid r_i)) \tag{4}$$

目标函数可以通过梯度上升进行优化,因此,打分函数的参数 $\boldsymbol{W}_{LH}^{c},\boldsymbol{b}^{c},\boldsymbol{W}_{LH}^{n},\boldsymbol{b}^{n}$ 可以通过反向传播进行训练.

通过将历史轨迹的路段向量通过 RSV 模型进行训练,可以得到一个 $f$ 将道路 $id$ 映射到道路向量 $\boldsymbol{rv}_i$ 中.通过对映射到向量空间的路段向量进一步观察发现:具有上下游关系的路段映射到向量空间后,向量间的距离比没有上下游关系的路段向量更近.因此,通过 RSV 编码能够很好地保留路段间的上下游依赖信息.

### 3.2 混合神经网络

LCL 模型的框架如图 5 所示,模型分为 3 个组件:输入组件主要用于结合外部特征、路段序列及其统计量(如出发时间、路段长度等)生成下一阶段模型的输入;深度特征提取组件主要用于提取输入数据中的时空特征;预测组件根据提取出的高维度特征,结合 LSTM 模型和均匀池化层对旅行时间进行预测.

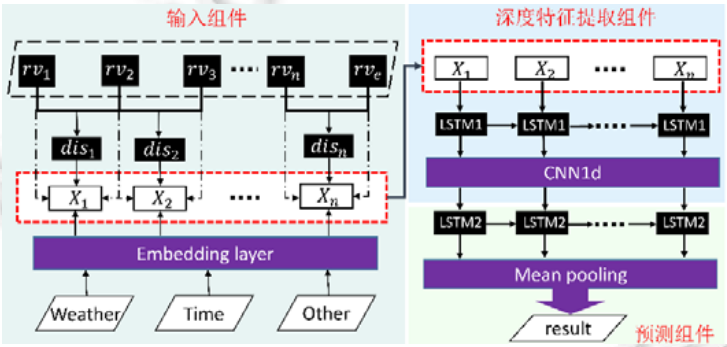



Fig.5 LCL architecture

图 5 LCL 模型框架图

#### 3.2.1 输入组件

前面使用 RSV 模型将每个路段 $id$ 映射到一个路段向量中,因此,将每个路段序列映射为路段向量序列 $\boldsymbol{RV}=\{\boldsymbol{rv}_1,\boldsymbol{rv}_2,\ldots,\boldsymbol{rv}_n,\boldsymbol{rv}_e\}$作为特征组件的部分输入.其中,为了标志轨迹序列的结尾,帮助模型对轨迹终点的判断,引入零向量 $\boldsymbol{rv}_e$ 作为每条轨迹的终止向量.由于交叉路口的特征通常由两个路口进行表示(例如图 1(b)中的 $RS_2\rightarrow RS_3$ 表示左转,$RS_3\rightarrow RS_4$ 表示右转),因此,这里将两个路段向量合并为一个向量作为输入,即 $\boldsymbol{RV}_{combine}=\{(\boldsymbol{rv}_1,\boldsymbol{rv}_2),(\boldsymbol{rv}_2,\boldsymbol{rv}_3),\ldots,(\boldsymbol{rv}_n,\boldsymbol{rv}_e)\}$,合并后的向量长度与输入的向量序列长度一致(为 $n$).考虑到轨迹在不同路段的行驶距离是一个重要因素,尤其对于起始路段以及终止路段.起始及终止路段通常位于道路的中间部分,直接使用路段长度会增大模型估计的旅行时间.鉴于此,输入组件计算目标轨迹在每个路段上的长度 $dis_i$,并将其作为特征输入.

模型的输入组建将一些外部因素(如天气、节假日等)与路段向量相结合,其中,天气数据包含最高温、最低温、天气状况.与此同时,由于旅行时间预测的请求通常带有出发时间,而出发时间是一个很重要的影响因素,考虑将一天划分成 28 800 个时间槽作为特征输入,每个时间槽对应了 3s 的时间间隔.由于城市的交通状况具有一定的周期性,通常周期为一周[16],因此引入每周的天数作为输入的特征.

上述特征除了最低温度和最高温度外都是类别变量,不能直接将其作为 LSTM 的输入,采用 Gal 的特征嵌入方法将类别特征转换为低维向量[20].类别特征 $c\in[C]$通过参数矩阵 $\boldsymbol{W}_{CE}$ 映射到空间 $R^E$ 中.其中,$C$ 为类别的数

量,$E$ 为低维向量的维数,通常情况下,$E<<C$.可以发现,这种类别特征的处理方式类似于 RSV 模型.但是需要注意,RSV 模型的映射向量单独通过路段序列进行训练,而类别映射的参数矩阵的相关参数随整个模型一起训练.

输入组件最后将以上特征进行整合,作为模型组件的输入,表示为 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_n\}$,其中,$\boldsymbol{X}_i=\{\boldsymbol{rv}_i,\boldsymbol{rv}_{i+1},dis_i,weather,time,\ldots\}$.此外,司机画像、车辆型号等信息同样可以通过特征嵌入的方式进行引入,实时的路况信息和道路限速信息的引入方式与路段长度的引入方式一致.

3.2.2 深度特征提取组件

深度特征提取组件由两层模型组成:第 1 层循环神经网络通过特征提取的方式挖掘路段向量之间的隐含关系,例如左转、右转、直行等特征;第 2 层卷积层能够挖掘更高层次的路段依赖关系,同时提取出更丰富的特征,如绿波带.

鉴于使用路段向量对轨迹进行编码,获得的路段向量序列长度是不固定的,为提取路段向量之间的前后依赖,考虑使用循环神经网络提取这部分的依赖特征.

LSTM 的整体结构如图 6 所示,它包括输入层、隐层和输出层,每个时间点共享同一个参数.当前时间点的隐层是由上一个时间点的隐层和当前时间点的输入所决定的.这样的结构使 LSTM 能够保存前序时刻输入的特征,从而提取输入数据中的时间依赖关系,并处理不定长度的输入序列.由于隐层的参数受到当前输入和前一个时间点隐层的共同影响,因此无法通过普通的反向传播算法进行训练,采用随时间的反向传播算法(back propagation through time,简称 BPTT)对模型进行训练.

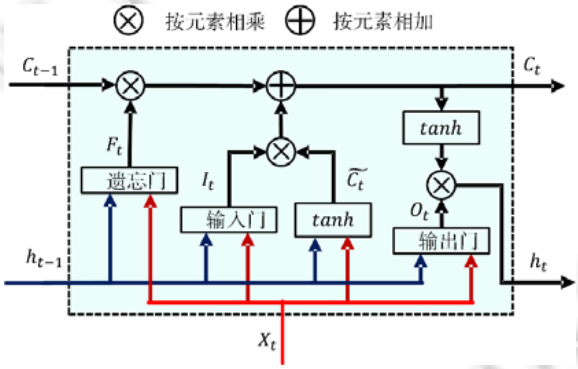



Fig.6 LSTM structure

图 6 LSTM 网络结构

图 6 是 LSTM 的内部结构图,与 RNN 相比多了一个记忆单元 $C_t$,用于记忆历史信息.LSTM 的内部结构主要由遗忘门、输入门、输出门组成.

- 遗忘门用于在原始记忆单元中丢失部分信息,根据上一时间点的隐层信息以及当前的输入,通过激活函数输出遗忘元素的概率:

$$F_t=\sigma(W_fX_t+W_fh_{t-1}+b_f) \tag{5}$$

- 输入门通过类似的结构筛选出输入数据中需保存至记忆单元的数据,并将其与通过遗忘门之后的记忆单元的数据相加作为记忆单元的输出:

$$I_t=\sigma(W_iX_t+W_ih_{t-1}+b_i) \tag{6}$$

$$\tilde{C}_t=\tanh(W_CX_t+W_Ch_{t-1}+b_C) \tag{7}$$

$$C_t=F_t\times C_{t-1}+I_t\times\tilde{C}_t \tag{8}$$

- 最后,隐层的数据通过输出门的数据与新的记忆单元的数据进行逐点相乘计算得到:

$$O_t=\sigma(W_OX_t+W_Oh_{t-1}+b_O) \tag{9}$$

$$h_t=O_t\times\tanh(C_t) \tag{10}$$

在实际训练过程中,将特征组件得到的特征序列 $\boldsymbol{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_n\}$依次输入到第 1 层的 LSTM 中,并将隐层数据 $\boldsymbol{h}^1=\{\boldsymbol{h}_1^1,\boldsymbol{h}_2^1,...,\boldsymbol{h}_n^1\}$ 提取出来作为下一层模型的输入.

通过 LSTM 提取上下路段间的时间依赖后,LCL 模型通过卷积神经网络挖掘多个路段间的依赖关系来丰富特征的多样性,同时挖掘出路段间更深层次的联系(如绿波带等信息).

考虑将卷积层引入,用以提取空间特征.由于输入的隐层序列是一维的序列,因此使用一维的卷积核对序列进行卷积操作.卷积的操作如图 7 所示,以大小为 $kn$=4 的卷积核为例,其中,填充向量 $\boldsymbol{pad}$ 用于保证输出序列的长度和输入序列的长度一致,个数为 $kn$–1.我们加入了与输入的隐层 $\boldsymbol{h}^1$ 同维数填充零向量,卷积层的输入为 $\boldsymbol{X}^c=\{\boldsymbol{X}_1^c,\boldsymbol{X}_2^c,...,\boldsymbol{X}_{n+kn-1}^c\}$,那么其输出为 $X^{L2}=\{X_1^{L2},X_2^{L2},...,X_n^{L2}\}$,其中,每个元素表示为

$$\boldsymbol{X}_j^{L2}=\sum_{i=0}^{kn}\sigma(K_i\times X_{j+i}^c+b_k) \tag{11}$$

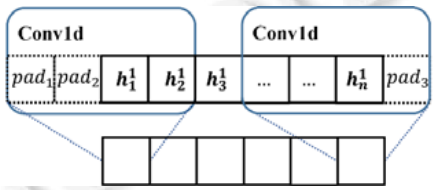



Fig.7 Conv1d diagram

图 7 Conv1d 示意图

3.2.3 预测组件

LSTM 模型不仅可以用于特征提取,其输出同样可以用于回归、分类等任务.在预测组件中,模型将上一层卷积层的输出作为第 3 层 LSTM 的输入.为了在旅行时间的预测过程中考虑到所有的路段,该层 LSTM 依旧使用与第 1 层的 LSTM 一致的多对多网络结构,并将隐层的信息作为均匀池化层的输入.

由于最后输出的序列长度与输入的路段序列长度相同,但旅行时间预测的目标是一个实值,因此使用均匀池化将 LSTM 隐层的输出映射为一个实值作为最终预测的旅行时间:

$$\hat{y}=sum(\boldsymbol{h}_{out})/len(\boldsymbol{h}_{out}) \tag{12}$$

3.2.4 损失函数

鉴于长度越长的轨迹预测误差时间越大,直接使用 *MAE* 作为损失函数会使模型更加偏重于旅行时间较长的轨迹.为保证模型同时优化短途和长途的轨迹,采用相对百分误差的绝对值(mean absolute percentage error,简称 MAPE)作为模型的损失函数:

$$MAPE(y,y_{pred})=\frac{1}{N}\sum_{i=0}^{N}\frac{|y_i-\hat{y}_i|}{y_i} \tag{13}$$

根据损失函数提供的梯度信息,模型参数可以通过反向传播不断学习调整.

## 4 实 验

为了验证 REDTTE 框架的有效性,本文基于出租车轨迹数据集进行对比实验以及各模块的有效性验证实验.通过与 AVG,KNN 等传统旅行时间预测算法的对比实验,验证深度学习模型处理旅行时间预测问题的优越性.通过与 DeepTTE 算法的对比实验来验证 REDTTE 模型结构的有效性.此外,为验证路段向量编码以及深度特征提取组件的有效性,分别执行了与 PLCL(point LCL)及 LC 算法的对比实验.

### 4.1 实验设置

4.1.1 数据集

- 轨迹数据:本文使用滴滴开放的盖亚计划(https://outreach.didichuxing.com/research/opendata/)数据集.

该数据集提取成都 2016 年 11 月 30 天的车辆轨迹数据,包含 10 亿 GPS 坐标点、400 多万条轨迹.由于数据集每天采取不同的哈希函数对司机 ID 进行脱敏,因此,实验中无法使用司机 ID 作为外部特征;

- 路网数据:为了保证数据的一致性,路网数据使用成都市 2016 年的 OpenStreetMap(https://www.openstreetmap.org)数据,对实验数据进行地图匹配后获得 5 856 条路段;
- 外部因素:成都市的天气数据通过爬虫抓取,包含天气状况(晴天、雨天等 4 种天气状态)、最低气温与最高气温等信息.

4.1.2 实验环境

本文使用 python 语言完成全部实验的编写,并使用基于 python 的深度学习框架 Pytorch 0.3 框架实现了部分模型的编写.硬件环境为 160G 内存,两个 8 核的 Intel Xeon Silver 4110 处理器以及一个 NVIDIA Tesla K80 GPU.模型的训练与测试均在 GPU 上进行.

4.1.3 评测指标

为了更全面地评估框架预测旅行时间的效果,本文使用了 4 种评测指标:平均绝对值误差(mean absolute error,简称 MAE)、均方根误差(root mean square error,简称 RMSE)、*MAPE*(见公式(13))、*MAE*/*D*.

$$MAE(y,y_{pred})=\frac{1}{N}\sum_{i=0}^{N}|y_i-\hat{y}_i| \tag{14}$$

$$RMSE(y,y_{pred})=\sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i-\hat{y}_i)^2} \tag{15}$$

$$MAE/D(y,y_{pred})=\frac{1}{N}\sum_{i=0}^{N}\left|\frac{y_i-\hat{y}_i}{dis_i}\right| \tag{16}$$

由于 MAE 会受到路径长度的影响,即越长的路径旅行时间越长,因此误差可能就会越大.考虑将 *MAE* 与轨迹长度的比值,即 *MAE*/*D*,作为衡量算法在每公里的预测误差时间.

4.1.4 参数设置

- RSV:考虑到部分路段长度较短,在采样的过程中轨迹可能会跳过这些路段,因此,基于实验数据集将滑动窗口大小设置为 3,正负样本的采样比为 1:5.映射到向量空间的维度设为 10 维,为加快训练速度,训练过程中采取了批梯度下降,批量设为 50,初始学习速率设为 0.025,样本训练一个周期;
- LCL:输入组件部分,将每周的天数映射到 3 维的向量,小时映射到 10 维的向量,时间槽映射到 40 维的向量.深度特征提取组件的 LSTM 的隐层维数为 84,CNN 的通道数为 36,卷积核大小为 4.预测组件的 LSTM 与上一层的 LSTM 参数一致.

为加速模型的训练,我们对数据集进行分批训练,模型的批量为 64,初始的学习速率为 0.002,模型训练 10 个周期.

4.1.5 对比算法

- Avg 算法是作为旅行时间预测对比的一种基准算法,算法将一天以 10 分钟的间隔划分成 144 个时间槽.通过计算轨迹长度与对应出发时间的历史轨迹平均速度的比值作为旅行时间.在计算过程中同样考虑到了每周的天数带来的影响,例如,周一的旅行时间会考虑历史的周一在该时间槽内出发的轨迹的平均速度;
- KNN 算法是一种非参数的算法,算法根据待查询轨迹的起点和终点查询与其起点和终点相近的历史轨迹,根据邻近轨迹与查询轨迹的出发时间和节假日、轨迹长度等信息,对邻近轨迹的旅行时间进行加权和作为最后预测的旅行时间.在实验过程中,选取了 $k$=10 作为具体的参数,当算法查询不到足够数量的邻近点时,会同时扩大起点和终点的搜索范围,直到找到足够的邻近轨迹;
- PLCL(point LCL)将 LCL 模型的输入由路段向量序列替换为每个路段的中心经纬度坐标点,其余的参数及特征设置与 LCL 一致;
- LC 为 LCL 模型的简化版本,LC 模型去掉了第 3 层的 LSTM,将卷积层的输出直接连接池化层,输出最

后的旅行时间.除此之外,其余参数特征与 LCL 一致;

- DeepTTE[1]是当前主流的旅行时间预测算法.算法使用经、纬度作为模型的输入,利用混合神经网络分别提取轨迹数据的时空依赖.同时,通过 Attention 机制以及特征嵌入的方式将外部因素引入.这里,我们使用算法开放的源代码与算法表现最优的参数,并在实验数据集下进行微调优化.模型使用的外部特征与 LCL 使用的外部特征一致.

### 4.2 实验效果

为了展示路段向量的表示效果,选取成都青羊区的部分区域进行可视化展示.图 8 所示为 RV1,RV2 两个路段向量的示意图.其中,横纵坐标分别表示经纬度.图中反映了该区域的路网情况,以颜色的深浅表示对应路段的路段向量与示意路段向量间的距离.黑色表示距离为零,颜色越淡表示距离越远.从图中可以看出:路段向量的表示方式保留了路段上下游的依赖关系,与目标向量具有上下游关系的路段,其路段向量距离较近,而不具有上下游关系的路段向量则距离较远.

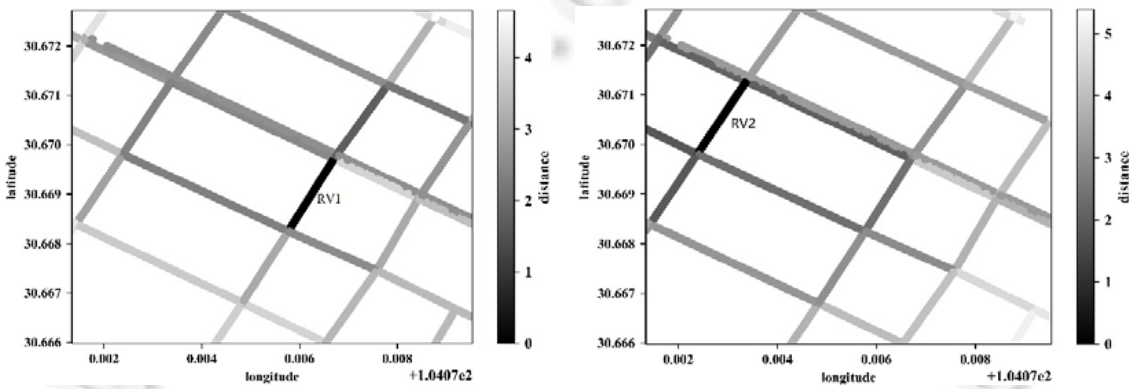



Fig.8 Schematic plot for road vector

图 8 路段向量示意图

为了验证算法的准确率,将数据集中前 23 天的数据作为模型的训练数据,后 7 天的数据作为验证数据.表 1 为算法及对比算法在该数据集下各项指标的预测结果,可以看出,本文提出的方法在各项误差指标中均拥有最低的误差.

**Table1** Error comparison

**表 1** 误差对比

| | MAPE (%) | MAE (s) | RMSE (s) | MAE/D (s/km) |
|---|---|---|---|---|
| AVG | 36.11 | 227.73 | 307.70 | 43.27 |
| KNN | 29.54 | 198.07 | 311.54 | 49.86 |
| PLCL | 14.86 | 101.63 | 155.14 | 21.04 |
| LC | 14.72 | 102.54 | 158.31 | 21.06 |
| DeepTTE | 14.80 | 109.45 | 194.79 | 30.03 |
| LCL | **13.66** | **92.50** | **142.12** | **19.29** |

通过 PLCL 和 LCL 的误差对比可以发现:将路段向量替换为经纬度坐标后,模型的误差值上升,这证明了 RSV 算法的有效性.同样,通过 LC 和其他算法的误差对比可以发现,直接使用深度特征提取组件的输出预测旅行时间具有较高的准确率.通过 LC 及 PLCL 与 DeepTTE 的对比可以发现:使用 REDTTE 的部分模型结构预测旅行时间具有较好的效果,甚至在部分指标的对比中优于 DeepTTE 的表现.

为了验证 DeepTTE,LCL,LC 及 PLCL 算法在不同影响因素下的敏感性以及预测精度,分别统计了不同轨迹长度以及不同出发时间对上述算法的影响.图 9 为不同轨迹长度对上述算法的影响,横坐标为轨迹的长度,纵坐标为 MAPE.可以发现:在路段长度较短时,真实的旅行时间往往较短,小的偏差会导致 MAPE 值增大.因此,上述

算法均在较短的轨迹上具有较大的 MAPE 值.

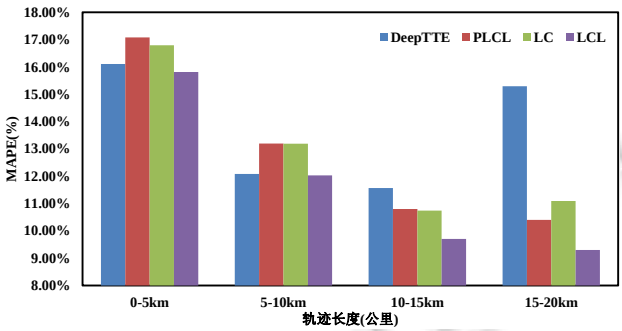



Fig.9 Influence of trajectory distance on each algorithm

图 9 轨迹长度对各算法的影响

为评测出发时间对各算法的影响,分别对比工作日与节假日的 MAE 和 MAPE 两个指标的算法效果.如图 10、图 11 所示,其中,横坐标是以小时为单位划分的时段,纵坐标为对应的各项指标值.

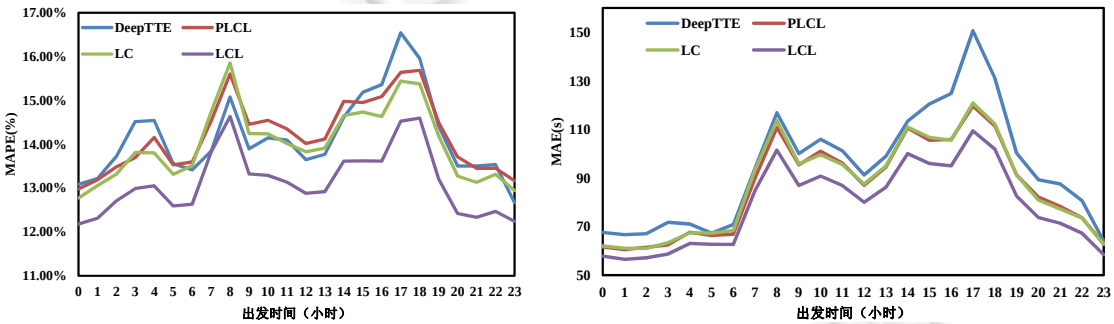



Fig.10 Influence of start time (weekday) on each algorithm

图 10 出发时间对各算法的影响(工作日)

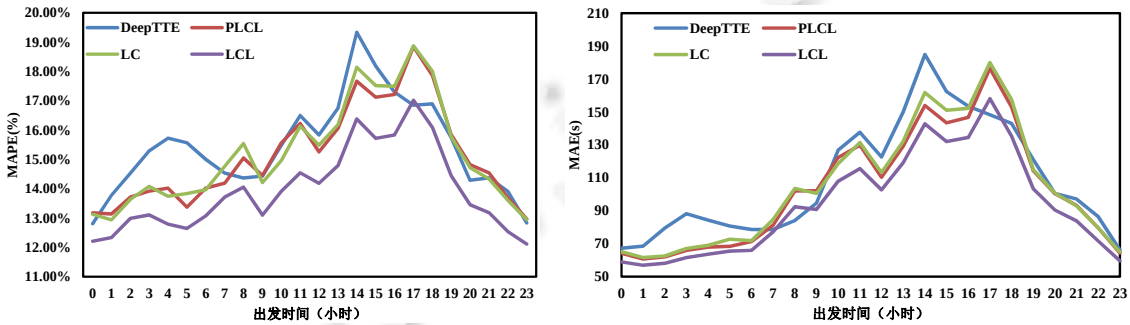



Fig.11 Influence of start time (weekend) on each algorithm

图 11 出发时间对各算法的影响(节假日)

可以看出:在工作日的早晚高峰期间,算法的误差普遍偏高.通过对早晚高峰期的数据分析,可以发现旅行时间在这一时间段拥有较高的方差,因此难以做到准确的预测.产生这一现象的主要原因是:早晚高峰期间道路情况多变,各模型因未引入路况信息作为外部特征,难以捕捉实时变化的道路拥堵状况.在节假日期间,由于路况复杂多变,各算法的误差要大于工作日的预测误差.

### 4.3 性能测试

旅行时间预测通常应用于实时性需求较强的环境中,因此其运行效率是一个重要的参考指标.为了验证算法的执行效率以及分布式场景下的可扩展性,通过在 GPU 上运行算法,设立了不同的批次模拟并行环境.同时,考虑到设备性能波动而产生的时间误差,实验分别在不同批次下对 100 万条轨迹进行查询,并统计其平均时间开销.查询实验的硬件环境与第 4.1.2 节的实验环境一致.

图 12 为算法在 64,80,96,112,128 批次下平均单条轨迹的查询时间效果图,横轴为批次大小,纵轴为单条轨迹的平均查询时间.从实验结果可以看出:算法对于单条轨迹的响应时间均为毫秒级,具有较高的实时性;并且随着批次的增多,平均单条轨迹的查询时间随着线性减少.这是因为算法在运行过程中仅需要进行路段序列的映射以及 LCL 模型的推断(inference),且整体模型可以离线存储至内存中.因此,算法能够在大规模分布式环境下进行拓展.

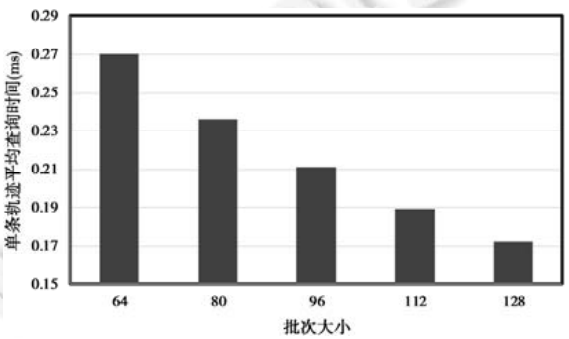



Fig.12 Average query time for a single trajectory on different batch size

图 12 不同批次下单条轨迹的平均查询时间

## 5 结束语

针对传统的旅行时间预测模型难以引入多源特征的问题,本文提出了一种两阶段的旅行时间预测框架 REDTTE.框架的第 1 阶段通过引入 Skip-Gram 模型将路段映射到向量空间,可以有效地捕捉路段之间的依赖关系;第 2 阶段针对路段向量序列的特性,整合天气日期(工作日/节假日)等外部特征,设计了深度模型用于预测旅行时间.基于海量轨迹数据的实验结果表明,REDTTE 框架因能较好提取路段间上下游依赖关系而具有较高的旅行时间预测精度.考虑到在节假日与工作日的早晚高峰时期模型的预测误差较高,未来拟通过对路况建模,预测未来路段序列中每个路段的速度信息,将其与路段向量共同输入以提高模型的预测精度.

**References**:

[1] Wang D, Zhang J, Cao W, *et al*. When will you arrive? Estimating travel time based on deep neural networks. In: Proc. of the 32nd AAAI Conf. on Artificial Intelligence. 2018.

[2] Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: Proc. of the Int'l Conf. on Neural Information Processing Systems. 2012. 1097−1105.

[3] Wang J, Gu Q, Wu J, *et al*. Traffic Speed prediction and congestion source exploration: A deep learning method. In: Proc. of the IEEE Int'l Conf. on Data Mining. 2016. 499−508.

[4] Zhang J, Zheng Y, Qi D, *et al*. Deep spatio-temporal residual networks for citywide crowd flows prediction. In: Proc. of the National Conf. on Artificial Intelligence. 2016. 1655−1661.

[5] Jozefowicz R, Zaremba W, Sutskever I. An empirical exploration of recurrent network architectures. In: Proc. of the Int'l Conf. on Machine Learning. 2015. 2342−2350.

[6] Dong W, Yuan T, Yang K, *et al*. Autoencoder regularized network for driving style representation learning. In: Proc. of the Int'l Joint Conf. on Artificial Intelligence. 2017. 1603−1609.

[7] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans. on Neural Networks, 2002,5(2):157−166.

[8] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks. Berlin, Heidelberg: Springer-Verlag, 2012.
[9] Song X, Kanasugi H, Shibasaki R. Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level. In: Proc. of the Int'l Joint Conf. on Artificial Intelligence. 2016. 2618−2624.
[10] Jia Z, Chen C, Coifman B, *et al*. The PeMS algorithms for accurate, real-time estimates of *g*-factors and speeds from single-loop detectors. In: Proc. of the IEEE Intelligent Transportation Systems. 2001. 536−541.
[11] Petty K, Bickel PJ, Ostland M, *et al*. Accurate estimation of travel times from single-loop detectors. Transportation Research Part A—Policy and Practice, 1998,32(1):1−17.
[12] Rice J, Van Zwet E. A simple and effective method for predicting travel times on freeways. IEEE Trans. on Intelligent Transportation Systems, 2004,5(3):200−207.
[13] Jenelius E, Koutsopoulos HN. Travel time estimation for urban road networks using low frequency probe vehicle data. Transportation Research Part B—Methodological, 2013. 64−81.
[14] Yang B, Guo C, Jensen CS, *et al*. Travel cost inference from sparse, spatio temporally correlated time series using Markov models. Very Large Data Bases, 2013,6(9):769−780.
[15] Rahmani M, Jenelius E, Koutsopoulos HN, *et al*. Route travel time estimation using low-frequency floating car data. In: Proc. of the Int'l Conf. on Intelligent Transportation Systems. 2013. 2292−2297.
[16] Wang H, Kuo YH, Kifer D, *et al*. A simple baseline for travel time estimation using large-scale trip data. In: Proc. of the ACM Sig-Spatial Int'l Conf. on Advances in Geographic Information Systems. 2016.
[17] Liu H, Jin C, Zhou A. Popular route planning with travel cost estimation. In: Proc. of the 21st Int'l Conf. on Database Systems for Advanced Applications. 2016. 403−418.
[18] Wang Y, Zheng Y, Xue Y, *et al*. Travel time estimation of a path using sparse trajectories. In: Proc. of the Knowledge Discovery and Data Mining. 2014. 25−34.
[19] Lam HT, Diazaviles E, Pascale A, *et al*. Blue) taxi destination and trip time prediction from partial trajectories. In: Proc. of the European Conf. on Principles of Data Mining and Knowledge Discovery. 2015. 63−74.
[20] Wang D, Cao W, Xu M, *et al*. ETCPS: An effective and scalable traffic condition prediction system. In: Proc. of the Database Systems for Advanced Applications. 2016. 419−436.
[21] Bengio Y, Ducharme R, Vincent P, *et al*. A neural probabilistic language model. Journal of Machine Learning Research, 2003,3(6): 1137−1155.
[22] Collobert R, Weston J, Bottou L, *et al*. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 2011, 2493−2537.
[23] Mikolov T, Sutskever I, Chen K, *et al*. Distributed repre-sentations of words and phrases and their compositionality. Neural Information Processing Systems, 2013,26:3111−3119.

**施晋**(1994－),男,浙江台州人,硕士生,主要研究领域为基于位置的服务.

**毛嘉莉**(1979－),女,博士,研究员,CCF 专业会员,主要研究领域为基于位置的服务.

**金澈清**(1977－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为海量数据管理.