利用深度时空残差网络预测城市范围的交通流量[1]

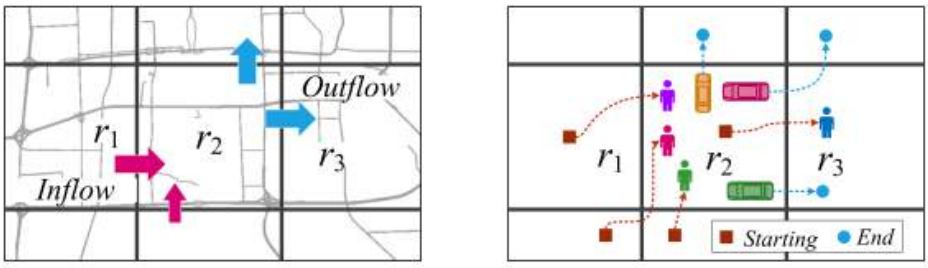
张钧波[a]，郑宇[a,b,c,d]，齐德康[b]，李瑞远[c]，易修文[b]，李天瑞[b]
[a]微软亚洲研究院，北京，中国
[b]信息科学技术学院，西南交通大学，成都，610031，中国
[c]计算机学院，西安电子科技大学，中国
[d]深圳先进技术研究所，中国科学院

摘要：预测城市交通流量对交通管理和公共安全非常重要，同时也非常具有挑战。交通流量受多种复杂因素的影响，包括：空间依赖（近处和远处），时间依赖（近期、周期和趋势），以及外部因素（例如：天气和事件）。我们提出了一种基于深度学习的方法来批量预测城市任何一个区域的两种交通流量类型（进入流和外出流），称为：ST-ResNet。我们基于时空特性设计了一种端对端结构的 ST-ResNet。具体而言，我们部署了残差人工网络框架来建模交通流量的时间邻近性，周期性和趋势性。对于每个特性，我们设计了一系列残差卷积单元，每个单元都建模了交通流量特性。ST-ResNet 试着根据数据动态聚合三个残差网络的输出，为不同的区域分配不同的权重。聚合结果更进一步与外部因素（例如：天气和星期）结合起来得到最终的预测结果。我们在微软 Azure 云平台上部署了一个实时系统，叫做 UrbanFlow，能够监测和预测贵阳的实时交通流量。此外，我们用北京和纽约的交通流量数据做了充分的实验，实验结果表明 ST-ResNet 要优于现有的九种注明的其他方法。

关键字：卷积神经网络，时空数据，残差学习，交通流量，云

1. 引言

预测城市交通流量对于交通管理、风险评估和公共安全非常重要[2]。例如，在 2015 年上海跨年晚会上，大量的人群涌入了一小块区域，造成了踩踏事故，导致了 36 人伤亡。在 2016 年 7 月中旬，上千的“口袋妖怪”玩家涌入了纽约中心公园，想要捕捉一种非常稀少的电子怪物，差点导致了踩踏事故。如果我们能够预测一个区域的人流量，并采取一些应急手段，比如：控制交通流量，发布警告信息，或者提前疏散人群，那么就可以避免此类悲剧的发生。



(a) Inflow and outflow　　(b) Measurement of flows

Figure 1: Crowd flows in a region

本文预测两种类型的交通流量[3]：进入流和外出流，例如图 1(a)所示。进入流是在给定时间区间内进入一个区域的交通总流量。外出流是一段时间内从一个区域流出的交通总流量。两种类型的流量记录了不同区域的人员流动情况。了解它们有助于风险评估和交通管控。进入流和外出流可以通过进出行人的数目、汽车的数目、公共汽车的数目估计出来。图 1(b)给出了一个例子，我们可以利用手机信号来估计行人的数目，表明 r2 的进入流和外出流分别是 3 和 1。类似的，使用车辆的 GPS 轨迹，进入流和外出流分别为 0 和 3。因此，总共的进入流和外出流分别是 3 和 4。显然，预测交通流问题

[1] 原文：Zhang J, Zheng Y, Qi D, et al. Predicting Citywide Crowd Flows Using Deep Spatio-Temporal Residual Networks[J]. arXiv preprint arXiv:1701.02543, 2017.

可以看成是一个时空预测问题[2]。

深度学习[4]已经成功运用到了很多应用中，并且被证明是最前沿的人工智能（AI）技术一直。将这些技术运用到时空数据对一系列的时空应用都非常重要，包括城市规划，城市交通，城市环境，能源，社会，经济，公共安全等[2]。尽管两种主流的深度神经网络考虑了部分空间或者时间属性：1）卷积神经网络（CNNs）考虑空间结构；2）递归神经网络（RNNs）考虑时间依赖关系。但是将这些 AI 技术运用到时空预测问题上仍然具有较大挑战，这主要是因为如下复杂原因：

1. 空间依懒性。
   近处的。如图 1(a)所示，区域 r2 的流入流受周围近处的区域（比如 r1）外出流影响。同样，r2 的外出流也影响这其他的区域（比如 r3）。区域 r2 的外出流同样也会影响本身的外出流。
   远处的。交通流量会受更远出区域的交通流影响。例如，住在离办公室非常远的人通常经由地铁或者高速公路去上班，意味着远处的外出流直接影响这办公区域的进入流。
2. 时间依懒性。
   短期的。一个区域的流量受最近一段时间的情形影响，包括远处的和近处的区域。例如，早上 8 点的交通拥堵将会影响早上 9 点的交通状况。同时，今天下午 4 点交通流量与昨天下午 4 点的交通流量类似。
   周期的。工作日中每天早上早高峰期间的交通流量类似。
   趋势的。冬天的时候早高峰的时间要推迟。因为温度下降，太阳升起的晚，人们起床的迟。
3. 外部因素影响。一些外部因素，比如天气状况或者事件能够影响城市不同区域的交通流。例如，暴风雨天气会影响车速，进而影响区域的流量。

为解决上述挑战，我们为时空数据设计出一种深度神经网络，提出了时空残差网网络（ST-ResNet）不断地预测城市每个区域的进入流和外出流。我们的贡献包括以下五个方面：

- ST-ResNet 基于卷积残差网络，建模近处的和远处的两个区域之间的空间依赖性，同时也保证了预测的精度不受神经网络的深度结构影响。
- 我们总结出交通流量的时间特性分成 3 类：短期的，周期的和趋势的。ST-ResNet 使用不同的残差神经网络建模不同的特性。
- ST-ResNet 动态聚合前述不同网络的输出，为不同的区域赋予不同的权值，聚合结果进一步与外部因素结合起来（比如：天气）。
- 我们利用北京的出租车轨迹数据和天气数据，以及纽约的自行车数据评估了我们的算法，结果表明，相对于其他 9 种方法，我们的方法更具有优势。
- 基于 ST-ResNet，我们部署了一个实时的交通流量监测、预测系统。这个系统基于云和 GPU 服务器，提供了一个强大和灵活的计算环境。

本人的组织结构如下：第二章描述了交通流量预测问题的基本概念。第三章大体描述了系统的基本框架。第四章描述了基于 DNN 的预测模型。第五章展示了实验结果，第六章总结了相关工作。

这篇工作与更早的工作[1]有四个不同。首先，我们部署了一个基于云平台的系统，能够不断地利用出租车轨迹数据预测贵阳市任何一个区域的交通流量，展示了 ST-ResNet 处理实际问题的能力。第三章介绍了基于云平台的实现。其次，我们将 ST-ResNet 从一步预测扩展成了多步预测，能够预测未来更长的交通流量（如第 4.4 节所述）。第三，我们做了更丰富的实验，暂时了 ST-ResNet 的有效性和鲁棒性：i) 与更多的方法作了对比（例如：三种不同变种的递归神经网络），如 5.2 节所述。ii) 测试了 ST-ResNet 更多的网络结构（如 5.3 节所述）；iii) 增加了多步预测的实验（第 5.5 节）。第四，我们探索了更多的相关工作（第 6 章），阐述了我们的工作与最前沿的工作的不同与联系。这更好地帮助我们的工作在整个研究社区的定位。

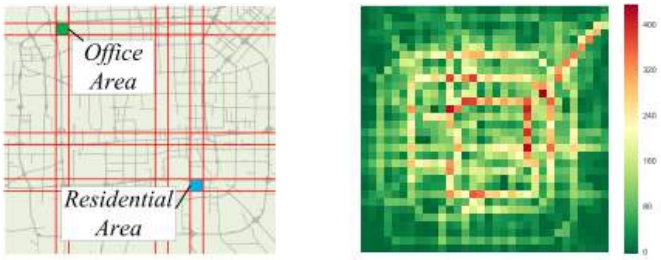
2. 先验知识

我们首先简要的回顾交通流量预测问题[3]，接着介绍深度残差网络[5]。

2.1 交通流量预测问题的形式化定义

定义 1 （区域[3]）。根据不同的粒度和语义，区域可以有多种定义。本文将城市化分成 I × J 的网

格，每个网格表示成一个区域，如图 2(a)所示。



(a) Map segmentation (b) Inflow matrix

Figure 2: Regions in Beijing: (a) Grid-based map segmentation; (b) inflows in every region of Beijing

定义 2（进入流和外出流[3]）。令 P 表示为时间区间 t^{th} 的轨迹集合。对于第 i 行和第 j 列的网格(i，j)，在指定时间区间 t 的进入流和外出流的分别计算如下：

$$x_t^{in,i,j} = \sum_{Tr \in \mathbb{P}} |\{k > 1 | g_{k-1} \notin (i,j) \wedge g_k \in (i,j)\}|$$

$$x_t^{out,i,j} = \sum_{Tr \in \mathbb{P}} |\{k \geq 1 | g_k \in (i,j) \wedge g_{k+1} \notin (i,j)\}|$$

其中 Tr：$g_1 -> g_2 -> \cdots -> g_{|Tr|}$ 是 P 中的一条轨迹，g_k 是空间坐标；$g_k \in (i, j)$ 表示点 g_k 在网格(i, j)中，反之亦然；$|\cdot|$ 表示集合的基数。

在时间区间 t^{th} 中，在所有 I × J 个区域的进入流和外出流可以表示成一个张量 $\mathbf{X}_t \in R^{2\times I\times J}$，其中 $(\mathbf{X}_t)_{0,i,j}=x_t^{in,j,j}$，$(\mathbf{X}_t)_{0,i,j}=x_t^{out,j,j}$。图 2(b)显示的是进入流矩阵。

形式化地，在一个 I×J 的网格地图中，每个网格有两种交通流量。任何时间的交通流量可以表示成一个张量 $\mathbf{X}\in R^{2\times I\times J}$。

问题 1. 给定历史交通流量 $\{\mathbf{X}_t | t=0,\cdots,n-1\}$，预测 $\mathbf{X}_n$。

2.2 深度残差学习

深度残差学习[6]允许卷积神经网络达到 100 层的深度，甚至超过 1000 层。深度残差学习能够解决多种棘手的识别任务，包括图像分类，目标识别，目标分段和定位[6]。

形式化地，一个拥有等同映射的残差单元定义为：

$$\mathbf{X}^{(l+1)} = \mathbf{X}^{(l)} + \mathcal{F}(\mathbf{X}^{(l)}) \quad (1)$$

其中 $\mathbf{X}^{(1)}$ 和 $\mathbf{X}^{(l+1)}$ 分别是第 1 层残差单元的输入和输出；F 是残差方程，例如，[6]文献中两个 3×3 的卷积层的一次堆叠。残差网络的中心思想是学习出 $\mathbf{X}^{(1)}$ 中的残差方程 F[5]。

3. 系统架构

图 3 显示了我们系统的框架，包括三个主要部分：本机 GPU 服务器，云平台和客户端（即：网站和二维码），分别产生在线数据流和离线数据流。本地 GPU 服务器存储历史数据，例如：出租车轨迹，天气数据。云平台接收实时的数据，包括一段时间的实时的交通数据（如轨迹）和天气数据。客户端能够获得进入流和外出流的数据，并在网页中或者手机中展示。

3.1 云平台

云平台不断地接收出租车的轨迹，并且爬取天气数据，然后将它们缓存到了 Redis 中。云平台上的一台虚拟机不断地从 Redis 中获取数据，然后根据 GPS 轨迹计算出城市中每个区域的车流量。同时，虚拟机从天气数据、事件数据及其他数据中抽取特征，之后，虚拟机将流量数据和抽取的特征数据存入存储器中（虚拟机的一部分）。为了节约云平台的资源（更多的存储需要花费更多的钱），我们只存储最近两天的车流量数据和特征数据。历史数据每隔一段时间备份到本地服务器中。

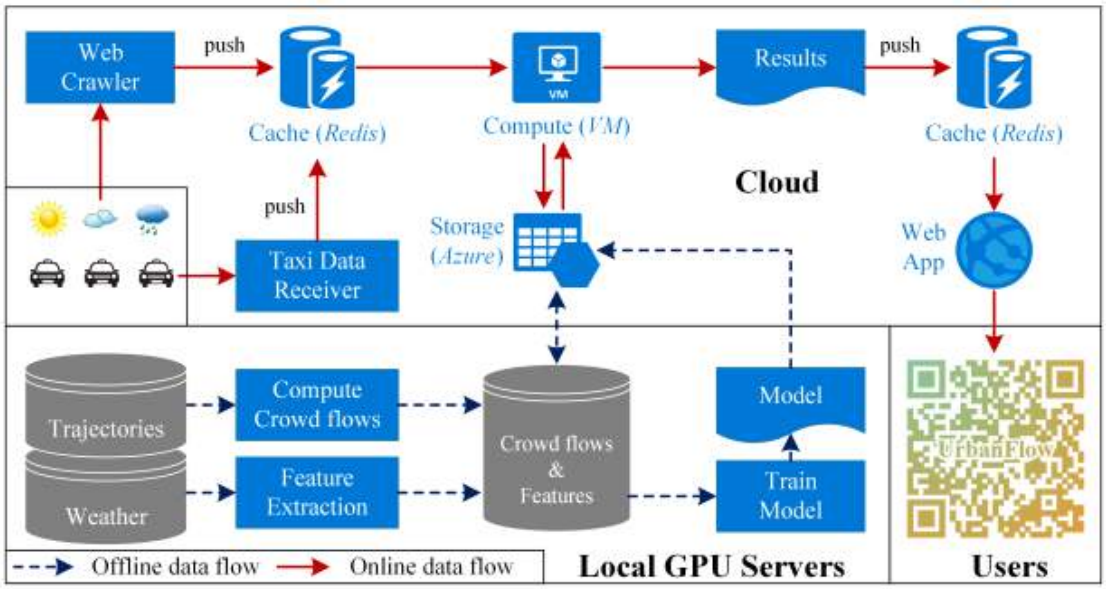


Figure 3: System Framework

我们使用了 Azure 云平台。表 1 详细介绍了我们系统使用的 Azure[2]资源以及价格[3]。我们部署了一台 A2 标准的虚拟机[4]（拥有 2 核 3.5GB 的内存）来预测未来短期的交通流量。考虑到可能来自多个用户的大量访问，网页服务共享一个 APP 服务。由于历史数据存储在了本地服务器中，一个 6GB 的 Redis 缓存便可存储最近半小时的实时轨迹数据、过去两天的交通流量数据和抽取的特征数据，以及预测的结果数据。

Table 1: The Azure resources used for our system

Azure Service	Configuration	Price
App Service	Standard, 4 cores, 7GB memory	$0.400/*hour*
Virtual Machine	A2 standard, 2 cores, 3.5 GB memory	$0.120/*hour*
Redis Cache	P1 premium, 6GB	$0.555/*hour*

3.2 本地 GPU 服务器

尽管所有的工作都可以在云平台上运行，部分地区（比如中国）的 GPU 云服务不支持。另一方面，我们需要为其他云服务付费，比如存储和 I/O 带宽。节约不必要的开支对于研究原型飞铲更重要。此外，考虑到有限的网络带宽，将大量的数据从本地服务器上传到云平台非常费时。例如，历史轨迹可能达到上百 GB，甚至 TB 级别，这需要非常长的时间将数据从本地服务器拷贝到云端。

因此，我们部署了一个混合框架，结合了本地 GPU 服务器与云平台。本地 GPU 服务器主要处理离线训练（学习），包括以下 3 个任务：

- 将轨迹转换成进入流和外出流。我们首先使用一个计算模型将大量的历史轨迹转化成交通流量数据，然后将结果存储在本地。
- 从外部数据中抽取特征。我们首先首先从不同的数据源中收集外部数据（例如：天气数据和节日事件），然后将它们放入特征抽取模型，得到一些连续的和离散的特征，最后存储在本地。
- 训练模型。我们利用上述产生的交通流量数据和外部特征通过提出的 ST-ResNet 来训练预测模型，然后将学到的模型上传到云端。注意，由于动态交通流量和特征存储在了 Azure 存储账号中，在每次训练之前，我们都会将在线的数据同步到本地服务器中。这样，我们可以尝试一下新的想法，同时也能减少研究原型的费用开销。

[2] https://azure.microsoft.com

[3] https://azure.microsoft.com/en-us/pricing/

[4] 为了加快计算，我们也可以选择一个更强的虚拟机，例如 D4 标准的 8 核 18GB 内存的虚拟机，每个小时 0.616 美元

3.3 用户界面

图 4(a)展示了网站 UrbanFlow[7]的用户界面，其中地图上的每个方格表示一个区域，方格上的数字表示其中的进入流和外出流。用户可以通过右上角的按钮查看进入流或者外出流。交通流的越小，颜色越浅，例如：红色表示越大的交通流，绿色表示较小的交通流。用户可以点击任何一个区域查看详细的流量情况，如图 4(b)所示，其中蓝色、黑色和绿色的曲线分别表示昨日、今天过去和今天未来的交通流情况。网页的下方是一个时间轴，用户可以点击特定的时刻，该时刻的交通流热力图就是现实处理啊。当用户点击时间轴的播放按钮时，交通流热力图就像播放电影一样播放（如图 4(c)所示）。目前，我们在中国贵阳市部署了 UrbanFlow[5]。

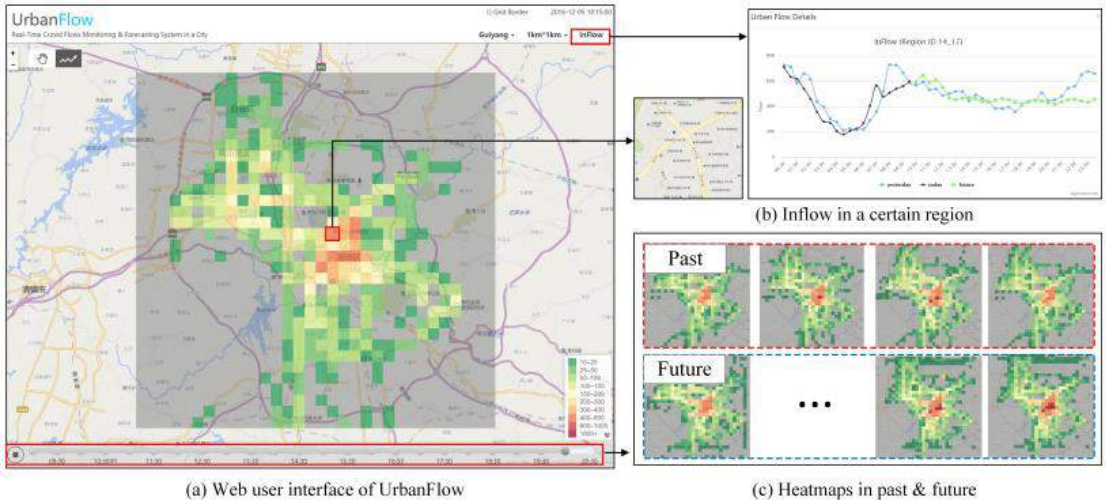


Figure 4: Web user interface of UrbanFlow

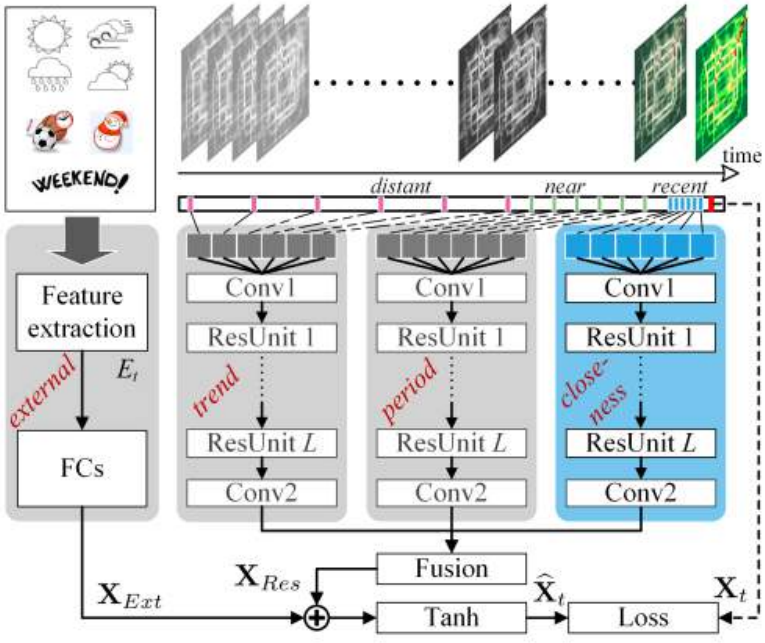


Figure 5: ST-ResNet architecture. Conv: Convolution; ResUnit: Residual Unit; FC: Fully-connected.

4. 深度时空残差网络

递归神经网络（RNNs），如同长短记忆单元一样（LSTM），能够学习较长范围的时间依赖性。然而，使用 RNNs 来建模时间周期性和趋势性，需要很长的输入序列（通常长度为 1344）[6]，这让整个训练过

[5] http://urbanflow.chinacloudsites.cn/

[6] 假设半小时是一个区间区间，4 个星期的序列长度为 48×7×4=1344

程变得更加艰难（5.2 节）。根据时空领域的知识，我们发现少数前面关键几帧的数据能够决定下面一帧的数据。因此，我们利用时间邻近性、周期性和趋势性来选择关键帧。图 5 展示了 ST-ResNet 的框架，包括四个主要的模块：分别建模邻近性、周期性、趋势性和外部影响因子。

图 5 的右上角，我们首先使用定义 1 和定义 2 的方法，将一段时间内城市中的进入流和外出流转化成一个类似于图像的 2 通道矩阵。然后将时间轴划分成 3 个分段：最近一段时间、较近历史时间和较长的历史时间。然后将每个时间段的 2 通道交通流矩阵输入到三个不同的组件中来建模前述的三种不同的时间特性：邻近的，周期的和趋势的。前三个组件共享相同的网络结构，都是卷积神经网络，后面加一个残差单元序列。这种结果能够捕捉近的和远的空间依赖性。在外部组件中，我们从外部数据集中（天气数据、事件数据）手动抽取一些特征，然后将特征输入到一个两层的全连接神经网络。前三个组件的输出基于参数矩阵融合成 X_{Res}，参数矩阵为不同的矩阵不同的区域赋予不同的值。X_{Res} 与外部组件 X_{Ext} 融合起来。最终，融合结果通过一个 Tanh 函数映射到[-1,1]区间中。Tanh 函数比后向传播学习中[8]标准的逻辑函数相比，能够更快的收敛。

4.1 前三层的结构

前三层的结构（即：邻近的，周期的和趋势的）共享相同的网络结构，每个网络结构包括两个子组件：卷积单元和残差单元，如图 6 所示。

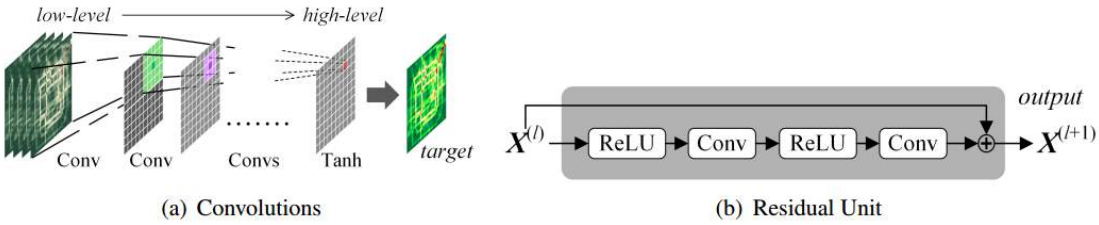


Figure 6: Convolution and residual unit

卷积单元。城市的面积通常非常大，拥有很多区域，区域之间距离不同。直观上来说，近距离的区域之间能够相互影响，卷积神经网络正好能够较好地层次化处理空间结构信息[9]。此外，地铁系统和高速公路能够连接非常远的两个区域，也让远的地区拥有空间依赖性。为了捕捉任意两点之间的空间依赖性，我们需要设计一个多层的卷积神经网络，因为一层卷积只是捕捉了近处的空间依赖性，且受核的大小限制。这个问题在输入和输出拥有相同分辨率的视频序列产生任务中也同样存在[10]。为避免在保留远处依赖性时欠采样而导致分辨率丢失，一些方法被提了出来[11]。与传统的 CNN 不同的是，我们并没有采用欠采样，而只是使用卷积操作[12]。如图 6(a)所示，在多层特征映射中，有多个卷积操作。我们发现高层特征映射的节点取决于中层特征映射的 9 个节点，而这些节点被更底层特征映射的节点覆盖。这意味着一个卷积操作能够捕捉空间邻近依赖性，多个卷积堆叠操作能够捕捉远处的甚至整个城市的空间依赖性。

图 5 的邻近性模块采用一些最近一段时间的 2 通道车流量矩阵来建模时间的邻近性依赖。令最近的一个分段为$[X_{t-lc}, X_{t-(lc-1)}, \cdots, X_{l-1}]$，称为邻近性依赖矩阵。我们首先将它们与第一个轴（即时间轴）拼接成一个张量 $X_c^{(0)} \in R^{2lc \times I \times J}$，之后跟一个卷积操作（即图 5 中的 Conv1）：

$$\mathbf{X}_c^{(1)} = f\left(W_c^{(1)} * \mathbf{X}_c^{(0)} + b_c^{(1)}\right) \quad (2)$$

其中*表示卷积操作，f 是一个激活函数，例如整流器 f(z):=max(0,z)[13]；$W_c^{(1)}, b_c^{(1)}$ 是第一层中需要学习的参数。

传统的卷积操作输出比输入要小，也就是说，是窄卷积，如图 7(a)所示。假定输入的大小为 5×5，过滤器的大小为 3×3，步长为 1，如果使用窄卷积，那么输出的大小为 3×3。在我们的工作中，最终的输出大小应该与输入的大小相同（即 I×J）。为此，我们采用了一种特殊的卷积，即等卷积（如图 7(b)所示），允许过滤器在输入边界的外部，边界外部的区域赋值为 0。

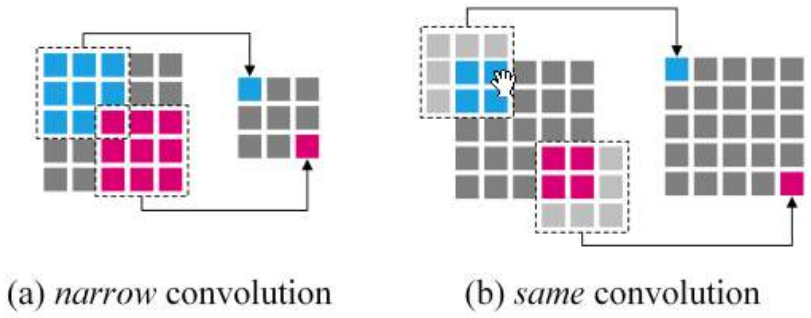


Figure 7: Narrow and same types of convolution. The filter has size 3×3.

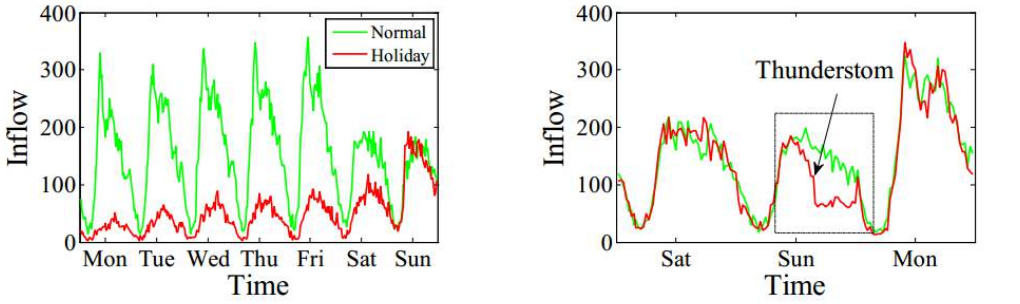
残差单元。众所周知，卷积网络层数越多，训练效果越差，尽管使用一些著名的激活函数（如 ReLU）和规则化技术[13][14][15]。另一方面，我们仍然需要一个深的网络来捕捉全城范围内的依赖。假设一个典型的交通流数据的输入大小为 32×32，卷积核的大小为 3×3，如果需要建模全城的空间依赖（也即：高层的每个节点依赖输入的每个节点），那么需要 15 层连续的卷积层。为了解决这个问题，我们的模型采用残差学习[6]。残差学习被证明训练 1000 层以上的深度神经网络都非常有效。

在我们的 ST-ResNet 中（见图 5），我们为 Conv1 堆叠 L 个残差单元，如下：

$$\mathbf{X}_c^{(l+1)} = \mathbf{X}_c^{(l)} + \mathcal{F}(\mathbf{X}_c^{(l)}; \theta_c^{(l)}), l = 1, \cdots, L \quad (3)$$

其中 F 是残差方程（即：两个 ReLU 和卷积的组合，见图 5），$\theta^{(1)}$包括所有第 1 层需要学习的参数。同样，在 ReLU 之前，我们使用批量规则化（Batch Normalization，BN）。在最顶端的第 L 层残差单元，图 5 中邻近性组件的输出为 $\mathbf{X}_c^{(L+2)}$。

类似的，使用上述操作，我们构建图 5 的周期性和趋势性组件，假设有 l_p 个时间段，周期为 p。那么，周期依赖性序列为$[\mathbf{X}_{t-l_p \cdot p}, \mathbf{X}_{t-(l_p-1) \cdot p}, \cdots, \mathbf{X}_{t-p}]$。利用像公式 2 和公式 3 的卷积操作及 L 层残差单元，周期性组件的输出为 $\mathbf{X}_p^{(L+2)}$。同样，周期性组件的输出为 $\mathbf{X}_{q(L+2)}$。其中 l_q 是趋势性依赖序列的长度，q 是周期长度。注意，p 和 q 是两种不同的周期。在具体实现中，p 是一天的时间，描述了每天的周期性，q 是一周的时间，揭露了一周的趋势。



(a) Feb 8-14 (red), Feb 15-21 (green), 2016 (b) Aug 10-12 (red), Aug 17-19 (green), 2013

Figure 8: Effects of holidays and weather in Office Area of Beijing (the region is shown in Figure 2(a)).

4.2 外部组件的结构

交通流量受多种复杂的外部因素影响，比如：天气状况和事件。图 8(a)显示假期（中国的春节）的交通流量与平常的交通流量非常不同。图 8(b)显示，与平日里相比，暴风雨能够减少办公区域的流量。令 Et 表示预测时间区间 t 的外部因素特征向量。实现中，我们主要考虑天气、假期以及其他的元数据（星期几或者工作日/周末）。表 2 介绍了相关细节。为了预测时间区间 t 的交通流量，假期和元数据能够直接获得，但是，未来时间区间 t 的天气是未知的。但是，我们可以使用 t-1 时刻的天气近似预测 t 时刻的天气。形式化而言，我们在 Et 上堆叠两个全连接层，第一层可以看做是每个子因素的嵌入层，后面接着一个激活函数。第二层将拥有与 Xt 相同形状的低维映射成高维。图 5 中外部

组件的输出表示成带参数 θ_{Ext} 的 X_{Ext}。

4.3 融合

本节讨论如何融合图 5 的四个组件。我们首先用参数矩阵融合算法融合前 3 个组件，然后与外部组件进一步融合。

图 9(a)和(d)展示了表 2 的北京轨迹数据的比率曲线，其中 x 轴是两个时间区间的程度，y 轴是任意两个拥有相同时间跨度的输入流的平均比率。不同区域的曲线都显示了时间相关性。也就是说，更近的时间区间的输入流比远的时间区间的输入流更相关，揭示了时间邻近性。两条区间有不同的形状，展示了不同的区间可能拥有不同的邻近性特性。图 9(b)和(e)描述了 7 天的输入流。我们可以看到两个区域都呈现出明显的日周期性。在办公区域，工作日的高峰值远远比周末的要高。居民区域在工作日和周末都拥有类似的峰值。图 9(c)和(f)展示了从 2015 年 3 月和 2015 年 6 月所有周二的特定时间区间(晚上 9 点到 9 点半)的输入流。从中我们可以看出，随着时间的推移，工作区域的输入流逐渐降低，但居民区域的输入流逐渐升高。这也说明不同的区间有不同的趋势。总之，两个区域的输入流都有邻近性、周期性和趋势性，但是受影响的程度大不相同。我们在其他的区域以及输入流上也发现了类似的特性。

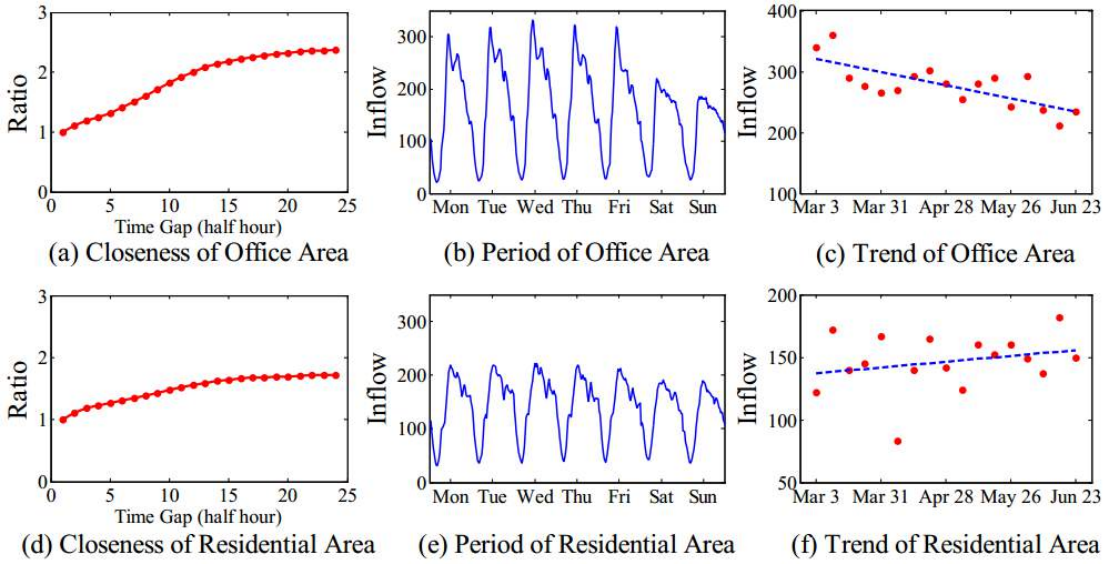


Figure 9: Temporal dependencies (Office Area and Residential Area are shown in Figure 2(a)).

总之，不同的区域都受邻近性、周期性和趋势性的影响，但是影响程度不同。受此观察启发，我们提出了一个基于参数矩阵融合的方法。

基于参数矩阵融合。我们使用下述方法融合图 5 的三个组件（邻近性，周期性和趋势性）：

$$\mathbf{X}_{Res} = \mathbf{W}_c \circ \mathbf{X}_c^{(L+2)} + \mathbf{W}_p \circ \mathbf{X}_p^{(L+2)} + \mathbf{W}_q \circ \mathbf{X}_q^{(L+2)} \quad (4)$$

其中∘是阿达玛乘法（即元素间相乘），Wc，Wp 和 Wq 是要学习的参数，分别表示邻近性、周期性和趋势性的影响程度。

融合外部组件。我们直接将前 3 个组件的输出与外部组件进行融合，如图 5 所示。最终，预测的低 t 时间区间的值 Xt 表示为：

$$\widehat{\mathbf{X}}_t = \tanh(\mathbf{X}_{Res} + \mathbf{X}_{Ext}) \quad (5)$$

其中 tanh 是双曲线切线函数，保证了输出的范围在-1 到 1 之间。

ST-ResNet 通过 3 个流矩阵和外部因素特征来预测 Xt，最小化预测的流矩阵和真实的流矩阵之间

的平均平方误差：

$$\mathcal{L}(\theta) = \|\mathbf{X}_t - \widehat{\mathbf{X}}_t\|_2^2 \quad (6)$$

其中， θ 是 ST-ResNet 需要学习的参数。

4.4 算法优化

算法 1 描述了 ST-ResNet 的训练过程。我们首先从原始序列中构建一个训练实例（第 1-6 行），然后通过后向传播和 Adam[16]进行训练（第 7-11 行）。

Algorithm 1: Training of ST-ResNet

Input: Historical observations: $\{\mathbf{X}_0, \cdots, \mathbf{X}_{n-1}\}$;
external features: $\{E_0, \cdots, E_{n-1}\}$;
lengths of *closeness*, *period*, *trend* sequences: l_c, l_p, l_q;
peroid: p; trend span: q.
Output: ST-ResNet model $\mathcal{M}$.

// construct training instances

1 $\mathcal{D} \longleftarrow \emptyset$
2 **for** *all available time interval* $t (1 \le t \le n-1)$ **do**
3 $\quad \mathcal{S}_c = [\mathbf{X}_{t-l_c}, \mathbf{X}_{t-(l_c-1)}, \cdots, \mathbf{X}_{t-1}]$
4 $\quad \mathcal{S}_p = [\mathbf{X}_{t-l_p \cdot p}, \mathbf{X}_{t-(l_p-1)\cdot p}, \cdots, \mathbf{X}_{t-p}]$
5 $\quad \mathcal{S}_q = [\mathbf{X}_{t-l_q \cdot q}, \mathbf{X}_{t-(l_q-1)\cdot q}, \cdots, \mathbf{X}_{t-q}]$
$\quad$ *// $\mathbf{X}_t$ is the target at time t*
6 $\quad$ put an training instance $(\{\mathcal{S}_c, \mathcal{S}_p, \mathcal{S}_q, E_t\}, \mathbf{X}_t)$ into $\mathcal{D}$

// train the model

7 initialize the parameters θ
8 **repeat**
9 $\quad$ randomly select a batch of instances $\mathcal{D}_b$ from $\mathcal{D}$
10 $\quad$ find θ by minimizing the objective (6) with $\mathcal{D}_b$
11 **until** *stopping criteria is met*
12 output the learned ST-ResNet model $\mathcal{M}$

Algorithm 2: Multi-step Ahead Prediction Using ST-ResNet

Input: Learned ST-ResNet model: $\mathcal{M}$;
number of look-ahead steps: k;
historical observations: $\{\mathbf{X}_0, \cdots, \mathbf{X}_{n-1}\}$;
external features: $\{E_n, \cdots, E_{n+k-1}\}$;
lengths of *closeness*, *period*, *trend* sequences: l_c, l_p, l_q;
peroid: p; trend span: q.

1 $\mathcal{X} \longleftarrow \{\mathbf{X}_0, \cdots, \mathbf{X}_{n-1}\}$ *// (i.e., $\mathcal{X}_t = \mathbf{X}_t, \forall t$)*
2 **for** $t = n$ **to** $n + k - 1$ **do**
3 $\quad \mathcal{S}_c = [\mathcal{X}_{t-l_c}, \mathcal{X}_{t-(l_c-1)}, \cdots, \mathcal{X}_{t-1}]$
4 $\quad \mathcal{S}_p = [\mathcal{X}_{t-l_p \cdot p}, \mathcal{X}_{t-(l_p-1)\cdot p}, \cdots, \mathcal{X}_{t-p}]$
5 $\quad \mathcal{S}_q = [\mathcal{X}_{t-l_q \cdot q}, \mathcal{X}_{t-(l_q-1)\cdot q}, \cdots, \mathcal{X}_{t-q}]$
6 $\quad \widehat{\mathbf{X}}_t \longleftarrow \mathcal{M}(\mathcal{S}_c, \mathcal{S}_p, \mathcal{S}_q, E_t)$
7 $\quad$ put $\widehat{\mathbf{X}}_t$ into $\mathcal{X}$, *i.e.*, $\mathcal{X}_t = \widehat{\mathbf{X}}_t$
8 output $\{\widehat{\mathbf{X}}_n, \cdots, \widehat{\mathbf{X}}_{n+k-1}\}$

训练完成后，学到的 ST-ResNet 模型 M 可以用来预测单步或者多步预测。预测的过程如算法 2 所示。一些外部特征（即天气）与算法 1 不同。在训练过程中，我们使用真实的天气数据，但在算法 2

中，我们使用的是预测的天气数据。

5. 实验

本章我们比较 ST-ResNet 与九种其他算法在北京和纽约的两种数据流预测上的性能。

5.1 实验设置

数据集。我们使用两个不同的数据集，如表 2 所示。每个数据集有包含两种不同的数据：轨迹和天气，具体描述如下。

Table 2: Datasets (holidays include adjacent weekends).

Dataset	TaxiBJ	BikeNYC
Data type	Taxi GPS	Bike rent
Location	Beijing	New York
Time Span	7/1/2013 - 10/30/2013 3/1/2014 - 6/30/2014 3/1/2015 - 6/30/2015 11/1/2015 - 4/10/2016	4/1/2014 - 9/30/2014
Time interval	30 minutes	1 hour
Gird map size	(32, 32)	(16, 8)
Trajectory data		
Average sampling rate (s)	~ 60	\
# taxis/bikes	34,000+	6,800+
# available time interval	22,459	4,392
External factors (holidays and meteorology)		
# holidays	41	20
Weather conditions	16 types (*e.g.*, Sunny, Rainy)	\
Temperature / °C	[−24.6, 41.0]	\
Wind speed / mph	[0, 48.6]	\

- **TaxiBJ**：北京的轨迹数据（出租车轨迹数据）和天气数据包括四个时间段：2013 年 6 月 1 日至 2013 年 8 月 30 日，2014 年 5 月 1 日至 2014 年 6 月 30 日，2015 年 5 月 1 日至 2015 年 6 月 30 日，2015 年 12 月 1 日至 2016 年 4 月 10 日。根据定义 2，我们获得两种类型的交通流。我们选择最后 4 个星期的数据作为测试集，所有其他的数据作为训练集。
- **BikeNYC**：纽约的轨迹数据来自纽约自行车系统，时间跨度为 2014 年 4 月 1 日到 9 月 30 日。一条数据包括：旅行时间、开始站点和结束站点的 ID、开始时间和结束时间。这些数据中，最后 10 天的数据作为测试集，其他数据作为训练集。

对比方法。我们将 ST-ResNet 与 9 种对比方法进行比较：

- **HA**：通过计算对应时间区间的所有历史输入流和外出流的平均流量。例如：周二上午的 9 点到 9 点半，对应所有历史数据中所有周二上午的 9 点到 9 点半。
- **ARIMA**：自回归滑动平均（ARIMA）是一个著名的预测时间序列的模型。
- **SARIMA**：季节性 ARIMA。在 ARIMA 上，SARIMA 还考虑了季节特性，能够同时学习时间邻近性和周期性。
- **VAR**：向量自回归（VAR）是一个更为先进的时空模型，能够捕捉所有流量的成对之间的关系，但由于大规模的参数，计算开销也大。
- **ST-ANN**：抽取空间（周围 8 个区域的值）和时间（前 8 个时间区间）的特征，然后放入人工神经网络。
- **DeepST**[3]：一个针对时空数据的深度神经网络（DNN）预测模型，在交通流数据预测问题上展示了较好的特性。

- **RNN**[17]：递归神经网络（DNN）是一种深度学习模型，能够捕捉时间依赖性。形式化地讲，RNN 能够训练任意长度的序列。在我们的实验中，我们固定输入序列的长度为{3, 6, 12, 24, 48, 336}其中的一个。例如当序列长度为 48 且时间区间为 30 分钟时，输入序列的时间依赖性为 1 天。因此，有六种 RNN 的变种，包括：RNN-3，RNN-6，RNN-12，RNN-24，RNN-48 和 RNN-336。
- **LSTM**[18]：长短记忆单元网络（LSTM）是一种特殊的 RNN，能够学习长时间的时间依赖。与 RNN 的设置相同，我们做了 6 种 LSTM 变种的实验，即：LSTM-3，LSTM-6，LSTM-12，LSTM-24，LSTM-48 和 LSTM-336。
- **GRU**[19]：门递归单元网络，是一种新的 RNN，能够捕捉长时间的时间依赖。与 RNN 的设置相同，有以下 6 种 GRU 变种作为对比方法：GRU-3，GRU-6，GRU-12，GRU-24，GRU-48，以及 GRU-336。

预处理。在 ST-ResNet 的输出中，我们使用 tanh 作为最后的激活函数（见公式 5），使得输出为-1 和 1。这里，我们使用最小-最大规则化方法将数据映射到[-1,1]范围中。在评估节点，我们将预测值恢复到正常的值，然后与真值进行对比。对于外部因素，我们使用单热点编码（one-hot coding）将元数据（即：一周的星期数，周末/工作日），节假日和天气状况转化成二进制向量，然后使用最小-最大规则化方法将气温和风速转化到[0,1]范围区间。

超参数。需要学习的参数用 Keras[20]的默认参数的均匀分布进行输出华。卷积操作 Conv1 和所有的残差单元使用 64 个 3×3 的过滤器，Conv2 使用 2 个 3×3 的过滤器。例如，一个 4 残差单元的 ST-ResNet 有一个 Conv1,4 个残差单元和一个 Conv2 组成。表 3 进行了详细的说明。我们使用 Adam[16]进行优化，批处理大小为 32。残差单元数目的设置中，TaxiBJ 设为 12，BikeNYC 设为 4。ST-ResNet 有 5 个超参数，其中 p 和 q 分别设为 1 天和一周。对于 3 个序列，我们设为：lc∈{1,2,3,3,5}，lp∈{1,2,3,4}，lq∈{1,2,3,4}。我们选择 90%的数据作为训练数据，身下的 10%的数据作为验证集。这样，当验证分数最佳时，我们训练算法可以提前终止。之后，我们使用所有的训练数据训练固定的轮数（即：10100 轮）。

Table 3: Details of convolutions and residual units

layer name	output size	closeness	period	trend
Conv1	32×32	$3 \times 3, 64$	$3 \times 3, 64$	$3 \times 3, 64$
ResUnit 1	32×32	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
ResUnit 2	32×32	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
ResUnit 3	32×32	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
ResUnit 4	32×32	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
Conv2	32×32	$3 \times 3, 2$	$3 \times 3, 2$	$3 \times 3, 2$

衡量标准：我们使用开方平均平方误差（RMSE）[7]来评估模型的有效性：

$$RMSE = \sqrt{\frac{1}{z}\sum_{i}(x_i - \hat{x}_i)^2} \tag{7}$$

其中 x 和 $\hat{x}$ 分别表示真值和对应的预测值。Z 是所有可获得的真值数目。

实验主要在 GPU 服务器上运行，详细信息见表 4。我们使用了 Python 库，包括 Theano[21]和 Keras[20]来训练我们的模型。

[7] 越小越好

Table 4: Experimental environment

OS	Windows Server 2012 R2
Memory	256GB
CPU	Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz
GPU	Tesla K40m
Number of GPU cards	4
CUDA version	8.0
cuDNN version	8.0
Keras version	1.1.1
Theano version	0.9.0dev

5.2 单步预测的评估

本节，我们评估单步预测，即使用历史数据预测时间 t 的车流量。表 5 显示 TaxiBJ 和 BikeNYC 所有方法的 RMSE。ST-ResNet 都明显优于所有的对比方法。特别地，TaxiBJ 数据集上的结果表明 ST-ResNet（有 12 个残差单元）比 ARIMA 好 26%，比 SARIMA 好 37%，比 VAR 好 26%，比 ST-ANN 好 14%，比 DeepST 好 7%，比 RNN 好 28%到 64%，比 LSTM 好 18.1%到 45.7%，比 GRU 好 17.4%到 46.1%。ST-ResNet-noExt 是 ST-ResNet 的一个降级版本，没有考虑外部因素（例如：天气数据）。我们可以看到，ST-ResNet-noExt 略逊于 ST-ResNet，表明外部因素的有效性。DeepST 采用时空 CNNs，明显优于其他对比方法。尽管 ST-ANN 和 VAR 使用时空信息以及不同流的关系，但是他们都比 DeepST 表现要差，因为他们只是考虑了比较近的时间信息以及最近的时间信息。对于那些时间模型，GRU 和 LSTM 的 RMSE 相近，但平均优于 RNN，因为 GRU 和 LSTM 都能够捕捉长时间的时间依赖性。然而，GRU-336 和 LSTM-336 以及 RNN-336 的性能最差，表明基于 RNN 模型不能捕捉长时间的依赖性（即：周期性和趋势性）。为更直观，我们将这些模型排序，如图 10(a)所示。

Table 5: Comparisons with baselines on TaxiBJ and BikeNYC. The results of ARIMA, SARIMA, VAR and DeepST on BikeNYC are taken from [3].

	RMSE	
Model	TaxiBJ	BikeNYC
HA	57.69	21.58
ARIMA	22.78	10.07
SARIMA	26.88	10.56
VAR	22.88	9.92
ST-ANN	19.57	7.57
DeepST	18.18	7.43
RNN-3	23.42	7.73
RNN-6	23.80	7.93
RNN-12	32.21	11.36
RNN-24	38.66	12.95
RNN-48	46.41	12.15
RNN-336	39.10	12.01
LSTM-3	22.90	8.04
LSTM-6	20.62	7.97
LSTM-12	23.93	8.99
LSTM-24	21.97	10.29
LSTM-48	23.02	11.15
LSTM-336	31.13	10.71
GRU-3	22.63	7.40
GRU-6	20.85	7.47
GRU-12	20.46	6.94
GRU-24	20.24	11.96
GRU-48	21.37	9.65
GRU-336	31.34	12.85
ST-ResNet	**16.89** (12 residual units)	**6.33** (4 residual units)
ST-ResNet-noExt	17.00 (12 residual units)	\

与 TaxiBJ 不同，BikeNYC 包括了两种类型的交通流量，包括新流和终止流[22]。我们采用 4 残差单元的 ST-ResNet，同时像 DeepST[3]一样考虑元数据作为外部特征。ST-ResNet 比这些对比方法要优 9%至 71%，提现出我们的模型在其他流量预测任务上具有较好的泛化性能。图 10(b)显示了这些模型的排序。

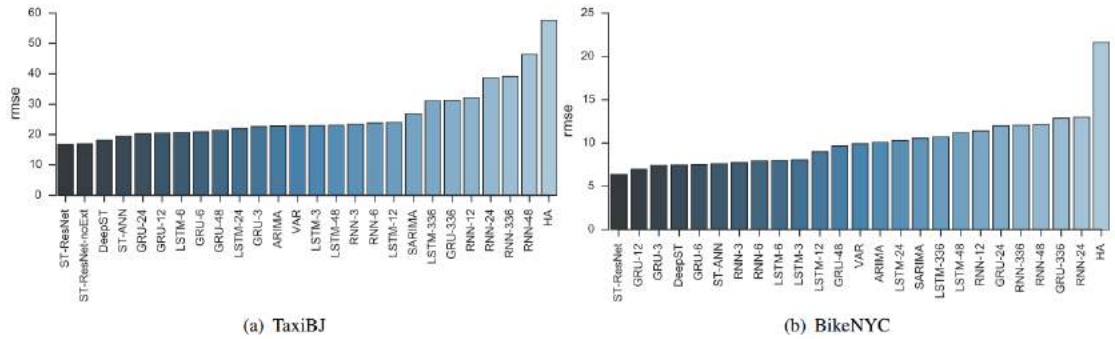


Figure 10: Model ranking on TaxiBJ and BikeNYC. The smaller the better.

5.3 不同 ST-ResNet 变种的结果

本节讨论不同 ST-ResNet 变种的性能比较结果，包括改变网络配置，网络深度和使用不同的组件。

5.3.1 不同网络配置的影响

图 11 展示了不同网络配置的影响。它们都有相同的超参数：lc=3，lp=1，lq=1，残差单元的数量=12。

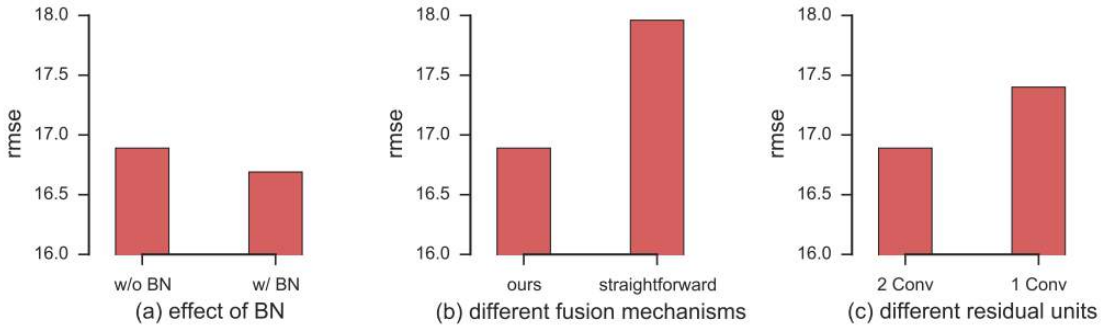


Figure 11: Results of different configurations

- 批规则化（Batch Normalization，BN）的效果：我们尝试在每个残差单元采用 BN，发现在单步预测时 RMSE 略微提升了一点，如图 11(a)所示。
- 参数矩阵融合的效果。我们使用了一种参数矩阵融合的机制（见等式 4）来融合时间邻近性，周期性和趋势性组件。更简单一点，我们也可以使用一种直接的方式，即：$\mathbf{X}_c^{(L+2)}+\mathbf{X}_p^{(L+2)}+\mathbf{X}_q^{(L+2)}$。图 11(b)显示了我们的方法显著的比直接的方法要好，这表明了参数矩阵融合方法的有效性。
- 残差单元的内部结构。残差单元包括包括 2 个卷积操作。我们测试了不同卷积单元的性能。从图 11(c)我们可以看到使用两个卷积比使用 1 个卷积效果更好。

5.3.2 网络深度的影响

图 12 显示的是不同网络深度的影响。随着网络深度的加深（即残差网络增加），RMSE 首先介绍，然后增加，表明，越深的网络通常拥有更好的性能，因为它能够同时捕捉近处和远处的空间依赖性。然而，当网络变得很深时（例如残差单元超过 14 时），训练变得非常困难。

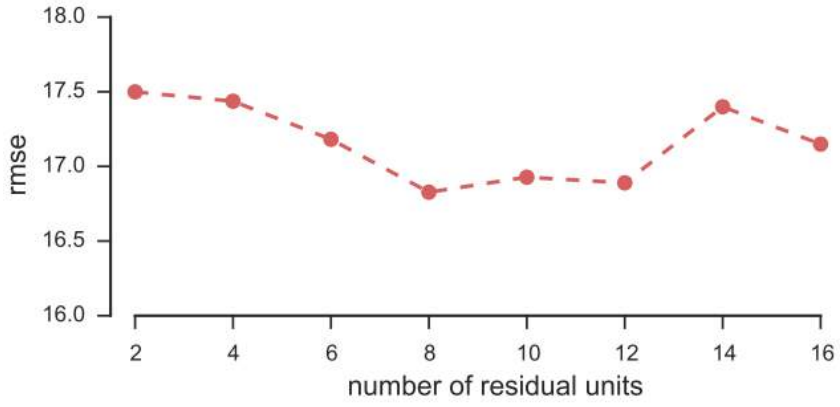


Figure 12: Impact of network depth

5.3.3 过滤器大小的影响

卷积的接受域取决于过滤器的大小。我们改变不同的过滤器的效果，从 2×2 到 5×5。图 13(a)显示了过滤器越大，RMSE 越小，表明大的接受域更能够建模空间的依赖性。从图 13(b)，我们可以看到更多的过滤器，结果越好。

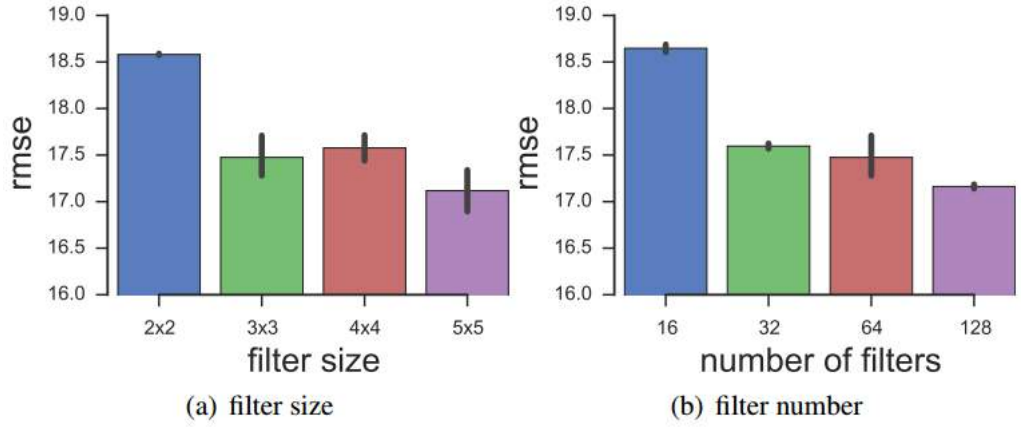


Figure 13: Impact of filter size and number.

5.3.4 时间邻近性、周期性和趋势性

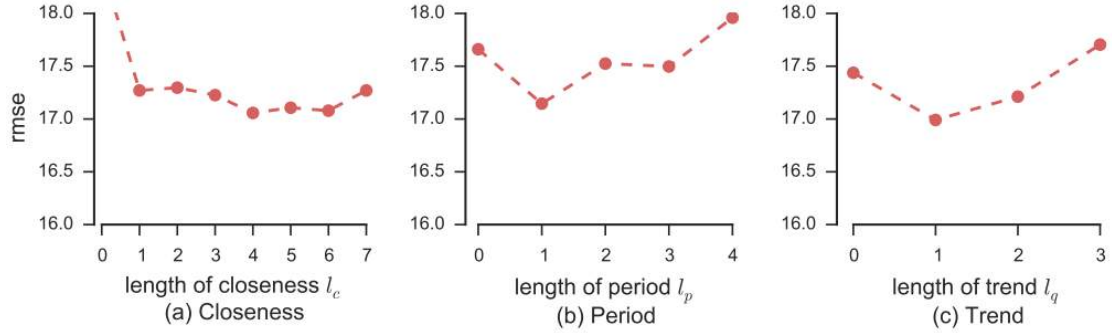


Figure 14: Impact of temporal *closeness*, *period*, *trend*

我们验证了 TaxiBJ 数据集上的时间邻近性、周期性和趋势性组件的影响，如图 14 所示。图 14(a)显示当固定 lp=1，lq=1，lc 变化的时间邻近性效果。我们观察随着考虑的时间增加，RMSE 首先下降，然后上升，且 lc=4 的时候性能最佳。图 14(b)展示了周期性效果的影响，此时，lc 和 lq 设为 1，lp 不断变化。我们可以看到当 lp=1 时，RMSE 最小。没有周期性模块的模型（即 lp=0）比 lp=2,3 的效果差些，但是由于 lp=4 的模型，意味着小范围的周期通常更有力，而长时间的周期难以建模，通常

无用。图 14(c)显示了趋势性的变化，此时 lc 和 lp 分别等于 3 和 2，lq 的取值从 0 到 3。曲线表明 lq=1 的时候模型最好。与周期性类似，趋势性能够增强模型，而且长趋势性难以捕捉，通常无效。

为了更好的了解事件邻近性、周期性和趋势性，我们展示了参数矩阵融合层的参数，这能够学习到城市不同的区域之间的时间影响程度，如图 15 所示。每个图中的单元表示一个特定区域学到的参数，反映了时间邻近性、周期性和趋势性的影响程度。我们设置了一个阈值（例如 0.3）来查看整个城市的时间属性。给定一个固定的阈值 0.3，我们发现邻近性的比率（参数值小鱼 0.3 的区域的数目）为 0，表明城市中所有的区域或多或少都有一些邻近性。周期性的比率显示 9%的区域有若周期模式。同样，图 15(c)显示 7%的区域没有时间趋势性。从图 15(a)中可以看出，一些主要干道对应区域（红曲线区域）的邻近性并不明显。一个原因是这些区域的流量可以使用周期性和趋势性，加一点邻近性便可预测出来。

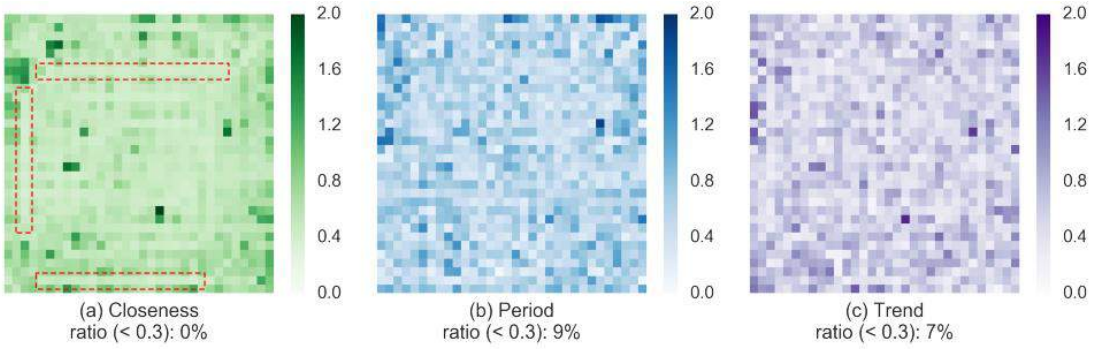


Figure 15: Visualization of parameters

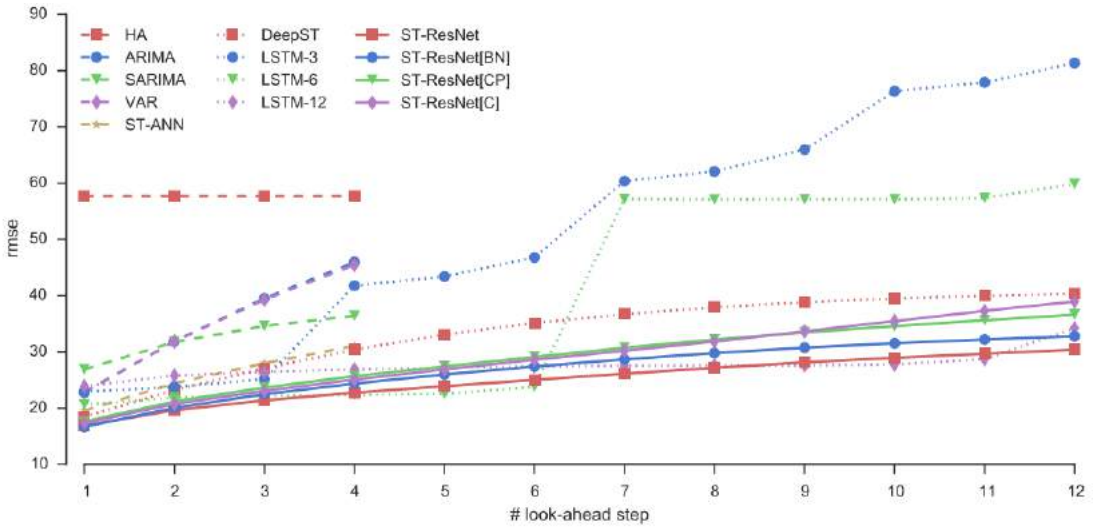


Figure 16: Multi-step ahead prediction

5.4 多步预测评估

根据算法 2，我们使用历史数据和最近的预测数据来预测接下来时间的交通流量，成为多步预测。图 16 显示在 TaxiBJ 数据集上 13 中不同的模型的预测结果。这些模型中，ST-ResNet[BN]，ST-ResNet[CP]和 ST-ResNet[C]是三种 ST-ResNet（12 残差单元）的变种，其中 ST-ResNet[BN]在所有的残差单元中采用 BN 方法，ST-ResNet[CP]没有部署趋势性组件，ST-ResNet[C]只是使用了邻近性和外部组件。LSTM-3，LSTM-6 和 LSTM-12 是 LSTM 模型的变种（具体参见 5.1 节）。在实际应用中，预测未来短时间的交通流量（如未来 2 小时）尤为重要。从 4 步预测的结果[8]可以看到，尽管 ST-ResNet[BN]方法在单步预测时表现较好（图 11(a)所示），但是 ST-ResNet 在多步预测中表现最佳。在 ST-ResNet、ST-ResNet[C]和 ST-ResNet[Cp]中，ST-ResNet 表现最好，表明周期性和趋势性在多步预测中非常重

[8] 在 TaxiBJ 数据集上进行 4 步预测表明预测未来两个小时后的交通流

要。同时，我们注意到当多余预测的不长超过 8 时，LSTM-12 比 ST-ResNet 更好，原因可能是 LSTM-12 考虑了过去 12 个记录，而 ST-ResNet 只是采用过去 3 小时的记录作为邻近性组件的输入。

5.5 效率和资源

我们测试了云平台上（即微软 Azure）两种不同的虚拟机的效率。如 3.1 节介绍，预测城市中每个区域的交通流量分成 4 步：(1)从 Redis 中读取轨迹数据；(2)将轨迹数据转化成交通流数据；(3)预测未来短时间的交通流量；(4)将预测结果放入 Redis 中。我们分别测试了以上 4 步的时间开销。整体上，A2 标准的虚拟机完成整个预测过程需要 18.56 秒，而 D3 标准的虚拟机（性能更强但是更贵）只需要 10.93 秒。我们可以选择 A2 标准的虚拟机，因为它仅仅只需要 20%的价格，但是能够获得 50%的性能。

Table 6: Configuration of virtual machines and performance

Virtual Machine (Azure)	A2 standard	D4 standard
price	$0.120/*hour*	$0.616/*hour*
OS	Ubuntu 14.04	Ubuntu 14.04
Memory	3.5GB	28GB
CPU	8 cores @ 2.20GHz	2 cores @ 2.20GHz
Keras version	1.1.1	1.1.1
Theano version	0.9.0dev	0.9.0dev
Time (s)		
Pulling trajectories from redis	2.71	1.64
Converting trajectories into flows	9.65	6.05
Predicting the crowd flows	5.79	2.97
Pushing results into redis	0.41	0.27
Total	18.56	10.93

6. 相关工作

6.1 交通流预测

一些前期工作基于用户的位置来预测个人的行为[23][24]。它们主要预测个人的轨迹，而不是一个区域的交通流。这些工作可能需要很多的计算资源，在公共安全的场景下也没有必要。其他一些研究员预测一条道路上的速度和交通流量[25][26][27]。这些工作大部分预测单条或者多条路段，而不是整个城市范围[27][28]。最近，一些研究员开始进行城市范围的交通流预测[22][29]。这两个工作与我们不同，它们主要是针对单个区域而不是城市，并且它们没有将整个城市划分成网格，而是使用了更为复杂的方法首先找到不规则的区域。

6.2 时间序列预测的传统模型

预测交通流可以看成是一种时间序列预测问题。有一些传统的线性模型来解决这个问题。历史平均模型比较简单，只是使用历史时间序列的平均值来预测未来的时间序列的平均值。然而，此模型无法对动态事件（如交通事故）作出响应[30]。滑动自回归平均模型（ARIMA）假设未来时间序列的值是历史值的线性组合和残差，因此，为了获得稳定的结果，非稳定的时间序列在分析之前需要区别对待[31]。ARIMA 不适合有缺失值的时间序列，因为它依赖于非缺失时间序列，随着应用场景的复杂度提高，数据填补技术将会出现问题[32]。在使用 ARIMA 之前，额外的季节性差异通常应用到季节性时间序列中以获得稳定的结果，称之为 SARIMA。SARIMA 的不足之处在于它非常耗时[32]。向量自回归模型（Vector Autoregressive，VAR）能够捕捉非相关时间序列的线性依赖[33]。然而，预测值和残差之间的关系却被忽略了。

与前述的线性模型不同，人工神经网络模型（ANN）是一个非线性模型，通常使用在时间序列的预测场景中[34][35][36]。ANNs 拥有较强的非线性建模能力，但是线性建模能力欠佳[37]。

6.3 深度神经网络

神经网络和深度学习[4][38][39]在许多应用场景中都获得了巨大成功，例如：计算视觉[13][40]，语音识别[41][42]，和自然语言处理[43]。例如，卷积神经网络在 2012 年之后就获得了 ImageNet[44]的冠军，同时帮助 AlphaGo[45]打败了人类冠军[9]。递归神经网络（RNNs）成功运用在了序列学习的任务上[46]。长短记忆单元（LSTM）的引入[18]或者门递归单元(GRU)[19]使得 RNNs 能够学习长时间依赖性。然而，两种神经网络都只能捕捉空间或者时间依赖性。最近，研究学者们将上述网络结合起来，提出了一种卷积 LSTM 网络[47]来同时学习时间和空间依赖性。但是这个网络无法建模非常长的时间依赖性（即周期性和趋势性），随着深度的增加，训练过程也变得更加困难。

针对时空数据，我们之前的工作[3]提出了一种基于 DNNs 的通用预测模型。为更加有效的建模时空预测（即城市范围的流量预测），本文提出使用残差学习和参数矩阵融合机制。关于数据融合方法的详细说明可以参见[48]。

6.4 城市计算

城市计算[2]是一个新兴研究领域，旨在使用城市中产生的多种数据（例如地理数据，交通流数据和人类数据）来解决城市问题（例如：交通拥堵，能量消耗和污染问题）。一个研究分支是首先将城市划分成网格，然后研究每个区域的交通流量，例如预测空气质量[49][50]，异常交通模式检测[51]，分析缺失空气质量[52]，预测时空数据[3]。此外，一些研究人员开始使用深度学习的方法来构建城市计算应用。例如，Song 等提出了一个递归神经网络模型来预测个人未来的行为[53]。Chen 等提出了一种深度学习模型来理解人类行为如何影响交通事故风险[54]。这两个工作在方法和问题上都与我们不同。据我们所知，在城市计算领域，针对城市范围交通流量预测的端对端深度学习还未出现。

7. 结论和未来工作

我们提出了一种新的深度学习模型，使用历史时空数据，天气数据和事件数据，能够预测城市每个区域的交通流量。提出的 ST-ResNet 模型能够学习所有的空间（近处和远处）依赖、时间（邻近性，周期性和趋势性）依赖和外部因素（例如：天气和事件）。在北京和纽约的两个数据机上，我们验证了模型的性能优于 9 种对比方法，更进一步确定了我们的模型在交通流量预测中更实用。代码和数据集已经发布在了：https://www.microsoft.com/en-us/research/publicatiop/deepspatio-temporal-residual-networks-for-citywide-crowd-flows-predpredic。我们也部署了一个基于云平台的系统，称为 UrbanFlow，能够监控实时交通流，预测未来短时间的交通流量。

未来工作中，我们将会考虑其他类型的流数据（例如地铁刷卡数据，出租车/卡车/公交车轨迹数据，和手机信息号数据），利用他们进行更多类型的交通流量预测，然后使用一个合适的融合模型将这些结果融合起来得到最终结果。

8. 致谢

本工作受到中国国家自然基金（基金号 61672399 和 U1401258）以及中国国家基本研究项目（973 项目，基金号 2015CB352400）的支持。

参考文献：

[1] J. Zhang, Y. Zheng, D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[2] Y. Zheng, L. Capra, O. Wolfson, H. Yang, Urban computing: concepts, methodologies, and applications, ACM Transactions on Intelligent Systems and Technology (TIST) 5 (3) (2014) 38.

[3] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, DNN-based prediction model for spatio-temporal data,

[9] https://en.wikipedia.org/wiki/AlphaGo_versus_Lee_Sedol

in: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2016, Burlingame, California, USA, October 31 – November 3, 2016, 2016, pp. 92:1–92:4. doi:10.1145/2996913.2997016.URL http://doi.acm.org/10.1145/2996913.2997016

[4] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[5] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks (2016) 630–645 doi:10.1007/978-3-319-46493-0_38.URL http://dx.doi.org/10.1007/978-3-319-46493-0_38

[6] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition (2016) 770–778 doi:10.1109/CVPR.2016.90.URL http://dx.doi.org/10.1109/CVPR.2016.90

[7] Urban Flow website: http://urbanflow.sigkdd.com.cn/.

[8] Y. A. LeCun, L. Bottou, G. B. Orr, K.-R. Muller, E ¨ fficient backprop, in: Neural networks: Tricks of the trade, Springer, 2012, pp. 9–48.

[9] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998)2278–2324.

[10] M. Mathieu, C. Couprie, Y. LeCun, Deep multi-scale video prediction beyond mean square error, arXiv preprint arXiv:1511.05440.

[11] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[12] V. Jain, J. F. Murray, F. Roth, S. Turaga, V. Zhigulin, K. L. Briggman, M. N. Helmstaedter, W. Denk, H. S. Seung, Supervised learning of image restoration with convolutional networks, in: 2007 IEEE 11th International Conference on Computer Vision, IEEE, 2007, pp. 1–8.

[13] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[14] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, 2015, pp. 448–456. URL http://jmlr.org/proceedings/papers/v37/ioffe15.html

[15] V. Nair, G. E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.

[16] D. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[17] Y. Bengio, I. J. Goodfellow, A. Courville, Deep learning, book in preparation for mit press (2015), URL http://www. iro. umontreal.ca/bengioy/dlbook 373 – 420.

[18] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[19] K. Cho, B. van Merrienboer, C¸ . Gulc¸ehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN ¨ encoder-decoder for statistical machine translation (2014) 1724–1734. URL http://aclweb.org/anthology/D/D14/D14-1179.pdf

[20] F. Chollet, Keras, https://github.com/fchollet/keras (2015).

[21] Theano Development Team, Theano: A Python framework for fast computation of mathematical expressions, arXiv e-prints abs/1605.02688. URL http://arxiv.org/abs/1605.02688

[22] M. X. Hoang, Y. Zheng, A. K. Singh, Forecasting citywide crowd flows based on big data, ACM SIGSPATIAL 2016, 2016. URL https://www.microsoft.com/en-us/research/publication/forecasting-citywide-crowd-flows-based-big-data/

[23] Z. Fan, X. Song, R. Shibasaki, R. Adachi, Citymomentum: an online approach for crowd behavior prediction at a citywide level, in: Proceedings of the 2015 ACM International Joint Confere

nce on Pervasive and Ubiquitous Computing, ACM, 2015, pp. 559–569.
[24] X. Song, Q. Zhang, Y. Sekimoto, R. Shibasaki, Prediction of human emergency behavior and th eir mobility following large-scale disaster, in: Proceedings of the 20th ACM SIGKDD internatio nal conference on Knowledge discovery and data mining, ACM, 2014, pp. 5–14.
[25] A. Abadi, T. Rajabioun, P. A. Ioannou, Traffic flow prediction for road transportation networks with limited traffic data, IEEE Transactions on Intelligent Transportation Systems 16 (2) (2015) 653–662.
[26] R. Silva, S. M. Kang, E. M. Airoldi, Predicting traffic volumes and estimating the effects of sh ocks in massive transportation systems, Proceedings of the National Academy of Sciences 112 (18) (2015) 5643–5648.
[27] Y. Xu, Q.-J. Kong, R. Klette, Y. Liu, Accurate and interpretable bayesian mars for traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems 15 (6) (2014) 2457–2469.
[28] P.-T. Chen, F. Chen, Z. Qian, Road traffic congestion monitoring in social media with hinge-los s markov random fields, in: 2014 IEEE International Conference on Data Mining, IEEE, 2014, pp. 80–89.
[29] Y. Li, Y. Zheng, H. Zhang, L. Chen, Traffic prediction in a bike-sharing system, in: Proceeding s of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Sy stems, ACM, 2015, p. 33.
[30] B. L. Smith, M. J. Demetsky, Traffic flow forecasting: comparison of modeling approaches, Jou rnal of transportation engineering 123 (4) (1997) 261–266.
[31] G. E. Box, G. M. Jenkins, G. C. Reinsel, G. M. Ljung, Time series analysis: forecasting and c ontrol, John Wiley & Sons, 2015.
[32] B. L. Smith, B. M. Williams, R. K. Oswald, Comparison of parametric and nonparametric mod els for traffic flow forecasting, Transportation Research Part C: Emerging Technologies 10 (4) (2002) 303–321.
[33] S. R. Chandra, H. Al-Deek, Predictions of freeway traffic speeds and volumes using vector aut oregressive models, Journal of Intelligent Transportation Systems 13 (2) (2009) 53–72.
[34] L. Florio, L. Mussone, Neural-network models for classification and forecasting of freeway traff ic flow stability, Control Engineering Practice 4 (2) (1996) 153–164.
[35] M. S. Dougherty, M. R. Cobbett, Short-term inter-urban traffic forecasts using neural networks, International journal of forecasting 13 (1) (1997) 21–31.
[36] G. P. Zhang, Time series forecasting using a hybrid arima and neural network model, Neuroco mputing 50 (2003) 159–175.
[37] G. P. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, European jo urnal of operational research 160 (2) (2005) 501–514.
[38] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks 61 (2015) 85 –117.
[39] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, An MIT Press bookin preparation. Draft chapters available at http://www.deeplearningbook.org/.
[40] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with regio n proposal networks, in: Advances in neural informationprocessing systems, 2015, pp. 91–99.
[41] A. Graves, A.-r. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: 2013 IEEE international conference on acoustics, speech and signal processing, IEEE, 2013, pp. 6645–6649.
[42] D. Yu, L. Deng, Automatic Speech Recognition, Springer, 2012.
[43] Q. V. Le, T. Mikolov, Distributed representations of sentences and documents., in: ICML, Vol.

14, 2014, pp. 1188–1196.
[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang,A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, ImageNet Large Scale Visual Recognition Challenge, International Journal of Computer Vision (IJCV) 115 (3) (2015) 211–252. doi:10.1007/s11263-015-0816-y.
[45] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[46] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Advances in neural information processing systems, 2014,pp. 3104–3112.
[47] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, W.-c. WOO, Convolutional lstm network: A machine learning approach for precipitation nowcasting, in: Advances in Neural Information Processing Systems, 2015, pp. 802–810.
[48] Y. Zheng, Methodologies for cross-domain data fusion: An overview, IEEE transactions on big data 1 (1) (2015) 16–34.
[49] Y. Zheng, F. Liu, H.-P. Hsieh, U-air: When urban air quality inference meets big data, in: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2013, pp. 1436–1444.
[50] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, T. Li, Forecasting fine-grained air quality based on big data, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 2267–2276.
[51] L. X. Pang, S. Chawla, W. Liu, Y. Zheng, On detection of emerging anomalous traffic patterns using gps data, Data & Knowledge Engineering 87 (2013) 357–373.
[52] X. Yi, Y. Zheng, J. Zhang, T. Li, ST-MVL: filling missing values in geo-sensory time series data, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016, 2016, pp. 2704–2710. URL http://www.ijcai.org/Abstract/16/384
[53] X. Song, H. Kanasugi, R. Shibasaki, DeepTransport: Prediction and simulation of human mobility and transportation mode at a citywide level, IJCAI, 2016.
[54] Q. Chen, X. Song, H. Yamada, R. Shibasaki, Learning deep representation from big and heterogeneous data for traffic accident inference, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.