Chapter 5

Data Preparation

Kristian Henrickson*, Filipe Rodrigues† and Francisco Camara Pereira†
**Department of Civil and Environmental Engineering, University of Washington, Seattle, WA, United States, †Department of Management Engineering, Technical University of Denmark (DTU), Lyngby, Denmark*

Chapter Outline

1 INTRODUCTION

This chapter is designed as an introduction to data preparation, with a focus on nonconventional data sources such as probe vehicles, internet, and web services. These nonconventional data promise to support new analysis methods which can provide a better understanding of our transport systems, and they are rapidly making their way into mainstream transport engineering, analysis, and decision making. However, such data come with a number of new challenges associated with their size and/or complexity, new acquisition channels, and added quality control considerations.

In a general sense, data preparation is the process of reading, organizing, transforming, and quality checking, to turn raw data into something accessible and useful in subsequent analysis. Of course, the end goal is to support transport analysis which accurately predicts or explains some aspect of the physical transport system. This goal is best served by a data preparation framework that is designed and documented to insure that the result is truly representative of the underlying system, and that the processing steps applied are transparent and interpretable to the end user. Developing such a framework requires an understanding of the nature of these emerging data sources and associated

Mobility Patterns, Big Data and Transport Analytics. https://doi.org/10.1016/B978-0-12-812970-8.00005-1

challenges. Toward this end, this chapter also serves as a broader introduction to several important and increasingly prevalent transport data sources. Additionally, extracting, structuring, and preparing these data will require a familiarity with the standard tools and methods for dealing with large-scale structured and unstructured datasets. Providing an introduction to these data sources, and to essential tools and methods required to prepare them for analysis, is the goal of this chapter.

This chapter is organized as follows. First, we describe some popular software tools for data management, processing, and analysis, and provide an introduction to web data formatting and extraction. Next, we introduce and discuss probe-vehicle based traffic data, with an emphasis on formatting and location referencing, data sources and characteristics, and quality control methods. Next, we introduce context data, that is, data which relates to the physical and human context in which the transportation system operates. This includes weather, web and social media, and event data, with a particular emphasis on internet sources and web services.

2 TOOLS AND TECHNIQUES

This section briefly introduces some important software tools and methods for managing and analyzing data relevant to traffic analysis. Specifically, we introduce popular scripting and analysis software, database management systems, common markup languages and tools for text and web data. We emphasize free and/or open source tools, as these typically provide the cheapest, most interoperable, and generally most accessible options for getting started with nonconventional transportation data.

2.1 Scripting and Statistical Analysis Software

A great variety of programing/scripting languages and analytical tools are available. Selecting the best alternative for a particular application will depend on a range of factors including the expertise of the development team or individual, the development vs execution time trade off, and the features of the analytical libraries currently available. Here we introduce a few popular choices for data processing and statistical analysis, with a focus on open source tools.

2.1.1 Python

Python is a popular object-oriented interpreted scripting language, and has been widely applied in statistical computing, machine learning, web development, and other applications. Having been designed to facilitate rapid development and human-readable code, Python is dynamically typed, relatively easy to learn, and very flexible. Python is open source and free (even for commercial applications), and it supports a wide variety of data processing, analysis, and visualization tools through third-party packages. Compared to R, Python is more of

a general purpose language, and may be preferable when performance becomes an issue or when data processing/analysis code needs to be integrated into a larger software framework. Some popular data analysis and statistical packages are listed below.

Pandas is a data analysis and modeling library that is very useful for analyzing and summarizing tabular data. At the core of Pandas are the DataFrame and Series data structures, which simplify restructuring, aggregating, indexing, and summarizing large datasets with mixed data types.

Scikit-learn is a statistical and machine learning library which offers a variety of powerful tools for regression, classification, clustering, and other methods.

Statsmodels is another library for statistical analysis and processing and, compared to scikit-learn, is focused more on statistical methods rather than machine learning.

SciPy is a general purpose library for scientific and engineering computing. In addition to core functionality such as numerical integration and optimization, SciPy includes a number of other Python packages including Pandas, Numpy (a popular numerical computing library) and Matplotlib (a general purpose plotting library).

2.1.2 R

R is an open source interpreted language that is focused on statistical analysis and visualization. It is widely used in data mining, statistics, and general data analysis and visualization tasks, and much of its core functionality and third-party packages are developed by the statistical community. R was initially developed for data analysis and statistical modeling, and it is supported by a large and active community of developers. Compared to Python, R offers much greater breadth of functionality in terms of new and novel statistical methods, domain-specific analysis tools, and data visualization. However, Python is more flexible and capable for general purpose programming. A few popular R packages for general data analysis are introduced briefly below.

dplyr is a popular R package for manipulating and restructuring data frames, the core tabular data structure in R, and database tables.

stats is a general package for statistical calculations, and includes a variety of modeling tools supporting time series analysis, generalized linear models, and statistical testing. The stats package is part of the base R installation, and can be used in R "out of the box."

forecast is a popular R package for multivariate time series analysis. This package includes the very useful auto.arima function for automatic order selection and training of a univariate ARIMA model.

ggplot2 is the most popular visualization package in R. Much of the details are abstracted away, which makes even complex plots relatively easy to construct.

2.1.3 MatLab

MatLab is a commercial mathematical computing environment and interpreted programming language. It is often compared with R due to its rich numerical and statistical computing functionality, though the MatLab environment is preferable in some cases for general programming tasks. Packages in MatLab are often more mature and robust than their R counterparts, but the software is quite expensive and each additional package represents added licensing costs. As is often the case for commercial software, the documentation and developer support for MatLab tends to be better than that for open source alternatives. As a general purpose programing language, MatLab is not as flexible, interoperable, or, in some cases, fast as Python. Both R and Python can achieve numerical computing performance comparable to that of MatLab, though this varies significantly depending on which tools/packages are used, and the ability of the user to parallelize and optimize code.

2.2 Database Management Software

A database management system (DBMS) is an indispensable tool for storing, organizing, and interacting with large volumes of data. In addition to providing an interface for programmatic remote access to data, a DBMS offers a range of desirable features including transaction management, user access controls, and others. A relational DBMS or RDBMS is the most common type of DBMS, where the word *relational* refers to the relationships between different entities represented in the database.

A relational DBMS is a structured data management framework, and it enforces whatever structure is specified by the user. For this reason, database design or *data modeling* is a crucial step in the development of any database with nontrivial structure. A data model describes the structure, relationships, and constraints in a database, and should be developed prior to building and populating tables. A data model is important both to plan and clearly specify the structure of a database, as well as to communicate this structure to potential users such as programmers and analysts. Using domain knowledge and information provided by the data provider, a data model should be designed with consideration of query efficiency, indexing, data integrity under ongoing data inserts, and updates, and deletions, as well as possible future changes to the geographic scope of the data.

Here we discuss some of the most popular database solutions and their associated strengths and weaknesses. With the exception of Microsoft SQL Server, all of the Relational Database Management Software described below runs on both windows and Linux (Linux support for Microsoft SQL Server is present in the 2017 version).

2.2.1 MySQL

MySQL is a very popular Relational Database Management Software (RDBMS) maintained by Oracle, and is available in both commercial and free (under the General Public License) versions. The open source Community edition does not include enterprise features such as encryption or sophisticated thread management, but it is an easy to use and surprisingly powerful solution for a variety of database applications. Current versions support geospatial data types and spatial indexing. Being relatively simple to set up and use, MySQL community edition is recommended for those getting familiar with the use of RDBMS, and in general for transactional rather than analytical applications. For more complex databases, or those requiring more sophisticated analytical capabilities, it may become necessary to switch to a more powerful solution such as PostgreSQL.

2.2.2 PostgreSQL

PostgreSQL is among the most powerful open source RDBMS available and, while not as popular as MySQL, it is widely used in commercial applications. PostgreSQL supports geospatial data types and indexing through the PostGIS extension, and offers excellent analytical query capabilities compared to other open source alternatives (e.g., MySQL). PostgreSQL is recommended as a general purpose RDBMS and, being somewhat more complex to set and use, in particular for complex databases or those requiring more powerful analytical features. Further, the geospatial capabilities available in PostGIS make PostgreSQL one of the best, if not the best, RDBMS solutions for managing and analyzing spatial data.

2.2.3 Commercial DBMS

There are a number of very powerful commercial DBMSs available, some of which provide very useful tools and substantial performance advantages over open source alternatives. For example, Microsoft SQL Server provides excellent versioning tools, sophisticated multithreading, and of course excellent support for other Microsoft products such as Azure cloud services. Other enterprise-scale commercial options include Oracle RDBMS, IBM DB2, and Sybase ASE. In general, commercial DBMSs are quite expensive, and may not provide significant advantages for many users. Microsoft Access is a less expensive and user-friendly option that is often used in business environments, but comes with a number of limitations (most notably a database size limit of 2GB) that make it less applicable for dealing with large datasets. Commercial DBMSs have the advantage of support provided by the manufacturer, though open source alternatives often have a very active and helpful user-base.

2.2.4 NoSQL Data Management

NoSQL or "Not only SQL" is a broad concept encompassing a range of database technologies. In general terms, NoSQL data systems are designed to support management and analysis of very large data, high-velocity streaming data, and/or data that does not fit into the conventional relational model. Much of the current generation of distributed and NoSQL data processing and management software are designed to allow databases to scale out rather than scale up. That is, rather than purchasing a small number of very powerfully servers, the storage and processing load can be distributed to a large number of commodity computers. Such systems are designed for redundancy and easy recovery, making it possible to continue operating as normal with the loss of one or even several machines in the network.

The benefits of a NoSQL data system vary by application and software, but generally include rapid scaling, high data throughput, built-in redundancy and recovery, and cost advantages associated with the ability to scale incrementally rather than make big upfront hardware investments. However, such systems have a number of characteristics that may be more or less problematic depending on the application. For example, NoSQL data systems often place less emphasis on data governance, and store data in an unstructured or variable schema form that may lead to quality and consistency issues. This may be an advantage for some applications which need to ingest large volumes of data with variable or undefined structure, but undesirable for maintaining the integrity of well-structured data. Also, unlike most relational DBMSs, many NoSQL systems do not support full ACID (Atomicity, Consistency, Isolation, and Durability) compliance. Such systems are described as "eventually consistent," which means that the result of a query may not reflect the most recent changes to the data. Other possible drawbacks of NoSQL databases include the use of lower level, declarative query languages and limited support for complex queries and joins.

Examples of NoSQL databases include column stores like Apache Cassandra and Amazon Redshift, Document databases such as MongoDB and CouchDB, and key-value stores like Berkeley DB and Amazon DynamoDB. The easiest and most cost effective way to get started with NoSQL is on a managed cluster through one of the several available platform-as-a-service providers. For example, Amazon provides a variety of NoSQL managed server options on AWS, as does Microsoft through Azure. However, unless one has access to an existing computing cluster and the staff expertise needed to build and use NoSQL databases, significant upfront monetary and time investment will be needed get started with NoSQL at scale. With careful indexing and a database server of reasonable performance, conventional relational database software will perform adequately for nearly any scale of interest for transportation analysis.

2.3 Working With Web Data

When working with web data, the most popular data formats that one is likely to encounter are XML, JSON, HTML, and plain natural language text. XML is a markup language that defines a set of rules for encoding objects in a format that is both human-readable and machine-readable. JSON is an open-standard format that also seeks human and machine-reliability for encoding objects. However, in JSON these objects consist of attribute-value pairs. Table 1 shows a sample of weather data encoded in the XML and JSON formats.

XML and JSON formats will be encountered mostly when gathering data from RSS feeds or APIs. In order to retrieve data from an RSS feed or an API, the first step is to construct the request URL with all the desired parameters. This varies considerably from source to source. Therefore, a careful read of the documentation is necessary. Moreover, most APIs require the users to sign up for an API key, which in many cases is free. Given an URL address with the request, it is then trivial to retrieve the data corresponding to the request using, for example, a popular programming language like Python. In Python 3.5, retrieving the data for the upcoming events in New York City can be done in just two lines of code using the requests library (Reitz, 2018) as shown in Fig. 1.

The retrieved data then needs to be parsed in order to make it machine-readable. Fortunately, many of the most common programming languages

TABLE 1 XML and JSON Examples

XML	JSON

<?xml version="1.0" encoding="unicode-escape"?>
<weather>
<timestamp>2017-01-20 12:28:32</timestamp>
<condition>Sunny</condition>
<temperature>10 °C</temperature>
<humidity>85%</humidity>
<wind>NNE 2 mph</wind>
</weather>
	{
	"weather": {
	"timestamp": "2017-01-20 12:28:32",
	"condition": "Sunny",
	"temperature": "10 °C",
	"humidity": "85%",
	"wind": "NNE 2 mph"
	}
	}

```
import requests
resp  = requests.get('https://world.timeout.com/api/events?lat_low=40.4& \
                                lat_high=41.0&lon_low=-74.9&lon_high=-72.9').text
```

FIG. 1 Example of an API request in Python.

```
import json
data = json.loads( resp )
print ("The temperature is: {}".format( data["weather"]["temperature"] ) )
```

FIG. 2 Example of JSON parsing in Python.

```
from bs4 import BeautifulSoup
soup = BeautifulSoup(resp, features = "html.parser")
print ("The temperature is: {}".format( soup.weather.temperature.text) )
```

FIG. 3 Exampling of XML parsing in Python.

provide good support for this. As an example, suppose that the "resp" object contains the JSON content in Table 1. Using the "json" package of Python (Python Software Foundation, 2018), one can easily process the JSON to obtain and print out the temperature using just 3 lines of code, as shown in Fig. 2.

The same could be done in the case of the XML version of data, by using the "BeautifulSoup" package from Python (Richardson, 2017), as shown in Fig. 3.

As the examples above demonstrate, it is fairly straightforward to access an RSS feed or an API. However, many websites don't provide such good interfaces. In these cases, one might need to resort to a technique called "screen scraping." In this context, screen scraping refers to the process of obtaining the source HTML code of a web page, parsing it, and extracting the desired content from it (although, legal terms of the website may apply). If using Python, one can once again rely on the "BeautifulSoup" package for this procedure, although the difficulty of scrapping the contents of a web page depends significantly on how well structured it is. For example, for a properly structured web site like AllEventsIn, retrieving the description of an event can be as easy as the example shown in Fig. 4.

Unfortunately, for poorly structured websites, extracting the relevant information from the source HTML code can be quite cumbersome. To make things worse, the structures of web sites change rather frequently. Hence, a perfectly working scrapper today, tomorrow may not work anymore.

In many cases, the use of regular expressions can significantly ease the process of scrapping the contents of a web page. A regular expression (regex or regexp for short) is a special text string for describing a search pattern. Regexes make use of special characters or symbols to represent certain patterns. For example, the symbol "." is used to match any character in a string of text, and the symbol "*" is used to indicate that the previous pattern can repeat many times or be absent. By exploiting these and other features of regexes, one can,

```
soup = BeautifulSoup(html, "html.parser")
description = soup.find("p", {"property":"schema:description"}).text
```

FIG. 4 HTML scraping example in Python.

for example, use the following regex to match all the emails in a text: "[A–Z0–9._%+−]+@[A–Z0–9.−]+\.[A–Z]{2,}". For further details on how to use regexes, please refer to (Goyvaerts, 2016). Using regexes in Python is quite easy through the "re" package.

3 PROBE VEHICLE TRAFFIC DATA

Mobile GPS and communications devices are an increasingly popular source of link-level speed and travel time data for transportation planning and analysis. A number of commercial providers offer a version of this type of data, typically as a combination of road link definitions, GIS file(s) specifying road link locations and topology, and a collection of travel time or speed measurements for each road link. Two of the world's leading providers of probe vehicle based traffic data products are INRIX, based in the United States, and HERE, based in the Netherlands. Throughout this discussion, repeated references to the National Performance Monitoring Research Dataset (NPMRDS) will be made. This is a large probe vehicle-based traffic dataset covering the national highway system in the United States, acquired by the Federal Highway Administration for use in performance measurement and management. The first version of the NPMRDS was provided by HERE North America, but since mid-2017 has been provided by a consortium of academic and industry partners including INRIX. Rather than describe the specific product offerings from different providers, which in general will be determined by a procurement agreement, a general introduction to the concepts, terminology, and data characteristics is given here.

3.1 Formats and Protocols

Historical traffic data will typically be delivered in the form of large text files containing traffic records, GIS shapefile(s) containing the road network, and TMC lookup table(s) describing the TMC segments and their relationship to the geospatial data. Due to the size and relational structure of link-level speed and travel time data, spread sheets and text files are not suitable data management and analysis tools. In most cases, relational database management software with geospatial extensions should be used, along with scripting and data analysis tools such as Python, R, and/or Matlab. Real-time speed and travel time data will typically be delivered via web service in the form of JSON or XML, as a response to a user request. Such data will typically require some scripting to retrieve and format for storage, analysis, and presentation.

With a relational table structure and industry standard units, working with commercial traffic data is similar to working with more conventional traffic data sources (e.g., fixed mechanical traffic sensors). The primary difference, and source of complexity when integrating such data into an existing traffic database, is in the location referencing methods used. While conventional traffic data is often encoded using linear referencing, commercial probe vehicle

data is generally referenced to locations using a combination of (a) predefined road segments defined in geospatial terms, and (b) traffic records which are linked to road segments through road link-specific codes. The Traffic Message Channel (TMC) is the most widely used location encoding standard in commercial traffic data, and nearly all providers offer TMC-based products. As an alternative to the TMC standard, TomTom develops and maintains the OpenLR standard for location referencing, an open source standard which offers greater coverage and flexibility than TMC-based referencing. Some vendors develop and maintain their own location referencing tables based on the TMC standard (e.g., OpenLR-compliant INRIX TMC XD segments), and in some cases use an alternate encoding in addition to TMC codes to enable sub-TMC resolution. A brief discussion of the two leading standards for traffic data location encoding is given in the following two subsections.

3.1.1 TMC Codes

Common to this type of data is the use of Traffic Message Channel (TMC) codes, a unique identifying code used to assign traffic information to specific points or segments on a road network. Initially developed for encoding and transmitting traffic information on radio side bands, TMC codes are now widely used in a variety of wireless and web-based traffic information systems. The Traveler Information Services Association (TISA) maintains the TMC standard, and is responsible for making sure that new TMC definitions conform to this standard. TMC definition tables (conforming to TISA standards) are maintained by various organizations for many countries around the world, and are licensed to map makers and data vendors for use in their services. Due to the use of different TMC tables, as well as the flexibility in the geometric interpretation of a TMC definition, significant differences may be present in the TMC segment geometry used by different providers.

TMC codes are used to assign traffic data to a road segment or point for both real time web service calls and in archived historical data. For a set of geospatial data describing a road network, a TMC may refer to one or more consecutive road segments, and each segment may be associated with zero or more TMCs. For example, the NPMRDS includes a GIS shapefile describing the traffic network and a TMC lookup table relating TMCs to road segment IDs in a many-to-many relationship. A few important notes about the use of TMC tables for encoding and maintaining traffic data are given below.

1. TMC tables used by a specific data vendor are proprietary, and do not cover every road segment. Typically, coverage is greatest on freeways and decreases for each subsequent functional class.
2. TMC length varies significantly, even within a single area and roadway functional class. There is pressure to develop TMCs with finer granularity, but once defined TISA standards require that a TMC should only be removed if the corresponding road segment is taken out of service.

3. TMC support for freeway ramps varies between providers, and in most cases traffic data will not be available for all ramps.
4. As with any such standard, significant time is required to develop, approve, and implement new TMC definitions. This is often cited as a drawback of TMC-based encoding.

3.1.2 Open Location Referencing

TomTom launched the Open Location Referencing (OpenLR) project in 2009 to develop an open source standard for location referencing, and has since been pushing for widespread industry adoption. Unlike TMC representation, OpenLR enables dynamic location encoding and a great deal of flexibility for representing different types of geospatial data including 2-dimentional shapes, points, and nonroad lines (TomTom, 2012). In OpenLR, a road segment is represented as a series of waypoints that can be combined, along with bearing and road attributes (functional class and way type), to represent a travel path. Even in a system that uses OpenLR for location encoding and exchange, the data are represented fundamentally as road link-level records (possibly with greater coverage and granularity than TMC encoding).

OpenLR has a number of advantages over TMC encoding. Here we list a few key considerations for data encoded using this standard.

- OpenLR segments can represent any road segment, as well as geographic features that are not associated with the road network. That said, the road segments and time periods for which traffic data is available will still vary by vendor and product.
- OpenLR can represent the traffic network at higher granularity than TMCs in general, but the resolution of the underlying data is still limited to that of the database maintained by each vendor.
- OpenLR is a dynamic standard, and as such requires no time to develop and approve a new segment definition as is the case for TMC segments. However, a certain amount of work and time delay should be expected for a given vendor to include a new road segment in their traffic data products.

3.2 Data Characteristics

3.2.1 Data Sources

Data sources vary between different vendors and within vendors by product. In cases where multiple sources are combined to create a single aggregate record, the exact source(s) of any particular record may not be presented to the end user. Instead, users will be given a single aggregate value representing the vendor's best guess at the true traffic state. The primary sources of commercial traffic data and briefly summarized below.

3.2.1.1 Mobile Location Services

For certain providers, much of the data they collect is from mobile location services. That is, when an individual uses a contributing mobile phone application and agrees to share their location data, this data is automatically shared with the owners of the application and used to generate traffic statistics. This type of crowd sourcing is used by nearly all commercial traffic data vendors, though the set of applications contributing data will vary between vendors. In some cases, application users are provided with some mechanism to opt out of data reporting.

3.2.1.2 Consumer GPS Devices

Consumer GPS units, including both stand-alone units and in-vehicle systems, provide a significant percentage of the traffic data available from commercial vendors. TomTom, for example, is a well-established manufacturer of consumer GPS units, and both their historical and real time data products are obtained (at least in part) through such devices (TomTom, 2011, 2014). Users contribute data by uploading their device GPS logs or in real-time via communications-enabled GPS units. Although commercial vendors are not always entirely forthcoming about all of their data sources, both HERE and INRIX source user GPS data through partners in the automotive and consumer GPS industries (also note that HERE is majority owned by a consortium consisting of BMW, Daimler, and Volkswagen).

3.2.1.3 Commercial Vehicle Transponders

Commercial vehicle fleets were one of the first sources of commercial probe vehicle data and, despite the ubiquity of mobile computing and consumer GPS, remain a crucial resource. Commercial fleets including taxis, shuttles, delivery vehicles, and freight trucks are often equipped with transponders which often contain a GPS receiver. By contracting with fleet operators, some commercial traffic vendors are able to obtain the data generated by these transponders. In addition to providing a reliable and accurate source of vehicle GPS data, transponder data can target a specific vehicle subpopulation. For example, through a partnership with the American Trucking Research Institute, freight truck transponder data was used to generate truck-specific travel times that were included in the NPMRDS.

3.2.1.4 Other Sources

Fixed traffic sensors, including inductance loops, traffic cameras, and others are listed among the data sources used by several commercial data vendors. For example, several traffic data vendors collect incident and congestion information based on manual observation of traffic cameras, public agency alerts, and emergency reporting systems. HERE runs several reporting centers in the United

States and Europe to collect such data, while TomTom contracts with third parties for this purpose.

3.2.2 Granularity

As noted previously, TMC segment lengths vary significantly even within a single region and roadway functional class. Most vendors have made efforts to move beyond the limitations of the TMC representation to achieve greater coverage, granularity, and responsiveness. In 2013, INRIX introduced the TMC XD encoding, which is maintained exclusively by INRIX (though it does conform to the OpenLR standard) and offers greater coverage and granularity compared to standard TMC definitions. While TMC segments are often several miles or more in length, TMC XD segments have a maximum length of 1.5 miles in the United States (INRIX, 2014). TomTom products are offered in both TMC and OpenLR encoding. Although HERE uses TMC encoding, they utilize an offset scheme to provide additional flexibility and detail. As of 2016, the NPMRDS contained 385,000 TMCs with an average length of 1.74 miles (Pu, 2018).

3.2.3 Vendor Quality Control and Imputation

The quality and completeness of link-wise probe vehicle data varies between vendors and procurement process. For example, the standard product offered by INRIX provides a fully complete data series for all TMCs, part of which represents imputation performed by the company. On the opposite end of the spectrum, the NPMRDS was specified in the procurement contract to contain only measured values, with no imputation or processing applied. Because of this, outliers are present and a great number of observations are missing.

Unprocessed and incomplete data may seem undesirable at first glance, but it allows the user to retain more control over the quality control, imputation, and uncertainty estimation which can result in more informed analysis and decision making. Some providers do include information that can aid in quality assessment in both web service responses and archived historical data. For example, in the I-95 vehicle probe project carried out by the University of Maryland, a confidence metric was developed which considers vehicle density and likelihood of current observation conditioned on the previous 45 min and on a longer time window based on historical data (INRIX, 2014). However, much of the processing applied to commercial data sets is typically not presented to the end user, which complicates any uncertainty and error assessment.

3.3 Challenges

3.3.1 Completeness

Data completeness, or missing data, is a significant challenge in a variety of scientific and engineering analysis. Probe vehicle data in particular relies on

observations from contributing vehicles on the transport network, and so has unique missing data challenges compared to, for example, mechanical sensor data. This primarily relates to the fact that higher volume, slower traffic conditions are more likely to be represented in the dataset, and so there is the potential for differences in speed or travel time distributions between the missing and observed data.

For most commercial probe vehicle data, the number and granularity of TMC-based observations has increased significantly in the last 5 years, as has the per-month average data completeness (Hosuri, 2017). This was illustrated in (Cambridge Systematics and Texas Transportation Institute, 2015), which analyzed NPMRDS travel time data for a set of 10 eastern US states from 2014. They showed that travel time data tends to be most complete during daytime hours and least complete during the night/early morning hours. For the states analyzed in the Cambridge Systematics study, interstate highway segments were approximately 58% complete for all time periods and vehicle types, and 22% Complete for all other highway classes in 2014. (Kim and Coifman, 2014) analyzed probe vehicle data from INRIX and compared it with loop detector data on a single corridor over a period of 2 months in 2014. They found that, despite data being complete for all sampling periods, there we a great number of instances of repeated measurements. This suggests that, at least at the temporal resolution at which the data is provided, the completeness was significantly lower than 100%.

Regardless of whether imputation is completed by a commercial data provider, by the end user, or not at all, it is clear that missing data will have a significant impact on the accuracy and usefulness of this data source, which underscores the importance of how such missing values are dealt with. Previous research has shown that the quantities of interest, both missing and observed, are correlated with the probability of being missing. Further, for data that is observed, there is a clear relationship between penetration rate, sampling frequency, and accuracy of the derived traffic measures (Bucknell and Herrera, 2014; Patire et al., 2015). All else being equal, longer segments are less likely to produce missing traffic records, but provide lower granularity in all records.

3.3.2 Data Quality and Accuracy

Due to the commercial nature of third party probe vehicle data, there are some restrictions on how validation and comparative analysis can be conducted and presented. Here, we summarize some previous work on the accuracy of probe vehicle data to give a general sense of the challenges associated with this type of data.

One notable study, part of the I-95 Corridor Coalition Vehicle Probe Project conducted by the University of Maryland, assessed the accuracy and bias of commercial probe vehicle data on different road types in the eastern United States (Young et al., 2015). The findings of this work indicate that probe vehicle

data is most reliable on roadways with low signal density, low access point frequency, and relatively high annual average daily traffic (AADT). The authors of this study state that probe vehicle data is not suitable for roadways with signal density greater than two/mile, roadways with annual average daily traffic (AADT) <20,000, or with fewer than two lanes per direction. However, it should be noted that this report was based on 2013–14 data, and there have been significant changes in the commercial traffic data products available since that time. In any case, some evaluation work will likely be needed prior to applying probe vehicle data to analysis on lower volume roadways or those with higher signal density or access point frequency. (Haghani et al., 2015) presented a validation procedure for probe vehicle data, and demonstrated their approach using commercial travel time data. However, their work was based on a small sample of road segments, relatively old data (from 2012), and did not really offer any overall quality assessment. (Florida Department of Transportation, 2012) completed some validation and comparison work on several different commercial data sources, again based on older data (2011) and without any far reaching conclusions. Other work (Kim and Coifman, 2014; Wang et al., 2018) indicates that common quality issues include significant data reporting time lag as well as periods of unexplainable drops in reported speed during generally free-flow conditions.

For the purpose of assessing highway performance and, more importantly, freight mobility performance, it is important to consider the difference between low travel speeds due to congestion and those attributable to other factors. For example, on a long inclines or declines many heavy vehicles will often travel well below the posted speed limit simply due to the vehicle performance characteristics or safety considerations, and the speeds reported by such vehicles does not represent the true operating state of the roadway (Hallenbeck and McCormack, 2015; Bitar, 2016) Further, slower moving vehicles are more likely to be present on a given road segment in multiple five-minute reporting intervals, and so travel time or speed values based on a mix of passenger and freight vehicles will often be biased toward slow moving heavy vehicles. Even on relatively flat roadways, it is important to remember that speed or travel time data derived from probe vehicles represents a collection of measurements from a sample of all vehicles on the road segment. Because slower moving vehicles spend more time on each segment, reported traffic parameters may tend to be biased toward these lower vehicle speeds. This issue is of course largely driven by the data sources used by any particular vendor, as well as the processing methods applied to generate the final values presented to the customer.

It should be noted that a major independent study analyzing and comparing the accuracy of different commercial traffic data products has not been completed within the last two years. Thus, because the quality and coverage of commercial traffic data has evolved along with the technology supporting it, determining the suitability of such data for a given application may require some preliminary assessment on the part of the user.

3.4 Data Preparation and Quality Control

3.4.1 Data Loading

Prior to data loading, the database should be designed with respect to, at minimum, table definitions and structure, primary and foreign key definitions, and data types. Ideally, a data model describing the structure, relationships, and constraints in a database should be developed prior to building and populating tables. A data model is important both to plan and clearly specify the structure of a database, as well as to communicate this structure to potential users such as programmers and analysts. Using domain knowledge and information provided by the data vendor, a data model should be designed with consideration of query efficiency, indexing, data integrity under ongoing data inserts, updates, and deletions, as well as possible future changes to the geographic scope of the data.

Some form of user interface and data loading utilities are available for most commercial relational database management software. However, especially when historical data is received at regular time intervals, it is advisable to develop scripts to automate the data loading process. Two possible data loading approaches are briefly outlined below.

Option 1: load CSV files into a temporary database table using a data import utility. Write a SQL statement (possibly a stored procedure) to convert data types and insert rows into the final table. This option will be the easiest for non-programmers, but will require a significant time investment if many files are to be loaded (though much of this process can be automated using command line tools).

Option 2: read text files using a scripting language (e.g., Python), convert data types and insert rows into final database table. The "csv" package in Python simplifies reading of csv files, and a variety of packages are available for database connectivity (e.g., psycopg2).

3.4.1.1 Geospatial Data

For historical data, vendors will generally provide regular updates to TMC/road link definition and GIS files. Geospatial vector data is typically encoded using one of (a) Esri shapefiles, (b) Well Known Text (WKT) strings, or (c) GeoJSON text strings. For real-time web service data, spatial information will generally be encoded as (b) or (c).

Platform-specific tools, such as shp2pgsql for PostgreSQL, can be used to convert Esri shapefiles into the appropriate spatial database representation. Alternatively, the Python package "pyshp" is a useful library for reading and writing Esri shape files, though this will not be needed if Esri software is used. WKT and GeoJSON can be inserted directly into a PostGIS or MySQL spatial database, using the ST_GeomFromText or ST_GeomFromGeoJSON respectively, though the user must supply an SRID number designating the coordinate system.

```
ST_GeomFromText('LINESTRING(-117.411234 47.664433,-117.413221 47.660511)',4269)
```

```
ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[-117.411234, 47.664433],[
-117.413221 47.660511]]}',4269)
```

FIG. 5 Examples for inserting WKT and GeoJSON data into PostGIS.

The examples in Fig. 5 show the Spatial SQL statements which convert WKT and GeoJSON to spatial database geometry. Note that both of these statements encode the same information, geometry type (Linestring), coordinates, and SRID (4326). Of course, there are a great number of geometry types and coordinate systems that can be used, for more information see (The PostGIS Development Group, 2018).

3.4.1.2 Tabular Traffic Data

Historical traffic records are typically delivered in the form of comma or tab separated text files. Loading these files into a database is straightforward, and can be completed using database command line tools, graphical data loading utilities included with most database administration software, or a simple script written in Python or other language. In the author's experience, when using data import utilities, it can save time and difficulty to import tabular data with generic data types (e.g., variable length character) and perform data type conversions using SQL statements. This is because it makes it easier to identify and troubleshoot data type conversion failures.

Depending on the vendor and product, missing data may result in either (a) NULL or 0 values in the place of missing records, (b) entire record(s) missing from the data file, or (c) imputed values in the place of missing records. In order to facilitate analysis of missing data patterns, imputation, and other investigation, it may be desirable to add records such that all locations and time intervals are represented in the database, whether or not data is available. This can be accomplished in a database with a cross join or common table expression, or in a scripting environment. For example, in Python pandas, DataFrames (a table-like data structure) can be indexed by one or more columns, for example, date, time, and road link identifier (e.g., TMC). To fill in missing rows, a new index can be built using the Cartesian product of these three fields, which will create a row for each unique combination present in the DataFrame. An example of this is shown in Fig. 6.

3.4.2 Outlier and Error Detection

Due to a range of factors including GPS error, vehicle roadway matching and other conflation errors, mid-segment stops, and others, outliers and erroneous observations are often present in probe vehicle-derived traffic data. Depending on the data vendor and product, outliers filtering may have been completed before the data is presented to the customer. If not, or if additional filtering

```
import pandas as pd

#Read data file into pandas dataframe
traffic_data = pd.read_csv('/Documents/probe_files/december_2015.csv')

#Set index to date, time, and tmc columns
traffic_data = traffic_data.set_index(['dt_stamp','tm_stamp','tmc'])

#Create a new index from the Cartesian product of tmc, date, and time
index = pd.MultiIndex.from_product(traffic_data.index.levels)

#Re-index the existing traffic dataframe
new_traffic_data = traffic_data.reindex(index)

#Fill NA values with zero
new_traffic_data = new_traffic_data.fillna(0).astype(int)
```

FIG. 6 Completing a DataFrame in pandas using a Cartesian product index.

is deemed necessary, the user may undertake an outlier detection step before completing any analysis. The term outlier is used here to indicate observations which are not representative of the true traffic state, and usually take the form of excessively high or low recorded speed/travel time values. Other outliers, such as those that describe actual but anomalous traffic conditions, are often of interest for detecting and analyzing traffic incidents. However, this relates to analysis rather than data preparation, and as such is not considered here. The frame of reference for what constitutes "excessive" will shift by location, time of day, and with the occurrence of traffic incidents. For this reason, a simple high/low thresholding may not be sufficient to capture the bulk of erroneous observations.

3.4.2.1 Visual Analysis

The first step in outlier detection should be a visual assessment of the recurring traffic patterns and the spread or distribution of the data for the location(s) of interest. Several previous studies have recommended using 24-h overlay plots for outlier and missing data assessment, and the authors have found it a useful tool for illustrating the ordinary traffic patterns for a particular location and time period and generating summary statistics for incomplete and unprocessed traffic data (Kaushik and Ernest Young, 2015). A 24-h overlay plot is simply multiple days of data superimposed onto a single 24-h plot, such that the denser regions indicate more "typical" traffic patterns. Usually, an overlay plot is created using data for days with similar expected traffic patterns, for example, all mid-week days (Tuesday—Thursday) in a month or quarter (Kaushik and Ernest Young, 2015).

3.4.2.2 Rule-Based Outlier Detection

Using domain knowledge and physical constraints, a rule set can be developed to distinguish between reasonable and unlikely or impossible observations. One of the simplest outlier detection methods simply to set upper and lower

thresholds for "reasonableness," outside of which data is assumed to be anomalous. Thresholds can be set based on known characteristics of the road segment, for example, one could define reasonable travel time observations as those corresponding to speeds below 85 miles/h and above 5 miles/h on a freeway segment with a free-flow speed of approximately 65 miles/h. However, this is a very naïve approach, and one that would fail to capture a great number of problematic records. Generally speaking, a simple threshold method such as this is an important but insufficient step toward removing the majority of outlying and erroneous observations.

3.4.2.3 Statistical Methods

(Hodge and Austin, 2004) defines three general categories of statistical outlier detection methods. The first category is unsupervised methods, which essentially separate or cluster a dataset into normal and anomalous regions based on the overall distribution of the data. The second is supervised classification methods, which require manually labeled data and train a predictive model to classify observations as regular or outlying/erroneous. The third category is semisupervised methods, which are trained only on normal, nonanomalous data to characterize normal conditions, labeling observations that lie outside the threshold for normality as novel or outlying. Due to the magnitude of traffic datasets and difficulty associated with manually identifying errors, here the focus is on unsupervised and semisupervised (type 1 and 3) methods.

A number of outlier detection methods have been developed which regard an outlier as a record which falls in an unlikely region based on the local or overall statistical properties of the data. In one such approach, upper and lower percentile thresholds as set as the outlier cutoff based on the interquartile range (Laurikkala et al., 2000). Similarly, in the z-score method, unlikely regions are defined in terms of the normal distribution, by setting a maximum z-score as the threshold for outliers (Hodge and Austin, 2004). Such methods can be extended to consider local behavior in a time series, for example, by setting an outlier threshold based on the absolute deviation from the local median calculated on a moving window (Basu and Meckesheimer, 2007) or applying the interquartile range method to a moving window (Bhaskar et al., 2009). More sophisticated variants of this approach, including methods for multivariate and/or non-Gaussian data and those considering temporal patterns, have been applied to traffic data (Shekhar et al., 2001; Park et al., 2003).

Much of the existing work on unsupervised outlier detection methods relies on some form of clustering, and a variety of clustering methods have been applied to this task including k-nearest neighbors (KNN), Density Based Spatial Clustering of Applications with Noise (DBSCAN), and k-means. In essence, such methods define outliers as observations which do not belong to a sufficiently large and/or dense group of observations based on some user-defined criteria. As an example, one method introduced by (Ramaswamy et al., 2000)

computes the distance D_k between each observation and its kth nearest neighbor, and defines the top m observations with respect to D_k as outliers. For a traffic dataset consisting of a set of consecutive observations from multiple neighboring road segments, the simplest way to approach clustering-based outlier detection would be to ignore temporal dependencies and define an observation or record as a vector of observations from a collection of road segments at a point in time. To consider temporal dependencies, distance based methods such as (Ramaswamy et al., 2000) have also been applied to features representing temporal subsequences (Gupta et al., 2014). Incorporating both temporal and spatial dependencies in an informative way is somewhat more complex (Li et al., 2009).

It should be noted that most existing distance-based outlier detection methods scale poorly with the dimensionality of the data. For this reason, carefully defining the spatial and temporal neighborhood representing the context for what defines an outlier is important. In some cases, some form of dimension reduction (e.g., Principle Component Analysis) may be needed to reduce both the noise and dimensionality of the data.

3.4.3 Imputation

A body of literature has been devoted to missing data theory and imputation, very little of which can be covered here. Instead, we introduce a few common methods that have been successfully applied to missing traffic data, starting with a brief introduction to the topic of missing data patterns.

3.4.3.1 Missing Data Patterns

Most current work considers the occurrence of missing data under a probabilistic framework, with the pattern described by a statistical distribution (Rubin, 1976). The mechanism driving the missing data pattern is assumed to be ignorable if data is Missing At Random (MAR), which is only true when the distribution of missingness is independent of the missing values given the observed values. That is, if the dataset defined as X is constituted of both observed and unobserved components (X_{obs} and X_{mis} respectively) the probability that a value is missing depends only on X_{obs} as shown in Eq. (1) (Rubin, 1976; Schafer and Graham, 2002).

$$\Pr(\text{missing}|\, X) = \Pr(\text{missing}|\, X_{obs}) \tag{1}$$

Missing completely at random (MCAR) is a special case of MCAR in which the probability of missingness is independent of both the observed and missing data. In this case, the probability of an observation being missing given X is simply the probability of being missing independent of X as shown in Eq. (2).

$$\Pr(\text{missing}|\, X) = \Pr(\text{missing}) \tag{2}$$

If data is neither MAR or MCAR, the missing data pattern is Not Missing At Random (NMAR) or "nonignorable," a term used to indicate that the mechanism

driving missingness in the data cannot be ignored in developing an imputation model. Nonignorable missing data patterns are defined as a systematic difference between the distribution of the observed and missing values which cannot be fully described by the (observed) predictor values (Rubin and Little, 1987).

With respect to the MAR assumption in probe vehicle data, note that in order for a traffic record to available for a road link, a vehicle contributing to the dataset (e.g., through the use of a contributing mobile phone application) must be present for the time period of interest. Intuitively, it is clear that completeness will increase as the traffic density increases, with the result that more congested time periods are less likely to be missing all else being equal. Of course, the impact of missing data and imputation method will depend on the overall rate of missingness. All of the methods introduced in this section, and in fact most of the widely used methods and software tools for imputation, are based on the MAR or MCAR assumption and in fact, even when the assumption is violated, data is often assumed to be MAR. This simplifies the imputation process, and the MAR assumption can be made more plausible by including additional predictors that can help to describe the distribution of missingness (Schafer, 1997).

3.4.3.2 Univariate Methods

The simplest univariate imputation approach would be to fill in missing observations with the sample mean, free-flow speed or travel time, or other fixed value. Though such methods are common, they will reduce the variance of the dataset and better accuracy will generally be achieved by considering the sequence of observations as a time series in order to take advantage of temporal correlation. Simple time series imputation methods include linear interpolation, moving average, and last observation carried forward. However, the better imputation accuracy will generally be achieved by building a statistical time series model to predict missing observations based on the previous n.

The state space ARIMA model (using the Kalman filter) is a robust and relatively simple univariate approach to time series imputation, and has been widely applied in a variety of fields including transportation. Compared to a standard ARIMA model, the state space representation easily handles missing values in model training and provides additional flexibility, for example, by allowing time varying parameters (Ravichandran, 2001). The arima and Kalman filtering functions in the stats library can be used for state space time series modeling in the R programing language, and there are a number of time series and state space modeling tools available in the statsmodels Python library (Hyndman, 2016). A very simple solution to univariate imputation using a state space ARIMA model can be developed using the auto.arima function in the R stats library which, although not without potential pitfalls, performs automatic model order selection and fitting. The na.kalman function in the imputeTS package (Moritz, 2017) can be used in combination the auto.arima function to select and train a state space ARIMA model on an incomplete dataset and then fill in the missing

```
library(stats)
library(imputeTS)

complete_data <- na.kalman(incomplete_data, model = "auto.arima", smooth = FALSE)
```

FIG. 7 Univariate time series imputation example using imputeTS in R.

values with the filtered estimates as shown in Fig. 7. The imputeTS library includes a number of other functions for univariate time series imputation, see (Moritz, 2017).

Another approach that has been applied to traffic prediction and imputation is k-nearest neighbors (KNN) (Liu et al., 2008; Zhong and Ling, 2015). This is a nonparametric method that generates predictions (or imputations) by identifying the most similar past observation with respect to some feature vector according to a distance metric (e.g., Euclidian). In a univariate imputation setting, the feature vector is simply the n most recent observations, and imputed values are obtained by aggregating over outcome associated with the n nearest feature vectors in a historical database.

3.4.3.3 Multivariate Methods

The key difference between multivariate and univariate traffic data imputation methods is the consideration of spatial correlation. That is, in treating a sequence of observations as a univariate time series, any information that can be gained from neighboring locations is ignored. Likewise, failing to model a multivariate time series dataset as such will fail to take advantage of any temporal correlation that is present, and can lead to auto-correlated errors and other problems.

Some of the methods introduced as univariate can be easily adapted to the multivariate setting. For example, the univariate time series features used in a KNN imputation scenario can be expanded to represent two-dimensional, time-space features as described in (Tak et al., 2016). When missing values are present in all predictors (as is most often the case for multivariate traffic data), an iterative method may be used to cycle through each variable and replace the missing values that are present, with the missing predictor values initially filled in with the variable-wise mean or median. Other nonparametric predictive methods have been applied to impute spatially and/or temporally correlated variables, including regression trees, random forests (RF), and others, though few examples of such methods being applied to link-level probe vehicle data exist. This is primarily a feature engineering challenge, as these nonparametric predictive modeling methods are included with most common statistical software packages. Multivariate state space time series models have also been applied in a variety of transportation prediction tasks. Adapting existing code to the imputation task may require some work, but the FKF and dlmodeler libraries in R and statsmodels library in python provide most of the fundamentals.

A variety of other multivariate methods exist including matrix and tensor completion, various regression methods, etc. (Chiou et al., 2014; Li et al., 2014; Ran et al., 2015). The choice of methods will most often be driven by the type of data (e.g., real-time vs historical), time and resource constraints, and some application-specific assessment of imputation accuracy.

3.4.3.4 Multiple Imputation

Multiple Imputation (MI) is a category of methods used to replace missing values and quantify the uncertainty attributable to missing data. Originally introduced by Rubin and Little (1987) as a way to deal with missing survey and census data, MI has since been applied in a variety of science and engineering fields including transport (Ni et al., 2005; Henrickson et al., 2015). In general terms, MI is a Monte Carlo approach which replaces each missing observation with $m > 1$ replacement values, resulting in m complete datasets. Each of the imputed datasets are then analyzed using complete data methods, and the results are combined to give a final estimate and confidence bounds for the desired quantities (model parameters, predicted values, etc.) which incorporate both the uncertainty inherent in the complete data and the uncertainty due to the missing values.

There are a number of multiple imputation libraries in R, although in most cases the treatment of time series data is limited to including lag terms for the variables of interest. The Amelia library in R provides access to an expectation maximization algorithm for generating multiple imputations, and offers some tools for working with time series (limited in this case to lagged variables and polynomial/spline fitting (Honaker et al., 2016)). The MICE R library performs multiple imputation by chained equations (hence the name MICE), an iterative imputation method which, unlike joint modeling approaches such as that offered by the Amelia package, does not assume that the variables of interest follow a specified joint distribution (Buuren and Groothuis-Oudshoorn, 2011). This allows a good deal of flexibility in model specification by allowing different models to be specified for each variable, and a variety of common prediction models for real valued, nominal, and ordinal data types are supported.

4 CONTEXT DATA

4.1 The Role of Context Data

The increasing availability of sensor data, such as that described in Section 3, provides a unique potential for developing great tools for traffic management and traveler decision making. However, the complex role of human behavior in the transportation system demands considerations that might not be captured with sensors that are focused on the network or vehicles. For example, the traffic manager needs to understand why certain congestion is formed (Is it an incident? A special event? A religious ceremony? Weather? School pick-up/drop-off?)

and to predict how it will evolve. A special event leads to different patterns and management procedures than an incident or a flooding event. In other words, besides knowing that a problem exists, traffic managers and prediction systems need to know its context.

We define context as any available semantic information that can be associated to observations from the traffic-sensing system (for example, cameras, loop counters, and GPS probes). Context can be important to explain and help predict many transport-related phenomena. For example, a sudden demand peak in an area can be due to special events, religious activities, political demonstrations, street fairs; general demand pattern changes can be associated to school holidays; and nonrecurrent supply changes can be caused by incidents, roadworks, road blockages, and harsh weather. For example, (Pereira et al., 2015) use contextual data from the internet to explain nonhabitual transport overcrowding. On a somewhat different perspective, context can be used to analyze aspects transversal to behavior and transport, such as wellbeing (for example, sentiment analysis on public transport) or environment (online reports on emissions). Context mining is therefore complementary to traffic sensing technologies. While the latter provides information on what is happening in traffic, the former helps understand why. When properly aligned in space and time, they become essential to understand how traffic will evolve, by becoming inputs to transport-prediction algorithms.

Knowing context is particularly relevant in nonhabitual scenarios. While in recurrent scenarios, traffic managers and commuters are aware of their evolution and available options, in nonrecurrent ones, they need good predictive capability to make decisions. Adding semantic causal information to the prediction process together with observation should contribute to improving its accuracy, especially in nonrecurrent scenarios.

4.2 Types of Context Data

There are several types of context data covering different external factors that can play a role on transport systems, such as information about weather, incidents, roadworks, road blockages, special events, natural disasters, acts of terrorism, etc. In the following, we describe a few of these data types individually and discuss some potential sources for obtaining this data.

4.2.1 Weather Data

Weather data is perhaps the most well-known type of contextual data. While in certain areas of the globe the weather has a small effect on the transport system, in other areas accounting for weather information is critical. For example, in colder regions accounting for the presence of snow or ice is essential from both a management (e.g., displaying alerts, dispatching snowplows or spreading salt) and a modeling perspective (e.g., predicting travels times). Similarly, it is well

known that precipitation affects road safety by increasing accident frequency and congestion, especially during peak hours (Koetse and Rietveld, 2009), and that dense fog conditions or the positioning of the sun in sky can cause poor visibility conditions, thus making drivers slow down. Indeed, the weather can affect transport systems in a multitude of ways, from road travel times to ultimately affecting people's mode choices.

The are two main sources for weather data: proprietary and online data. Proprietary data is typically collected by public agencies and it usually consists of very detailed information about the weather conditions. This can include road surface temperature, relative humidity above road surface, water film height, ice percentage, friction, etc. This data is often stored in a raw format in databases and used internally. On the other hand, online data is publicly available. There are several popular websites that provide application programming interfaces (APIs) to access this data for most places in the world, such as OpenWeatherMap (OpenWeatherMap Inc., 2018) or Wunderground (TWC Product and Technology LLC, 2018). Alternatively, one could rely on RSS feeds from local authorities, although their quality can vary quite significantly between different countries. The weather data from these sources usually comes in XML or JSON format.

4.2.2 Incident, Roadworks, and Road Blockages Data

Traffic incidents have been identified as one of the major contributors to increased congestion, causing about one-quarter of the congestion on US roadways (Haas, 2006). They are estimated to cause >50% of delay experienced by motorists in total for all urban areas. Furthermore, for every minute that the primary incident remains a hazard, the likelihood of a secondary crash increases by 2.8% (Karlaftis et al., 1999).

In order to support a timely response to incidents, traffic management centers establish workflows that consist of collecting information, analyzing it and executing the chosen strategy, continuously using updated information to control traffic, disseminate information and manage incident response resources. In this process, a lot of contextual information is gathered (usually in a database or in the form of logs), which can be used to enrich our understanding of the observed traffic behavior. This information is then (partially) made publicly available usually through broadcasting systems, such as official RSS feeds or social networks (e.g., Twitter or Facebook).

Like incidents, roadworks and road blockages are another major contributor to road congestion, since they can severely reduce road capacity and they can last for weeks or even months. With the exception of roadworks and road blockages that are caused by an incident, this information is typically made publically available online well in advance by official governmental websites or through the use of RSS feeds.

More recently, with the development of community-driven location-based services like Waze (Waze Mobile, 2018), sharing this kind of information has become much easier. Waze is mobile application that allows users to share the traffic conditions of the road that they are travelling on (e.g., travel time information) automatically, as well as to report accidents, traffic jams, speed and police traps, etc. As the size of the community of Waze users grows, so does the quality of the information provided. However, although all this information is available to the users of the mobile application, it is not readily available for the general public, thus making it harder to use, for example, when developing transportation models.

4.2.3 Events Data

Over the last decade, the Internet has become the preferred resource to announce, search, and comment about social events such as concerts, sports games, parades, demonstrations, sales, or any other public event that potentially gathers a large group of people. These planned special events often carry potentially disruptive impacts on the transportation system, because they correspond to nonhabitual behavior patterns that are hard to predict and plan for.

There is a myriad of websites that provide rather detailed descriptions of most planned special events that take place in cities around the world, especially in larger metropolitan areas. Particularly popular examples are: Eventful (Entercom, 2018), Timeoutworld (Time Out Group, 2018) and AllEventsIn (Alleventsin, 2018). These are typically user-contributed and have the advantage of providing world-wide coverage, rather than being limited to a certain region. Furthermore, these typically provide APIs to the users, which greatly simplifies the data collection. However, it is important to note that local public authorities typically maintain their own official event directories (e.g., New York City (Nyc.com, 2018)).

A popular alternative to event websites are social networks. In particular, Facebook has become a great source for people to create and publicize events. Furthermore, the social nature of Facebook gives the potential for understanding the online popularity of events, which then may or may not be reflected in the transportation system.

4.2.4 Social Media Data

Social media data is by far the most widely available on the Internet, with social networks like Facebook and Twitter being two of the most popular examples. It is widely understood that user-generated messages in social media platforms now play a determinant role in different areas such as politics, business and entertainment. Although perhaps to a lesser extent, the transport sector is no exception. Indeed, researchers have realized the potential of social media in providing a better understanding the behavior of the transport system, and several research works seek to explore that potential. Examples of this range from

using social media to understand the opinion of the users toward the public transport system in certain cities (Schweitzer, 2012), to trying to detect incident in the road network by analyzing user tweets (D'Andrea et al., 2015).

All of the most popular social media platforms provide APIs, which make accessing their data quite easy. However, data access in some of these APIs is quite restrictive due to privacy issues. For example, Facebook only allows a user to access the posts of their friends or other users who have public profiles.

4.3 Formats and Data Collection

Depending on its source, context data can come in very different formats and flavors. The most popular ones are XML, JSON, HTML, relational databases and plain natural language text. XML and JSON formats will be encountered mostly when gathering data from RSS feeds or APIs. As we saw in Section 2.3, these greatly simplify accessing and processing contextual data from the web. However, many websites don't provide such good interfaces. In these cases, one might need to resort to working the source HTML code of web pages directly, using techniques such as "screen scraping."

Although most contextual data sources provide a fair amount of structure that can be exploited (e.g., with formats like XML, JSON, relational databases, or even HTML tags), we quite often encounter context data in the form of plain natural language text. A good example of this are incident logs/reports as the ones used in (Pereira et al., 2013) for incident duration prediction. In these situations, one might need to consider natural language processing (NLP) techniques (Manning and Schütze, 1999). Furthermore, even when using a well-structured format like XML or JSON, some text-valued attributes might require further processing of the natural language text. A good example of this are event descriptions. These usually provide a significant amount of important information about the characteristics of the event. Therefore, NLP techniques need to be applied in order to turn that text into features (or attributes) that could be used, for example, in demand prediction models.

Social networks, like Facebook or Twitter, are another contextual information source where a significant part of the data is raw text. Hence, if one considers using, for example, Twitter as an incident detection mechanism or as a feedback tool for the understanding the opinion of the public toward the transportation system, the use of NLP techniques is essential. In the following section, we provide some guidelines on how to approach this type of data and turn it into useful information for transport models.

4.4 Data Cleaning and Preparation

In the previous sections, we discussed different types of context data, their sources and typical formats, as well as how to access those sources in order to retrieve the data. However, in most situations, having access to the raw data

isn't sufficient. We need to translate such data into features that are useful for ITS applications. We want to extract information about what (for example, an incident, concert, sports game, or religious celebration), when, where, and other relevant attributes (including how many lanes are blocked in an incident, the cost of concert tickets, the public's age range, or a temple's size). Sometimes this information can be directly obtained through APIs or screen scraped from well-structured websites (e.g., specific fields in XML). This was the technique used earlier, for example, for demand prediction in special events scenarios in (Pereira and Rodrigues, 2015; Rodrigues et al., 2016), where the event type, location, and start time were used as input to a neural network model. Unfortunately, quite often, well-structured, ready-to-use information won't be available, deeming it necessary to extract information from unstructured text using natural language processing (NLP) techniques, such as Information Extraction, Named Entity Recognition, Topic Modeling and Sentiment Analysis. In the following, we briefly discuss each of these individually, how they can be relevant for ITS applications and software packages that apply them.

4.4.1 Information Extraction

Information Extraction (IE) corresponds to the general task of automatically extracting structured information from unstructured text. An example consists on identifying the names of persons and locations that are mentioned in a document and the relationships between them. A particularly well-known subtask of IE is Named Entity Recognition (NER). The goal of NER is locate named entities in text, and classify them into a set of preestablished categories such as the names of persons, organizations, locations, dates, quantities, monetary values, etc.

In the context of ITS application, IE techniques can be used for example to identify where an event will take place or the name of its performers from its textual description. Other uses include identifying the number of lanes blocked or vehicles involved from an incident report, or the location of an incident being reported on Twitter.

There are several software packages that provide IE functionalities. For example, the "Stanford NER" is very popular implementation of a Named Entity Recognizer in Java. In Python, the "NLTK" package provides a lot of NLP functionality, including NER and many others such as text classification, tokenization (breaking a sentence into tokens), stemming (reducing words to their root; e.g., "smoking" reduces to "smoke") and part-of-speech tagging (identifying nouns, verbs, adjectives, adverbs, etc.). Using the NLTK package in Python (Bird et al., 2017), splitting a sentence into tokens, assigning each its corresponding part-of-speech (POS) tag and identifying named entities is relatively easy, as shown in Fig. 8. For further details, the NLTK documentation is a great resource (Bird et al., 2017).

```
import nltk
tokens = nltk.word_tokenize("A vehicle broke down close to the National Stadium.")
tagged = nltk.pos_tag(tokens)
entities = nltk.chunk.ne_chunk(tagged)
```

FIG. 8 Tokenizing, POS tagging, and NER example using nltk in Python.

4.4.2 Topic Modeling

Quite often, instead of being interested in extracting a specific information from a text, we are interested in obtaining a representation of the text as whole, which we can easily interpreted by a machine and fed, for example, as additional features/attributes to a prediction model. In such situation, a popular simple solution is to use a bag-of-words representation of the text, in which we disregard grammar and even word order and keep only track of the number of occurrences of each word in a document. In this way, a document is represented as a long vector of counts, whose length is the size of the vocabulary and where each entry corresponds to the number of occurrences of each word in that document.

The simplicity and efficiency of the bag-of-words models makes it very appealing for practical applications. However, in some scenarios, its limitations can be problematic. For example, the size of the representations scales linearly with the size of the vocabulary, which means that even for small corpora, vocabulary sizes in the order of 20,000 or 30,000 words are frequent. Furthermore, the bag-of-words representation is not able to capture the semantics of words. This means that the words "car" and "automobile" are represented as completely independent, even though they represent the exact same concept.

A common solution to some of these problems is to use topic models. In fact, the growing need to analyze large document corpora has led to great developments in topic modeling. Topic models, such as latent Dirichlet allocation (LDA) (Blei et al., 2003), allow us to analyze large collections of documents, by revealing their underlying themes, or topics, and how each document exhibits them. In LDA, each topic is a pattern represented by a distribution over the words present in the vocabulary. Its result is a set of topics and topic proportions associated with each document, meaning that documents are mixtures of topics and topics mixtures of words. Hence, by using LDA, a document can be represented as vector of size K, where K is the number of topics, and each element in the vector corresponds to the topic proportion of a given topic.

A great number of open source and proprietary software packages provide implementations of LDA (and variants), including the Matlab (MathWorks, 2018), R (Chang, 2015; Grün and Hornik, 2017), Python (Rehurek and Sojka, 2010), and Python/Scala in Apache Spark's MLlib (Apache Software Foundation, 2018). In Python, one can easily convert a collections of texts to the bag-of-words representation and then apply LDA by making use of the "gensim" package as shown in Fig. 9. For further details, see the gensim tutorials and documentation available at (Řehůřek, 2018).

```
From gensim import corpora, models
dictionary = corpora.Dictionary(texts)
corpus = [dictionary.doc2bow(text) for text in texts]
m = gensim.models.ldamodel.LdaModel(corpus, num_topics=10, id2word = dictionary)
print ("Topics:", m.print_topics(num_topics=10, num_words=5)))
```

FIG. 9 LDA model example using gensim in Python.

```
from nltk import sentiment
sentence = "I feel good."
sent_analyzer = sentiment.vader.SentimentIntensityAnalyzer(lexicon_file = \
"sentiment/vader_lexicon.zip/vader_lexicon/vader_lexicon.txt")
print "The sentiment is:", sent_analyzer.polarity_scores(sentence)
```

FIG. 10 Sentiment analysis example using nltk in Python.

4.4.3 Sentiment Analysis

Sentiment analysis corresponds to the process of identifying the sentiment associated with a piece of text. It usually relies on applying machine learning techniques to classify documents based on a collection of features extracted from the text using other NLP techniques, such as the presence of certain words or the coverage of some topics. In the context of ITS, sentiment analysis can be applied, for example, to large scale collections of tweets in order to determine how people feel about the transport system in general, or certain aspects of it in particular.

There are various packages that provide sentiment analysis functionality, such as the "RSentiment" package of R (Bose and Goswami, 2017) or the "nltk" package of Python (Bird et al., 2017). Most of these, actually allow you to train the user to train their own sentiment classifiers, by providing a dataset of texts along with their corresponding sentiments. However, pretrained versions also exist. For example, in Python on can easily use "nltk" package to identify the sentiment in texts as shown in Fig. 10.

REFERENCES

Alleventsin (2018) All Events in. Available at: https://allevents.in/. Accessed 29 March 2018.

Apache Software Foundation (2018) Machine Learning. Library (MLlib) Main Guide—Spark 2.3.0 Documentation. Available at: https://spark.apache.org/docs/latest/ml-guide.html. Accessed 29 March 2018.

Basu, S. and Meckesheimer, M. (2007) 'Automatic outlier detection for time series: an application to sensor data', Knowl. Inf. Syst. Available at: http://www.springerlink.com/index/442WP9XQ56286245.pdf. Accessed 1 February 2017.

Bhaskar, A., Chung, E. and Dumont, A.-G. (2009) 'Integrating cumulative plots and probe vehicle for travel time estimation on signalized urban network', in *9th Swiss Transport Research Conference*. Monte Verita/Ascona. Available at: http://www.strc.ch/conferences/2009/Bhaskar.pdf. Accessed 1 February 2017.

Bird, S., Klein, E. and Loper, E. (2017) Natural Language Processing with Python—Analyzing Text with the Natural Language Toolkit. Available at: http://www.nltk.org/book/. Accessed 29 March 2018.

Bitar, N. (2016) 'Big Data Analytics In Transportation Networks Using The Npmrds'. Available at: https://www.researchgate.net/profile/Naim_Bitar/publication/305994432_Big_Data_Analytics_in_Transportation_Networks_Using_the_NPMRDS/links/57a92b4608aef20758cd124c.pdf. Accessed 31 January 2017.

Blei, D., Ng, A. and Jordan, M. (2003) 'Latent dirichlet allocation', J. Mach. Learn. Res. Available at: http://www.jmlr.org/papers/v3/blei03a.html. Accessed 24 February 2017.

Bose, S. and Goswami, S. (2017) Package 'RSentiment'. Available at: https://cran.r-project.org/web/packages/RSentiment/RSentiment.pdf. Accessed 29 March 2018.

Bucknell, C., Herrera, J.C., 2014. A trade-off analysis between penetration rate and sampling frequency of mobile sensors in traffic state estimation. Transp. Res. Part C: Emerg. Technol. 46, 132–150. https://doi.org/10.1016/J.TRC.2014.05.007. Pergamon.

Buuren, S. and Groothuis-Oudshoorn, K. (2011) 'Mice: Multivariate imputation by chained equations in R', J. Stat. Softw. Available at: http://doc.utwente.nl/78938/. Accessed 27 January 2017.

Cambridge Systematics and Texas Transportation Institute, 2015. NPMRDS Missing Data and Outlier Analysis. https://www.regulations.gov/document?D=FHWA-2013-0054-0103.

Chang, J. (2015) Package 'lda': Gibbs Sampling Methods for Topic Models. Available at: https://cran.r-project.org/web/packages/lda/lda.pdf. Accessed 29 March 2018.

Chiou, J.-M., et al., 2014. A functional data approach to missing value imputation and outlier detection for traffic flow data. Transportmetrica B: Transp. Dyn. 2 (2), 106–129. https://doi.org/10.1080/21680566.2014.892847. Taylor & Francis.

D'Andrea, E., Ducange, P. and Lazzerini, B. (2015) 'Real-time detection of traffic from twitter stream analysis', IEEE transactions on. Available at: http://ieeexplore.ieee.org/abstract/document/7057672/. Accessed 24 February 2017.

Entercom (2018) Eventful. Available at: www.eventful.com. Accessed 29 March 2018.

Florida Department of Transportation (2012) 'Probe Data Analysis Evaluation of NAVTEQ, TrafficCast, and INRIX Travel Time System Data in the Tallahassee Region'. Available at: http://www.fdot.gov/traffic/ITS/Projects_Deploy/2012-03-26_Probe_Data_Analysis_v2-0.pdf. Accessed 31 January 2017.

Goyvaerts, J. (2016) Regular-Expressions.info—Regex Tutorial, Examples and Reference—Regexp Patterns. Available at: http://www.regular-expressions.info/. Accessed 29 March 2018.

Grün, B. and Hornik, K. (2017) Package 'topicmodels'. Available at: https://cran.r-project.org/web/packages/topicmodels/topicmodels.pdf. Accessed 29 March 2018.

Gupta, M., Gao, J. and Aggarwal, C. (2014) 'Outlier detection for temporal data: A survey', *on Knowledge and Data* …. Available at: http://ieeexplore.ieee.org/abstract/document/6684530/. Accessed 2 February 2017.

Haas, K., 2006. Benefits of Traffic Incident Management. National traffic incident management coalition.

Haghani, A., Zhang, X., Hamedi, M., 2015. Validation and Augmentation of INRIX Arterial Travel Time Data Using Independent Sources. Maryland State Highway Administration.

Hallenbeck, M. and McCormack, E. (2015) 'Developing a System for Computing and Reporting MAP-21 and Other Freight Performance Measures'. Available at: http://wadot.wa.gov/NR/rdonlyres/4869900F-9E88-4B2E-968B-EF2CA3B3D1FD/107207/8441.pdf. Accessed 31 January 2017.

Henrickson, K., Zou, Y. and Wang, Y. (2015) 'Flexible and robust method for missing loop detector data imputation', J. Transp. Available at: http://trrjournalonline.trb.org/doi/abs/10.3141/2527-04. Accessed 24 February 2017.

Hodge, V. and Austin, J. (2004) 'A survey of outlier detection methodologies', Artif. Intell. Rev. Available at: http://link.springer.com/article/10.1007/s10462-004-4304-y. Accessed 27 January 2017.

Honaker, J., King, G., Blackwell, M., 2016. Amilia: A Program for Missing Data.

Hosuri, S. R. (2017) Congestion Quantification Using the National Performance Management Research Dataset. University of Alabama at Birmingham. Available at: https://search.proquest.com/docview/1914681659?pq-origsite=gscholar. Accessed 28 March 2018.

Hyndman, R. (2016) 'Package "forecast" Title Forecasting Functions for Time Series and Linear Models'. Available at: http://github.com/robjhyndman/forecast. Accessed 26 January 2017.

INRIX (2014) 'I-95 Vehicle Probe Project II Interface Guide'.

Karlaftis, M., Latoski, S. and Richards, N. (1999) 'ITS impacts on safety and traffic management: an investigation of secondary crash causes', J. Intell. Available at: http://www.tandfonline.com/doi/abs/10.1080/10248079908903756. Accessed 24 February 2017.

Kaushik, K., Ernest Young, S., 2015. Computing performance measures using National Performance Management Research Data set (NPRMDS) data. Transp. Res. Record J. Trans. Res. Board 2529, 10–26.

Kim, S., Coifman, B., 2014. Comparing INRIX speed data against concurrent loop detector stations over several months. Transp. Res. Part C: Emerg. Technol. 49, 59–72. https://doi.org/10.1016/J.TRC.2014.10.002. Pergamon.

Koetse, M. and Rietveld, P. (2009) 'The impact of climate change and weather on transport: an overview of empirical findings', Transp. Res. Part D: Transp. Available at: http://www.sciencedirect.com/science/article/pii/S136192090800165X. Accessed 24 February 2017.

Laurikkala, J. et al. (2000) 'Informal identification of outliers in medical data', *on Intelligent Data* …. Available at: http://www.academia.edu/download/30622241/All.pdf#page=24. Accessed 27 January 2017.

Li, X. et al. (2009) 'Temporal outlier detection in vehicle traffic data', Data Engineering, 2009. ICDE'09. Available at: http://ieeexplore.ieee.org/abstract/document/4812530/. Accessed 2 February 2017.

Li, L., Li, Y., Li, Z., 2014. Missing traffic data: Comparison of imputation methods. IET Intell. Transp. Syst. 8 (1), 51–57. https://doi.org/10.1049/iet-its.2013.0052.

Liu, Z., Sharma, S., Datla, S., 2008. Transportation planning and technology imputation of missing traffic data during holiday periods. Transp. Plan. Technol. 31 (5), 525–544. https://doi.org/10.1080/03081060802364505.

Manning, C. and Schütze, H. (1999) Foundations of Statistical Natural Language Processing. Available at: http://www.mitpressjournals.org/doi/pdf/10.1162/coli.2000.26.2.277. Accessed 24 February 2017.

MathWorks (2018) Latent Dirichlet allocation (LDA) model, MATLAB 9.4 Documentation. Available at: https://www.mathworks.com/help/textanalytics/ref/ldamodel.html?requestedDomain=true. Accessed 29 March 2018.

Moritz, S. (2017) 'Package "imputeTS" Title Time Series Missing Value Imputation Description Imputation (replacement) of missing values in univariate time series'. Available at: https://github.com/SteffenMoritz/imputeTS/issues. Accessed 26 January 2017.

Ni, D. et al. (2005) 'Multiple imputation scheme for overcoming the missing values and variability issues in ITS data', J. Transp. Available at: http://ascelibrary.org/doi/abs/10.1061/(ASCE)0733-947X(2005)131:12(931). Accessed 24 February 2017.

Nyc.com (2018) New York Events and Event Calendar | NYC.com—New York's Box Office. Available at: https://www.nyc.com/events/. Accessed 29 March 2018.

OpenWeatherMap Inc. (2018) Current weather and forecast—OpenWeatherMap. Available at: https://openweathermap.org/. Accessed 29 March 2018.

Park, E., Turner, S. and Spiegelman, C. (2003) 'Empirical approaches to outlier detection in intelligent transportation systems data', Res. Record J. Available at: http://trrjournalonline.trb.org/doi/abs/10.3141/1840-03. Accessed 31 January 2017.

Patire, A.D., et al., 2015. How much GPS data do we need? Transp. Res. Part C: Emerg. Technol. 58, 325–342. https://doi.org/10.1016/J.TRC.2015.02.011. Pergamon.

Pereira, F. and Rodrigues, F. (2015) 'Using data from the web to predict public transport arrivals under special events scenarios', J. Intell. Available at: http://www.tandfonline.com/doi/abs/10.1080/15472450.2013.868284. Accessed 24 February 2017.

Pereira, F., Rodrigues, F. and Ben-Akiva, M. (2013) 'Text analysis in incident duration prediction', Transp. Res. Part C: Available at: http://www.sciencedirect.com/science/article/pii/S0968090X13002088. Accessed 24 February 2017.

Pereira, F.C., et al., 2015. Why so many people? Explaining nonhabitual transport overcrowding with internet data. IEEE Trans. Intell. Transp. Syst. 16 (3), 1370–1379. https://doi.org/10.1109/TITS.2014.2368119.

Pu, W., 2018. Interstate speed profiles. Transp. Res. Rec. https://doi.org/10.1177/0361198118755713. SAGE Publications, Sage: Los Angeles, CA, p. 36119811875571.

Python Software Foundation (2018) 19.2. json—JSON encoder and decoder—Python 3.6.5 documentation. Available at: https://docs.python.org/3/library/json.html. Accessed 29 March 2018.

Ramaswamy, S., Rastogi, R. and Shim, K. (2000) 'Efficient algorithms for mining outliers from large data sets', SIGMOD Rec. Available at: http://dl.acm.org/citation.cfm?id=335437. Accessed 1 February 2017.

Ran, B. et al. (2015) 'Traffic Speed Data Imputation Method Based on Tensor Completion', Comput. Intell. Neurosci. Hindawi Publishing Corporation, 2015, pp. 1–9. https://doi.org/10.1155/2015/364089.

Ravichandran, S., 2001. State space modelling versus ARIMA time-series modelling. J. Ind. Soc. Agric. Stat. 54 (1), 43–51.

Řehůřek, R. (2018) Gensim: Topic modelling for humans. Available at: https://radimrehurek.com/gensim/. Accessed 29 March 2018.

Rehurek, R. and Sojka, P. (2010) 'Software Framework for Topic Modelling with Large Corpora', Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.695.4595. Accessed 29 March 2018.

Reitz, K. (2018) Requests: HTTP for Humans—Requests 2.18.4 documentation. Available at: http://docs.python-requests.org/en/master/. Accessed 29 March 2018.

Richardson, L. (2017) Beautiful Soup Documentation—Beautiful Soup 4.4.0 documentation. Available at: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. Accessed 29 March 2018.

Rodrigues, F., Borysov, S. and Ribeiro, B. (2016) 'A Bayesian additive model for understanding public transport usage in special events', IEEE transactions on. Available at: http://ieeexplore.ieee.org/abstract/document/7765036/. Accessed 24 February 2017.

Rubin, D. (1976) 'Inference and missing data', Biometrika. Available at: http://biomet.oxfordjournals.org/content/63/3/581.short. Accessed 18 January 2017.

Rubin, D., Little, R.J.A., 1987. Statistical Analysis with Missing Data, first ed. Wiley and Sons, New York.

Schafer, J. (1997) Analysis of Incomplete Multivariate Data. Available at: https://www.google.com/books?hl=en&lr=&id=3TFWRjn1f-oC&oi=fnd&pg=PR13&dq=Analysis+of+incomplete+multivariate+data&ots=2pLNsCBcq6&sig=l40JIymHlDU9os09foooXl5Rnn8. Accessed 26 January 2017.

Schafer, J. and Graham, J. (2002) 'Missing data: our view of the state of the art.', Psychol. Methods. Available at: http://psycnet.apa.org/journals/met/7/2/147/. Accessed 18 January 2017.

Schweitzer, L. (2012) 'How are we doing? opinion mining customer sentiment in us transit agencies and airlines via twitter', *Transportation Research Board 91st Annual Meeting*. Available at: https://trid.trb.org/view.aspx?id=1129878. Accessed 24 February 2017.

Shekhar, S., Lu, C. and Zhang, P. (2001) 'Detecting graph-based spatial outliers: algorithms and applications (a summary of results)', Proceedings of the seventh ACM SIGKDD. Available at: http://dl.acm.org/citation.cfm?id=502567. Accessed 31 January 2017.

Tak, S., Woo, S. and Yeo, H. (2016) 'Data-driven imputation method for traffic data in sectional units of road links', IEEE Trans. Intell. Available at: http://ieeexplore.ieee.org/abstract/document/7444178/. Accessed 27 January 2017.

The PostGIS Development Group (2018) PostGIS 2.4.4dev Manual, http://postgis.net. Available at: https://postgis.net/docs/. Accessed 26 March 2018.

Time Out Group (2018) Time Out World. Available at: https://world.timeout.com/. Accessed 29 March 2018.

TomTom, 2011. TomTom Real Time Traffic Information. http://www.tomtom.com/lib/img/REAL_TIME_TRAFFIC_WHITEPAPER.pdf.

TomTom, 2012. OpenLR White Paper Version 1.5 revision 2. http://www.openlr.org/data/docs/OpenLR-Whitepaper_v1.5.pdf.

TomTom, 2014. Historical Traffic Information. http://www.tomtom.com/lib/img/HISTORICAL_TRAFFIC_WHITEPAPER.pdf.

TWC Product and Technology LLC (2018) Weather Forecast & Reports—Long Range & Local | Weather Underground. Available at: https://www.wunderground.com/. Accessed 29 March 2018.

Wang, Z. et al. (2018) 'A Cross-Vendor and Cross-State Analysis of the GPS-Probe Data Latency', in Trans Res Board Ann Meeting. Washington DC: Transportation Research Board. Available at: http://amonline.trb.org/2017trb-1.3983622/t005-1.4000488/200-1.4001272/18-05026-1.3997223/18-05026-1.4001275#tab_0=1. Accessed 29 March 2018.

Waze Mobile (2018) Free Community-based GPS, Maps & Traffic Navigation App | Waze. Available at: https://www.waze.com/. Accessed 29 March 2018.

Young, S.E., et al., 2015. I-95 Corridor Coalition Vehicle Probe Project: Validation of Arterial Probe Data. http://i95coalition.org/wp-content/uploads/2015/02/I-95_Arterial_Validation_Report_July2015-FINAL.pdf?fe2c99.

Zhong, J. and Ling, S. (2015) 'Key factors of k-nearest neighbours nonparametric regression in short-time traffic flow forecasting', Proceedings of the 21st International Conference on. Available at: http://link.springer.com/chapter/10.2991/978-94-6239-102-4_2. Accessed 27 January 2017.