International Journal of Information Management Data Insights 2 (2022) 100086

Contents lists available at ScienceDirect

International Journal of Information Management Data Insights

journal homepage: www.elsevier.com/locate/jjimei

Particle swarm optimization and RBF neural networks for public transport arrival time prediction using GTFS data

Eva Chondrodima[a,*], Harris Georgiou[a], Nikos Pelekis[b], Yannis Theodoridis[a]

[a] Department of Informatics, University of Piraeus, Greece
[b] Department of Statistics and Insurance Science, University of Piraeus, Greece

ARTICLE INFO

Keywords:
Estimated time of arrival (ETA)
Fuzzy means
General transit feed specification (GTFS)
Intelligent transportation systems
Neural networks (NN)
Particle swarm optimization (PSO)
Public transport

ABSTRACT

Accurate prediction of Public Transport (PT) mobility is important for intelligent transportation. Nowadays, mobility data have become increasingly available with the General Transit Feed Specification (GTFS) being the format for PT agencies to disseminate such data. Estimated Time of Arrival (ETA) of PT is crucial for the public, as well as the PT agency for logistics, route-optimization, maintenance, etc. However, prediction of PT-ETA is a challenging task, due to the complex and non-stationary urban traffic. This work introduces a novel data-driven approach for predicting PT-ETA based on RBF neural networks, using a modified version of the successful PSO-NSFM algorithm for training. Additionally, a novel pre-processing pipeline (CR-GTFS) is designed for cleansing and reconstructing the GTFS data. The combination of PSO-NSFM and CR-GTFS introduces a complete framework for predicting PT-ETA accurately with real-world data feeds. Experiments on GTFS data verify the proposed approach, outperforming state-of-the-art in prediction accuracy and computational times.

1. Introduction

Cities worldwide are facing a number of serious challenges as their resources and infrastructure are increasing (Breetzke & Flowerday, 2016) due to various reasons, such as population growth (Hassan, Shabbir, Iqbal, Said, Kamiran, Nawaz & Saif, 2021; Israilidis, Odusanya & Mazhar, 2021; Manfreda, Ljubi & Groznik, 2021). An emerging trend to manage these challenges is the utilisation of Information and Communication Technology (ICT) (Ismagilova, Hughes, Dwivedi & Raman, 2019), in order for the cities to become more automated and 'smarter'. The concept of 'smart' cities (Chen, Lu, Bulysheva & Kataev, 2022; Pee & Pan, 2022; Wu & Chen, 2021) has drawn increased attention in recent years (Chen & Silva, 2021; Lytras, Visvizi, Chopdar, Sarirete & Alhalabi, 2021) and has attracted significant attention by researchers from various fields, including information systems (Ismagilova et al., 2019).

One of the key aspects of efficient services and transportation of people and goods depends on the level of ICT automation and decision support systems (Arjun, Kuanr & Suprabha, 2021; Carter, Yoon & Liu, 2022; Quijano-Sanchez, Cantador, Corts-Cediel & Gil, 2020) in terms of data analytics (Dwivedi, Hughes, Kar, Baabdullah, Grover, Abbas, Andreini, Abumoghli, Barlette, Bunker et al., 2022; Kar & Kushwaha, 2021; Yadav, Kar & Kashiramka, 2021) regarding human flows, mobility patterns and events detection. In other words, 'smart' modalities contribute in the sense that the urban grid is not viewed as mere ensemble of individual statistical processes but rather a holistic complex environment of interconnected and interacting actors, services and associations between them. The more these ICT automation and data analytics become a natural part of everyday urban life, the 'smarter' the city becomes. In practice, 'smart' cities employ ICT to improve various aspects of city operation and management (Ismagilova et al., 2019), as well as deal with high-impact challenges such as energy supply, urban mobility, route management and traffic (Jafari, Kavousi-Fard, Niknam & Avatefipour, 2021). Hence, an essential part of the concept and operations in a 'smart' city is the 'smart' transport system (Chen & Silva, 2021; Kar, Gupta, Ilavarasan & Dwivedi, 2017).

A 'smart' transport system revolutionizes how cities approach mobility and manage traffic congestion, making Public Transport (PT) tasks a key area for such advancements. This incorporates all the typical efficiency factors from the viewpoint of logistics, i.e., optimal route design, accurate trip planning, capacity maximization, better marginal gain/cost ratio for each vehicle and overall for the PT service, as well as predictive maintenance and predictive-preventive failure management for the vehicle fleet. Even more importantly, these efficiency factors are multiplied by the fact that in this case the 'consumer' of this optimized PT service is the citizen, who in turn becomes more efficient in terms of net effort and productivity, as well as being of course less entangled in everyday problems when moving in the city, i.e., traffic jams,

* Corresponding author.
E-mail address: evachon@unipi.gr (E. Chondrodima).

https://doi.org/10.1016/j.jjimei.2022.100086
Received 20 October 2021; Received in revised form 30 May 2022; Accepted 30 May 2022

long waiting times, congestion points in central PT hubs (highly non-recommended in times of pandemic), etc. Overall, having 'smart' PT solutions helps passengers in managing their transit time and safety, and contributes to a more efficient transportation network as the 'veins' of modern cities. Furthermore, 'smart' PT solutions can, also, contribute to address challenges related to the spread of infectious diseases (Costa & Peixoto, 2020), such as COVID-19 pandemic, which has brought large challenges globally in several areas (Bangyal, Qasim, Ahmad, Dar, Rukhsar, Aman, Ahmad et al., 2021c; Chakraborty & Kar, 2021; Gupta, Tuunanen, Kar & Modgil, 2022; Mehra, Sarin, Singh, Sawhney & Kar, 2021; Murano, Ueno, Shi, Kawashima, Tanoue, Tanaka, Nomura, Shoji, Shimizu, Nguyen et al., 2021; Sarkar, Shankar & Kar, 2021).

Intelligent transportation systems may revolutionize the passengers' experience by developing safe, secure and comfortable travels with predictable timetables and delays (Ravi, Tigga, Reddy, Hakak & Alazab, 2022). This includes the provision of real-time information on bus timings, service disruptions and estimated arrival times at different stations. Accurate prediction of Estimated Time of Arrival (ETA) in PT stops, or PT-ETA for short, is vital for the PT agencies to deliver efficient services, as well as for the passengers planning their daily trips with minimal uncertainty. Moreover, the proper treatment of the PT-ETA task is the basis for efficient fleet management in terms of required vehicles, optimization of routes and time schedules (including seasonal trends), while additionally assisting in failure-preventive and/or failure-predictive vehicle maintenance based on actual usage figures instead of projected round-trip distances.

The task of PT-ETA is very challenging in modern urban environments. There are many factors introducing uncertainty and highly volatile behavior, ranging from Global Positioning System (GPS) signal errors to unpredictable traffic conditions, thus disturbing the scheduled time plan. In order to effectively predict PT-ETA, various studies have applied data-driven techniques over GPS traces derived from transport vehicles (Liu, Xu, Yan, Cai, Sun & Li, 2020; Petersen, Rodrigues & Pereira, 2019; Ranjitkar, Tey, Chakravorty & Hurley, 2019; Celan & Lep, 2017). Valuable information resides in the generated transit data, which can be fully exploited through data-driven techniques and provide the means to adapt and improve the PT-ETA prediction quality. The motive is to derive actionable insights with the volume, variety, and veracity of data (Kushwaha, Kar & Dwivedi, 2021) to improve the PT-ETA prediction quality.

Over the past few years, transit data have become increasingly available by the PT agencies all over the world. The PT industry has benefited from a broadly accepted open source data format, the General Transit Feed Specification (GTFS) (Google, 2021; Velasquez Ortiz, Álvarez Rodríguez, Vargas Martin & Ponce Gallegos, 2019), which has enabled PT agencies to publish their transit data and associate them with spatial and temporal characteristics. Today PT agencies can publish static transit timetabling information through the GTFS static (GTFS-s) feed or incorporate the transit network's real-time data through the GTFS real-time (GTFS-rt) feed.

Open data are essential for the 'smart' cities to operate efficiently, while accurate and up-to-date data are important to any information system dealing with such operational tasks. Pereira, Macadar, Luciano & Testa (2017) found that open data can enhance the delivery of public value in 'smart' city contexts. Nevertheless, according to Kar, Ilavarasan, Gupta, Janssen & Kothari (2019) more data might not increase the 'smartness' of citizens and various biases can occur. This possible deficiency also applies to the GTFS data.

Despite the wide availability of transit data with the GTFS format, the problem of extracting useful knowledge and usable information content inherently resides on the quality of the data (Velasquez Ortiz et al., 2019). GTFS data are often saturated with GPS noise and errors, inconsistencies and missing information. In order to address this problem, various open-source platforms (Google, 2021) have introduced validation tools for verifying GTFS-s and GTFS-rt feeds. However, these tools can not guarantee that the validated data are directly usable in data-driven methods and algorithms, as they require further cleansing and refinement (Chondrodima, Georgiou, Pelekis & Theodoridis, 2021). In addition, many of the current state-of-the-art methods in ETA prediction address the task via a closely associated one, e.g. trajectory prediction, rather than modeling the ETA task itself; as a result, the trained models may perform well for this associated task, but with no guarantee for the ETA performance in the sense of generalization level.

In this work we propose a complete pipeline for Cleansing and Reconstructing GTFS data, called CR-GTFS. A preliminary version was presented in Chondrodima et al. (2021), where the PT-ETA task was addressed at the context of the next vehicle's stop, applying various standard data-driven models on processed GTFS data; the raw data were processed by a much more limited version of the CR-GTFS pipeline. The conducted experimental analysis there showed that the Neural Network (NN) -based method outperformed other alternatives in terms of prediction accuracy.

In comparison to Chondrodima et al. (2021), in this work we investigate a more complex and challenging task, as we extend the PT-ETA look-ahead window of prediction by including more vehicle stops, i.e., in a generalized manner and much longer time horizon. Additionally, based on the previous findings in Chondrodima et al. (2021), we focus our current study primarily on NN-based methods. More specifically, we propose a novel data-driven modeling approach based on Radial Basis Function (RBF) NNs trained with an optimally adapted version of the successful Particle Swarm Optimization method using the Non-Symmetric Fuzzy Means (PSO-NSFM) algorithm (Alexandridis, Chondrodima & Sarimveis, 2013). The proposed method is applied to GTFS data processed by the currently upgraded version of the CR-GTFS pipeline. The combination of the proposed modified PSO-NSFM method with the upgraded CR-GTFS pipeline presents a novel and complete framework, capable of predicting PT-ETA accurately and efficiently.

As part of the experimental work, a case study was conducted with GTFS data collected from the American Public Transportation Association named 'Metro Transit' (MetroTransit, 2021). The results showed that the use of the modified PSO-NSFM helps in significantly improving the performance of the RBF NNs and provide increased accuracy on the PT-ETA prediction task, outperforming other comparable state-of-the-art methods (Ali, 2020; Breiman, Friedman, Olshen & Stone, 2017; Cervantes, Garcia-Lamont, Rodriguez-Mazahua & Lopez, 2020; Hagan & Menhaj, 1994; Hastie, Tibshirani & Friedman, 2008; Huang, Zhu & Siew, 2006; Klesk & Korzen, 2021; Sarimveis, Alexandridis, Tsekouras & Bafas, 2002; Theodoridis & Koutroumbas, 2008; Vapnik, 2013). Hence, the proposed framework can be applied as a robust and reliable approach for accurate prediction of PT-ETA on GTFS static and real-time data feeds.

In summary, the main contributions of this work are listed as follows:

- A new CR-GTFS pipeline for optimized pre-processing, focused on cleansing and reconstructing GTFS static and real-time data.
- An optimally adapted PSO-NSFM algorithm for training RBF NNs in order to predict PT-ETA accurately based on GTFS data.
- Comparative results with competitive state-of-the-art alternative methods, proving the increased accuracy and performance of the proposed framework.

The rest of this paper is organized as follows: Section 2 reviews the related literature; Section 3 formulates the problem definition; Section 4 presents a short description of the GTFS data; Section 5 presents the applied techniques and introduces the proposed framework for the PT-ETA prediction task; Section 6 describes the experimental protocol; Section 7 presents the results of the proposed approach, as well as comparison to other solutions; Section 8 discusses the results produced by the employing methods; Section 9 concludes by outlining the advantages of the proposed framework and setting directions for future work.

2. Literature review

A number of studies covering 'smart' cities have been published in the relevant literature (Akpinar, 2019; Chatterjee & Kar, 2015; Chen, Zou, Li, Li, Yang & Chen, 2021; Madakam, 2020; Nevado Peña, López Ruiz & Alfaro Navarro, 2020). In Chatterjee, Kar & Gupta (2018) the authors attempted to predict factors that can influence 'smart' cities highlighting the critical aspects of produced information and system quality. Chen, Guo, Su, Chen & Chang (2015) proposed a city-level data exchange system (citizen-card system, intelligent transportation system and urban regional health system), that was successfully implemented and applied in Zhenjiang city (China). The study (Corbett & Mellouli, 2017) developed a conceptual model that expands the concept of sustainable 'smart' cities. In Rybnytska, Burstein, Rybin & Zaslavsky (2018) a decision support tool was created to coordinate drivers in selecting the optimal path for garbage collection; the results highlighted the potential for a reduction in distance covered and associated CO2 emissions. Furthermore, Ang, Seng, Ngharamike & Ijemaru (2022) investigated the impact of data-driven approaches in the context of 'smart' city transportation.

Motivated by Ang et al. (2022), in this paper we employ data-driven techniques in order to address problems related to 'smart' PT. Particularly, we address the PT-ETA problem by combining the advantages of trajectory analysis (Brisaboa, Faria, Galaktionov, Rodeiro & Rodriguez, 2022; Gu, Jiang, 'David' Fan & Chen, 2022; Sun, Zhao, Zhang, Chen & Yu, 2022; Xiao, He, Yang, Liu & Liu, 2022), NNs (Bangyal, Ahmad, Shafi & Abbas, 2012; Korovesis, Kandris, Koulouras & Alexandridis, 2019; Stogiannos, Alexandridis & Sarimveis, 2018) and swarm intelligence (Alexandridis, Paizis, Chondrodima & Stogiannos, 2017; Beni, 2020; Chakraborty & Kar, 2017) in order to train the NN models. Significant amount of current state-of-the-art research work is concentrated on training NNs by using meta-heuristic techniques (Bangyal, Nisar, Ibrahim, Bin, Haque, Rodrigues & Rawat, 2021b; Ji, Zheng, Zhuang & Lin, 2021; Şen, Dönmez & Yıldırım, 2020; Thorat, Parekh & Mangrulkar, 2021) such as PSO (Bangyal, Ahmad, Rauf & Shakir, 2018b; Bangyal, Hameed, Alosaimi & Alyami, 2021a; Pervaiz, Ul-Qayyum, Bangyal, Gao & Ahmad, 2021; Rauf, Bangyal, Ahmad & Bangyal, 2018), bat algorithm (Bangyal, Ahmad & Rauf, 2019; Bangyal, Ahmad, Rauf & Pervaiz, 2018a; Bangyal, Ahmed & Rauf, 2020; Haider Bangyal, Hameed, Ahmad, Nisar, Haque, Ibrahim, Asri, Rodrigues, Khan, Rawat et al., 2022), simulated annealing (Alexandridis & Chondrodima, 2014), tabu search (Karamichailidou, Kaloutsa & Alexandridis, 2021; Zhang, Liu, Zhou & Zhang, 2020), differential evolution (Karamichailidou, Alexandridis, Anagnostopoulos, Syriopoulos & Sekkas, 2022) and many others (Batra, Jain, Tikkiwal & Chakraborty, 2021; Chakraborty & Kar, 2016).

In the core task of trajectory prediction of moving objects (Yu, Zhou, Wang, Pu, Cheng & Chen, 2021), a number of algorithms have been applied in various domains, including aviation, maritime and urban traffic (Georgiou, Pelekis, Sideridis, Scarlatti & Theodoridis, 2020a; Georgiou, Petrou, Tampakis, Sideridis, Chondrodima, Pelekis & Theodoridis, 2020b; Petrou, Tampakis, Georgiou, Pelekis & Theodoridis, 2019). Moreover, some studies have addressed the ETA prediction task, in the aviation domain, where NNs have been used successfully for the climb/vertical trajectory prediction (Le Fablec & Alliot, 1999) or in relation to the air traffic flows for ETA at the destination (Cheng, Cui & Cheng, 2003).

Several studies use Machine Learning (ML) approaches (Tripathi, Goswami, Trivedi & Sharma, 2021) to predict travel times based on GPS traces from vehicles (Ghanim, Shaaban & Miqdad, 2020; Liu et al., 2020; Celan & Lep, 2017), or the so-called live Automatic Vehicle Locations (AVL) data (Hua, Wang, Wang & Ren, 2018; Petersen et al., 2019; Ranjitkar et al., 2019). In Larsen, Yoshioka & Marte (2020) an NN was employed to predict the travel times of buses using open real-time data derived from Sao Paulo City bus fleet location, real-time traffic data and traffic forecast from Google Maps. In Alam, Kush, Emami & Pouladzadeh (2020) a Recurrent NN (RNN) architecture predicted the ETA irregularities by exploring live AVL data from buses, provided by the Toronto Transit Commission, along with schedules retrieved from GTFS and weather data.

Nevertheless, there is a limited amount of work in the literature regarding specifically the PT-ETA prediction task when using GTFS static and real-time feeds, most importantly during the transit (along the trip) and not only for a selected final destination. For example, Sun, Pan, White & Dubey (2016) combined clustering analysis with Kalman filters to predict arrival times at various bus stops in Nashville (TN, USA) by estimating the actual delay versus the pre-planned scheduled ETA, using GTFS static and real-time data, as well as historical bus timing data.

Besides (Alam et al., 2020), Long Short Term Memory (LSTM) models for predicting ETA of busses were also explored in Celan & Lep (2017), where heterogeneous information about the transport environment were taken into account; the model evaluation was conducted by using operational bus data from Samara (Russia). Petersen et al. (2019), proposed a method for bus travel time prediction that leverages the non-static spatio-temporal correlations presented in urban bus networks, allowing the discovery of complex patterns, not easily captured by more traditional methods. The model was composed of convolutional and LSTM layers and it was evaluated with a dataset from Copenhagen's 'Movia' public transport authority. Furthermore, in Liu et al. (2020) an LSTM-based comprehensive prediction model was proposed by taking into account spatio-temporal feature vectors and data provided by Xingtai bus company.

All the aforementioned LSTM-based approaches present a very promising alternative to automatic pattern discovery in trajectories and ETA prediction. However, there are three main drawbacks and negating factors in employing LSTM in such tasks, namely: (a) requiring long training sequences in order to converge properly, (b) requiring increased susceptibility to noise and poor data quality, and (c) requiring increased computational complexity in terms of training time. In the context of PT-ETA prediction based on stops (i.e., not on the entire full-resolution GPS trajectory data), the input sequences are in the order of much less than a hundred data points, i.e. much less than what a typical LSTM requires for proper convergence. Moreover, running PT-ETA predictions as the trip evolves within very narrow time frames, instead of batch/offline mode with plenty of time available for computations, makes the employment of LSTM very problematic in practice.

Besides LSTM networks, researchers have employed other NN architectures as well. Wang, Zuo & Fu (2014) combined historical data and real-time transit information to forecast the bus arrival time based on two phases: a) an RBF NN approach for modeling the historical data, and b) an online-oriented adaptive method that uses the currently available information to modify the RBF. Also, some studies exploit dwell times, e.g. Jeong & Rilett (2004) developed a model to predict bus arrival time by using automatic vehicle location data and by taking into account schedule adherence and dwell times, i.e., the durations of the transit vehicle stops when serving passengers. In Amita, Singh & Kumar (2015) NNs were applied for bus travel time prediction, taking into account dwell time, delays and the distance between the bus stops.

Other studies focus more on the online aspect of processing, e.g. in Larsen et al. (2020) the purpose was to train a NN for predicting travel times of buses based on real-time open data, exploiting traffic data and forecasts from Google Maps. Similarly, in Ghanim et al. (2020) data from public bus routes were used to predict transit travel time for entire journeys using NNs. Kodiyan & Francis (2020) focused on predicting delays and grouped them into classes based on similarities on real-time bus transits. Furthermore, Ranjitkar et al. (2019) focused on improving the bus passenger experience in terms of bus ETA prediction by investigating various time series and regression-based techniques suitable for bus arrival time modeling.

It is evident that in the current state-of-the-art in the PT-ETA prediction task there are only few studies working directly with GTFS data. More specifically, Sun et al. (2016) combined clustering analysis with Kalman filters to predict arrival times at various bus stops by calculat-

ing the delay versus a scheduled time. Information is drawn from GTFS static and real-time data, as well as historical bus timing data and the model was deployed in Nashville (TN, USA).

3. Problem formulation

Using a GTFS dataset, the PT-ETA prediction task can be stated as follows:

- **Given**: an input vector $\mathbf{u}' = [u'_{t_s-k_s}, \dots, u'_{t_s-1}, \tilde{u}'_{t_s,\, t_s+q}]$, where t_s is the current vehicle stop, u'_{t_s-b} contains sequential information about passing through stop $t_s - b$, $b \in \{0, \dots, k_s\}$ and $\tilde{u}'_{t_s,\, t_s+q}, q > 0$ contains information about current stop t_s and future vehicle stop $t_s + q$,
- **Predict**: the arrival time or $dT_{t_s,\, t_s+q}$ towards a future vehicle stop q in sequence.

Further analysis and details concerning the problem formulation can be found in Appendix A.

4. General transit feed specification (GTFS) data

As already mentioned, GTFS data exist in two main variants: the GTFS-s feed and the GTFS-rt feed. The former includes static information, with updates occurring periodically, such as time schedules that might be updated only a couple of times in a year. The latter refers to transit data collected during the vehicle trips and are usually based on GPS tracking. Both GTFS feeds are described in detail in Appendix B.

In this work real-world GTFS (static and real-time) data were used, which were collected from the American Public Transportation Association named 'Metro Transit' (MetroTransit, 2021). More information about the PT agency and the exact data feed details can be found in Appendix C.

5. Methods

In this section, we present all the applied techniques, including the proposed CR-GTFS pipeline, a description of the main concepts behind the PSO-NSFM algorithm and the modifications made in the proposed approach. Finally, we present the proposed framework's robustness.

5.1. Data collection

GTFS has been widely adopted in the past decade by transit agencies as the standard format for sharing transit data. However, several issues and quality degradation need to be resolved before GTFS input can used by data-driven methods for predictive analytics such as the PT-ETA prediction task. These issues can be identified and resolved effectively by applying the proposed CR-GTFS pipeline to the static and real-time data feeds.

The CR-GTFS environment operates across multiple processes, whereas an overview of the pipeline is presented in Fig. 1. The process concerning the GTFS-s feed is executed in parallel to the process concerning the GTFS-rt feed. First, both processes retrieve the respective data periodically and store them depending on whether the data have changed or not. Particularly, the GTFS-s feed is downloaded every two hours and the GTFS-rt data are downloaded every five seconds. In both processes, the download module is followed by the version-check module, which compares the version of the downloaded data with the version of the stored data. Then the downloaded data pass through the next steps if the two versions differ. The data are stored and structured in a PostgreSQL database following the GTFS data format, with the addition of executing spatial operations implemented with PostGIS. Next, we describe the aforementioned processing steps.

5.1.1. GTFS static data processing

In the PostgreSQL database each one of the.txt files of the GTFS-s feed constitutes a table with the same name as the original.txt file. Once the GTFS feeds have been downloaded, the processing detects incomplete, incorrect or inaccurate information and reconstructs them as follows.

First, the CR-GTFS tool detects and fixes errors in each table:

- Each of the following fields should be unique in table 'routes': 'route_id', 'route_short_name', 'route_long_name', 'route_url'. The duplicated fields are eliminated.
- Each of the following fields should be unique in table stops: 'stop_id', 'stop_code', 'stop_name', 'stop_url'. Also, each 'stop_id' should correspond to a unique point defined by 'stop_lat' and 'stop_lon'. Any duplicate fields are eliminated.
- In table 'trips' each 'trip_id' and 'trip_headsign' should be unique for each 'route_id'. Any duplicate fields are eliminated.
- In table 'stops_times' the values of the field 'stop_sequence' should increase and be unique along the trip, otherwise the out of sequence or the duplicated stops are eliminated. Also, the 'arrival_time' and 'departure_time' should increase along the trip and should not be the same at three or more consecutive stops (otherwise the stop is eliminated). In addition, each 'trip_id' should comprise an adequate number of stops (otherwise the 'trip_id' is eliminated).
- In table 'shapes' the sequence of points in each 'shape_id' should be in ascending order (otherwise the out of order points are eliminated), two consecutive points within a 'shape_id' should be different (otherwise the second point is eliminated).

Subsequently, the GTFS-s data tables are merged according to common keys as shown in Fig. 2, where only the values of the matching keys are maintained (inner join). Also, in Fig. 2 the different colors indicate different processes, which can be executed in parallel. For instance, one such process is the following: the table 'stops' is merged with the table 'stop_times' by using as key the 'stops_id'. Hence, the table trips include the full information provided by the GTFS-s and the redundant values are eliminated.

Next, the CR-GTFS pipeline detects spatial problems related to the actual road network generated by the OpenStreetMap contributors (2017). Specifically, (a) the shapes coordinates and sequence should match the OpenStreetMap road network coordinates and direction respectively, otherwise the faulty points are eliminated, and (b) the stops coordinates and sequence should match the coordinates and direction, respectively, generated by the available shapes; the stops coordinates match the shapes when the stops lie within a predefined small distance (a few meters) of the shape for the respective trip. Also, the shapes must comply with the boundaries of the PT agency's service area. The exact numbers and thresholds for these tests are GTFS-dependent and in this experimental work they have been extensively explored and validated.

Furthermore, the CR-GTFS pipeline creates two new fields called 'shape_dist' and 'trip_stop_dist'. For the first, for each 'shape_id' and for each point it defines the network distance traveled from the first point of the specific 'shape_id' by using information from OpenStreetMap road network. Based on 'shape_dist', the 'trip_stop_dist', for each 'trip_id' and for each 'stop_id', defines the network distance traveled from the first stop of the specific 'trip_id'.

The resulting processed GTFS-s data provide complete timetable information of high quality (error-free) and usable as training datasets for data-driven methods.

5.1.2. GTFS real-time data processing

Following the GTFS-s processing step, the process of cleansing and reconstructing the GTFS-rt feed for PT-ETA prediction purposes is implemented. In the PostgreSQL database each one of the real-time feeds constitutes a table with the same name as the original feed. Once the GTFS-rt feeds have been downloaded, the processes of detecting incomplete, incorrect or inaccurate information and reconstructing the data also applies here.

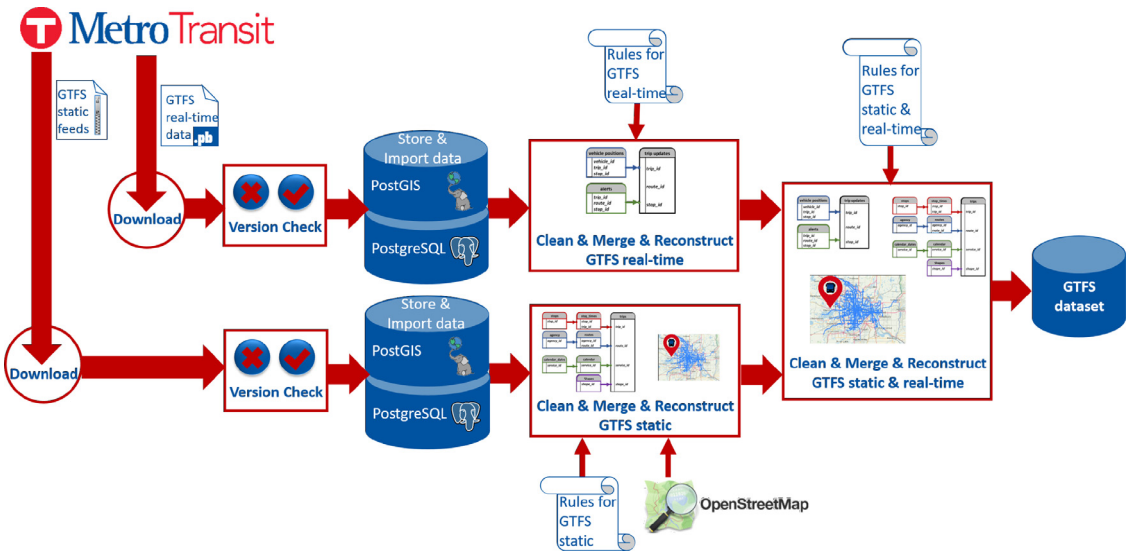

Fig. 1. Overview of proposed CR-GTFS pipeline.

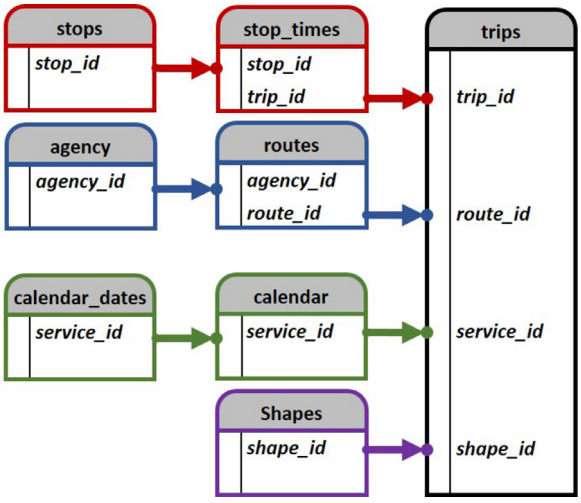

Fig. 2. GTFS static merging process.

First, the TripUpdate and the VehiclePosition tables are merged. Due to the fact that both real-time feeds often include missing or faulty values related to their primary keys, the merging procedure is not trivial. We address this problem by employing multiple key combinations, including 'trip_id', 'route_id', 'direction_id', 'vehicle_id' and 'vehicle_label'. For instance, if 'trip_id' is not available in the feed, it can be replaced by 'route_id' and 'direction_id'. By using the aforementioned merging procedure, the CR-GTFS pipeline fills the common real-time missing information and eliminates redundancies. Then, the CR-GTFS pipeline detects and eliminates any records with invalid timestamps. Specifically, each new valid timestamp generated by a specific vehicle operating on a specific trip, should increase compared to its previous valid timestamp (small positive increments).

In order to reconstruct real-time data of high quality, the CR-GTFS pipeline merges the resulted real-time data with the reconstructed static data according to two common keys, the 'trip_id' and the 'route_id' (inner join). Hence, only the records generated by vehicles operating in accordance with the transit schedule are maintained.

Subsequently, the ServiceAlerts feed is merged with the scheduled data in order to: (a) delete the stops from the specific routes that are characterised by the alerts as closed, and (b) delete the part of the route that is indicated in the alerts as detoured. Next, the CR-GTFS pipeline detects errors related to mismatches and inconsistencies between the real-time data and the scheduled data (changed by the alerts) and resolves them. First, incorrect position data are eliminated via spatial filtering, e.g. GPS jittering. Specifically, each vehicle's new broadcasted position should reside within a small distance (a few meters) from the respective GTFS shape. Also, each vehicle's new broadcasted position compared to its previous valid position should match the direction of the respective GTFS shape and the calculated road network distance between theses points should result in a reasonable value (a few meters). In addition, for each vehicle operating on a specific trip, each new recorded timestamped position should correspond to a reasonable speed (depending on the vehicle type) calculated from the previous timestamped position. The vehicle speed, which is calculated by using the road network distance between the current and the previous positions and the corresponding time horizon, should also follow the minimum and maximum speed limits defined by the PT data provider, otherwise it is eliminated as error.

The CR-GTFS tool also detects inconsistencies related to trip start/end times and positions. Specifically, GTFS-rt feeds report updates only for the trips facilitated by vehicles in service. Thus, the vehicles and the trips that are not in progress are not included in the real-time feed updates. Following this, a trip starts when:

- a vehicle is assigned to the specific trip, broadcasts timestamped positions and no valid records have reported in previous real-time updates;
- the broadcasted positions fall within a small distance (50 meters) from the position of the first scheduled stop of the specific trip;
- the broadcasted timestamp matches the scheduled arrival time of the first stop of the specific trip, where a small time difference is permitted, which is equal to the calculated maximum delay time of the trips of the same road at the first stop.

Similarly, a trip ends when:

- the assigned vehicle on the specific trip does not broadcast updates for a long time, or reports a different trip/route/direction for consecutive timestamps;
- the broadcasted positions fall within long distances from all the scheduled stops of the specific trip;
- the broadcasted timestamp is higher than the scheduled departure time of the last stop of the specific trip and their time difference is higher than a specific amount of time, which is equal to the calcu-

lated maximum delay time of the trips of the same road at the last stop.

Although the processed GTFS data provide usable knowledge for building data-driven models for PT purposes, they cannot facilitate PT-ETA prediction on stops yet, because they lack the information concerning the actual arrival and departure times of the vehicles at the stops. However, this is a challenging task due to a number of reasons, such as the fact that sometimes no location is reported near a stop as the vehicle passes the stop location too quickly (no stop). The CR-GTFS pipeline also addresses this problem of discovering and reconstructing the information concerning the actual arrival/departure time of the vehicles at stops in real-time mode.

More specifically, the CR-GTFS pipeline matches each new valid timestamped position broadcasted by a vehicle operating on a specific trip to the closest scheduled stop of the specific trip along the shape path. A broadcasted position-stop-match is valid when the network distance between the broadcasted position and the scheduled stop falls within the range of 30 meters. As the vehicle moves, multiple broadcasted positions may match the same scheduled stop within the range of 30 meters. Thus, for each new valid broadcasted timestamped position, the CR-GTFS compares the matches and maintains only the broadcasted position-stop-match that indicates the minimum network distance from this stop location. Also, as the vehicle moves, a valid broadcasted timestamped position-stop-match may occur while the previous stop in sequence remains without a valid match, i.e., the vehicle passed by the stop but no valid timestamp was reported for it. In this case, the timestamp for the remaining stop (that does not have a matched timestamp) can be estimated by using the road network distance and the speed of the vehicle from the previous and next positions of the specific stop's location.

After tackling all these challenges and quality degradation factors via this extensive CR-GTFS pipeline, the resulting data provide complete static and real-time information of high quality regarding the transit system, capable of being used for training PT-ETA predictive models.

5.2. Data analysis

5.2.1. RBF training

RBF networks (Moody & Darken, 1989) are described in Appendix D. Standard approaches decompose the problem of RBF network training in two steps. In the first step the determination of the hidden layer dimension takes place, while in the second step the synaptic weights are calculated by using linear least squares regression in matrix form as:

$$\mathbf{w}^T = \mathbf{Y}^T\mathbf{Z}(\mathbf{Z}^T\mathbf{Z})^{-1} \tag{1}$$

where $\mathbf{Z} = [\mathbf{z}(1), \mathbf{z}(2), \dots, \mathbf{z}(K)]^T$ and $\mathbf{Y} = [\mathbf{y}(1), \mathbf{y}(2), \dots, \mathbf{y}(K)]$ are matrices containing the hidden layer outputs and the target values for all data points, respectively.

As far as the first step of the RBF training procedure is concerned, the determination of the hidden layer dimension is a difficult task with no guarantee of strict optimality. Conventional training methods such as the popular *k*-means algorithm often suffer from multiple disadvantages (Alexandridis, Chondrodima & Sarimveis, 2016; Kushwaha, Pant & Sharma, 2019). An alternative to these methods is provided by the fuzzy means (FM) algorithm (Sarimveis et al., 2002) and its variants (Alexandridis et al., 2013; Alexandridis, Sarimveis & Ninos, 2011), which have found many successful applications (Alexandridis & Chondrodima, 2014; Alexandridis, Chondrodima, Efthimiou, Papadakis, Vallianatos & Triantis, 2014a; Alexandridis, Chondrodima, Paivana, Stogiannos, Zois & Sarimveis, 2014b; Alexandridis, Stogiannos, Papaioannou, Zois & Sarimveis, 2018; Karamichailidou et al., 2021; Stogiannos et al., 2018). A brief discussion about the FM algorithm and its variants is provided next.

5.2.2. Symmetric fuzzy means algorithm

The FM algorithm has the ability to automatically determine the RBF network size in one step (non-iteratively), i.e., the number of RBF kernel centers and their positions. The FM algorithm is based on a symmetric fuzzy partition of the N-dimensional input space and, thus, it is also called symmetric FM (SFM). Due to the symmetric partitioning concept, the domains of all input variables are partitioned into equal fuzzy sets and the optimum fuzzy partition s' is necessary to be determined in order for the SFM to operate optimally.

In order to find s', in this work an exhaustive search is employed, where all partitions are tested ranging from $s_{\min}$ to $s_{\max}$, which correspond to the lower and upper bounds on the number of fuzzy sets, respectively. Procedures based on exhaustive search are usually time consuming, but in this work we take advantage of the extremely fast training procedure adopted by the SFM. Thus, finding the optimal symmetric fuzzy partition is a rather fast and easy task to optimize, as we only need to test a relatively small number of RBF networks, which are trained very fast and their total number is equal to $1 + s_{\max} - s_{\min}$.

5.2.3. Non-symmetric fuzzy means algorithm

A non-symmetric variant of the FM, namely NSFM algorithm (Alexandridis et al., 2011), utilizing a non-symmetric fuzzy partition of the input space, has been proposed in Alexandridis et al. (2011) offering increased accuracy over the SFM approach. Particularly, the NSFM is based on a non-symmetric fuzzy partitioning of the N-dimensional input space, which implies that the domains of all input variables are partitioned into s_j fuzzy sets, where $j = 1, 2, \dots, N$. The SFM algorithm can be considered as a special case of the NSFM when all input variables are partitioned into the same number of fuzzy sets, i.e., $s_1 = s_2 =, \dots, = s_N$. Following the SFM approach, the NSFM also requires only a single pass of the training data to perform the centers estimation stage.

Furthermore, in order for the NSFM to operate successfully, the optimal non-symmetric fuzzy partition is necessary to be determined. The NSFM algorithm defines a more complicated network design problem contrary to the SFM approach. Thus, finding the optimal fuzzy partition in the NSFM algorithm is a more challenging problem. For low-dimensionality problems the optimal partition can be estimated by using an exhaustive search, as previously mentioned. However, in high-dimensionality problems we need to employ a slightly different approach in order to optimally determine an increased number of operational parameters. Although the NSFM algorithm also adopts a fast training procedure, the increased number of operational parameters makes the use of an exhaustive search procedure prohibitive. Hence, the PSO approach is employed instead for NSFM, as is being described in the following section.

5.2.4. Modified PSO-NSFM algorithm

In this work, a modified version of the PSO-NSFM algorithm is proposed to train the RBF networks. The proposed algorithm is based on the original PSO-NSFM method (Alexandridis et al., 2013) and the SFM approach, where the former is based on a combination of the NSFM approach and the PSO (Engelbrecht, 2007) method. A description of the main concepts behind the PSO-NSFM algorithm is provided next, along with the proposed modifications that were made in order to build RBF-based models with increased prediction capabilities concerning the specific PT-ETA prediction task.

In Alexandridis et al. (2013), the PSO-NSFM method was employed as a search method for exploring the space of solutions by encoding each individual one to reflect the input space partitioning. Specifically, the standard PSO is based on a population of possible solutions coded as particles, where each particle has two vectors, namely position **s** and velocity **v**. The position vector corresponds to the number of fuzzy sets. The PSO-NSFM algorithm comprises of a swarm that includes a total number of P particles and at iteration t each particle i with $i = 1, 2, \dots, P$ contains $\mathbf{s}_i(t) = [s_1(t), s_2(t), \dots, s_N(t)]^T$ elements, which represent the number of fuzzy sets assigned to each one of the N dimensions.

The original PSO-NSFM algorithm starts by initializing P randomly with different partitions of the N-dimensional input space. In this work we modify the initialization procedure by selecting randomly $P-1$ particles. The remaining particle $\mathbf{s}_P(t)$ contains the optimum symmetric fuzzy partition s', which is defined by the SFM algorithm using an exhaustive search testing all partitions ranging from $s_{\min}$ to $s_{\max}$. The remaining particle at iteration t is encoded as: $\mathbf{s}_P(t) = [\underbrace{s'(t), s'(t), \ldots, s'(t)}_{N}]^T$. It should be noted that by initializing a particle of the swarm with the optimum fuzzy partition of the SFM ensures that the PSO will start its search from an already 'good' solution.

Next, in each iteration t, each particle in the swarm brings to action an RBF network by determining its centers with the NSFM algorithm and calculating the synaptic weights according to Eq. (1). Thus, P RBF networks are trained and provide P fitness values. These values are estimated by applying the error-related criterion, which in this case is the Root Mean Square Error (RMSE).

Based on these P fitness values, the particles of the swarm interact with each other to optimize their search target. These interactions depend on the individual best position $\mathbf{y}_i(t)$ of each particle i and the global best position $\hat{\mathbf{y}}(t)$ of the swarm, which are calculated as follows:

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t), & \text{if } f(\mathbf{s}_i(t+1)) \geq f(\mathbf{y}_i(t)). \\ \mathbf{s}_i(t+1), & \text{otherwise.} \end{cases} \tag{2}$$

$$f(\hat{\mathbf{y}}(t)) = \min(f(\mathbf{y}_1(t)), \ldots, f(\mathbf{y}_P(t))) \tag{3}$$

where $f()$ is the fitness function. At each iteration t each particle i in the swarm updates its position by adding the corresponding velocity vector $\mathbf{v}_i(t)$ as:

$$\mathbf{s}_i(t+1) = \mathbf{s}_i(t) + \mathbf{v}_i(t+1) \tag{4}$$

As far as the velocity calculation is concerned, the original PSO-NSFM updates the velocity $v_{i,j}(t+1)$ of particle i in dimension j at iteration $t+1$ by using the standard formula (Alexandridis et al., 2013). However, the proposed modified PSO-NSFM updates the velocity $v_{i,j}(t)$ by employing an adaptive inertia weight strategy (Abbas, Gu, Farooq, Asad & El-Hawary, 2017), along with the classical method of 'crazy' particles (Abbas et al., 2017):

$$v_{ij}(t+1) = \begin{cases} \text{rand}(0, V_{\max}), & \text{if } \rho > r_{3,ij}(t) \\ \text{round}\big(\omega v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - s_{ij}(t)] \\ \ldots + c_2 r_{2j}(t)[\hat{y}_j(t) - s_{ij}(t)]\big), & \text{otherwise.} \end{cases} \tag{5}$$

where $V_{\max}$ is the velocity clamping constant, ρ is the probability of 'craziness', ω is the inertia weight, c_1 and c_2 are acceleration constants, $r_{1,j}(t)$, $r_{2,j}(t)$ and $r_{3,ij}(t)$ are random values in the range [0, 1], sampled at each iteration t from a uniform distribution.

The inertia weight strategy affects the position of the particle, where larger values of ω result in stronger global exploration and weaker local mining ability. By practice, good results can be obtained when the ω is dynamically adjusted compared to a fixed value (Li, Wu, He, Bashir, Liping & García, 2020), where a larger ω is applied at the beginning of the search and is gradually decayed (Abbas et al., 2017). Thus, in this work we define ω through the adaptive inertia weight (Abbas et al., 2017) as follows:

$$\omega(t) = \omega_{\max} - (\omega_{\max} - \omega_{\min})(t/t_{\max}) \tag{6}$$

Such gradually decreasing inertia weight strategies in the first few iterations increase the probability of locating the global optimum peak. The original PSO-NSFM lacks the use of inertia, which implies that ω is equal to 1, i.e., it is a large value that leads to strong global searchability and weak local mining ability during the optimization procedure.

Additionally, the method of 'crazy' particles was employed to maintain momentum in the PSO search and avoid saturation. Classic PSO algorithms tend to converge prematurely to a local minimum solution. The method of 'crazy' particles randomize some of the particle velocities based on the probability ρ of 'craziness', where in this study is given by:

$$\rho(t) = \omega_{\min} - \exp\Big(-0.4\,\big(\omega(t)/\omega_{\max}\big)\Big) \tag{7}$$

Note that ρ decreases as t increases. Hence, by using 'crazy' particles we also help the algorithm to avoid premature convergence, due to the proposed initialization procedure that involves SFM.

Furthermore, like the original PSO-NSFM algorithm, the proposed modified PSO-NSFM controls the exploration-exploitation trade-off by a 'velocity clamping' mechanism, bounding the elements of the velocity vector between predefined values:

$$v_{ij}(t+1) = \begin{cases} v_{ij}(t+1), & \text{if} ||v_{ij}(t+1)|| < V_{\max} \\ \pm V_{\max}, & \text{otherwise} \end{cases} \tag{8}$$

Similarly to the original PSO-NSFM algorithm, the proposed modified PSO-NSFM, after updating the positions of all particles in the swarm, returns to the RBF network creation stage until one of two stopping conditions has been met: (a) reaching a maximum number of iterations ξ_1, or (b) the normalized swarm radius R_{norm} becomes smaller than the minimum normalized swarm radius value ξ_2, which is calculated by:

$$R_{norm}(t) = \frac{R_{\max}(t)}{R_{\max}(1)}, \text{ where } R_{\max}(t) = \max_{1 \leq i \leq P} \big[||\mathbf{s}_i(t) - \hat{\mathbf{y}}(t)||\big] \tag{9}$$

In summary, the proposed modified PSO-NSFM approach provides the following significant improvements: (a) PSO is initialized by including the optimum fuzzy partitioning provided by SFM to ensure that the PSO will start its search from an already 'good' solution, (b) PSO includes the adaptive inertia weight enhancement to dynamically adjust its search speed and stability, and (c) PSO uses the 'crazy' particles enhancement to maintain momentum in the PSO search and avoid premature stopping.

5.2.5. Complete framework for accurate prediction of PT-ETA

The combination of the modified PSO-NSFM algorithm and the CR-GTFS pipeline forms a novel complete framework for predicting PT-ETA accurately. This framework constitutes of two phases, the offline phase where the learning procedure takes place and the online phase where the predictions of the PT-ETA model are produced.

In the offline mode the upgrated CR-GTFS tool downloads, clears, merges, reconstructs and stores the processed GTFS data. The data are then fed to the RBF network, which is trained with the modified PSO-NSFM approach. The trained RBF network is stored in the system.

In the online mode the CR-GTFS pipeline executes some of its tasks in parallel with the RBF-based process. Specifically, the RBF-based process is executed after: (a) the CR-GTFS pipeline has completed the static GTFS processing, (b) the CR-GTFS pipeline has completed all the real-time GTFS processing except the data storing stage, and (c) the PT vehicle has moved along a pre-defined number of stopping points of the trip in question. The trained RBF model can generate predictions when the information of a pre-defined number of stops of the trip in question is available. To this end, the online RBF-based procedure predicts the PT-ETA of the vehicle and the trip in question. This online procedure is repeated every 5 s, as the GTFS real time feed is also downloaded every 5 s.

Combined together, these offline and online steps of the proposed approach provide a complete and unified framework for the PT-ETA prediction task based on GTFS data feeds.

5.3. Robustness checks

It is evident from the descriptions in the previous sections that the GTFS data management is by far a non-trivial task. Besides all the typical data quality deficiencies that are expected in real-world data sources, like missing data, sensing errors, statistical biases, etc, in the case of GTFS there are numerous additional problems that have to be addressed before the data are usable for proper model training. These deficiencies

include out-of-order data updates in real-time feeds, transition via stops without actually stopping (no passenger embarkation or disembarkation), GPS tracks that do not match the referenced location of bus stops (too far away, e.g. due to temporary detours), as well as other statistical limitations. For example, PT-ETA prediction essentially translates in processing a sequence of only a very limited set of reference points (bus stops), rather than entire full-resolution GPS tracks, e.g. as in the trajectory prediction tasks.

Considering all these limitations and constraints, the proposed approach introduces a dedicated pre-processing pipeline as described above. The purpose is to correct typical errors in mobility data (as it is usually the case with trajectory reconstruction pipelines), as well as to 'discover' the actual movement of a vehicle between specific pre-defined reference points, estimating with a reliable and accurate way the corresponding transition distances and timestamps. These data present the true information content provided by the GTFS feeds; this is the input, which the model training can actually use for addressing the PT-ETA task itself rather than any other closely associated task, e.g. performing trajectory prediction and then matching it to bus stops.

In this sense, this whole study was designed around the GTFS-based PT-ETA task and the inherent limitations, data quality degradation and statistical constraints. Standard cross-validation procedures, enhanced with additional bias-removal (multiple randomized iterations) and more conservative generalization estimation (multiple validation levels), were employed throughout all the stages of the experimental work. The CR-GTFS pipeline was designed separately as a stand-alone processing stage, i.e., regardless of the subsequent model training stage. However, even at this stage, maximum effort was put to include only minimal and 'loose' constraints regarding the removal of data errors and deficiencies; this is an intentional design specification, i.e., to accept a moderate level of extremes, outliers and errors in the training data, in order to increase the quality and the generalization level of the trained models, even at a (marginal) cost of their precise performance in this specific GTFS data provider or PT service.

More details regarding the exact experimental protocol, including dataset handling (pertitioning, cross-validation, etc), is provided in Section 6 (more importantly in Section 6.2).

6. Experimental study

This section presents the evaluation procedure regarding all the implemented approaches, as well as comparative experiments.

6.1. GTFS dataset

The proposed modified PSO-NSFM was evaluated on real-world GTFS data and compared with various compatible state-of-the-art methods. Real-world GTFS (static and real-time) data provided by Metro Transit PT agency for one month period (March 2021) were downloaded and processed with the improved CR-GTFS pipeline, as described in Section 5.1. Figure 3 depicts an overview of the extracted data routes of the processed dataset. March was chosen as the target month because it did not include any major holidays or exceptional weather events that would have impacted transit service in any of the cities in the dataset, i.e., 'exceptional' days that would constitute statistical outliers. For the PT-ETA task the processed dataset was prepared according to the problem formulation described in Section 3. Particularly, in order to predict the PT-ETA at the 6th stop beyond the current one (look-ahead span n=6), the models used the following information as input:

a) Sequential information from the previous four PT stops of the current stop:
   - target stop location expressed by longitude and latitude
   - target stop sequence, which corresponds to the stop order in the trip
   - actual distance travelled between 5 consecutive stops (current and 4 previous)
   - time travelled between 5 consecutive stops (current and 4 previous)
   - mean estimated speed (last stops pair)

b) Sequential information for the current and future stop:
   - current and future stop location expressed by longitude and latitude
   - current and future stop sequence
   - actual network distance between the current stop and the future target stop

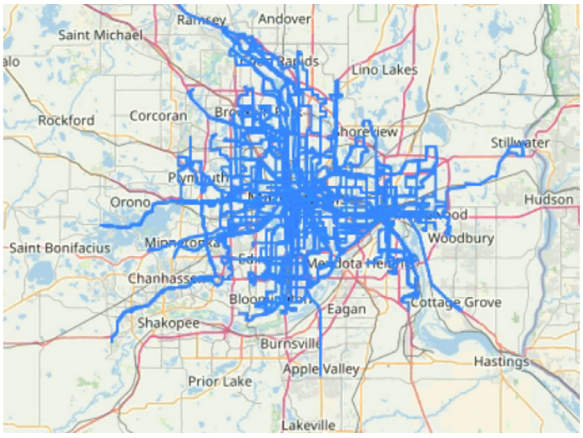

Fig. 3. MetroTransit dataset generated by the CR-GTFS pipeline.

6.2. Performance evaluation and hyper-parameter optimization

For evaluation purposes, standard k-fold cross-validation (kfCV) process (Theodoridis & Koutroumbas, 2008) was employed for all trained models. The kfCV method can be used for both model selection/hyper-parameter optimization and for model performance evaluation. However, when the same dataset is employed for both procedures simultaneously, the kfCV method may provide upwardly biased model evaluation, i.e., over-estimation of its true generalization capabilities.

In order to overcome this bias, a slightly different type of kfCV was employed. Specifically, the validation procedure involved dividing the available data into k non-overlapping partitions, using one fold as the testing set and the remaining folds as the training set, according to the kfCV. Hence, k different training sets and k corresponding testing sets were created. Then, within the kfCV method, a hyper-parameter optimizing procedure (such as grid search, or PSO algorithm) employed the kth training set for network parameter calculation and the kth testing set for estimating the optimality. Subsequently, the selected parameters were used for training and evaluating $k-1$ models on the $k-1$ combination of training and testing sets, i.e., excluding the kth training and testing sets that where used in the network selection procedure. Thus, the model performance is evaluated as the mean value over the remaining $k*(k-1)$ splits.

A kfCV process was applied with $k=5$ folds to all the trained models. As a result, the initial dataset was partitioned in 80% training and 20% testing subsets. Furthermore, experimental run included 10 instances for every algorithm tested and the accuracy and error reported are the averages, in order to obtain further reliable performance estimation.

6.3. Comparative experiments

For comparison purposes, a wide range of different ML techniques was tested; these include the RBF networks trained with the SFM algorithm as described earlier, Multi-layer Perceptron (MLP) neural networks (Hagan & Menhaj, 1994), Support Vector Machines for regression (SVR) (Cervantes et al., 2020; Vapnik, 2013), Extreme Learning

Table 1
Operational parameters for the PSO-NSFM -based methods.

Parameter	Symbol	Value
Maximum number of iterations	ξ_1	1500
Minimum normalized swarm radius	ξ_2	0.1
Population	P	20
Nostalgia	c_1	0.5
Envy	c_2	0.5
Velocity clamping constant	$V_{\max}$	30
Minimum inertia weight	$\omega_{\min}$	0.7
Maximum inertia weight	$\omega_{\max}$	1.0
Minimum number of fuzzy sets	$s_{\min}$	4
Maximum number of fuzzy sets	$s_{\max}$	50

Machine (ELM) (Huang et al., 2006; Klesk & Korzen, 2021), Classification and Regression trees (CART) (Breiman et al., 2017; Theodoridis & Koutroumbas, 2008) and the regression ensemble learning with least squares boosting (LSBoost) (Ali, 2020; Hastie et al., 2008). In all cases, the model parameter selection and model evaluation were performed based on the kfCV process as previously described.

Regarding the model parameterization, several aspects were taken into account and optimized accordingly. The SFM-based models were optimized with respect to the number of fuzzy sets. Also, a two-hidden layer MLP was trained with the Levenberg-Marquardt algorithm (Hagan & Menhaj, 1994), with its parameters optimized by using an exhaustive search for all possible combinations with the number of nodes ranging from 15 to 50 and 10 to 45, for the 1st and the 2nd hidden layer, respectively, with step equal to 5 and provided that the 2nd layer dimension is less than the 1st layer dimension, i.e., 36 combinations were produced. In addition, the SVR models were employed with Gaussian kernel function and exhaustive search procedure based on grid search adopted for optimizing the width of the Gaussian kernels σ and the penalty factor C. The ELM models were optimized based on the number of hidden neurons, with exhaustive search applied to the number of nodes ranging from 10 to 2000 with step 10, and implemented by applying different activation functions including hard-limit function (har), radial basis function (rad), sigmoid function (sig), sine function (sin) and triangular basis function (tri). CART were optimized with respect to the depth of the tree and the minimum leaf size for prunning threshold. Finally, the LSBoost ensemble models were optimized with respect to the number of trees in the ensemble and the minimum leaf size.

In order to further evaluate the proposed method specifically in terms of the PSO performance, we implemented three more variants: (a) the original PSO-NSFM algorithm (Alexandridis et al., 2013), (b) the modified PSO-NSFM without using the adaptive inertia weight and 'crazy' particles strategies (see Section 5.2.4), and (c) the modified PSO-NSFM without using the 'crazy' particles strategy (see Section 5.2.4). The operational parameters used by the PSO-NSFM -based approaches, which were applied to all experiments, are given in Table 1.

7. Results

This section presents the results produced by the proposed approach and its rivals in terms of training times, prediction accuracy and training stability.

7.1. Training times

The processing time estimations for all the implemented methodologies were made in the context of the experimental setup described in the previous sections, including kfCV and averaging over multiple iterations. All computational times were measured on the same PC with Intel Core i9 processor (3.60 GHz) with 16 CPU cores and 64 GB of RAM, running Ubuntu Linux 18.04 server edition.

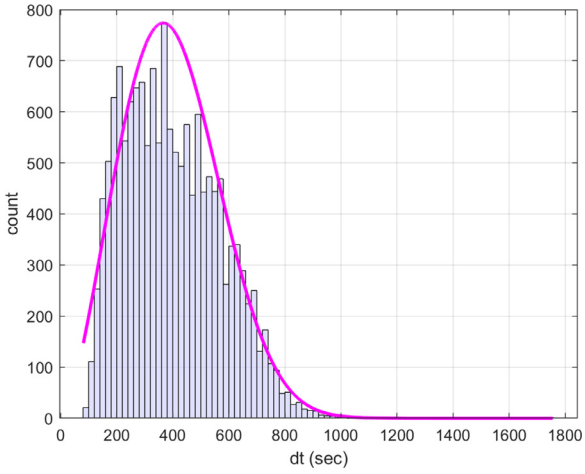

Fig. 4. Distribution of the PT-ETA regression target in the GTFS dataset, i.e., true dT_{t_s,t_s+6}. The implied pdf is clearly GEV-like with thin long tail due to the inclusion of extreme cases.

Overall, the methods that are capable of predicting more accurately and with less computational times are the modified PSO-NSFM, the SFM and the LSBoost algorithms. Although the MLP model ranked in the 3rd position regarding the prediction accuracy, its training time for each one of the $k = 5$ folds in the kfCV process took more than 10 min, i.e., roughly an hour for the complete dataset. The increased training time requirement makes the MLP less practical for the PT-ETA problem in real-world applications, where the predictive model would normally have to be re-trained often in a changing environment. As far as the LSBoost is concerned, it can be trained within approximately 10 s in this PT-ETA setup when using predefined optimal parameters, while in comparison the training times for RBF-based methods require less than 17 s. Regarding the proposed modified PSO-NSFM approach, it is important to note that the resulted model requires less training time than the model produced by the SFM, due to the former's less complicated network topology. Specifically, training RBF networks with the NSFM algorithm by using the optimal parameters estimated by the PSO requires about 14 s, while the SFM algorithm using the optimal parameters estimated by exhaustive search requires about 17 s.

7.2. Prediction accuracy

Figure 4 presents the distribution of the target values of PT-ETA at the 6th PT stop, i.e., dT_{t_s,t_s+6}, based on the available training dataset. As explained earlier, this is the maximum look-ahead window for the CR-GTFS processed trips with an average of 20-25 stops in total for more than 3/4 of the available trips, i.e., given the fact that roughly no more than 1/4 of the trip length (measured in number of stops) should be used as input buffer for the predictions. The error distribution of the predicted ETA values conforms to a Generalized Extreme Value (GEV) profile (Spiegel, Schiller & Srinivasan, 2009), i.e., a highly skewed Gaussian-like or Weibull-like profile (Spiegel et al., 2009) for the underlying probability distribution function (pdf) implied by the corresponding histogram, as depicted in Fig. 4. There are a few large time differences in specific trips, i.e., having excessive distances between few pairs of (sparse) stops in their PT routes, producing a thin long positive tail in the distribution. In order to test the robustness of the models and their training, it was decided not to remove any such extreme values but instead use it as-is, simulating a real-world requirement of having to produce robust PT-ETA predictions for any given input vector, including such extremes and non-typical PT routes.

The results obtained from the testing process, employing the kfCV method described in Section 6.2, were averaged over 10 repetitions for all the implemented methods and the exact values are summarized in

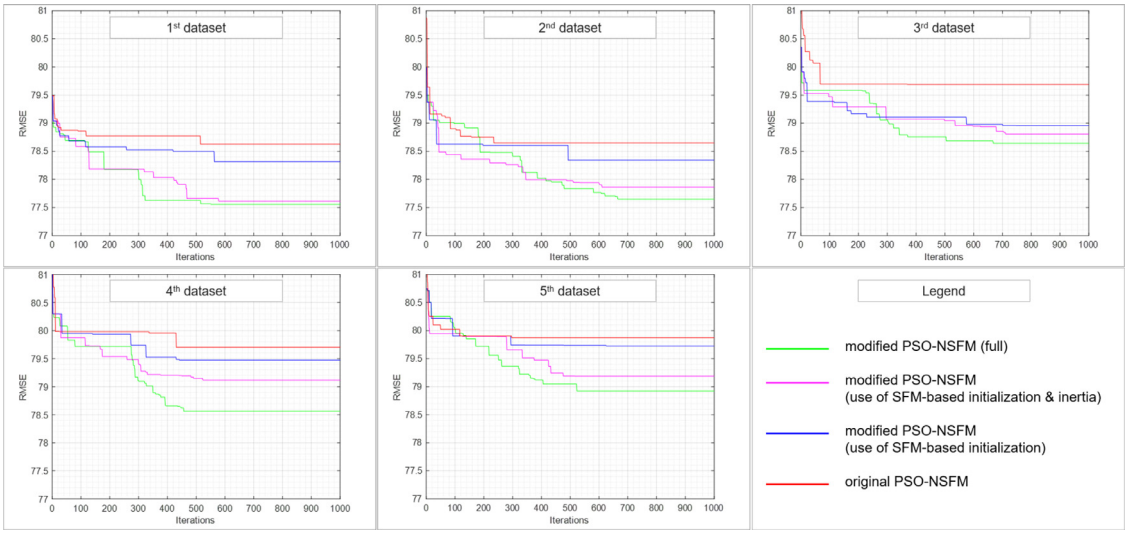

Fig. 5. Evolution of the RMSE in the five test sets of the kfCV procedure for the best solution produced by the PSO-NSFM -based methods.

Table 2
Results for all the implemented methods.

Method	RMSE	$\mathbf{R^2}$	MAE
ELM (SIG)	88.596±2.084	0.733±0.009	65.584±1.333
CART	86.943±0.669	0.744±0.004	63.497±0.462
SVR	83.969±1.669	0.763±0.009	60.799±0.618
LSBoost	81.773±0.163	0.774±0.002	59.872±0.172
MLP	81.148±1.378	0.777±0.008	59.451±0.454
RBF-SFM	80.539±0.125	0.781±0.001	59.372±0.127
modified PSO-NSFM	**79.067**±0.147	**0.786**±0.001	**58.978**±0.141

Table 2 in terms of Root Mean Squared Error (RMSE), R-Squared (R^2) and Mean Absolute Error (MAE). Particularly, Table 2 depicts the mean and standard deviation values from the 10 test runs; bold indicates the overall-best performance.

7.3. Training stability

Figure 5 presents the evolution of the RMSE in the five folds of the kfCV procedure for the best solution produced by each PSO-NSFM -based method; the best solution corresponds to the best kfCV performance found among the 10 runs for each method, as an example of the overall training process.

Regarding the PSO-NSFM hyper-parameter optimization, the maximum number of iterations and the population size are parameters that significantly affect the computational load of the algorithm. As expected, experiments showed that increasing these parameters may result in improving prediction accuracy, albeit at the cost of additional computational burden. For the purposes of this work, we kept these parameters at small values (see Table 1), as our experiments showed that satisfactory results can be obtained in relatively fast computational times.

8. Discussion

8.1. Contributions to literature

There are three main aspects worth noting in the comparative results of this study, namely (a) the training times, (b) the prediction accuracy and (c) the overall stability and robustness of the proposed framework.

First, regarding the training time, it should be noted that the exact numbers and statistical margins for the processing time estimations for the implemented methodologies (presented in Section 7.1) are extremely difficult to attain. This is an expected and common issue, since each individual training instance is affected by various system-level factors that can not be disabled entirely, e.g. CPU scheduling by the OS (task switching), volatile background tasks of lower priority, associated memory swapping to disk, etc. Nevertheless, the numbers presented in Section 7.1 are very useful in terms of general comparisons and valid ranking of alternative approaches, providing a valid baseline for the real-world PT-ETA task. Overall, the proposed approach for both the offline and the online GTFS data feeds was proven to be a very efficient and balanced trade-off of speed-versus-accuracy. According to Table 2, the modified PSO-NSFM method achieves marginal overall-best accuracy compared to the other state-of-the-art alternatives, but (equally important) does so with the minimum possible complexity and acceptable training time. Inherently to all PSO-based approaches, the exact accuracy can be improved further depending on more complex and time-consuming training, if allowed.

Second, regarding the prediction accuracy and the numbers presented in Table 2, it can be verified that the proposed modified PSO-NSFM approach produces the overall-best ETA prediction accuracy in terms of average RMSE. This is probably the most prominent improvement over the current state-of-the-art in related literature, although not always the most important one. When comparing against the RBF networks trained with the SFM algorithm that is the most competitive of the alternatives, the proposed method still provides a slight advantage and similar stability in R^2. Additionally, it should be noted that the modified PSO-NSFM favors networks with smaller topology while exhibiting low standard deviations; this indicates better generalization capability and robustness for model training using the proposed algorithm. The modified PSO-NSFM and SFM methods achieve an advantage over the LSBoost models, but with somewhat higher training times, as described in Section 7.1. Furthermore, the modified PSO-NSFM method achieves an advantage over the MLP networks, which are clearly ranked in 3rd position in terms of average error, despite using exhaustive search in MLP network topology optimization. Regarding the comparison with LSBoost models, the MLP prediction accuracy is slightly better; however, LSBoost requires significantly smaller training times than MLP. Besides the RBF-based, MLP and LSBoost methods, the performances of the rest of the models differ more evidently. The ELM models produce the worst prediction results, although still comparable. As expected, the CART model outperforms all the implemented alternatives in terms of computational times, but ranks in the second-worst position regarding the prediction accuracy. Finally, the SVR technique is significantly less capable on pre-

diction accuracy compared to LSBoost. Overall, the preference over both accuracy and processing time requirements for the PT-ETA task in realistic experimental setups favours the modified PSO-NSFM over any other alternative method.

Third, regarding the overall stability and robustness of the proposed framework, Fig. 5 illustrates that in all cases the modified PSO-NSFM outperforms the other variants in a consistent and reliable way, outlining the merits of extending the original algorithm using SFM-based initialization, as well as the adaptive inertia weight and the 'crazy' particles strategies. The modified PSO-NSFM variant including only the modifications of SFM-based initialization and adaptive inertia weight (see Section 5) ranks second after the proposed (full) modified PSO-NSFM approach. The remaining two PSO-NSFM variants produce less accurate results, whereas, as it was expected, the PSO-NSFM variant with only SFM-based initialization clearly gives better results within the first few iterations, in contrast to the original PSO-NSFM algorithm. Due to the fact that the SFM outperforms all the non-RBF based implemented methods, the PSO-NSFM -based variants that use the SFM-based initialization procedure manage to outperform all the other implemented methods right from the first iteration. Therefore, the proposed modified PSO-NSFM approach generates a highly accurate model within the first few training steps and additional improvements are being made until convergence. In addition, the robustness of the proposed approach is further enhanced by the highly adaptive and error-resilient design of the CR-GTFS pipeline for pre-processing, as described in detail in Section 5.3.

8.2. Practical implications

Given the three main aspects discussed in the previous Section, this study contributes to the current state-of-the-art in urban ETA prediction tasks in the corresponding ways. As noted in Section 1, the main novelties and improvements include: (a) the CR-GTFS pipeline for efficient and robust pre-processing of the data feeds, (b) the modified PSO-NSFM algorithm optimally adapted for the PT-ETA task providing overall-best accuracy, (c) comparative experiments and results against other state-of-the-art alternatives regarding both speed and accuracy.

One additional implication is that designing a highly adaptive pipeline for robust and error-resilient pre-processing, although it may seem a drawback in terms of added complexity and processing time, in reality it improves the quality of the training data and, hence, the required complexity for the trained models (see Section 5.3). This means that, even if the proposed PSO-NSFM method is not used, the CR-GTFS pipeline as presented in this work is a standalone significant merit and a proven quality-enhancement stage of data pre-processing in real-world PT-ETA applications.

In addition, using methods of scalable complexity like PSO, as well as proper 'hot start' initialization as with SFM, enables the fine-tuning of the optimal balance between training speed and model accuracy. In other words, coarse setups with fast training can be used for massively parallel implementations with online constraints, while fine-grained setups with longer training can be used for offline applications that favour accuracy over speed. These advantages are of utmost importance in challenging problems like the PT-ETA task, especially in the era of big data for 'smart' cities.

Finally, it should be noted that given that the scale, volume and velocity of the transit data generated in modern urban environments is increasing in super-linear rates, it is extremely important that the proposed solutions to everyday practical problems like the PT-ETA task follow this trend and evolve in being more and more robust, scalable and modular by design.

9. Conclusion

The intelligent transportation systems are said to revolutionize the travel experience by making it safe, secure and comfortable for the people. Nowadays, PT mobility and vehicle flow information has become possible largely due to recent advances in the transit data format standard, namely GTFS, which has been adopted by thousands of PT agencies around the world to publish their scheduled and real-time public transit data, i.e., transportation route and timetable information. PT movement information used by data-driven methods to address major PT-related problems can improve the quality and the reliability of the PT services and 'smart' cities management. One of the most urgent yet challenging tasks is the accurate PT-ETA prediction task for the service's vehicles.

In this work, we investigated the problem of accurate prediction of ETA for PT vehicles after a fixed number of stops, at the maximum look-ahead horizon for the available data, by: (a) presenting an improved version of the CR-GTFS pipeline as a unified pre-processing process and (b) introducing a novel modified version of the successful PSO-NSFM data-driven method for the core prediction task. Despite the increasing use of GTFS data format by PT agencies, these data are often noisy and with inconsistencies. The combination of the PSO-NSFM algorithm with the CR-GTFS pipeline produces a novel and complete framework for both accurate and fast PT-ETA predictions. Experiments on real-world GTFS data showed that the proposed approach outperforms several competitive state-of-the-art alternatives in terms of prediction accuracy and in comparable or better computational times.

Future enhancements of this work include the exploitation of localized weather information as possible additional input in the PT-ETA prediction task; although this may seem intuitively correct and straightforward, proper factor analysis is required for assessing the actual relevance and statistical correlation, especially against other, potentially more important factors, such as traffic jams unrelated to weather conditions, localized weekday patterns, stops with special importance (e.g. interlink hubs), etc. Other enhancements will focus on GTFS fields that are still experimental and may be formally adopted in the future by the PT agencies, such as the 'occupancy status' (degree of passenger occupancy for the vehicle), the 'congestion level' (congestion level that the vehicle is currently experiencing), etc. Finally, we plan to validate the proposed method on GTFS data provided (when available) from PT agencies from different countries, most importantly from Europe, since it may reveal quite different inherent properties in terms of urban environment, PT time schedules, trip distances, etc.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Eva Chondrodima: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Harris Georgiou:** Conceptualization, Data curation, Investigation, Methodology, Writing – original draft, Writing – review & editing. **Nikos Pelekis:** Formal analysis, Supervision, Writing – review & editing. **Yannis Theodoridis:** Project administration, Supervision, Writing – review & editing.

Acknowledgement

This paper is one of the deliverables of the project with MIS 5050503 of the Call entitled 'Support for researchers with emphasis on young researchers - cycle B' (Code: EDBM103) which is part of the Operational Program 'Human Resources Development, Education and Lifelong Learning' which is Co-financed by Greece and the European Union (European Social Fund).

Appendix A. Problem formulation analysis

For each of the previous $k_s + 1$ vehicle stops (including the current one at t_s), the sequential information gathered in each element u' is 'stop lon' (longitude), 'stop lat' (latitude) and 'stop sequence' (increasing number) for the route stops in each trip, the actual distance $dS_{t_s-b,\ t_s-b+1}$ travelled between the adjacent stops b and $b+1$, the actual time $dT_{t_s-b,\ t_s-b+1}$ for this transition and the GPS-based estimated (mean) speed, i.e., $dS_{t_s-b,\ t_s-b+1}/dT_{t_s-b,\ t_s-b+1}$. Hence, each element u' is essentially a sub-vector by itself, including the semantic information about vehicle stops and the transitions between them, as provided in the GTFS data.

It can be noted that each sub-vector u' includes some redundant information, i.e., mean speed $dS = dT$ along with transition distance dS and time dT. The reasoning behind this is that with similar techniques in feature generation some regression models become much more simplified and easier to train, as the pair distance-and-time often introduces non-linearities in the input compared to distance-and-speed. Furthermore, speed may also be included in the next-step sub-vector u'_{t_s}, where it is typically not available ($dT_{t_s,\ t_s+1}$ not realized yet), if instead some globally available estimation of it can be attained from historic data, i.e., the mean time it usually takes to travel between this specific pair of stops and in that specific direction. For more accurate comparison of the examined models in this study, no such additional estimations were made for next-speed elements and, hence, the future-step sub-vector $\tilde{u'}_{t_s,\ t_s+q}$ contains 'stop lon', 'stop lat', 'stop sequence' for vehicle stops t_s and $t_s + q$, and actual distance $dS_{t_s,\ t_s+q}$ that are available at any given vehicle stops.

As mentioned above, a relevant previous work (Chondrodima et al., 2021) using different algorithms employed only next-stop prediction, i.e., with $q = 1$. The purpose was to compare those methods in the pure short-term sense, i.e., limit the effects of degraded data quality and stationarity shifts caused by exogenous localized factors like unstable road traffic. In contrast, in this work the look-ahead window is maximized to $q = 6$, based on the average length of the vehicle routes (about 25 stops) and given the fact that no more than 1/4 or so of the entire trip should be used as input buffer, i.e., not producing any predictions for the beginning of the trip. Furthermore, training the predictors directly for the target $q = 6$, instead of using a feedback loop with six single-step predictions, is the actual experimental test and a realistic baseline assessment of how each algorithm performs when stretching the purely short-term single-step setup to predicting further into the future. In other words, direct prediction for the next qth stop is at least as good as applying the short-term single-next-stop prediction q times sequentially, given the fact that in the second case the prediction errors are propagated and accumulated with each prediction iteration over q steps.

Appendix B. GTFS-s feed files

The GTFS data exist in two main variants: the GTFS-s feed and the GTFS-rt feed. The former is usually a collection of comma-separated values (CSV) files with relatively static information, with updates occurring periodically, such as time schedules that might be updated only a couple of times in a year. The latter refers to transit data collected during the vehicle trips and are usually based on GPS tracking. Some agencies provide additional information by adding extensions to the GTFS standard.

There are five files that are mandatory for sufficiently describing a PT network through the GTFS-s feed: (i) *'agency.txt'* that includes information for the transit agencies providing the data in the feed, each identified via an agency id; (ii) *'routes.txt'* that includes the routes in the PT network, each identified via a route id; (iii) *'trips.txt'* that includes the trips in the PT network, each identified via a trip id which is associated with a route id - a trip describes the movement of a PT vehicle through a pre-defined sequence of stopping points; (iv) *'stops.txt'* that includes the stops in the PT network, each identified via a stop id and described by the respective coordinates based on the WGS 84 datum; and (v) *'stop_times.txt'* that includes the scheduling information by providing the time that each trip is at a stop, identified via the associated trip id and stop id. There are also conditionally required files: (vi) *'calendar.txt'* that includes the service days, each identified via a service id; and (vii) *'calendar_dates.txt'* that includes the exceptions to the default service patterns associated with a service id. Regarding common optional files that may also be available, two of these are: (viii) *'feed_info.txt'* that includes information about the dataset itself; and (ix) *'shapes.txt'* that includes the path that the vehicle travels within trips, identified via a shape id which can be associated with a trip id and described by the respective coordinates based on the WGS84 datum.

The combination of these GTFS files provides the complete static information for the PT network. Specifically, in a PT network stops represent specific locations at which the riders are allowed to board or disembark from the PT vehicles. A fixed sequence of stops defines a route, i.e., it represents a predetermined path within the transit network. Hence, each individual stop can serve multiple routes, as a node in a graph. Additionally, a trip describes the movement of a PT vehicle executing one complete pass of a route. Obviously, a trip occurs at a specific time frame in a day along a route, while a route itself is time-independent and is associated to many trips.

Moreover, PT agencies publish their real-time information through the GTFS-rt feed. This information refers to updates from the transit system including current locations of PT vehicles, route status and system alerts. The GTFS-rt feed supports three different types of information: (a) trip updates, which include predicted arrival and/or departure events; (b) vehicle positions, which include updates regarding the location of individual transit vehicles; and (c) alerts, which include updates regarding the disruptions in the transit network in the form of human-readable messages. Moreover, the GTFS-rt data exchange format is based on the protocol buffer (pb) format, which is an open-source standard for efficiently serializing structured data.

Appendix C. Metro transit

Metro Transit, the largest transportation provider in Minnesota, USA, uses the GTFS format to publish its transit data for bus, light rail and commuter rail service for the Minneapolis / St. Paul metropolitan area. GTFS-s and real-time data can be retrieved from Metro Transit's webpage (MetroTransit, 2021). The scheduled data are provided by the GTFS-s feed[1], which is generally updated every week, but Metro Transit recommends to perform checks on a daily basis. Metro Transit publishes 11 files through the GTFS-s feed, which are the following: agency, routes, trips, stops, stop times, calendar, shapes, calendar dates, feed info, linked datasets, vehicles.

Additionally, Metro Transit provides real-time data by using three GTFS-rt feeds of version 2.0, which are refreshed every 15 s: the TripUpdate feed[2], the VehiclePosition feed[3] and the ServiceAlerts feed[4]. In Metro Transit's webpage there is extensive documentation that describes the available information and data schemas provided by each real-time feed. More specifically, the Metro Transit TripUpdate feed includes information about vehicle's timestamp, trip id, route id, direction id, start time, start date, vehicle id, vehicle label, stop sequence, stop id, arrival time, departure time. Note that the arrival/departure times at stops are the ones provisioned in the time schedule; the actual arrival/departure times are not included in this feed. Also, the Metro Transit VehiclePosition feed includes information about vehicle's timestamp, trip id, route id, direction id, start time, start date, vehicle id, vehicle label, position latitude, position longitude, bearing, odometer, speed. Finally, the the Metro Transit ServiceAlerts feed provides updates whenever there is disruption on the network, such as delays and cancellations of individual

[1] https://svc.metrotransit.org/mtgtfs/gtfs.zip
[2] https://svc.metrotransit.org/mtgtfs/tripupdates.pb
[3] https://svc.metrotransit.org/mtgtfs/vehiclepositions.pb
[4] https://svc.metrotransit.org/mtgtfs/alerts.pb

trips. Particularly, the ServiceAlerts feed specifies exactly, which entities of the network an alert affects; these are (i) the whole network, (ii) the whole route, (iii) a particular trip, (vi) a particular stop.

Appendix D. Radial basis function (RBF) network architecture

RBF networks (Moody & Darken, 1989) are a special variant of three-layer feed-forward neural networks. An RBF network contains an input layer of the same dimensionality N of the input space, a hidden layer with L neurons and an output layer. The input layer is a pass-through layer that translates the input sample $\mathbf{u}$ to the hidden layer, which in turn applies a non-linear transformation to the input variables. More specifically, each hidden neuron l is characterized by a center vector $\hat{\mathbf{u}}_l^T = [\hat{u}_{1,l}, \hat{u}_{2,l}, \dots, \hat{u}_{N,l}]$ and for each kth input vector the lth neuron triggers the activation $\mu_l(\mathbf{u}(k)) = ||\mathbf{u}(k) - \hat{\mathbf{u}}_l||$. As a result, the overall hidden layer response can be defined as:

$$\mathbf{z}(k) = [g(\mu_1(\mathbf{u}(k))), \dots, g(\mu_L(\mathbf{u}(k)))], k = 1, 2, \dots, K \tag{D.1}$$

where K is the total number of data and g is the activation function, which in this work is expressed in thin plate spline function: $g(\mu) = \mu^2 \log(\mu)$.

A linear combination of the hidden nodes creates a response through the vector of the connection weights $\mathbf{w}^T = [w_1, w_2, \dots, w_L]$ and produces the final output of the RBF network $\mathbf{y}$, which for the for the kth data point is:

$$\hat{\mathbf{y}}_l = \mathbf{z}(k) \cdot \mathbf{w} = \sum_{l=1}^{L} w_l g(\mu_l(\mathbf{u}(k))) \tag{D.2}$$

References

Abbas, G., Gu, J., Farooq, U., Asad, M. U., & El-Hawary, M. (2017). Solution of an economic dispatch problem through particle swarm optimization: A detailed survey - Part I. *IEEE Access, 5*, 15105–15141. 10.1109/ACCESS.2017.2723862.

Akpinar, M. T. (2019). Smart city applications in digital age: State-of-art review and critique. *Journal of Information Systems and Management Research, 1*(1), 37–42.

Alam, O., Kush, A., Emami, A., & Pouladzadeh, P. (2020). Predicting irregularities in arrival times for transit buses with recurrent neural networks using GPS coordinates and weather data. *Journal of Ambient Intelligence and Humanized Computing*, 1–14.. 10.1007/s12652-020-02507-9.

Alexandridis, A., & Chondrodima, E. (2014). A medical diagnostic tool based on radial basis function classifiers and evolutionary simulated annealing. *Journal of Biomedical Informatics, 49*, 61–72. 10.1016/j.jbi.2014.03.008.

Alexandridis, A., Chondrodima, E., Efthimiou, E., Papadakis, G., Vallianatos, F., & Triantis, D. (2014). Large earthquake occurrence estimation based on radial basis function neural networks. *IEEE Transactions on Geoscience and Remote Sensing, 52*(9), 5443–5453. 10.1109/TGRS.2013.2288979.

Alexandridis, A., Chondrodima, E., Paivana, G., Stogiannos, M., Zois, E., & Sarimveis, H. (2014). Music genre classification using radial basis function networks and particle swarm optimization. In *6th Computer science and electronic engineering conference (CEEC)* (pp. 35–40). 10.1109/CEEC.2014.6958551.

Alexandridis, A., Chondrodima, E., & Sarimveis, H. (2013). Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization. *IEEE Transactions on Neural Networks and Learning Systems, 24*(2), 219–230. 10.1109/TNNLS.2012.2227794.

Alexandridis, A., Chondrodima, E., & Sarimveis, H. (2016). Cooperative learning for radial basis function networks using particle swarm optimization. *Applied Soft Computing, 49*, 485–497. 10.1016/j.asoc.2016.08.032.

Alexandridis, A., Paizis, E., Chondrodima, E., & Stogiannos, M. (2017). A particle swarm optimization approach in printed circuit board thermal design. *Integrated Computer Aided Engineering, 24*(2), 143–155. 10.3233/ICA-160536.

Alexandridis, A., Sarimveis, H., & Ninos, K. (2011). A radial basis function network training algorithm using a non-symmetric partition of the input space - application to a model predictive control configuration. *Advances in Engineering Software, 42*(10), 830–837. 10.1016/j.advengsoft.2011.05.026.

Alexandridis, A., Stogiannos, M., Papaioannou, N., Zois, E., & Sarimveis, H. (2018). An inverse neural controller based on the applicability domain of RBF network models. *Sensors, 18*(1), 315. 10.3390/s18010315.

Ali, A. (2020). Ensemble learning model for prediction of natural gas spot price based on least squares boosting algorithm. In *International conference on data analytics for business and industry: Way towards a sustainable economy (ICDABI)* (pp. 1–6). 10.1109/ICDABI51230.2020.9325615.

Amita, J., Singh, J. S., & Kumar, G. P. (2015). Prediction of bus travel time using artificial neural network. *International Journal of Transportation Engineering, 5*(4), 410–424. 10.1016/j.trpro.2016.11.091.

Ang, K. L.-M., Seng, J. K. P., Ngharamike, E., & Ijemaru, G. K. (2022). Emerging technologies for smart cities' transportation: Geo-information, data analytics and machine learning approaches. *ISPRS International Journal of Geo-Information, 11*(2), 85. 10.3390/ijgi11020085.

Arjun, R., Kuanr, A., & Suprabha, K. R. (2021). Developing banking intelligence in emerging markets: Systematic review and agenda. *International Journal of Information Management Data Insights, 1*(2), 100026. 10.1016/j.jjimei.2021.100026.

Bangyal, W. H., Ahmad, J., & Rauf, H. T. (2019). Optimization of neural network using improved bat algorithm for data classification. *Journal of Medical Imaging and Health Informatics, 9*(4), 670–681. 10.1166/jmihi.2019.2654.

Bangyal, W. H., Ahmad, J., Rauf, H. T., & Pervaiz, S. (2018). An overview of mutation strategies in bat algorithm. *International Journal of Advanced Computer Science and Applications, 9*(8), 523–534. 10.14569/IJACSA.2018.090866.

Bangyal, W. H., Ahmad, J., Rauf, H. T., & Shakir, R. (2018). Evolving artificial neural networks using opposition based particle swarm optimization neural network for data classification. In *International conference on innovation and intelligence for informatics, computing, and technologies* (pp. 1–6). 10.1109/3ICT.2018.8855772.

Bangyal, W. H., Ahmad, J., Shafi, I., & Abbas, Q. (2012). A forward only counter propagation network-based approach for contraceptive method choice classification task. *Journal of Experimental and Theoretical Artificial Intelligence, 24*(2), 211–218. 10.1080/0952813X.2011.639091.

Bangyal, W. H., Ahmed, J., & Rauf, H. T. (2020). A modified bat algorithm with torus walk for solving global optimisation problems. *International Journal of Bio-Inspired Computation, 15*(1), 1–13. 10.1504/IJBIC.2020.105861.

Bangyal, W. H., Hameed, A., Alosaimi, W., & Alyami, H. (2021). A new initialization approach in particle swarm optimization for global optimization problems. *Computational Intelligence and Neuroscience, 2021*. 10.1155/2021/6628889.

Bangyal, W. H., Nisar, K., Ibrahim, A., Bin, A. A., Haque, M. R., Rodrigues, J. J., & Rawat, D. B. (2021). Comparative analysis of low discrepancy sequence-based initialization approaches using population-based algorithms for solving the global optimization problems. *Applied Sciences, 11*(16), 7591. 10.3390/app11167591.

Bangyal, W. H., Qasim, R., Ahmad, Z., Dar, H., Rukhsar, L., Aman, Z., Ahmad, J., et al., (2021). Detection of fake news text classification on COVID-19 using deep learning approaches. *Computational and Mathematical Methods in Medicine, 2021*. 10.1155/2021/5514220.

Batra, J., Jain, R., Tikkiwal, V. A., & Chakraborty, A. (2021). A comprehensive study of spam detection in e-mails using bio-inspired optimization techniques. *International Journal of Information Management Data Insights, 1*(1), 100006. 10.1016/j.jjimei.2020.100006.

Beni, G. (2020). *Complex social and behavioral systems: game theory and agent-based models* (pp. 791–818). New York, NY: Springer US. Chapter Swarm Intelligence

Breetzke, T., & Flowerday, S. V. (2016). The usability of IVRs for smart city crowdsourcing in developing cities. *Electronic Journal of Information Systems in Developing Countries, 73*(1), 1–14. 10.1002/j.1681-4835.2016.tb00527.x.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (2017). *Classification and regression trees*. Routledge.

Brisaboa, N. R., Faria, A., Galaktionov, D., Rodeiro, T. V., & Rodriguez, M. A. (2022). Improved structures to solve aggregated queries for trips over public transportation networks. *Information Sciences, 584*, 752–783. 10.1016/j.ins.2021.10.079.

Carter, L., Yoon, V., & Liu, D. (2022). Analyzing e-government design science artifacts: A systematic literature review. *International Journal of Information Management, 62*, 102430. 10.1016/j.ijinfomgt.2021.102430.

Cervantes, J., Garcia-Lamont, F., Rodriguez-Mazahua, L., & Lopez, A. (2020). A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing, 408*, 189–215. 10.1016/j.neucom.2019.10.118.

Chakraborty, A., & Kar, A. K. (2016). A review of bio-inspired computing methods and potential applications. In *Proceedings of the international conference on signal, networks, computing, and systems* (pp. 155–161). Springer. 10.1007/978-81-322-3589-7_16.

Chakraborty, A., & Kar, A. K. (2017). *Nature-inspired computing and optimization: Theory and applications* (pp. 475–494). Cham: Springer International Publishing. Chapter Swarm Intelligence: A Review of Algorithms

Chakraborty, A., & Kar, A. K. (2021). How did COVID-19 impact working professionals–a typology of impacts focused on education sector. *International Journal of Information and Learning Technology*. 10.1108/IJILT-06-2020-0125.

Chatterjee, S., & Kar, A. K. (2015). Smart cities in developing economies: A literature review and policy insights. In *2015 International conference on advances in computing, communications and informatics (ICACCI)* (pp. 2335–2340). IEEE. 10.1109/ICACCI.2015.7275967.

Chatterjee, S., Kar, A. K., & Gupta, M. (2018). Success of IoT in smart cities of India: An empirical analysis. *Government Information Quarterly, 35*(3), 349–361. 10.1016/j.giq.2018.05.002.

Chen, J., Guo, Y., Su, C., Chen, J., & Chang, S. (2015). A smart city system architecture based on city-level data exchange platform. *Journal of Information Technology Research, 8*(4), 1–25. 10.4018/JITR.2015100101.

Chen, Y., Lu, Y., Bulysheva, L., & Kataev, M. Y. (2022). Applications of blockchain in industry 4.0: A review. *Information Systems Frontiers*, 1–15. 10.1016/j.bcra.2021.100027.

Chen, Y., & Silva, E. A. (2021). Smart transport: A comparative analysis using the most used indicators in the literature juxtaposed with interventions in english metropolitan areas. *Transportation Research Interdisciplinary Perspectives, 10*, 100371. 10.1016/j.trip.2021.100371.

Chen, Y., Zou, X., Li, K., Li, K., Yang, X., & Chen, C. (2021). Multiple local 3D CNNs for region-based prediction in smart cities. *Information Sciences, 542*, 476–491. 10.1016/j.ins.2020.06.026.

Cheng, T., Cui, D., & Cheng, P. (2003). Data mining for air traffic flow forecasting: A hybrid model of neural network and statistical analysis. In *Proceedings of the 2003 IEEE international conference on intelligent transportation systems: Vol. 1* (pp. 211–215). 10.1109/ITSC.2003.1251950.

Chondrodima, E., Georgiou, H., Pelekis, N., & Theodoridis, Y. (2021). Public transport arrival time prediction based on GTFS data. In *7th International conference on machine learning, optimization and data science (LOD 2021)* (pp. 481–

495). Grasmere, Lake District, England, UK: Springer International Publishing. 10.1007/978-3-030-95470-3_36.
Corbett, J., & Mellouli, S. (2017). Winning the SDG battle in cities: How an integrated information ecosystem can contribute to the achievement of the 2030 sustainable development goals. *Information System Journal, 27*(4), 427–461. 10.1111/isj.12138.
Costa, D. G., & Peixoto, J. P. J. (2020). COVID-19 pandemic: A review of smart cities initiatives to face new outbreaks. *IET Smart Cities, 2*(2), 64–73. 10.1049/iet-smc.2020.0044.
Dwivedi, Y. K., Hughes, L., Kar, A. K., Baabdullah, A. M., Grover, P., Abbas, R., … Bunker, D., et al., (2022). Climate change and COP26: Are digital technologies and information management part of the problem or the solution? An editorial reflection and call to action. *International Journal of Information Management, 63*, 102456. 10.1016/j.ijinfomgt.2021.102456.
Engelbrecht, A. P. (2007). *Computational intelligence: An introduction*. John Wiley & Sons.
Georgiou, H., Pelekis, N., Sideridis, S., Scarlatti, D., & Theodoridis, Y. (2020). Semantic-aware aircraft trajectory prediction using flight plans. *International Journal of Data Science and Analytics, 9*(2), 215–228. 10.1007/s41060-019-00182-4.
Georgiou, H., Petrou, P., Tampakis, P., Sideridis, S., Chondrodima, E., Pelekis, N., & Theodoridis, Y. (2020). Future location and trajectory prediction. In *Big data analytics for time-critical mobility forecasting: From raw data to trajectory-oriented mobility analytics in the aviation and maritime domains* (pp. 215–254). Cham: Springer.
Ghanim, M., Shaaban, K., & Miqdad, M. (2020). An artificial intelligence approach to estimate travel time along public transportation bus lines. In *International Conference on Civil Infrastructure and Construction* (pp. 588–595). 10.29117/cic.2020.0074.
Google, 2021. Testing GTFS feeds. https://developers.google.com/transit/gtfs/guides/tools.
Gu, J., Jiang, Z., David Fan, W., & Chen, J. (2022). Short-term trajectory prediction for individual metro passengers integrating diverse mobility patterns with adaptive location-awareness. *Information Sciences, 599*, 25–43. 10.1016/j.ins.2022.03.074.
Gupta, S., Tuunanen, T., Kar, A. K., & Modgil, S. (2022). Managing digital knowledge for ensuring business efficiency and continuity. *Journal of Knowledge Management*. 10.1108/JKM-09-2021-0703.
Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks, 5*(6), 989–993. 10.1109/72.329697.
Haider Bangyal, W., Hameed, A., Ahmad, J., Nisar, K., Haque, M. R., Ibrahim, A., … Rawat, D. B., et al., (2022). New modified controlled bat algorithm for numerical optimization problem. *Computers, Materials & Continua, 70*(2), 2241–2259. 10.32604/cmc.2022.017789.
Hassan, S.-U., Shabbir, M., Iqbal, S., Said, A., Kamiran, F., Nawaz, R., & Saif, U. (2021). Leveraging deep learning and SNA approaches for smart city policing in the developing world. *International Journal of Information Management, 56*, 102045. 10.1016/j.ijinfomgt.2019.102045.
Hastie, T., Tibshirani, R., & Friedman, J. (2008). The elements of statistical learning. *Springer series in statistics*. New York: Springer.
Hua, X., Wang, W., Wang, Y., & Ren, M. (2018). Bus arrival time prediction using mixed multi-route arrival time data at previous stop. *Transport, 33*(2), 543–554. 10.3846/16484142.2017.1298055.
Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: Theory and applications. *Neurocomputing, 70*(1-3), 489–501. 10.1016/j.neucom.2005.12.126.
Ismagilova, E., Hughes, L., Dwivedi, Y. K., & Raman, K. R. (2019). Smart cities: Advances in research–an information systems perspective. *International Journal of Information Management, 47*, 88–100. 10.1016/j.ijinfomgt.2019.01.004.
Israilidis, J., Odusanya, K., & Mazhar, M. U. (2021). Exploring knowledge management perspectives in smart city research: A review and future research agenda. *International Journal of Information Management, 56*, 101989. 10.1016/j.ijinfomgt.2019.07.015.
Jafari, M., Kavousi-Fard, A., Niknam, T., & Avatefipour, O. (2021). Stochastic synergies of urban transportation system and smart grid in smart cities considering V2G and V2S concepts. *Energy, 215*, 119054. 10.1016/j.energy.2020.119054.
Jeong, R., & Rilett, R. (2004). Bus arrival time prediction using artificial neural network model. In *The 7th international ieee conference on intelligent transportation systems (IEEE Cat. No.04TH8749)* (pp. 988–993). 10.1109/ITSC.2004.1399041.
Ji, H., Zheng, W., Zhuang, X., & Lin, Z. (2021). Explore for a day? Generating personalized itineraries that fit spatial heterogeneity of tourist attractions. *Information & Management, 58*(8), 103557. 10.1016/j.im.2021.103557.
Kar, A. K., Gupta, M. P., Ilavarasan, P. V., & Dwivedi, Y. K. (2017). *Advances in smart cities: Smarter people, governance, and solutions*. CRC Press.
Kar, A. K., Ilavarasan, V., Gupta, M., Janssen, M., & Kothari, R. (2019). Moving beyond smart cities: Digital nations for social innovation & sustainability. *Information Systems Frontiers, 21*(3), 495–501. 10.1007/s10796-019-09930-0.
Kar, A. K., & Kushwaha, A. K. (2021). Facilitators and barriers of artificial intelligence adoption in business–insights from opinions using big data analytics. *Information Systems Frontiers*, 1–24. 10.1007/s10796-021-10219-4.
Karamichailidou, D., Alexandridis, A., Anagnostopoulos, G., Syriopoulos, G., & Sekkas, O. (2022). Modeling biogas production from anaerobic wastewater treatment plants using radial basis function networks and differential evolution. *Computers & Chemical Engineering, 157*, 107629. 10.1016/j.compchemeng.2021.107629.
Karamichailidou, D., Kaloutsa, V., & Alexandridis, A. (2021). Wind turbine power curve modeling using radial basis function neural networks and tabu search. *Renewable Energy, 163*, 2137–2152. 10.1016/j.renene.2020.10.020.
Klesk, P., & Korzen, M. (2021). Can boosted randomness mimic learning algorithms of geometric nature? Example of a simple algorithm that converges in probability to hard-margin SVM. *IEEE Transactions on Neural Networks and Learning Systems, 32*(9), 3798–3818. 10.1109/TNNLS.2021.3059653.
Kodiyan, A. A., & Francis, K. (2020). Prediction & classification of anomalies in transport network using dublin bus transit feeds. *Technincal Report*. 10.13140/RG.2.2.34048.05129.
Korovesis, N., Kandris, D., Koulouras, G., & Alexandridis, A. (2019). Robot motion control via an eeg-based brain–computer interface by using neural networks and alpha brainwaves. *Electronics, 8*(12), 1387. 10.3390/electronics8121387.
Kushwaha, A. K., Kar, A. K., & Dwivedi, Y. K. (2021). Applications of big data in emerging management disciplines: A literature review using text mining. *International Journal of Information Management Data Insights, 1*(2), 100017. 10.1016/j.jjimei.2021.100017.
Kushwaha, N., Pant, M., & Sharma, S. (2019). Electromagnetic optimization-based clustering algorithm. *Expert Systems*, e12491.. 10.1111/exsy.12491.
Larsen, G. H., Yoshioka, L. R., & Marte, C. L. (2020). Bus travel times prediction based on real-time traffic data forecast using artificial neural networks. In *International conference on electrical, communication, and computer engineering (ICECCE)* (pp. 1–6). 10.1109/ICECCE49384.2020.9179382.
Le Fablec, Y., & Alliot, J.-M. (1999). Using neural networks to predict aircraft trajectories. In *International Conference on Information Systems (ICIS)* (pp. 524–529).
Li, X., Wu, D., He, J., Bashir, M., Liping, M., & García, C.-A. (2020). An improved method of particle swarm optimization for path planning of mobile robot. *Journal of Control Science Engineering, 2020*. 10.1155/2020/3857894.
Liu, H., Xu, H., Yan, Y., Cai, Z., Sun, T., & Li, W. (2020). Bus arrival time prediction based on LSTM and spatial-temporal feature vector. *IEEE Access, 8*, 11917–11929. 10.1109/ACCESS.2020.2965094.
Lytras, M. D., Visvizi, A., Chopdar, P. K., Sarirete, A., & Alhalabi, W. (2021). Information management in smart cities: Turning end users' views into multi-item scale development, validation, and policy-making recommendations. *International Journal of Information Management, 56*, 102146. 10.1016/j.ijinfomgt.2020.102146.
Madakam, S. (2020). Would be smart balinese cities (Indonesia): An exploratory Indonesian case. *Journal of Information Systems and Technology Management, 17*. 10.4301/S1807-1775202017005.
Manfreda, A., Ljubi, K., & Groznik, A. (2021). Autonomous vehicles in the smart city era: An empirical study of adoption factors important for millennials. *International Journal of Information Management, 58*, 102050. 10.1016/j.ijinfomgt.2019.102050.
Mehra, V., Sarin, P., Singh, P., Sawhney, R. S., & Kar, A. K. (2021). Impact of COVID-19 pandemic on e-participation of fans in sports events. In *Conference on e-Business, e-Services and e-Society* (pp. 692–703). Springer. 10.1007/978-3-030-85447-8_57.
MetroTransit, 2021. https://svc.metrotransit.org/.
Moody, J., & Darken, C. J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation, 1*(2), 281–294. 10.1162/neco.1989.1.2.281.
Murano, Y., Ueno, R., Shi, S., Kawashima, T., Tanoue, Y., Tanaka, S., … Nguyen, H., et al., (2021). Impact of domestic travel restrictions on transmission of COVID-19 infection using public transportation network approach. *Scientific Reports, 11*(1), 1–9. 10.1038/s41598-021-81806-3.
Nevado Peña, D., López Ruiz, V. R., & Alfaro Navarro, J. L. (2020). An analysis of the key role of human and technological development in the smart specialization of smart European regions. *Information Technology for Development, 26*(4), 728–741. 10.1080/02681102.2019.1704675.
OpenStreetMap contributors, 2017. Planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org.
Pee, L. G., & Pan, S. L. (2022). Climate-intelligent cities and resilient urbanisation: Challenges and opportunities for information research. *International Journal of Information Management, 63*, 102446. 10.1016/j.ijinfomgt.2021.102446.
Pereira, G. V., Macadar, M. A., Luciano, E. M., & Testa, M. G. (2017). Delivering public value through open government data initiatives in a smart city context. *Information Systems Frontiers, 19*(2), 213–229. 10.1007/s10796-016-9673-7.
Pervaiz, S., Ul-Qayyum, Z., Bangyal, W. H., Gao, L., & Ahmad, J. (2021). A systematic literature review on particle swarm optimization techniques for medical diseases detection. *Computational and Mathematical Methods in Medicine, 2021*. 10.1155/2021/5990999.
Petersen, N. C., Rodrigues, F., & Pereira, F. C. (2019). Multi-output bus travel time prediction with convolutional LSTM neural network. *Expert Systems with Applications, 120*, 426–435. 10.1016/j.eswa.2018.11.028.
Petrou, P., Tampakis, P., Georgiou, H., Pelekis, N., & Theodoridis, Y. (2019). Online long-term trajectory prediction based on mined route patterns. In *International Workshop on Multiple-Aspect Analysis of Semantic Trajectories* (pp. 34–49). Springer, Cham. 10.1007/978-3-030-38081-6_4.
Quijano-Sanchez, L., Cantador, I., Corts-Cediel, M. E., & Gil, O. (2020). Recommender systems for smart cities. *Information Systems, 92*, 101545. 10.1016/j.is.2020.101545.
Ranjitkar, P., Tey, L.-S., Chakravorty, E., & Hurley, K. L. (2019). Bus arrival time modeling based on auckland data. *Transportation Research Record, 2673*(6), 1–9. 10.1177/0361198119840620.
Rauf, H. T., Bangyal, W. H., Ahmad, J., & Bangyal, S. A. (2018). Training of artificial neural network using PSO with novel initialization technique. In *2018 International conference on innovation and intelligence for informatics, computing, and technologies* (pp. 1–8). 10.1109/3ICT.2018.8855743.
Ravi, C., Tigga, A., Reddy, G. T., Hakak, S., & Alazab, M. (2022). Driver identification using optimized deep learning model in smart transportation. *ACM Transactions on Internet Technology*. 10.1145/3412353.
Rybnytska, O., Burstein, F., Rybin, A. V., & Zaslavsky, A. (2018). Decision support for optimizing waste management. *Journal of Decision Systems, 27*(sup1), 68–78. 10.1080/12460125.2018.1464312.
Sarimveis, H., Alexandridis, A., Tsekouras, G., & Bafas, G. (2002). A fast and efficient algorithm for training radial basis function neural networks based on a fuzzy partition of the input space. *Industrial & Engineering Chemistry Research, 41*(4), 751–759. 10.1021/ie010263h.
Sarkar, B. D., Shankar, R., & Kar, A. K. (2021). A scenario-based interval-input output model to analyze the risk of COVID-19 pandemic in port logistics. *Journal of Modelling in Management*. 10.1108/JM2-12-2020-0312.

Şen, D., Dönmez, C. Ç., & Yıldırım, U. M. (2020). A hybrid bi-level meta-heuristic for credit scoring. *Information Systems Frontiers, 22*(5), 1009–1019. 10.1007/s10796-020-10037-0.

Spiegel, M., Schiller, J., & Srinivasan, R. (2009). *Probability and statistics (3rd/Ed.)*. McGraw-Hill.

Stogiannos, M., Alexandridis, A., & Sarimveis, H. (2018). Model predictive control for systems with fast dynamics using inverse neural models. *ISA Transactions, 72*, 161–177. 10.1016/j.isatra.2017.09.016.

Sun, F., Pan, Y., White, J., & Dubey, A. (2016). Real-time and predictive analytics for smart public transportation decision support system. In *IEEE international conference on smart computingv(SMARTCOMP)* (pp. 1–8). 10.1109/SMARTCOMP.2016.7501714.

Sun, T., Zhao, K., Zhang, C., Chen, M., & Yu, X. (2022). PR-LTTE: link travel time estimation based on path recovery from large-scale incomplete trip data. *Information Sciences, 589*, 34–45. 10.1016/j.ins.2021.12.091.

Theodoridis, S., & Koutroumbas, K. (2008). *Pattern recognition* (4th ed.). Academic Press.

Thorat, O., Parekh, N., & Mangrulkar, R. (2021). TaxoDaCML: Taxonomy based divide and conquer using machine learning approach for DDoS attack classification. *International Journal of Information Management Data Insights, 1*(2), 100048. 10.1016/j.jjimei.2021.100048.

Tripathi, A., Goswami, T., Trivedi, S. K., & Sharma, R. D. (2021). A multi class random forest (MCRF) model for classification of small plant peptides. *International Journal of Information Management Data Insights, 1*(2), 100029. 10.1016/j.jjimei.2021.100029.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer Science & Business Media.

Velasquez Ortiz, R. A., Álvarez Rodríguez, F. J., Vargas Martin, M., & Ponce Gallegos, J. C. (2019). Mapping of the transportation system of the city of aguascalientes using GTFS data for the generation of intelligent transportation based on the smart cities paradigm. In *International conference on advances in emerging trends and technologies* (pp. 177–185). Cham: Springer. 10.1007/978-3-030-32022-5_17.

Wang, L., Zuo, Z., & Fu, J. (2014). Bus arrival time prediction using RBF neural networks adjusted by online data. *Procedia Social and Behavioral Sciences, 138*, 67–75. 10.1016/j.sbspro.2014.07.182. The 9th International conference on traffic and transportation studies (ICTTS 2014)

Wu, Y. J., & Chen, J.-C. (2021). A structured method for smart city project selection. *International Journal of Information Management, 56*, 101981. 10.1016/j.ijinfomgt.2019.07.007.

Xiao, Y., He, X., Yang, C., Liu, H., & Liu, Y. (2022). Dynamic graph computing: A method of finding companion vehicles from traffic streaming data. *Information Sciences, 591*, 128–141. 10.1016/j.ins.2022.01.022.

Yadav, H., Kar, A. K., & Kashiramka, S. (2021). How does entrepreneurial orientation and SDG orientation of CEOs evolve before and during a pandemic. *Journal of Enterprise Information Management*. 10.1108/JEIM-03-2021-0149.

Yu, J., Zhou, M., Wang, X., Pu, G., Cheng, C., & Chen, B. (2021). A dynamic and static context-aware attention network for trajectory prediction. *ISPRS International Journal of Geo-Information, 10*(5), 336. 10.3390/ijgi10050336.

Zhang, H., Liu, F., Zhou, Y., & Zhang, Z. (2020). A hybrid method integrating an elite genetic algorithm with tabu search for the quadratic assignment problem. *Information Sciences, 539*, 347–374. 10.1016/j.ins.2020.06.036.

Celan, M., & Lep, M. (2017). Bus arrival time prediction based on network model. *Procedia Computer Science, 113*, 138–145. 10.1016/j.procs.2017.08.331.

Eva Chondrodima received the PhD degree in computational intelligence from the National Technical University of Athens (NTUA), Greece, in 2017, the MSc degree in advanced information systems from the University of Piraeus, Greece, in 2013, and the BSc degree in electronic engineering from the Technological Educational Institute of Athens, Greece, in 2011. She is currently a Post-Doctoral researcher with the Department of Informatics, University of Piraeus. Her research interests include computational intelligence with emphasis on neural networks and evolutionary computation methods, and applications to geoscience, mobility data science, and semantic Web.

Harris Georgiou received the PhD degree in machine learning/medical imaging and the MSc degree in digital signal processing/computer systems from NKUA, Greece, in 2009 and 2000, respectively, and the BSc degree in informatics from the University of Ioannina, Greece, in 1997. He is currently a Data Scientist specializing in mobility analytics, signal/image processing, Bioinformatics and Artificial Intelligence. He has published 76 peer-reviewed journal & conference papers plus 72 independent works, contributed in five major books and one U.S. patent in related R&D areas. He is an Active Member of HRTA as a certified scuba diver and emergency medical care-first responder.

Nikos Pelekis received the PhD degree from UMIST, United Kingdom, in 2002. He is currently an Assoc. Professor in the Department of Statistics and Insurance Science, University of Piraeus. His research interests include all topics of data science with a particular interest for the field of mobility data management and mining.

Yannis Theodoridis is Professor of Data Science with the Department of Informatics, University of Piraeus, Greece. His research interests include Data Science (big data management & analytics) for humans-mobility-related information. He has co-authored three monographs and over 100 refereed articles in scientific journals and conferences, with over 12,500 citations ($h = 50$), according to Google Scholar. He is senior member of ACM.