A big data architecture for traffic forecasting using multi-source information

Yannis G. Petalas, Ahmad Ammari, Panos Georgakis, and Chris Nwagboso

Sustainable Transport Research Group, Department of Civil Engineering, Faculty of Science and Engineering,
University of Wolverhampton, Wolverhampton, UK
{I.Petalas,A.Ammari,P.Georgakis,C.Nwagboso}@wlv.ac.uk

Abstract. An important strand of predictive analytics for transport related applications is traffic forecasting. Accurate approximations of the state of transport networks in short, medium or long-term future horizons can be used for supporting traveller information, or traffic management systems. Traffic forecasting has been the focus of many researchers over the last two decades. Most of the existing works, focus on single point, corridor, or intersection based predictions with limited efforts to solutions that cover large metropolitan areas. In this work, an open big-data architecture for road traffic prediction in large metropolitan areas is proposed. The functional characteristics of the architecture, that allows processing of data from various sources, such as urban and inter-urban traffic data streams and social media, is investigated. Furthermore, its conceptual design using state-of-the-art computing technologies is realised.

Keywords: Big Data, Intelligent Transportation Systems, Twitter, Social Media, Natural Language Processing, Forecasting Models

1 Introduction

Traffic forecasting plays a significant role in the transportation domain. Precise and accurate predictions of road traffic can improve the information services to travelers, lead to more efficient traffic information systems and improve road safety by reduction of accidents. Forecasting of the state of transport networks can result in the provision of mobility services that can reduce congestion, travel costs and CO_2 emissions, as well as lead to more efficient control and planning of traffic infrastructure.

Until now many models have been employed for traffic prediction [1], however the majority of them are applied and evaluated in an offline, rather static mode. Furthermore, their inputs are unidimensional and limited to single-point loop detectors, or clusters of detectors in close proximity. Thus, they lack the potential of utilizing dynamic information from other parts of the transportation network, or from other types of data sources.

Initial methodologies used for traffic forecasting were based on statistical analysis, and were followed by computational intelligence-based approaches for handling more efficient complex traffic conditions. The existing methods belong to three broad categories, parametric, non-parametric and hybrid and a review of the most prominent ones is presented in the next section.

It is believed that nowadays, traffic forecasting can benefit from a distributed big data architecture [2] that utilises data from multiple heterogeneous sources. This paper proposes an open big-data architecture that has the potential to meet this emerging need. Big data applications are characterized by the use of high volume, high variety and high velocity data. The proposed solution consists of a suite of forecasting models and a social media data mining component that processes data from twitter and relates them to the transport domain. Therefore, in this study high volume data in a variety of formats are retrieved in real-time from urban and inter-urban clusters of loop detectors, as well as through a stream of twitter feeds within the area of interest.

The paper is organized as follows. In section 2, a review of forecasting models and uses of twitter data in the transport domain is presented. The data requirements for the operation of the proposed architecture is shown in section 3. The models used for traffic forecasting are presented in section 4, followed by a data mining approach for tweets in section 5. Section 6 outputs the results of experimentation with fused data from traffic sensors and social media. The physical instantiation of the application, composed of a number of components and a distributed system implementation is discussed in section 7. Section 8 concludes the paper with a reflective narration for the study.

2 Literature Review

2.1 Traffic Forecasting Models

Parametric Methodologies. This category involves statistical methodologies under the field of time series forecasting. An Autoregressive Integrated Moving Average (ARIMA) model was used for the short-term prediction of freeway traffic in the early 70s [3]. Since this initial effort, different variants of ARIMA were proposed to improve the prediction accuracy. These included, Kohonen ARIMA (KARIMA) [4], subset ARIMA [5], ARIMA with explanatory variables (ARIMAX) [6], spacetime ARIMA [7] and seasonal ARIMA (SARIMA) [8] to better handle the seasonality of the traffic data. Another set of parametric methodologies exploited Kalman Filtering (KF) and state space models. KF was used in [9] for incorporating an extra dimension (spatial) as part of a time-series model. Flow input from successive sensors on a transport corridor formed a multivariate time series state space model, which resulted to better predictions compared to that of a univariate ARIMA model. An enhanced time-series analysis integrated spatial characteristics and data as part of an ARIMA model [10]. Such state space models offer a multivariate approach, which makes KF methodologies suitable for network wide predictions [11]. In regards to other multivariate methodologies, a structural time-series (MST) technique, using the seemingly unrelated time-series equation (SUTSE), modeled time-series from flow observations in a network of junctions within a congested urban environment [12].

Non Parametric methodologies. A limitation of the parametric methods presented above is the deterioration of their prediction accuracy when the data lack linearity. Since traffic related parameters exhibit stochastic and nonlinear characteristics the application of non-parametric methods in the field of traffic prediction gained momentum among researchers. The k-Nearest Neighborhood (k-NN) method was used for short-term freeway traffic forecasting and it was argued that it performed comparably with but not better than the linear time-series approach [13]. A dynamic multi interval traffic volume prediction model based on the k-NN non-parametric regression can be found in [14], while functional estimation techniques were applied as part of a kernel smoother for the auto-regression function in [15]. A local linear regression model and a Bayesian network approach for traffic flow forecasting were discussed in [16] and [17] respectively. Finally, an online learning weighted Support Vector Regression (SVR) model was presented in [18].

The majority of work in non-parametric methods is related to Artificial Neural Networks (ANN). Prediction outputs include Annual Average Daily Traffic (AADT) [19, 20], flow, speed and occupancy [21, 22], as well as mean travel times [23]. The State Space Neural Network (SSNN) is considered as a variant of Elman Neural Network and has been applied to predict urban travel time [24–27]. The result demonstrates that the SSNN is superior to other prevailing algorithms in terms of accuracy [27]. A Long Short-Term Neural Network (LSTM) was used to predict travel speed and outperformed the other recurrent neural networks (SSNN, Time Delay Neural Network) [28].

Hybrid Systems. There are many researchers that combined ANNs with other methodologies like optimization and Fuzzy Logic to improve the performance of ANNs. A Genetic Algorithm (GA) has been used for optimising the structure and hyperparameters of an ANN for predicting flows and travel times in urban networks [29, 30]. Chaos Optimisation techniques have improved the performance of a GA supported ANN in [31]. A layered forecasting algorithm using ANN and Self-Organizing Maps (SOM) can be found in [32], while SOM has been adopted for the classification of traffic data as input to an ANN in [33]. In the former study, the layered structure was used to effectively support the forecasting task in situations where external events (e.g. incidents, social events, etc.) affected greatly recurrent traffic patterns.

A fuzzy neural model which classified input data using fuzzy rules and mapped input to output data using an ANN has been developed in [34], while a fuzzy rule base system that combined ANN and KF as part of a short-term forecasting model is presented in [35]. A deep-learning based coefficients optimization algorithm based on fuzzy ANN has been devised in [36]. Finally an aggregation approach for traffic flow prediction based on the Moving Average (MA), Exponential Smoothing (ETS), ARIMA, and ANN models has been proposed in [37].

2.2 Applications of Twitter data in transportation

More recently, there have been few emerging works on the exploitation of Intelligent Transport Systems (ITS) related information embedded in user-generated content of social data resources to support predictive analytics in traffic domain. Linear regression models have been employed for freeway traffic speed prediction using twitter data, weather and traffic information [38]. Twitter data had a relatively high sensitivity for predicting inclement weather (i.e., snow) during daytime. Twitter-based weather variables improved the predictive accuracy of the forecasting models. Traffic incident clustering and prediction was used for the development of a traffic management dashboard application in [39]. Tweets were classified to either insightful, or irrelevant to improve the system's performance in terms of traffic insights and alerts provision. Traffic incident management and detection on Dutch highways has been studied in [40]. In addition, spatio-temporal clusters of tweets on roads supported congestion detection in a study of Australian cities [41].

Prediction of traffic flow during sport events using a variety of methodologies like auto-regressive models, ANNs, SVRs and k-NN models has been examined in [42]. The authors reported that model enrichment with Twitter-engineered features improved prediction accuracy of regression models. An agent-based architecture for the detection of the movement of general public has been shown in [43]. Based on a city simulation framework, agent-based broadcasting of events extracted from tweets was claimed to be a source of human travel information useful for predicting traffic flow. Finally, twitter-engineered features extracted by KF models and semantic analysis of tweets were found to be able to overcome data processing limitations in existing baseline models, thus improving the accuracy of bus arrival time predictions [44].

3 Data Sources

The data sources that have been used for informing the development of the specifications for the proposed architecture are offered by Highways England [45], Birmingham City [46] and through Twitter's streaming APIs [47].

3.1 Highways England

Highways England, through its National Traffic Information Service (NTIS) initiative, provides real time traffic data (e.g. flow, average speed, travel time, headway, occupancy) to subscribed organisations. The format used is an extension of DATEX II, which is the European standard for traffic information exchange. For the purposes of this study, data from the following type of sensors, located on motorways enclosing the city of Birmingham, are being received:

1. "ANPR Journey Time Data": raw (unprocessed) real-time Automatic Number Plate Recognition (ANPR) journey time data, measured between ANPR camera sites.

2. "MIDAS Gold Loop Traffic Data": raw (unprocessed) real-time traffic data measured by Motorway Incident Detection and Automatic Signaling (MIDAS) loop sensors.
3. "Fused Sensor-only PTD": real-time Processed Traffic Data (PTD), calculated from raw sensor traffic data.
4. "TMU Loop Traffic Data": raw (unprocessed) real-time traffic data measured by Traffic Monitoring Unit (TMU) loop sensors.

Data from MIDAS (unprocessed and fused) are being pushed to subscribers every one minute, while data from ANPR and TMU systems every five minutes.

3.2 Birmingham City Council

Birmingham City Council provides traffic data through their open data infrastructure. REST endpoint calls are used to collect the traffic data in a proprietary format. The provided data are pulled every 5 to 10 minutes and include flows, average speeds and travel times, from loop detectors and ANPR installations.

3.3 Twitter

The Twitter APIs provide access to four main objects that have been considered for this work. These include:

1. "Tweets": The basic atomic building block of all things. Tweets are also known as status updates. Tweets can be embedded, replied to, liked, unliked and deleted. The key tweet fields streamed/retrieved for analysis are coordinates, creation timestamp, tweet id, language, place, and text.
2. "Users": Users can be anyone or anything. They tweet, follow, create lists, have a home timeline, can be mentioned, and can be looked up in bulk. The key user fields considered are account creation timestamp, description, number of followers, number of friends, whether the account has enabled the possibility of geotagging, account id, language, location, and number of tweets issued by the user.
3. "Entities": Provide metadata and additional contextual information about content posted on Twitter. Entities are never divorced from the content they describe. Entities are returned wherever tweets are found in the API. Entities are instrumental in resolving URLs. The key entity keys considered are hashtags and user mentions.
4. "Places": Specific, named locations with corresponding geo-coordinates. They can be attached to Tweets by specifying a place id when tweeting. Tweets associated with places are not necessarily issued from that location but could also potentially be about that location. Places can be searched for and Tweets can be requested by place id. Considered place fields are attributes, bounding box of coordinates which encloses this place, country, country code, human-readable representation of the places name (e.g. Birmingham), type of location represented by this place (e.g. city).

The volume, velocity and variety of the information received in real time by the aforementioned sources can be seen in Table 1.

Table 1. Volume and Velocity for the available data sources in the region of Birmingham city

Data Source	Volume	Velocity	Variety
Tweets	300MB per day	1 minute	Twitter objects in json format
Urban Traffic	600MB per day	5-10 minutes	Proprietary json documents
Motorway Traffic	3GB per day	1-5 minutes	Datex II compliant XML messages

4 Forecasting Models

Before using these data as input to the forecasting models, a pre-processing procedure takes place to enhance their quality. This procedure involves the following steps:

- *Data cleaning*, which involves detecting samples with values that do not lie in a specific range and removing duplicates. For example, traffic measurements with zero values are being discarded.
- *Computation of the percent of missing values per day*, which if is above a threshold (very high) the respective days are removed from the dataset.
- *Examination for big plateau*, where if a lot of measurements with the same value for consecutive time periods are received, is an indication of invalidity for the specific set of data. For example, consecutive sensor readings of 80 km/h for average speed over a period of many hours.
- *Imputation of missing values*, when valid measurements for the whole duration that the loop detectors operate (e.g. measurements per minute) are not possible. There are cases where measurements are not available due to malfunction of the loop detectors, or a possible break down in the data infrastructure (e.g. database). There are different strategies to handle missing values, such as to use the mean historical value of the available measurements, or their most frequent value. In this study linear interpolation has been used to replace the missing values.
- *Aggregation of the data* that are being received in various frequencies. Aggregation in different time scales e.g. per $5, 10, 15$ minutes has to be adopted before the application of the forecasting models.

After the pre-processing procedures, data with improved quality can be employed from the forecasting models. Various models have been used until now for traffic forecasting, ranging from statistics and computational intelligence as described in section 2. A number of such models have been incorporated in the proposed architecture in order to cover this wide range. From statistical sciences the methods considered include Auto-Regressive Integrated Moving Average (ARIMA) [48], Exponential Smoothing models (ETS) [49], Dynamic linear models [50, 51], Linear Regression [52] and Random forest Regression [53]. From computational intelligence, Artificial Neural Networks (ANNs) [54] and Support

Vector Regression (SVR) [55, 56] have been used. Regarding ARIMA and ETS, packages from R project [57] were explored, e.g. the forecasting package [58]. From DLM, the two basic models, the local level and the local linear trend were applied, using Kalman Filtering for the estimation of the models [59]. Regarding ANNs, Time Delay Neural Networks (TDNN) [54] were used in the experiments, since they are appropriate for applications with sequential data. They have the same structure as multilayer feed forward neural networks with a delay tape of observations. If the predicted value is $x(n)$, the inputs of the TDNN will be the s previous values $x(n-1), x(n-2), \dots, x(n-s)$. The ANN model was implemented in Python using the Keras library [60]. Keras is a library that allows the development of deep learning applications and it includes a suite of training methodologies (optimisers) that are suitable for online / real time learning. For linear regression, Random Forest Regression and Support Vector Regression the implementation in Python was supported by the scikit-learn library [61].

There are two phases related to the development of forecasting models. The first deals with the training of each model and it usually takes place offline, while the second phase is where the trained models receive their required input and provide predictions. For some models like DLM, ANNs and SVR the two phases may overlap such that they can be updated per sample of data. Regarding the training phase, requests are made to the storage of historical data, which may span across a period of months. For the statistical models (ARIMA, ETS, DLM) the used dataset is split in two parts, the training subset (80%) and the testing subset (20%). The fit of the models takes place using the training set and their evaluation using the testing set. Well known measures in the literature such as Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are used for the evaluation of the models [49].

For the rest of the forecasting models, which are primarily regression models, the dataset is split in 3 parts. Training set 60%, validation set 20% and testing set 20%. The validation subset is used for the computation of parameters specific to each method. These are, in random forest the number of trees/estimators, in SVR the type of kernel and its associate parameters and in ANNs the number of hidden layers. The final evaluation takes place with the usage of the testing set and the measures that have been mentioned above (MAPE,MAE,RMSE). The output of these models is the prediction for the next time step, while the predictors are recent values (in this study 2 previous observations) from traffic data, the mean historical value per hour and minute and other variables such as the current day, hour and minute. Additional predictors could use information generated from tweets related to traffic conditions.

There are two basic approaches for multistep ahead prediction. The first one is the incremental, where one step ahead prediction is made and the result is fed back as an input to the model for the next step prediction. This process is repeated until the number of desired steps ahead is reached. The second one is to use a separate model for each step ahead that needs to be predicted. For ANNs there is a third option to have as outputs the number of the steps that we want to predict ahead. If we want to predict 12 steps ahead, the ANN will have 12 neurons at the output layer. Regarding ANNs this was found to be the best approach and was adopted in the experiments. For the other regression models the second approach was used, e.g. building a separate model for each step ahead.

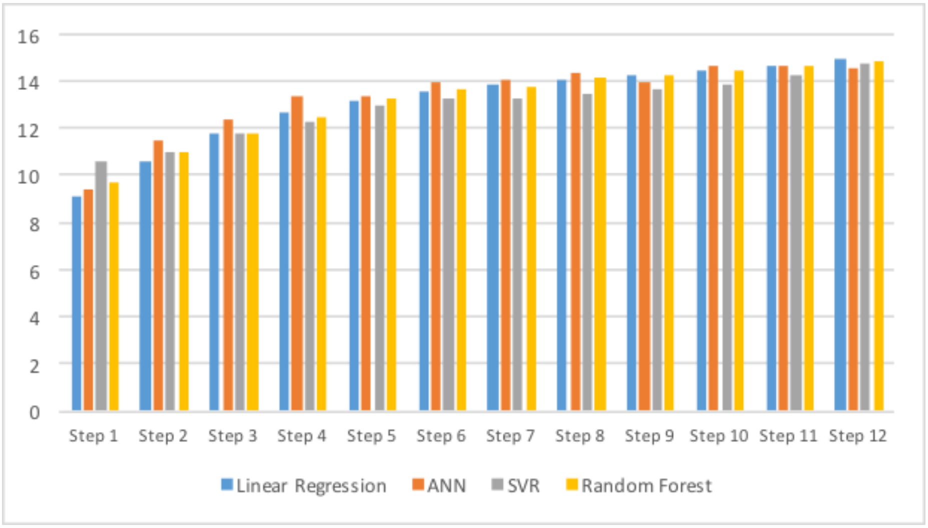


Fig. 1. MAPE 12 step ahead flow forecasting for a motorway sensor (M6)

In Figures 1, 2 some experimental results are presented. Traffic flow forecasting per 1 hour ahead (12 steps) for one sensor located on a motorway (M6) and average speed forecasting per 1 hour ahead for one sensor located close to the city of Birmingham. It can be noticed that the error increases for longer steps ahead and that per step of prediction the model with the best accuracy may be different.

5 Social Information Miner

Information generated by social media, and in particular from Twitter, can support traffic prediction applications in locations where sensors are sparsely located. The social media data mining approach used in this study employs different text pre-processing, Natural Language Processing (NLP), text mining, and text classification algorithms for the realisation of the sub-components presented in Figure 3. A brief description of each sub-component can be found below:

– Two data streaming adapters have been developed to collect tweets from the public Twitter Streaming API. The first is using a list of traffic informing twitter accounts, while the second is receiving public tweets using a geo-location filter.

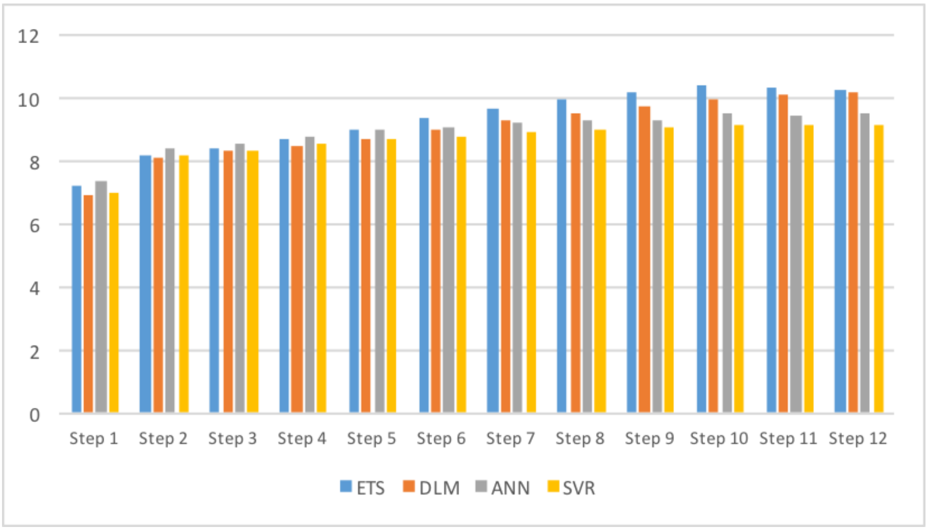


Fig. 2. MAPE 12 step ahead average speed forecasting for an urban sensor (N30162Y)

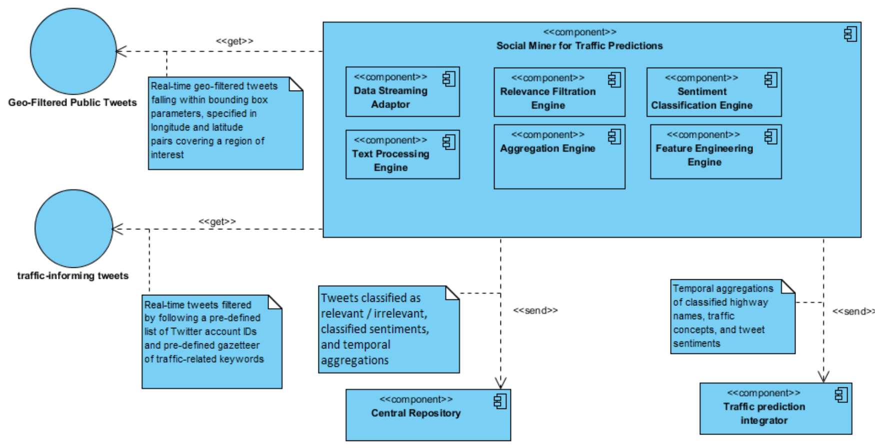

<<component>>
Social Miner for Traffic Predictions
<<component>> Data Streaming Adaptor
<<component>> Relevance Filtration Engine
<<component>> Sentiment Classification Engine
<<component>> Text Processing Engine
<<component>> Aggregation Engine
<<component>> Feature Engineering Engine
<<get>>
Geo-Filtered Public Tweets
Real-time geo-filtered tweets falling within bounding box parameters, specified in longitude and latitude pairs covering a region of interest
<<get>>
traffic-informing tweets
Real-time tweets filtered by following a pre-defined list of Twitter account IDs and pre-defined gazetteer of traffic-related keywords
Tweets classified as relevant / irrelevant, classified sentiments, and temporal aggregations
<<send>>
<<component>> Central Repository
Temporal aggregations of classified highway names, traffic concepts, and tweet sentiments
<<send>>
<<component>> Traffic prediction integrator

Fig. 3. Social media miner architecture

- The text processing engine adopts NLP mechanisms to pre-process and clean the tweets received from streaming API. The pre-processing pipeline is depicted in Figure 5.
- The relevance filtration classifier identifies tweets that are related to the transport domain. Different machine learning techniques and N-gram range sets have been trialed for generating a classification model with high accuracy percentages. In more particular, Multinomial Naïve Bayes (MNB), Support Vector Classification (SVC), Random Forest (RF), Artificial Neural Networks (ANN) were used, and the derived results can be seen in Table 2.
- A Sentiment classifier has been developed for identifying the polarity (positive, neutral, negative) of pre-processed tweets. The classifier enriches tweets with sentiment terms (good, bad, excellent, etc.) based on a mapping gazetteer that maps the traffic-related terms to the sentiment terms that will bias the overall sentiment scores of the tweets as obtained from a lexical database. For example, the tweet *"All lanes are now open on A21 Southbound btwn A25 near Chipstead and A225 near Sevenoaks following an earlier collision"* will be enriched with sentiment terms according to the mapping:

  "now open" $\longrightarrow$ "excellent great", "collision" $\longrightarrow$ "bad"

  Following enrichment, the tweet is fed into the SentiWordNet lexical resource for opinion mining [62] to determine its overall sentiment score. The boxplots depicted in Figure 4 illustrate the median sentiments scores of the tweets that include traffic-related terms in the training dataset. There is an obvious discrepancy of sentiment median scores between terms implying positive context (e.g. open, moving, cleared, released) and terms implying negative context (e.g. accident, congestion, flooding, delays), whereas terms implying neutral context (e.g. traffic, management) are mentioned in tweets with median sentiment scores around zero.
- Aggregation engine to produce time series aggregations of traffic mentions and sentiment polarities from the pre-processed tweets.
- Feature engineering sub-component to derive extended features from the pre-processed tweets, such as extracting the named entities (people, organizations, locations). The objective of the feature engineering sub-component is to extract entities from the pre-processed tweets. The extracted entities can be classified into three main classes, depending on the entity type as well as the technique(s) used in the extraction process:
  1. **Names of UK highways** which are extracted using regular expression rules.
  2. **Traffic concepts** which are extracted using a traffic gazetteer that has been generated based on text summaries and analysis of the tweets received from the traffic informing accounts and which includes word frequency, collocation, and concordance analysis.
  3. **Names of recognized locations, organizations, and people**, which are extracted using the Stanford Named Entity Recognition (NER) recognizer. For the purposes of mining information for supporting traffic predictions the general pre-trained Stanford recognizers for *PERSON*, *ORGANIZATION* and *LOCATION* entities has been used.

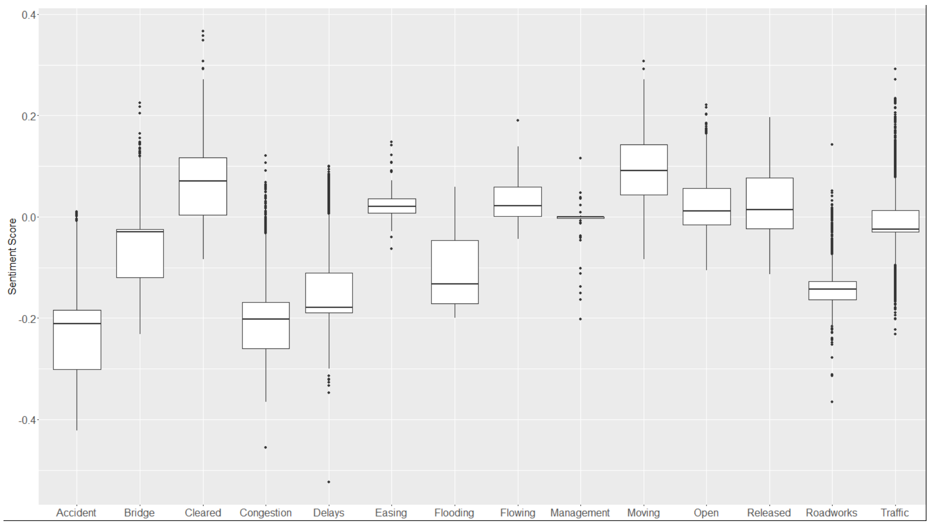


Fig. 4. Comparison between the sentiment scores of 15 traffic-related terms

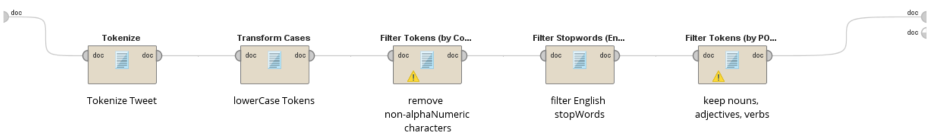


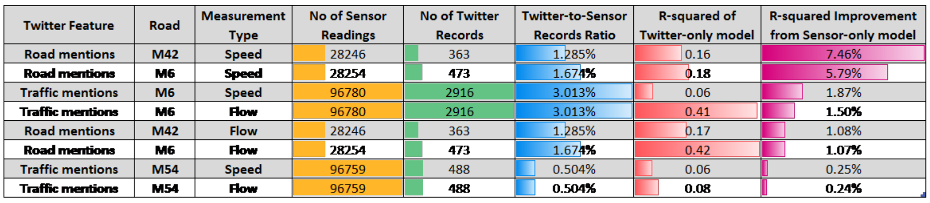
Fig. 5. Text processing and cleaning pipeline

Table 2. Confusion matrix for relevant (positive) and noise (negative) testing tweets with varying classification models

Algorithm	Actual Class	Predicted Negative	Predicted Positive	Class Precision	Class Recall	Class F1 score
MNB	Negative	TN: 2,370	FP: 27	99.04%	98.87%	98.95%
	Positive	FN: 23	TP: 2,580	98.96%	99.12%	99.04%
SVC	Negative	TN: 2,378	FP: 19	99.08%	99.21%	99.14%
	Positive	FN: 22	TP: 2,581	99.27%	99.15%	99.21%
RF	Negative	TN: 2,307	FP: 90	80.95%	96.25%	87.94%
	Positive	FN: 543	TP: 2,060	95.81%	79.14%	86.68%
ANN	Negative	TN: 2,324	FP: 73	98.64%	96.95%	97.79%
	Positive	FN: 32	TP: 2,571	97.24%	98.77%	98.00%

6 Data fusion experimentation

Following the development of the forecasting models and the social miner component, linear regression experiments were performed for investigating the prediction potential when information from heterogeneous sources is fused. Figure 6 summarizes the outputs from the experiments performed. In particular, the table highlights the percentage of the improvement of R-squared values after adding the Twitter features as predictors to the sensor-only models, the R-squared of the Twitter-only models, the ratio of the number of Twitter records to the number of sensor records, and the measurement type, road, and the Twitter features derived for each experiment. It is observed that in general, Twitter-derived fea-



Twitter Feature	Road	Measurement Type	No of Sensor Readings	No of Twitter Records	Twitter-to-Sensor Records Ratio	R-squared of Twitter-only model	R-squared Improvement from Sensor-only model
Road mentions	M42	Speed	28246	363	1.285%	0.16	7.46%
Road mentions	**M6**	**Speed**	**28254**	**473**	**1.674%**	**0.18**	5.79%
Traffic mentions	M6	Speed	96780	2916	3.013%	0.06	1.87%
Traffic mentions	**M6**	**Flow**	96780	2916	3.013%	**0.41**	**1.50%**
Road mentions	M42	Flow	28246	363	1.285%	0.17	1.08%
Road mentions	**M6**	**Flow**	**28254**	**473**	**1.674%**	0.42	**1.07%**
Traffic mentions	M54	Speed	96759	488	0.504%	0.06	0.25%
Traffic mentions	**M54**	**Flow**	96759	**488**	**0.504%**	**0.08**	**0.24%**

Fig. 6. Summary output of the linear regression experiments

tures as predicting variables to traffic forecasting models have contributed to an increase in the R-squared values from the values obtained by sensor-only models, thus contributed to a better explanation of the variability in the sensor readings as a response variable. However, the summary shows that the significant improvements occur, firstly, when the measurement type is the average speeds of the vehicles rather than the vehicles flows, and secondly, when using road mentions, rather than traffic mentions, as Twitter-derived features. The range of the absolute differences of the R-squared values between the Twitter-aided models and the sensor-only models is between 0.038 and 0.032 in the maximum two increased cases of 7.46% and 5.79%, and 0.002 in the minimum case of 0.24%. Although these differences are relatively small, they are not very far from the differences obtained by the analysis of similar studies, such as the study of [38], where the absolute differences of the R-squared values for one of the tried datasets (NFIA) between some Twitter and No-Twitter models developed to predict average speeds in inclement weather situations were between 0.05 and 0.07 (Daytime), and between 0.01 and 0.02 (Night-time).

On the other hand, using Twitter-derived features alone as explanatory attributes to the variability in the sensor readings, whether these features are extracted from the road mentions or the traffic-related concepts, provides the best R-squared results when the measurement type is the vehicles flows, rather than the average speeds. It is also noticed that having too insignificant Twitter-to-sensor records ratios (e.g. less than 1%) due to relatively low volumes of tweets about traffic events in particular roads, such as M54, is correlated with obtaining a poor explanation of the variability in the sensor readings. This applies on the two measurements types (flows and speeds) and when Twitter-derived features are used in both Twitter-only models and Twitter-and-sensor models. This motivates the utilization of more techniques, such as the use of social network analysis (SNA), to find more traffic-related tweets by discovering further Twitter accounts that post tweets about traffic events and road status.

7 Big Data Architecture

In this section, the proposed distributed architecture for traffic forecasting, as shown in Figure 7, will be described. Any sustainable architecture should be modular, extensible, fault-tolerant and be able to satisfy the specifications related to traffic forecasting as these change following technological developments.

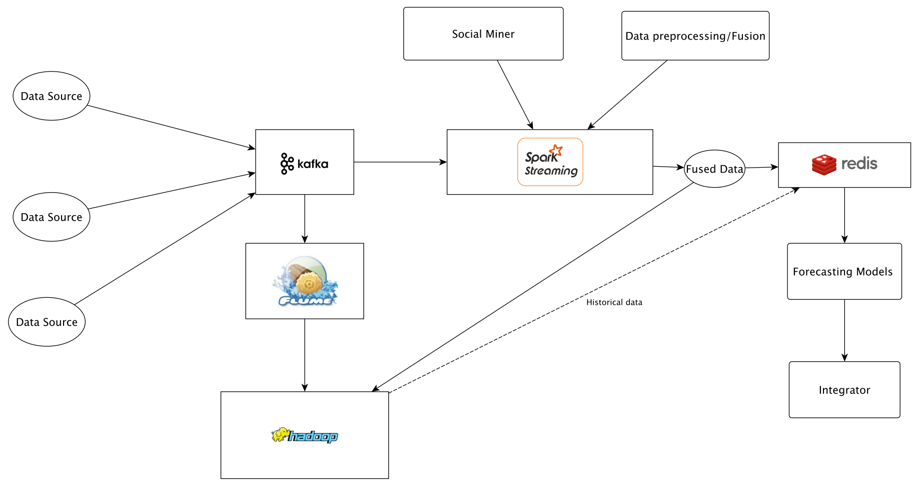


Fig. 7. Distributed Architecture For Traffic Forecasting

The inputs to the system are received through various available data sources and are being ingested in Apache Kafka [63]. Apache Kafka is an open source, distributed publish-subscribe (pub-sub) messaging system. Its main characteristics are persistence and replication messaging, high throughput, low latency, fast real time consumption of messages, high availability and no loss of data.

In a publish-subscribe system, messages persisted to a topic are produced by **publishers** and consumed by **subscribers**. In this study, each available data source is a publisher and can produce data to a topic in Kafka. Therefore, there will be a topic for traffic data from Highways England, another topic for traffic data from Birmingham City Council and a third topic for Twitter data. One can notice that a new data source can be easily integrated in this architecture.

This will require the creation of a new topic in Kafka and from there data can be consumed from the other components of the architecture. With the usage of a pub-sub system, like Kafka, the decoupling of the data sources from the other parts of the architecture is achieved. Form Kafka, many different pipelines can be initiated towards other components of the architecture.

As mentioned in section 4, the first requirement for traffic forecasting is the training of the models using historical data, while the second is the performance of predictions using the trained models. For the second requirement, real time data are needed to serve as input to the forecasting models. Thus, in order to fulfill these requirements two pipelines/flows in the architecture are proposed. In the first pipeline, the data from Kafka are transferred to Hadoop where they can be stored for lengthy periods of time. The volume of data per year can reach terabytes, thus the usage of the Hadoop ecosystem [64] is proposed. Hadoop allows the storage of data on a distributed file system (HDFS) that can be expanded by the integration of additional commodity hardware.

The data can reach Hadoop through Flume which can store them to HDFS, or to another storage supported in Hadoop. For the proposed platform, HBase, which is a distributed, column-oriented database that uses HDFS for its underlying storage is the preferred solution. This is due to its ability to store and retrieve structured, or semi-structured data using random access mechanisms [65]. Apache Flume [66] is a distributed, reliable, and available system for the efficient collection, aggregation, and movement of streaming data. Flume is more suitable for data ingestion in Hadoop. It has very easy configuration and can handle many common issues in writing data to HDFS or HBase, such as reliability, optimal file sizes, file formats, updating metadata, and partitioning. Kafka and Flume can complement each other and often the synergy of the two modules appear in big data architectures [67]. If besides traffic forecasting, it is desired to provide analytics/insights related to the archived traffic data, Apache Hive could be used to build a distributed data warehouse, along with a visualization/dashboard tool. Hive manages data stored in HDFS or HBase and provides a query language based on SQL to perform easily and in a more friendly way, MapReduce operations.

In the second pipeline/flow, Spark Streaming [68] is proposed to receive the streaming data from Kafka and serve them to the forecasting models as input. Spark Streaming is Sparks module for handling data as soon as they arrive from various data sources. It is a very similar API to batch processing jobs, and thus code written for batch jobs can be reused. Spark Streaming receives data streams every n seconds in a structure called DStream or discretized stream. Internally, each DStream is represented as a sequence of Resilient Distributed datasets (RDDs which are the primary abstraction in Spark), arriving at each time step. Thus, the APIs that Spark provides (Spark SQL or Spark MLib), can be incorporated in Spark Streaming. Spark Streaming uses a “micro-batch” architecture. This allows the streaming computation to be treated as a continuous series of batch computations on small batches of data. Since the data will not be used immediately (milliseconds after they are available) from the forecasting models, but be aggregated per 5 minutes or more, the “micro-batch” architecture is not a limitation for such a case and the dynamic load balancing and high throughput that Spark Streaming provides can be advantageous [69].

A DStream can be created for each topic in Kafka which, as previously mentioned, is associated with each of the available data sources. The window feature of Spark Streaming, provides the capability to perform cleaning, aggregation and enrichment to the received data. A window contains many batches of data and could intuitively have data from the last $5 - 15$ minutes or more. For example, related to the traffic data sources, the pre-processing that was described in section 4 can take place during the period that a window lasts.

Transformations of DStreams, like joins, can be accomplished to align and fuse the different DStreams in order to create a common dataset that will be used from the forecasting models. This dataset could have in each row the following information sensorId, datetime, traffic measurements (flow, average speed), number of tweets related to accidents/congestion/delay, visibility (weather info), precipitation (weather info). This fused dataset from Spark Streaming can be stored to the Hadoop/HBase to be used for the offline training of forecasting models and to an in memory database for providing real time input to the trained forecasting models. The role of the in memory database is to serve as a cache, storing the most recent data, thus allowing for rapid retrieval and real-time processing. Redis [70], which is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker is proposed for this task. In the presented case, Redis will be used as in memory database in order to make predictions in near real time. Redis has data structures such as, sorted set and hash, for handling efficiently time series and a connector is available for communicating with Spark.

The last part of the architecture refers to the component that contains the forecasting models and the social media miner. This component is composed from a number of forecasting models as described in section 4. The forecasting models can be trained using the data stored in Hadoop/HBase and make predictions using recent observations from the Redis database. In the case of the models that can work on-line, these models will be updated using the real time data from Redis. Similarly, the twitter processing pipeline presented in section 5 can be fed with tweets from the same components of the architecture. An integrator component is proposed to receive forecasts from the set of models and use assemble techniques for providing a final more accurate prediction. The final output from the integrator can be used to feed external traveler information systems (for example route planners), or traffic management systems (for example traffic signal control systems).

8 Conclusions

Experimentations presented in this paper demonstrated that the performance of different forecasting models varies depending on the type of traffic data used, forecasting timesteps, time of the day and nature of the transport network that a sensor is located. Therefore, and to achieve optimal predictions, the use of a multiple of models must be available in a traffic forecasting application. Such availability results in further complexities due to the diverse requirements for training and evaluation of each model in real-time. In addition, the use of crowd sourcing data enrichment has the potential to improve the forecasting outcomes.

Due to the above, modern traffic forecasting requires real-time collection, processing and storage of high volume, variety and velocity data. In this paper, an open big data distributed architecture has been proposed for traffic forecasting. It offers optimised data aggregation (cleaning, filtering, etc.), it allows integration and usage of real-time and historical data and enables cluster-based processing.

Acknowledgments. This research was supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. $636160 - 2$, the Optimum project, `http://www.optimumproject.eu`.

References

1. Vlahogianni, E., Karlaftis, M., Golias, J.: Short-term traffic forecasting: Where we are and where we're going. Transportation Research Part C: Emerging Technologies 43, 3–19 (2014)
2. Vlahogianni, E.I., Park, B.B., van Lint, J.W.C.: Big data in transportation and traffic engineering. Transportation Research Part C: Emerging Technologies 58, 161 (2015)
3. Ahmed, M. S., Cook, A. R.: Analysis of freeway traffic time-series data by using BoxJenkins techniques. Transp. Res. Rec. 722, 1-9 (1979)
4. VanderVoort, M., Dougherty, M., Watson, S.: Combining Kohonen maps with ARIMA time series models to forecast traffic flow. Transp. Res. C, Emerging Technologies 4(5), 307-318 (1996)
5. Lee,S., Fambro,D.: Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. Transp. Res. Rec. 1678, 179-188 (1999)
6. Williams,M.: Multivariate vehicular traffic flow prediction Evaluation of ARIMAX modeling. Transp. Res. Rec. 1776, 194-200 (2001)
7. Kamarianakis,Y., Prastacos,P.: Forecasting traffic flow conditions in an urban networkComparison of multivariate and univariate approaches. Transp. Res. Rec. 1857, 74-84 (2003)
8. Williams,M., Hoel,L. A: Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. J. Transp. Eng. 129(6), 664-672 (2003)
9. Stathopoulos,A., Karlaftis,M. G.: A multivariate state space approach for urban traffic flow modeling and prediction. Transp. Res. C Emerging Technol. 11(2), 121-135 (2003)
10. Kamarianakis, Y., Prastacos, P.: Spacetime modeling of traffic flow. Computers and Geosciences 31(2). 119–133 (2005)
11. Whittaker, J., Garside, S., Lindveld, K.: Tracking and predicting a network traffic process. International Journal of Forecasting, 51–61 (1997)
12. Ghosh,B., Basu, B., O' Mahony,M.: Multivariate short-term traffic flow forecasting using time-series analysis. IEEE Trans. Intell. Transp. Syst. 10(2), 246-254 (2009)

13. Davis,G. A., Nihan,N. L.: Nonparametric regression and short-term freeway traffic forecasting. J. Transp. Eng. 117(2),178-188 (1991)
14. Chang,H., Lee,Y., Yoon,B., Baek,S.: Dynamic near-term traffic flow prediction: System oriented approach based on past experiences. IET Intell. Transport Syst. 6(3), 292-305 (2012)
15. El Faouzi,N. E.: Nonparametric traffic flow prediction using kernel estimator. In: Proc. 13th ISTTT,pp. 41-54, (1996)
16. Sun,H. Y., Liu,H. X., Xiao,H., He,R. R., Ran,B.: Use of local linear regression model for short-term traffic forecasting. Transp. Res. Rec. 1836, 143-150 (2003)
17. Sun,S., Zhang,C., Guoqiang,Y.: A Bayesian network approach to traffic flow forecasting. IEEE Intell. Transp. Syst. Mag. 7(1), 124-132 (2006)
18. Jeong,Y. S., Byon,Y. J., Castro-Neto,M. M., Easa, S. M.: Supervised weighting-online learning algorithm for short-term traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 14(4), 1700-1707 (2013)
19. Faghri, A., Hua, J.: Roadway seasonal classification using neural networks. Journal of Computing in Civil Engineering 9(3), 209–215 (1995)
20. Lingras, P., Adamo, M.: Average and peak traffic volumes: Neural nets, regression, factor approaches. Journal of Computing in Civil Engineering 10(4), 300 (1996)
21. Dougherty, M. S., Cobbett, M. R.: Short-term inter-urban traffic forecasts using neural networks. International Journal of Forecasting 13(1), 21–31 (1997)
22. Dia, H.: An object-oriented neural network approach to short-term traffic forecasting. European Journal of Operational Research 131(2), 253–261 (2001)
23. Li, R., Rose, G.: Incorporating uncertainty into short-term travel time predictions. Transportation Research Part C: Emerging Technologies 19(6), 1006–1018 (2011)
24. Van Lint, J.W.C., Hoogendoorn, S.P., Zuylen, H.V.: Freeway travel time prediction with state-spaced neural networks: modeling state-space dynamics with recurrent neural networks. Transport. Res. Rec.: J. Transport. Res. Board 1811, 30-39 (2002)
25. Van Lint, J.W.C.: Reliable real-time framework for short-term freeway travel time prediction. J. Transport. Eng. 130 (12), 921-932 (2004)
26. Van Lint, J.W.C., Hoogendoorn, S.P., Zuylen, H.V.: Accurate freeway travel time prediction with state-space neural networks under missing data. Transport. Res. Part C 13, 347-369 (2005)
27. Liu, H., Zuylen, H.V., Lint, H.V., Salomons, M.: Predicting urban arterial travel time with state-space neural networks and Kalman filters. Transport. Res. Rec.: J. Transport. Res. Board 1968, 99-108 (2006)
28. Ma,X., Tao,Z., Wang,Y., Yu,H., Wang,Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C 54, 187-197 (2015)
29. Vlahogianni, E. I., Karlaftis, M. G., Golias, J. C.: Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. Transportation Research Part C 13(3), 211–234 (2005)
30. Khosravi, A., Mazloumi, E., Nahavandi, S., Creighton, D., Van Lint, J. W. C.: A genetic algorithm-based method for improving quality of travel time prediction intervals. Transportation Research Part C 19(6), 1364–1376 (2011)
31. He, W., Lu, T., Wang, E.: A new method for traffic forecasting based on the data mining technology with artificial intelligent algorithms. Research Journal of Applied Sciences, Engineering and Technology 5(12), 3417–3422 (2013)
32. Zhu, J., Zhang, T.: A layered neural network competitive algorithm for short-term traffic forecasting. Computational Intelligence and Software Engineering,1–4 (2009)

33. Chen, H., Grant-Muller, S., Mussone, L., Montgomery, F.: A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. Neural Computing and Applications 10(3), 277 (2001)
34. Yin, H.: Urban traffic flow prediction using a fuzzy-neural approach. Transportation Research Part C: Emerging Technologies 10, 85–98 (2002)
35. Dimitriou,L., Tsekeris,T., Stathopoulos,A.: Adaptive hybrid fuzzy rule-based system approach for modeling and predicting urban traffic flow. Transp. Res. C, Emerging Technol. 16(5), 554-573 (2008)
36. Lu, H. P., Sun, Z. Y., Qu, W. C., Wang, L.: Real-Time Corrected Traffic Correlation Model for Traffic Flow Forecasting. Mathematical Problems in Engineering 501, 348036 (2015)
37. Tan,M.C., Wong,S. C., Xu,J.M., Guan,Z. R., Peng,Z.: An aggregation approach to short-term traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 10(1), 60-69 (2009)
38. Lin, L., Ni, M., He, Q., Gao, J., Sadek, A. W., Director, T. I.: Modeling the Impacts of Inclement Weather on Freeway Traffic Speed: An Exploratory Study Utilizing Social Media Data. In: Transportation Research Board 94th Annual Meeting 15, 4749, (2015)
39. Tejaswin, P., Kumar, R., Gupta, S. Tweeting Traffic: Analyzing Twitter for generating real-time city traffic insights and predictions. In: Proceedings of the 2nd IKDD Conference on Data Sciences, p. 9, (2015)
40. Steur, R. J.: Twitter as a spatio-temporal source for incident management. Thesis, Utrecht University, Utrecht (2015)
41. Gong, Y., Deng, F., Sinnott, R. O.: Identification of (near) Real-time Traffic Congestion in the Cities of Australia through Twitter. In: Proceedings of the ACM First International Workshop on Understanding the City with Urban Informatics, pp. 7–12 (2015)
42. Ni, M., He, Q., Gao, J.: Using social media to predict traffic flow under special event conditions. In: The 93rd Annual Meeting of Transportation Research Board.,(2014)
43. Pathania, D., Karlapalem, K.: Social Network Driven Traffic Decongestion Using Near Time Forecasting. In: Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, pp. 1761–1762, (2015)
44. Abidin, A. F., Kolberg, M., Hussain, A.: Integrating Twitter Traffic Information with Kalman Filter Models for Public Transportation Vehicle Arrival Time Prediction. In: Big-Data Analytics and Cloud Computing, pp. 67–82, Springer International Publishing,(2015)
45. Highways England, `http://www.highways.gov.uk/`
46. Birmingham City Council Open Data, `http://butc.opendata.onl/AL_OpenData`
47. Twitter, `https://dev.twitter.com/`
48. Box, G. E. P., Jenkins, G. M: Time series analysis: Forecasting and control. 2nd ed. San Francisco: Holden-Day (1976)
49. Hydman, R., Athanasopoulos, G.: Forecasting principles and practice. OTexts (2013)
50. Petris, G., Petrone, S., Campagnoli, P.: Dynamic Linear Models with R. Springer Verlag New York (2009)
51. Commander, J., Koopman, S.J.:, An Introduction to State Space Time Series Analysis. Oxford University Press (2007)
52. Hastie, T., Tibshirani, R.,Friedman, J.: The Elements of Statistical Learning: Data mining, Inference and Prediction. Springer, Second Edition (2009)
53. Breiman, L.: Random Forests. Machine Learning 45(1), 5–32 (2001)

54. Haykin, S.: Neural Networks: A comprehensive foundation. Pearson, Prentice Hall, Second Edition (1999)
55. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1996)
56. Vapnik, V.N.: Statistical Learning Theory, Wiley. New York (1998)
57. R project, `https://www.r-project.org`
58. Hyndman, R.J., Yeasmin Khandakar,Y.: Automatic time series forecasting: the forecast package for R. Journal of Statistical Software 26(3), 1–22 (2008)
59. PyKalman, `https://pykalman.github.io`
60. Keras Library, `https://github.com/fchollet/keras`
61. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
62. Esuli, A., Sebastiani, F.: Sentiwordnet: A publicly available lexical resource for opinion mining. In: Proceedings of LREC 2006 6, pp. 417–422, (2006)
63. Apache Kafka, `https://kafka.apache.org/`
64. Apache Hadoop, `https://hadoop.apache.org`
65. Dimiduk, N., Khurana, A.: HBase in Action. Manning, (2013)
66. Apache Flume, `https://flume.apache.org`
67. Grover, M., Malaska, T., Seidman, J., Shapira, G.: Hadoop Application Architectures. O' Reilly Media, (2015)
68. Apache Spark, `https://spark.apache.org`
69. Karau, H., Konwinski,A., Wendell, P., Zaharia, M.: Learning Spark. O' Reilly Media (2015)
70. Redis, `https://redis.io`