Transportation Research Part C 153 (2023) 104195

Contents lists available at ScienceDirect

Transportation Research Part C

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

Forecasting passenger flows and headway at train level for a public transport line: Focus on atypical situations

Thomas Bapaume [a,b,*], Etienne Côme [a], Mostafa Ameli [a], Jérémy Roos [b], Latifa Oukhellou [a]

[a] *Univ. Gustave Eiffel, COSYS, GRETTIA, Champs-Sur-Marne, France*
[b] *Régie Autonome des Transports Parisiens (RATP), Paris, France*

A R T I C L E I N F O

Keywords:
Public transport
Forecasting
Machine learning
Computer vision
Inpainting
Transformer
Atypical situation

A B S T R A C T

This paper proposes a computer vision framework based on deep learning approaches for the real-time prediction of passenger loads and train headways in a metro line in an urban transit network. The short-term prediction problem is formulated as an image completion task. The train journeys on the metro line are represented as images, with the pixels denoting the train data, including the departure location, time, and load. Metro line applicative constraints, e.g., irregular time sampling of trains and univariate space, are considered in the images. Several deep learning-based architectures are investigated, including two new architectures based on transformers. Finally, an in-depth analysis and comparison of the models is performed based on a real test case of Paris metro line 9, for which a large database was collected over three years. This allowed us to include numerous instances with atypical transport system performance indicators (e.g., strikes, lockdowns, and disruptions) and to design a methodology (based on a latent space representation of the dataset) to identify and label interesting test cases. We evaluate and demonstrate the robustness of the proposed approaches numerically and study their limits for passenger loads and headways in atypical scenarios.

1. Introduction

The sustainable development of large cities requires shifting from individual and polluting modes of transport to more environmentally friendly transport modes. Within this context, mass transit railways are becoming increasingly important alternatives for passenger traffic. This shift toward public transport with an increasing number of passengers requires the development of intelligent and efficient traffic flow management tools. Indeed, the number of passengers that use a transit system impacts the quality of transport, especially during peak hours (An et al., 2020). Forecasting the evolution of the system performance indicators is thus crucial for transport operators and travelers. Operators use real-time forecasting methods to anticipate and optimize service levels. Travelers can obtain robust information related to schedules and comfort to plan their journey. Moreover, a reliable short-term prediction model must be able to address complex events such as congestion or disruptions to anticipate the evolution of passenger flows and adjust timetables during these perturbed periods.

Many approaches for representing and solving short-term prediction problems involve nonlinear inference and recursive data with spatial and temporal dependencies, such as the autoregressive integrated moving average (ARIMA) method (Marchev et al., 2022), support vector machine (SVM) (Meng et al., 2022), and probabilistic approaches (Roos et al., 2016; Colace et al., 2020; Chen

* Corresponding author at: Univ. Gustave Eiffel, COSYS, GRETTIA, Champs-Sur-Marne, France.
E-mail address: thomas.bapaume@univ-eiffel.fr (T. Bapaume).

https://doi.org/10.1016/j.trc.2023.104195
Received 16 December 2022; Received in revised form 22 April 2023; Accepted 2 June 2023
Available online 28 June 2023

et al., 2022a). Nevertheless, deep learning (DL) and machine learning (ML) approaches have recently been widely used to address forecasting problems in transportation applications. Compared to conventional time series methods, DL and ML approaches have obtained better accuracy in modeling sequential multivariate data and offer better scalability (Kim et al., 2016). The main techniques deployed in the literature are based on recurrent neural networks such as long short-term memory (LSTM) or gated recurrent unit (GRU) frameworks (see, e.g., Hochreiter, 1998; Toqué et al., 2016). These approaches can capture the short-term and long-term evolution of the target time series to be predicted. In addition, the spatial aspects can be treated in different ways depending on the scale of the network or the study objectives. Some approaches are limited to single stations and do not consider spatial relationships, while other methods attempt to represent roads or transport networks via appropriate data structures, such as graphs. In recent deep learning studies (Cui et al., 2020; Wang et al., 2020), urban networks have been designed as graphs representing interconnections of roads or bus travel routes to capture spatial correlations. Most of these hybrid methods use LSTM networks to address temporal dependencies and graphs to address spatial dependencies.

Most studies focus on forecasting one performance indicator concerning either passenger flows (e.g., passenger occupancy, ridership) or train operation (e.g., travel time, headway). This limitation occurs because data-driven approaches are not designed to solve two prediction problems simultaneously. Moreover, the developed approaches are specific to the data and the target variable(s) to be predicted. Thus, existing approaches do not provide a flexible methodology. Therefore, a public transport metro line forecasting approach must consider the following aspects: (i) interactions among all trains moving along the line, (ii) encoding the irregular displacement of trains in time and space, and (iii) the robustness of the proposed frameworks for critical situations such as disruptions.

The approach proposed here follows the research we performed in Bapaume et al. (2021). We extend our previous work by addressing short-term forecasting for the public transport of passenger loads or headway based on image representations of public transport lines rather than only one target. The architecture for the inpainting forecasting framework for solving the two forecasting tasks is designed considering recent advanced computer vision models, such as attention mechanisms and transformers. Inpainting is a well-known computer vision technique that aims to complete an image's missing or cropped areas. Adapting inpainting to a forecasting task is a recent approach used by Ma et al. (2017) to extract features and forecast the level of traffic on highway roads. In addition to using 2D images, Liu et al. (2022) used inpainting to forecast the performance of electric energy systems by using time series as 3D images. Here, we propose to use advanced methods, such as attention-based models, to improve the prediction performance. The efficient performance of attention-based models, e.g., the Transformer, has been proven in various computer vision tasks, and these models have become the state of the art (Zhai et al., 2022). The transformer architecture, which was introduced in 2017 by Vaswani et al. (2017), has shown promising performance in various domains, such as natural language processing with BERT architectures (Devlin et al., 2018) and computer vision tasks with Vision Transformer (ViT) architectures (Dosovitskiy et al., 2021). This model uses attention (e.g., similar to human behavior) to enhance relationships between elements in an image or a sequence. In addition, self-attention mechanisms have been used in short-term prediction tasks by Chen et al. (2022b), Wu et al. (2020a), Hao et al. (2019) to leverage the features that are highly correlated with the forecasting performance. These studies applied attention mechanisms in the state and gate equations of LSTM networks.

First, this paper proposes applying transformer-based models to image representations of metro traffic data to achieve short-term forecasting for a metro line. In addition, the methodology is flexible enough to be easily adapted to every target on the public metro line, such as the passenger load or headway. Second, the use of traffic images allows us to apply different statistical and deep learning processes to extract several test bases to evaluate the models in various operational contexts. The contributions of this paper can be summarized as follows:

- We develop two deep learning attention-based architectures, the U-transformer and the channel vision transformer, for the short-term prediction of train loads and headway. With the attention mechanism, the models can extract dependencies among all train departures from the image by considering the specific details in the metro line traffic image.
- We perform benchmark evaluations based on a real dataset of Paris metro line 9 collected over three years of operation to compare the forecasting ability of these two architectures. An extensive comparison of the forecasting performance of different models is carried out, including eight deep learning and basic machine learning approaches.
- We exploit the latent space of the prediction models to interpret the forecasting process. We use attention scores to explain the results and highlight the relationships between the predictions of one train departure (one pixel) and all train departures in an image.
- We label images to create various test sets representing atypical situations observed on metro lines, such as strikes or disruptions. The goal is to extract features from images or the latent space of models to build several test sets.
- Finally, we evaluate the proposed models based on the various test sets representing atypical situations to reveal the weaknesses and strengths of the models in real metro situations. The goal is to evaluate the robustness and generalizability of the proposed models with complex images.

The remainder of this paper is organized as follows. Section 2 presents a literature review of methods for forecasting passenger flows and train operation in public transport scenarios. In Section 3, an image representation of a metro line is defined. Then, the forecasting frameworks and transformer-based models are presented in Section 4. Section 5 introduces the experimental setup and details the forecasting performance of the proposed models based on two target variables, namely, the passenger load and the headway. In addition, we present the results over five months of public transport images in Section 5. Section 6 describes the details of various test sets used to evaluate the prediction robustness of the proposed models in particular contexts. Finally, Section 7 outlines the main findings of this paper and discusses some future research directions.

Table 1
A comparison of different passenger flow forecasting papers.

Research	Operation status		Spatial information			Time sampling		Methodology					Interpretability	Data
	Normal	Atypical	Single station	Transit line	Full network	Regular	Irregular	Deep learning				Other methods		
								LSTM	Graph	Transf.	U-net			
Toqué et al. (2016)	x		x			x		x						15 months
Zhao et al. (2017)	x				x	x		x						6 months
Liu et al. (2019)	x		x			x		x						1 month
Liyanage et al. (2022)	x			x		x		x						1 month
Wang et al. (2018)	x				x	x		x	x					1 month
Colace et al. (2020)	x		x			x						x	x	-
Li et al. (2020)	x				x	x			x					3 months
Peng et al. (2020)	x				x	x		x	x					4 months
Pasini et al. (2020)	x		x				x	x						1 year
Du et al. (2019)	x	x			x	x		x	x					4 months
Wu et al. (2020b)	x					x						x		-
Egu and Bonnel (2021)	x		x			x						x		4 years
Roos et al. (2016)	x		x			x						x	x	1 month
Heydenrijk-Ottens et al. (2018)	x			x			x					x	x	1 month
Hao et al. (2019)	x					x				x				1 week
Chen et al. (2022b)	x	x				x				x				3 months
Bapaume et al. (2021)	x			x			x				x			1 year
This work	x	x		x			x			x	x		x	3 years

2. Related work

This section is divided into two main parts: a survey on forecasting frameworks developed for passenger flow prediction, such as passenger loads or ticketing logs, and a second survey on forecasting train operations, such as the headway or travel time.

2.1. *Passenger flow forecasting*

Deep learning and LSTM models are standard approaches for forecasting passenger flows. Liyanage et al. (2022) used an LSTM network to predict bus passenger flows in Melbourne. Toqué et al. (2016) proposed an LSTM-based model to forecast the dynamic origin–destination matrices of a subway line in Rennes. Zhao et al. (2017) proposed a two-dimensional LSTM network for the same task, considering the spatiotemporal correlations between the origins and destinations in a transit network. Many studies have proposed combinations of two deep learning architectures, using an LSTM network to address temporal dependencies and another approach to address spatial dependencies. For example, Wang et al. (2018) developed a deep convolutional LSTM (ConvLSTM) network to extract spatiotemporal information to forecast travel passenger flows for Chengdu City in China. However, LSTM-based models require time-sequential data and are designed to focus mainly on temporal dependencies. There are various strategies to address spatial dependencies, such as graph-based or image-based approaches. In these approaches, the spatiotemporal series of public transport networks, including the history of all train movements, are represented as images. In recent years, several graph-based methodologies have been proposed to represent public transport networks and encode spatiotemporal features. For example, Colace et al. (2020) used a three-level graph representation of the London metropolis to forecast bicycle usage. Li et al. (2020) applied probabilistic graph convolutions to extract spatiotemporal features and forecast the performance of a Sydney public transit network. Peng et al. (2020) merged a recurrent neural network with graph convolutions to learn spatiotemporal information to predict bus and taxi flow distributions in Beijing. Several other conventional machine learning strategies have been developed to forecast the time series of public transport networks, such as gradient boosting (Wu et al., 2020b; Egu and Bonnel, 2021) and dynamic Bayesian networks (Roos et al., 2016; Heydenrijk-Ottens et al., 2018), which transform public transport load forecasting into a supervised classification task. With this approach, prediction tasks can be defined more precisely for all moving buses within a public transport system. We used an image-based approach in the present study to consider spatiotemporal dependencies and define the prediction tasks.

To compare the results of different studies, we present a characterization of the existing prediction frameworks in the literature in Table 1. A forecasting framework is often constructed according to the prediction objectives and the data available for its implementation. Thus, we have defined several categories to classify the various transit forecasting models, as shown in Table 1, including the operation status, spatial information, time sampling, methodology, and interpretability.

The operation status denotes whether the study investigated normal or rare and complex forecasting situations. The spatial information defines the spatial scope of the forecasting task, ranging from local to global information (e.g., station, bus, train line, transit network). The time sampling denotes the data aggregation approach. Regular sampling means that the data were aggregated at specific time intervals (e.g., every 10 min). The data column denotes the amount of data used for the forecasting task.

In contrast, irregular sampling means that the data were sampled according to real events or changes within the network (e.g., train departures, taxi stops). The methodology column refers to the models deployed for forecasting. Finally, the interpretability column indicates whether the study focused on interpreting the models' results.

Table 2
A comparison of different train operation indicator forecasting papers.

Research	Operation status		Spatial information			Time sampling		Methodology					Interpretability	Data
								Deep learning				Other methods		
	Normal	Atypical	Single station	Transit line	Full network	Regular	Irregular	LSTM	Graph	Transf.	U-net			
Wu et al. (2020a)	x		x			x		x						2 months
Marchev et al. (2022)	x		x			x		x						6 months
Petersen et al. (2019)	x		x			x		x						6 months
Ma et al. (2022)	x		x			x		x						1 month
Comi et al. (2020)	x		x			x		x						2 months
Meng et al. (2022)	x		x			x						x		1 month
Serin et al. (2021)	x		x			x		x						-
Coulaud et al. (2022)	x		x			x						x		1 month
Chen et al. (2022a)	x			x			x					x	x	1 month
Ricard et al. (2022)	x		x			x						x		3 weeks
Lee and Rhee (2022)	x				x	x		x	x	x				1 month
Chen et al. (2022c)	x				x	x		x	x	x				3 months
Zheng et al. (2022)	x				x	x		x	x					1 month
This work	x	x		x			x			x	x		x	3 years

2.2. *Train operation forecasting*

Methods for forecasting train movement usually rely on the same approaches as methods for forecasting passenger flows. The target variable to be predicted can take many forms, such as the dwelling time, the departure time of a bus or train, the headway (data that inform passengers about the delay between two train departures), or the travel time. The main objective is to forecast the next vehicle's travel time following the chosen transport mode. For passenger flow prediction, similar machine learning and deep learning approaches (Lee and Rhee, 2022; Zheng et al., 2022; Chen et al., 2022c) are commonly used in the literature, but some studies in particular are worth mentioning due to their originality or novelty. Marchev et al. (2022) described a framework to evaluate on-demand public transport travel time called INNOAIR. They combined three models: an autoregressive model to predict the bus schedule, a Bayesian model to simulate flow demand, and a neural network to forecast the travel time for the Sofia Municipality in Bulgaria. Petersen et al. (2019) proposed a multistep forecasting method for bus travel times with a hybrid model for the Greater Copenhagen area. They combined a convolutional neural network and an LSTM model to extract spatiotemporal dependencies. Ma et al. (2022) introduced a new approach for predicting the travel time of buses. Instead of directly forecasting the bus travel time, the authors used short-term forecasting based on all roads within the network to determine the bus travel time. First, the city network was represented as a graph. Then, a multiattention graph neural network was applied to consider spatial dependencies, while an LSTM network with a transformer was used to extract temporal dependencies to forecast the edge of the graph (travel time for one road segment).

Comi et al. (2020) presented an analysis of bus travel time for Lviv in Ukraine to build a long-term forecasting method using a statistical model called seasonal and trend decomposition and locally estimated scatterplot smoothing (LOESS). However, deep learning methods require a large amount of data. To overcome this constraint, Meng et al. (2022) proposed a short-term travel time prediction method that used a small amount of data based on nonparametric models for an expressway in Singapore. They used an SVM and the nearest neighbor method to address the nonlinear relationships in the traffic data and provide a robust forecasting method based on only a few days of data. Other studies focused on time series approaches, such as Serin et al. (2021), who proposed the use of three residual layers to forecast bus travel time in Istanbul. Coulaud et al. (2022) forecasted the dwelling time and arrival time for a train line with an L-shaped regression approach.

More recently, probabilistic approaches have been developed to predict the headway and assess uncertainties in travel time prediction (Chen et al., 2022a). The prediction results are provided as an explainable distribution of the headway per bus stops. In this case, the forecasting task aims to offer the passengers a distribution of the target value with robust forecasting performance. The authors proposed a Bayesian probabilistic model to forecast the distribution of the bus travel time over a line for a short-term horizon. They captured the interactions between adjacent buses and travel times to sample the future travel times through a Markov chain Monte Carlo approach and defined the probabilistic distribution of the headway for a bus line. Ricard et al. (2022) focused on estimating the probability density function for long-term bus travel time based on the bus schedule. They used two approaches, similarity-based density estimation and smoothed logistic regression for probabilistic classification of the bus travel time.

Similar to the literature review on passenger flow forecasting, Table 2 presents all the above references, with the same categorization used to define the scope of the papers.

Based on previous literature, this paper proposes a deep learning prediction framework that combines computer vision approaches and attention mechanisms. According to the results presented in Tables 1 and 2, this paper aims to integrate operational constraints, as described below:

- Unlike many similar prediction studies focusing on forecasting average values aggregated over a given time window (such as passenger traffic or load), this paper adopts a train/station-oriented approach. This modeling approach enables us to consider the actual operation of a metro line.

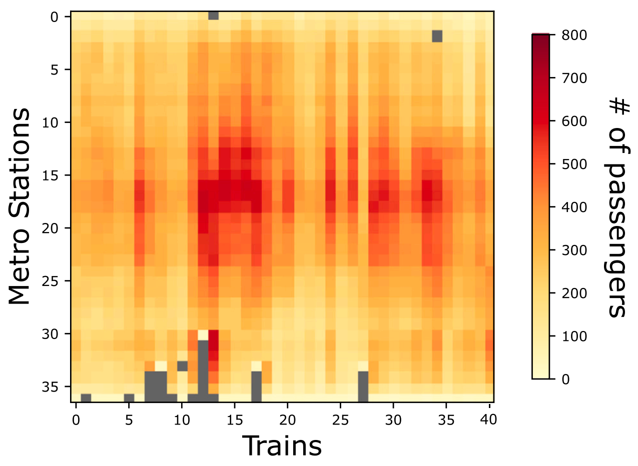


Fig. 1. Image $I(t)$ of Paris metro line 9, which has 40 trains and 36 stations. The gray pixels represent missing values.

- Most recent studies have employed graph neural networks and assumed that they have access to either the data for the entire transportation network or a significant portion of the data. However, an approach based on individual lines in the Parisian metro system, which includes both modern and older lines, seems more appropriate because only three out of the 14 lines in the system are equipped with load sensors.
- We consider a database collected over three years to develop and test our models. This approach allowed us to include numerous instances with atypical traffic conditions and transport passenger flows. In comparison, most studies in this field (as shown in Table 3) utilize data collected for only several months, which is insufficient for a thorough evaluation of methods for atypical situations that occur infrequently.
- The use of image formalism has proven to be successful for encoding spatial correlations. Here, we extend this approach to labeling atypical situations. Moreover, this formalism is straightforward and easy to comprehend from an operational perspective. By projecting the images into the latent space and using the image format for manual labeling, we created distinct test sets for evaluating the model's performance in atypical situations.

3. Problem and statement

This section presents the transformation of the short-term prediction problem into an image completion task. This is achieved by representing the train journeys on a metro line as images.

3.1. Metro traffic image representation

In Bapaume et al. (2021), we defined an image representation for trains moving along a metro line, with each column representing the course of the train along the entire metro line and each row representing the sequence of departures at each metro station (see Fig. 1). Thus, each pixel in the image denotes a train departure from one station. The pixel encodes train or station variables linked to this departure, including the passenger load, the dwelling time, and the headway. Fig. 1 depicts an image of forty trains over a metro line denoted I. Here, the color represents the number of passengers inside the train at its departure, which is denoted as the train load or load. The gray pixels in the traffic image represent either a missing completely at random (MCAR) departure, meaning an error in the data acquisition, or a missing not at random (MNAR) departure, which is related to the train schedule (e.g., closed stations, early train stopping). Fig. 1 offers a clear example of the interest in encoding missing data. In the image, the 12th train stops earlier at the 31st station, which is represented by the gray pixels toward the end of the column. This event leads to the passengers transferring to the following trains, thus significantly increasing their loads.

3.2. Dimension and channel encoding

In computer vision, we can define an image over 3 dimensions: the height h, the width w, and the channels d. A channel is a subimage that encodes one variable. Standard RGB (red, green, blue) images have 3 channels, one for each color. For the forecasting case, the height is the number of stations in the studied metro line, and the width represents the number of trains considered to generate the image, which is linked to the maximum capacity of the public transport line and the forecasting task (i.e., the maximum number of trains that can operate simultaneously). Here, the height and width are set to 37 (stations) and 40 (trains). The metro traffic image channels represent data from the train network. For this study, the following four variables are defined:

- Passenger load: The number of passengers inside a train at its departure

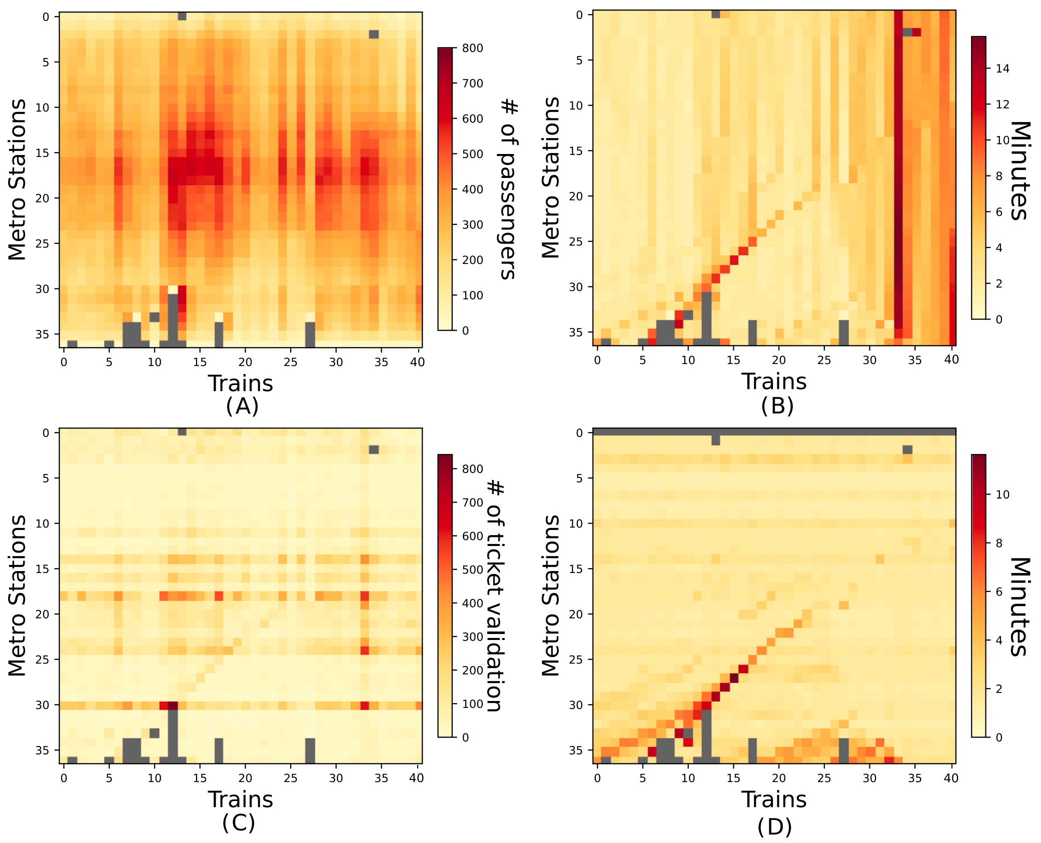


Fig. 2. Channels used to generate the metro traffic images. (A) Train passenger load channel, (B) headway channel, (C) tap-in ticketing channel, (D) travel time channel.

- headway: The number of minutes between the current train departure and the previous train departure at the same station.
- Tap-in ticketing: The number of passengers entering the network at a station during the time between the current train and the previous train
- Travel time: The duration in minutes for a train traveling from the previous station to the current station.

Fig. 2 presents a sample image of all the aforementioned variables. Although the names of the time variables may imply a strong correlation, their linear correlation coefficient is only 0.07. It is important to note that each variable represents a different aspect of the system. The headway provides insight into the train frequency and delay patterns, while most trains' travel times remain relatively constant. However, variations in travel times can be informative for differences in parking times. Therefore, these two variables define distinct physical realities and should be analyzed separately. The proposed framework is flexible with respect to other forecasting target(s) or data source(s). The image channels allow us to change the target to be predicted without modifying any other framework elements. In addition, this framework can be adapted to any other public transit line by simply changing the size of the input image. Moreover, it is possible to add categorical data as channels to address exogenous data.

3.3. Real time images and mask

Fig. 1 represents an image at the end of the day when all the trains have stopped running. However, from a real-time perspective, the trains are moving along the metro line. Thus, at time t, many pixels are missing in the image, as depicted by the gray pixels in Fig. 3(A). We denote the real-time image as $I^{in}(t)$ at time t.

The image $I^{in}(t)$ is composed of two parts: a past area with the traffic realized at time t and a future area representing the train departures to be forecasted. In addition, the size of the missing area can change depending on the time of the day (e.g., peak hours, off-peak hours), as different times involve various numbers of moving trains. Thus, forecasting the next train departures involves completing the missing area introduced by constructing the image $I(t)$ for the target variable. This method is called inpainting in computer vision. The forecasting horizon is based on the number of pixels or train departures to be predicted for each station. A short-term forecast focuses on a few pixels per station, whereas a long-term forecast focuses on the whole image. A mask m_p is defined to encode valid pixels, which must be predicted. Fig. 3(B) depicts an example of a mask to forecast the next 4 departures.

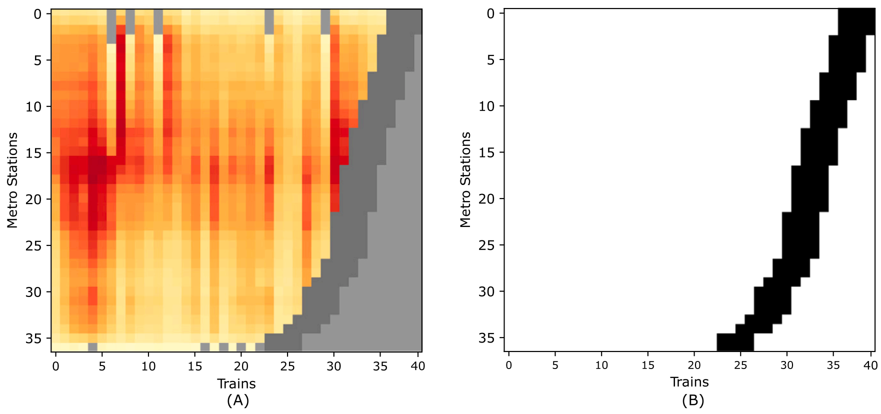


Fig. 3. (A) Real-time image $I^{in}(t)$ of Paris metro line 9, including 37 stations and 40 trains, at time t; the gray pixels represent future train stops, and darker pixels represent the next 4 departures to be predicted. (B) Mask m_p of the pixels to be forecasted (cluster pixels) within the image.

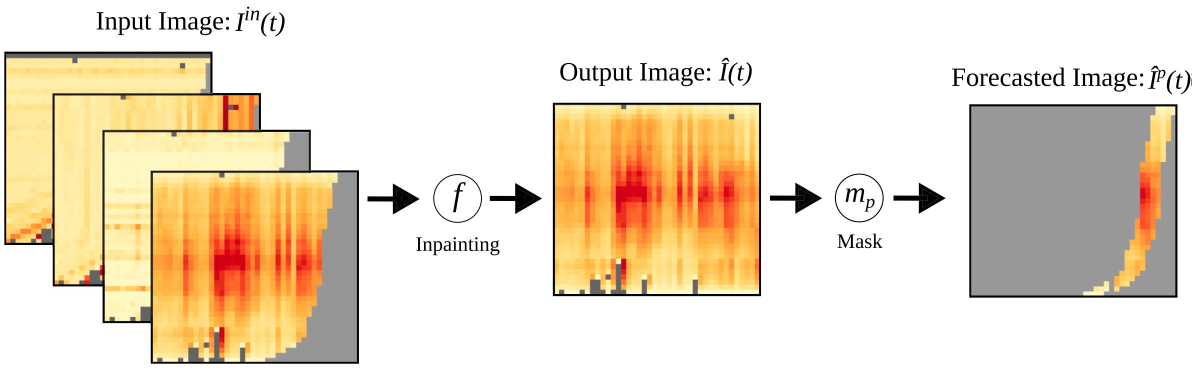


Fig. 4. A two-step forecasting framework: inpainting and forecasting. The inpainting image is the result of the inpainting function f. The forecasted image represents only the pixel considered in the prediction task.

The image forecasting framework is divided into two serial steps, an inpainting task and a forecasting task, as shown in Fig. 4. Forecasting is performed only in the first step. The second step selects and trains the model based on the target pixels. We denote the inpainted image and forecasted image as $\hat{I}(t)$ and $\hat{I}^p(t)$, respectively. In addition, the input image $I^{in}(t)$ is composed of several subimages called channels that encode several metro line variables.

4. Methodology

This section introduces the inpainting framework for the short-term forecasting of an urban public transport line. Two architectures were designed for the inpainting function based on the attention mechanism.

4.1. *Inpainting framework*

In computer vision, image inpainting is a well-known approach used to complete missing areas in images (Bertalmio et al., 2000; Goodfellow et al., 2014). The idea is to extract visual features and contexts from the image to replace all the missing pixels in the target image. In this study, inpainting is equivalent to painting the pixels corresponding to the future train departures based on $I^{in}(t)$, as shown in Fig. 1. Our inpainting framework includes two main steps: (i) reconstruction of the missing part in image $I^{in}(t)$, denoted by $\hat{I}(t)$, and (ii) extraction of the future target data from $\hat{I}(t)$, as illustrated in Fig. 4. In the first step, we apply an inpainting function f to a multichannel (four-channel) input image to infer a target image. The generated image includes only the channel related to the prediction task, and the other channels are present only in the input images. The second step aims to extract relevant pixels for the prediction task. Depending on the scope of the forecasting task, the prediction area varies in a time dependent manner. Therefore, we dynamically generate a mask m_p for each image to select the corresponding pixels from $I(t)$ and $\hat{I}(t)$. The idea is to extract the future target training variable, denoted by $I^{in}(t)$ for the input image and $\hat{I}^p(t)$ for the output image. The following equations summarize the framework tasks performed to obtain the prediction image:

$$\hat{I}(t) = f(I^{in}(t), \theta) \tag{1}$$

where f denotes the forecasting function associated with a neural network with parameters θ. The deep learning architecture is built according to the inpainting task. The resulting image $\hat{I}(t)$ is processed through the following equation:

$$\hat{I}^p(t) = \hat{I}(t) \odot m_p \tag{2}$$

where $\odot$ denotes the Hadamard product. Thus, the model must minimize the following loss function L:

$$L = \alpha MSE(I(t), \hat{I}(t)) + \beta MSE(I^p(t), \hat{I}^p(t)) \tag{3}$$

The loss function in Eq. (3) is a pixelwise loss function that is divided into two components, an inpainting term and a forecasting term. $I(t)$ and $I^p(t)$ denote the target images, which include only real data before inpainting and forecasting. $I^p(t)$ is the result of the application of m_p to $I(t)$, similar to Eq. (2). The model is trained according to the mean squared error (MSE) to consider pixelwise errors. In Eq. (3), α and β are weights, which are used to determine the importance of the task during the training phase. Thus, $\alpha, \beta \in [0, 1]$ and $\alpha + \beta = 1$. In other words, these weights represent the trade-off between the exploration and exploitation processes, in which inpainting is the exploration process and forecasting is the exploitation process.

In summary, we presented the model starting with the input and output images. We also introduced prediction masks that allow us to transform the inpainting task to a target pixel prediction task. However, the approaches that are implemented to perform the inpainting function f must still be introduced.

4.2. Forecasting transformer models

This section introduces the vision transformer framework and two models that perform the inpainting task. They rely on attention mechanisms and transformers to perform the image completion task.

In recent years, there has been a major evolution in deep learning with the development of the novel transformer architecture (Vaswani et al., 2017), which is based on attention mechanisms. This model attempts to simulate the human behavior of attention. From a computer science point of view, attention mechanisms enhance the dependencies between data sequence or image elements. In addition, they can manage temporal dependencies without any recursive architecture. The Transformer was originally used in natural language processing (NLP) to map the dependencies between words in a sentence or text. In computer vision, the vision transformer (ViT) (Dosovitskiy et al., 2021) model was proposed to adapt the attention mechanisms to the image classification task by constructing a visual word sentence called a patch (where a visual word is a small group of neighboring pixels). The transformer model is now considered the state-of-the-art in many computer vision tasks, such as classification (Yu et al., 2022; Khan et al., 2021; Liu et al., 2020), segmentation (Gao et al., 2021; Cao et al., 2021), object detection (Beal et al., 2020; Carion et al., 2020) and image generation (Lin et al., 2018). Recently, transformers have been applied to short-term forecasting tasks in the transport domain. For example, Chen et al. (2022b) built a bidirectional encoder–decoder transformer model to estimate future and past traffic conditions to forecast transport flows. Wu et al. (2020a) forecasted travel times by integrating attention mechanisms in a ConvLSTM, while (Hao et al., 2019) forecasted metro passenger flows by merging a sequence-to-sequence architecture with an attention mechanism.

The transformer uses a deep learning component that relies on attention to build various dependency representations of all pixels regardless of their coordinates. Following (Vaswani et al., 2017), a transformer block similar to those used in our two architectures is composed of three main elements: positional encodings to propagate information based on the coordinate of a patch or pixel, a multihead layer to apply attention mechanisms and a multilayer perceptron (MLP).

The core of such a module is the multihead attention layer. This layer uses several heads with distinct weights to compute self-attention according to Eq. (4):

$$Attention(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{4}$$

where d_k denotes the projection dimension of the transformer, $Q \in \mathbb{R}^{n \times d_k}$ is the query vector, $K \in \mathbb{R}^{n \times d_k}$ is the key vector, and $V \in \mathbb{R}^{n \times d_k}$ is the value vector. The goal of an attention mechanism is to compute the similarity between the query vector Q and the key vector K. Then, the pixel features are passed through the value vector V.

In our computer vision task, the Q, K, and V vectors are computed based on the pixels in image $I(t)$ through projection matrices denoted by W^Q, W^K, and W^V. Several heads are simply concatenated to form the final layer, as shown in Eqs. (5) and (6).

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{5}$$

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \tag{6}$$

Each $head_i$ in the multihead layer has its own set of projection matrices W_i^Q, W_i^K, and W_i^V, which are used to map multiple representations of the same input patch.

In computer vision, a non-overlapping patch decomposition technique introduced by Dosovitskiy et al. (2021) is commonly used to reduce the size of input vectors. This involves creating an ordered sequence of small, flattened pictures, known as patches, from an image. The patches are created with a predetermined size, such as 16×16 or 4×4 pixels. To maintain the patch order, the 1D linear positional encoding used in Dosovitskiy et al. (2021) is applied to help the transformer obtain information about the location of the patch within the image. The transformer propagates the position data between the different model layers with skip connections. Transformer models have the potential to achieve good performance in many domains. However, these models can be expensive to train in terms of computing resources, as a large amount of computer memory (graphic processing unit, GPU) is required to calculate attention scores that are quadratic with respect to the input sequence. Thus, creating a patch encoding of a single pixel considerably increases memory usage and impacts the architecture and training. This work proposes two deep learning architectures using a transformer as a primary component. The first is built as a U-net, while the second works independently on each of the channels. They share two common elements: an image-to-image framework and transformer block . The image-to-image framework is designed to perform load or headway prediction only from images without the use of recursive or graph units.

4.2.1. U-Transformer

The first models proposed integrating attention mechanisms in the inpainting function f to capture the context for any part of the image. Most image processing models rely on convolutional layers to observe visual patterns locally without considering the image's general context. In the case of our train traffic image, it seems relevant to attempt to extract context information from the metro traffic image for the inpainting task. Recall that the transformer aims to compute the attention between all pixels or patches, while the convolutional layer approach addresses only features related to the pixel and its neighbors. Thus, for each patch, the attention scores represent long-term and short-term dependencies for both spatial and temporal aspects within the image, making it possible to consider both local and global information for each train departure.

The first model, denoted U-Transformer, is an evolution of the U-net model. U-net (Ronneberger et al., 2015) is a deep learning architecture used to perform image-to-image tasks such as image segmentation, image generation, and inpainting. Fig. 5 presents the architecture of the model used in the forecasting process. The model consists of consecutive convolutional and pooling layers (i.e., dimension reduction) and symmetrical deconvolutional and unpooling layers, resulting in the U-shaped architecture. The convolutional layers aim to extract visual features at different scales from a pixel and its neighbors. The deconvolutional layers aim to reconstruct the images based on the extracted features. To help the model, skip connections between the corresponding convolutional and deconvolutional layers are used to propagate the information between model levels. For this model, the idea is to replace the convolutional and pooling layers with transformers, as shown in Fig. 5, to extract contextual information from the whole image and avoid focusing on local convolutions.

The architecture is based on the Swin-Unet proposed by Cao et al. (2021) for image segmentation. The proposed architecture is a variation of a standard U-net, where transformer blocks are used in place of convolutional blocks for the compression and expansion phases in the U-model. In the case of the Swin-Unet model, the transformer blocks are designed to build hierarchical representations of features that are relevant to the image segmentation task; however, this approach is not well suited to our prediction task. The Swin block consists of two subblocks, a window-based multihead and a shifted window-based multihead, which increases the computational resources required for training, even if the self-attention mechanism is limited to small windows. In our proposed model, we retained the global structure of the network and the approach of merging and extending the patches, but we implemented these features using standard transformer blocks instead of Swin blocks. The patch merging process involves concatenating patches followed by a linear layer to reduce the dimension of the image and increase the feature dimension d_k by a factor of 2. In the expanding phase, the opposite operation is performed, increasing the image dimension while reducing the feature dimension d_k by a factor of 2. This modification enables us to retain the advantages of the U-net architecture while replacing the convolutional layers with transformer blocks.

It is worth mentioning that other strategies for merging transformer blocks and U-net models have been developed, such as the approach proposed by Chen et al. (2021), in which an attention mechanism is applied to the bottleneck to enhance the visual features, or the approach presented by Petit et al. (2021), which uses an attention mechanism only in the decoder.

4.2.2. Channel vision transformer

Our second approach that is a channel vision transformer (CViT). This approach aims to consider the individual image channels. Each variable is linked to either a train (passenger load or travel time) or a station (ticketing data or headway). The architecture is built to consider these characteristics and computes an independent attention score for each variable and each pixel (i.e., patch size of 1).

The model follows the idea of channelwise methodologies (Sheng et al., 2021; Chen et al., 2017). The inpainting function f is decomposed into a series of encoders for each channel and a single decoder to build the complete image. Each transformer encoder captures features and the traffic context for one variable in $I^{in}(t)$, as shown in Fig. 6. The number of encoders d is equal to the channel dimension. The results of each component are concatenated and passed to a decoder. The decoder reconstructs the image and performs inpainting based on the features extracted by the first part of the model, as shown in Fig. 6. The decoder uses a local inference function to predict the target value for one pixel in $I_y(t)$.

5. Experimental details

To evaluate the performance of our methods, we applied the methods to a real dataset collected from Line 9 in the Paris subway network. This metro line is operated by the Régie Autonome des Transports Parisiens (RATP) and connects the west and east sides of Paris. As shown in Fig. 7, metro line 9 includes 37 stations, making it the longest Paris metro line, and the line was used by 136 million passengers before the pandemic in 2019. The experiment aims to separately forecast passenger loads and the headway for one traffic direction from the first station, Pont de Sévres, to Mairie de Montreuil. As a reminder, the passenger load is the number of people on a train departing from one station, and the headway is the time gap between two trains at one station. We forecast

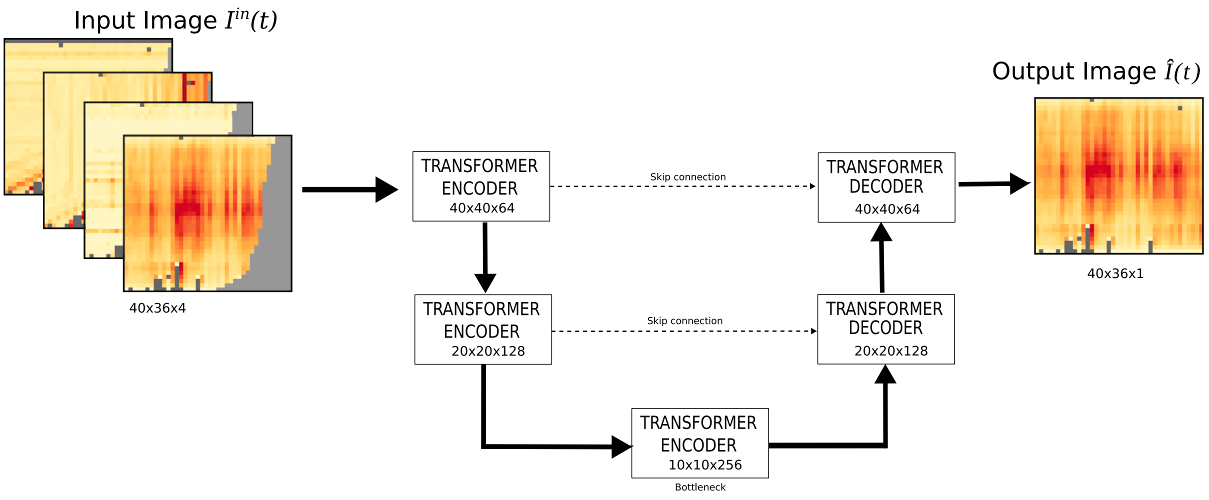

Input Image $I^{in}(t)$
40x36x4
TRANSFORMER ENCODER 40x40x64
Skip connection
TRANSFORMER DECODER 40x40x64
Output Image $\hat{I}(t)$
40x36x1
TRANSFORMER ENCODER 20x20x128
Skip connection
TRANSFORMER DECODER 20x20x128
TRANSFORMER ENCODER 10x10x256
Bottleneck

Fig. 5. Architecture of the U-Transformer model.

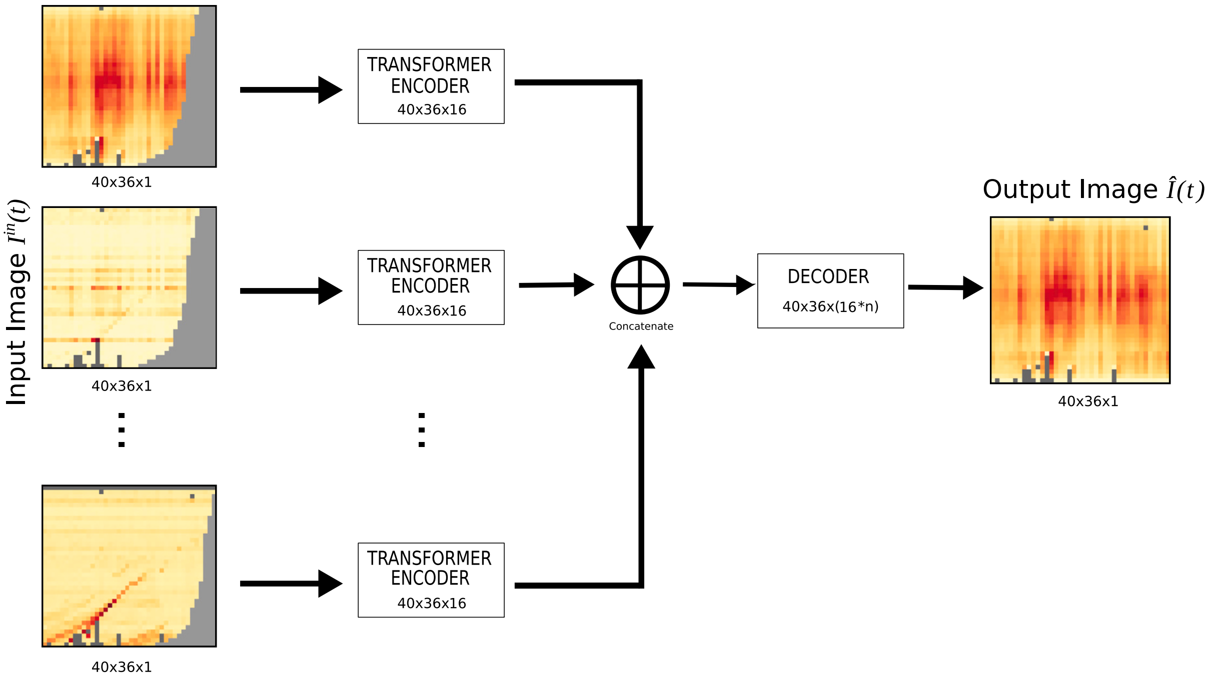

Input Image $I^{in}(t)$
40x36x1
40x36x1
40x36x1
TRANSFORMER ENCODER 40x36x16
TRANSFORMER ENCODER 40x36x16
TRANSFORMER ENCODER 40x36x16
Concatenate
DECODER 40x36x(16*n)
Output Image $\hat{I}(t)$
40x36x1

Fig. 6. Architecture of the Channel Vision Transformer model.

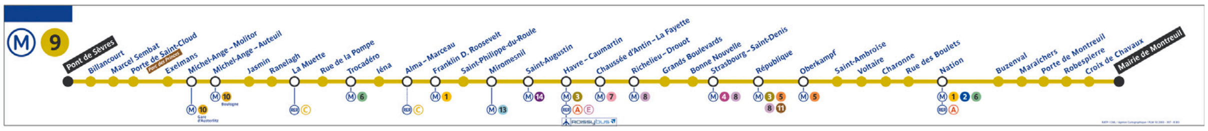
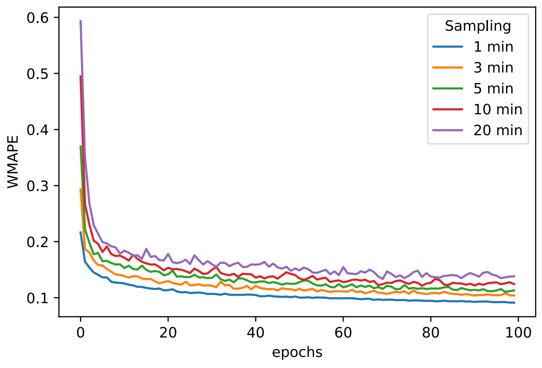


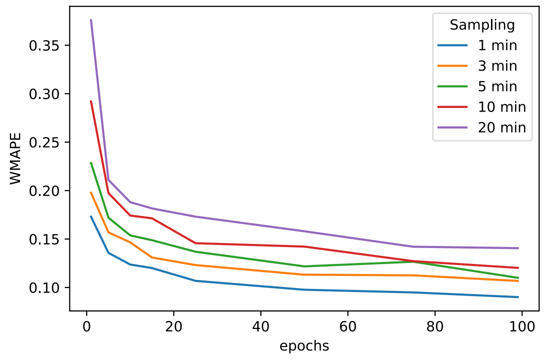
Fig. 7. Plan of metro line 9. Stations with information below the line represent a connection with other public transport lines.

the two variables with the same architecture. These two variables were chosen by the transport operator to focus on the passenger information.

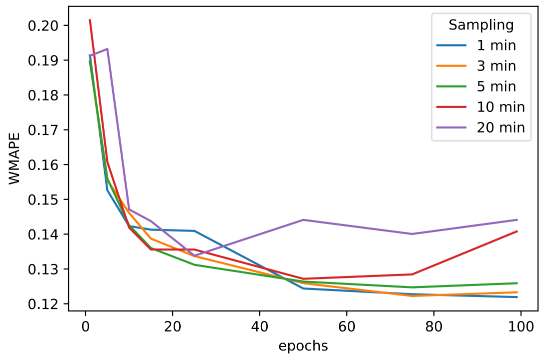
The forecasting process predicts the passenger load and headway for the stations' next four departures. We reduce the prediction horizon to the short term, although long-term forecasting can be considered in the inpainting task. The choice of four future stations



(a) Training dataset



(b) Test dataset (5 min sampling)



(c) Test dataset (1 min sampling)

Fig. 8. Evolution of the forecasting model errors at different sampling rates.

was determined by the metro supervision system used by the transport operator RATP, which provides four theoretical values of the headway for the station based on the train's position in the line.

5.1. Data and images

The dataset used in this study is composed of real data for metro line 9 collected from January 2019 to April 2021. An image is a snapshot of the considered network at the acquisition time t. The size of the image $I(t)$ is $40 \times 36 \times 4$, meaning that we consider 40 trains, 36 stations, and 4 data channels to generate an image. The choice of 40 trains was made based on the size of both the network and the departures to be predicted. It corresponds to the maximum number of trains that can be moving simultaneously along the line plus the number of departures to be forecasted from the first station in the line. Although the line is composed of 37 stations, we omit the last station because there are no train departures with passenger links to this station. The final dimension of the image is determined by the number of channels used as input variables for the prediction task. Four main variables are used: passenger train loads, ticket validations, headway, and travel times.

To create an effective image database, the sampling method is crucial since it affects both modeling and operational aspects. The sampling rate can be chosen based on whether the operator aims for short-term (every minute) or mid-term prediction (every 10 min) or according to the trains (one image per train departure). Several insights can be derived from the results presented in Fig. 8, which evaluates the performance of several U-net models with varying sampling rates. First, the models' performance improves significantly with a high sampling rate, even when the sampling rate of the test dataset differs. However, it is crucial to train the models with a sampling rate that is equal to or lower than that used to generate the dataset to prevent overfitting and achieve the best performance.

Furthermore, a high sampling rate introduces similarities for many distinct input images with the same target image in a close time range. The differences among these input images are minimal and involve only a few pixels since most of the trains have the same features. Thus, the images are generated every minute from the starting time of the public transport operation, 5:30 am, to the end of the operation at 1:30 am the following day, for a total of 20 h. Thus, the total number of images generated per day is 1200. More than 900,000 images were obtained from January 2019 to April 2021.

To evaluate our proposed approaches, we split the data into a training set and a test set to assess the model generalizability. To achieve a fair evaluation while reducing bias, the test set must meet some conditions to cover as many transport operational situations as possible. Therefore, the test set is composed of several months over the 3 years of data. It should be noted that several

Table 3
The settings of all the benchmark models.

Models (reference)	# Parameters	Encoding	Decoding	Batch size	Head	patch size
Naive	–	–	–	–	–	–
NN	27,354,528	24,403,968	2,950,560	128	–	–
CNN	20,254,465	11,173,632	9,080,833	128	–	–
CNN + NN (Ma et al., 2017)	25,418,272	22,467,712	2,950,560	128	–	–
Transformer (Vaswani et al., 2017)	16,616,200	25,960	16590240	8	16	1
VIT (Dosovitskiy et al., 2021)	232,449	232,400	49	8	64	3
U-net (Bapaume et al., 2021)	16,552,065	4,112,512	12,439,553	128	–	–
U-Transformer (ours)	1,911,985	618,128	1,293,857	8	64	3
Channel Vision Transformer (ours)	588,417	117,664	470,753	4	16	1

months with atypical scenarios, such as strikes, lockdown (i.e., due to the COVID-19 crisis), or holidays, are included in the test set. In addition, the data for the full month is included in the test set to ensure that the models do not learn any context or information from the images. This constraint was due to the image format, where overlap could occur between neighboring images during training and bias the outputs. Thus, over three years, the test set consists of 5 full months of images (2 for 2019, 2 for 2020, and 1 for 2021) and a total of approximately 220,000 images (i.e., around 25% of the images).

5.2. Benchmark models

We compared the two proposed models with other inpainting models adapted to our framework. The first model is a naive approach based on the last record at every station. The model completes each pixel by using the record of the current train (i.e., dynamic train approach) and the dynamics of the target variable at the station for the previous train (i.e., dynamic station approach). It is not relevant to attribute the first load to the second load in the prediction task. For example, consider two successive trains with loads of 200 and 600 passengers, where the load of the first train has increased by ten passengers compared to the load at the previous station. It is more relevant to add the variation in the number of passengers between the stations to the load of the next train.

[1] We also considered several deep learning approaches, including two conventional methods - a neural network (NN) and a fully convolutional neural network (CNN) - and four other methods - a convolutional neural network combined with a neural network (CNN+NN) (Ma et al., 2017), a U-net model (Bapaume et al., 2021), a transformer model (Vaswani et al., 2017), and a vision transformer (VIT) (Dosovitskiy et al., 2021). All deep learning models share several common hyperparameters. The first is the output function, which is the softplus function adapted to the targeted values. The passenger load and headway are defined as positive values between 0 and the maximum number of passengers or the largest headway (800 passengers in the case of the passenger load and approximately 1 h for the headway). The learning rate is set at 0.0001 and gradually decreased to 0.000001 during the 10 epochs.

The training is performed on an NVIDIA RTX 3090 graphics processing unit (GPU) with 24 GB of RAM using the TensorFlow framework and Python 3.8. The gradient descent method is based on the Adam optimizer (Kingma and Ba, 2014). Table 3 presents the settings of all nine models. It presents the number of parameters and the batch size used for training all the models. The transformer-based models have two additional columns, the number of heads and the patch size used to build the models. Due to the RAM usage required by the attention mechanism, there is a significant difference in the batch size and parameters between the transformer-based approaches and standard approaches. The quadratic computation and GPU memory cost of the attention mechanism impact the size of the transformer-based models compared to the size of the CNN or NN models. This constraint results in a trade-off between the patch size, the number of heads, the batch size, and the projection dimension. Thus, the number of heads and the projection dimension of the multihead attention layer were set to 16 and 64, respectively.

6. Results and discussion

In this section, we first introduce the measures used to evaluate the prediction errors. Then, we detail the global results of the proposed and benchmark models. To provide a detailed performance evaluation of the prediction results, we use the attention scores of the transformer models. Finally, we build multiple atypical datasets to evaluate the robustness of the proposed methodology.

[1] We can define the naive rule as follows: $y_{s,c} = x_{s-1,c} + \Delta x_{s,c-1}$ where $y_{s,c}$ represents the forecast value, $x_{s-1,c}$ denotes the current state (passenger load or headway) of the train at the last station, $\Delta x_{s,c-1}$ denotes the variation in the target value of the last train between the current and previous stations and c and s represent the trains and stations, respectively.

Table 4
Inpainting and forecasting results based on the weighted mean absolute percentage error (WMAPE) and root mean square error (RMSE).

Models (reference)	Inpainting		Forecasting	
	RMSE	WMAPE	RMSE	WMAPE
Naive	63	14.2	50	19.9
NN	60	22.4	48	18.9
CNN	41	12.3	33	13.1
CNN + NN (Ma et al., 2017)	44	15.9	36	14.8
Transformer (Vaswani et al., 2017).	30	8.3	43	18.2
VIT (Dosovitskiy et al., 2021)	42	12.9	33	12.4
U-net Bapaume et al. (2021)	**29**	**6.6**	31	12.3
U-Transformer (ours)	34	9.9	**31**	**11.2**
Channel Vision Transformer (ours)	32	9.2	32	12.1

6.1. Global forecasting results

We used two standard evaluation metrics, the root mean square error (RMSE) and weighted mean average error (WMAPE), which are defined as:

$$RMSE = \sqrt{\frac{1}{N}\Sigma_{i=1}^{N}\left(I(t) - \hat{I}(t)\right)^2} \tag{7}$$

The first indicator is based on the measure used during model training. It is a pixelwise indicator over an image. It leverages large errors over small errors, thus highlighting train departures with high passenger loads or long headway.

$$WMAPE : \frac{\sum |I(t) - \hat{I}(t)|}{\sum |I(t)|}. \tag{8}$$

The WMAPE metric offers a more interpretable indicator for the forecasting results and is expressed as a percentage. The impact of extreme values is reduced by scaling the error by the real value. Both metrics are employed to evaluate the two main steps, inpainting and forecasting.

Table 4 displays the passenger load forecasting results. The U-net model outperforms all other models in inpainting, with a difference of 3% in the WMAPE metric and 5 in the RMSE metric compared to the results of the second best model, the U-Transformer. The U-net model better predicts the loads in a long-term horizon (i.e., pixels not selected for prediction). Regarding the forecasting performance, the U-transformer presents the best results, with a slight difference of 1% in the WMAPE compared to the WMAPE values of the other best models, such as the U-net and Channel Vision Transformer. However, it should be noted that transformer models are trained less efficiently (i.e., cost and technique limitations) than U-net models. Nevertheless, two transformer models achieve good forecasting results. It is worth noting that the transformer model tended to overfit during inpainting (a WMAPE of 8.3%) and performed poorly for forecasting (a WMAPE of 18.2%).

Table 5 presents the headway forecasting results. The results highlight that headway forecasting is a more complex task than passenger load forecasting (e.g., regulation, interim train assignment). All models present WMAPE values that are at least two times greater than the corresponding WMAPE values in the load prediction task. U-net outperforms the benchmark models for both inpainting and forecasting. The WMAPE value for the U-net model differs from the WMAPE values of the VIT and Channel Vision Transformer models by 5% for inpainting. Moreover, the WMAPE value of the U-net model differs from that of the CNN model by 1.6% for forecasting. Although U-net outperforms all benchmark models for headway forecasting, the attention score used in the transformer-based models offers an explainable metric to interpret the results.

Tables 4 and 5 demonstrate that while the transformer approaches obtain improved forecasting results, they do not necessarily achieve significant gains. In fact, for headway forecasting, U-net outperforms the other models. These findings raise the question of whether these models are suitable for real-world applications. Deep learning models such as transformers are complex and require substantial resources for optimization. When applying these models, it is important to consider the trade-off between performance gains and resource requirements.

6.2. Attention score analysis

Transformer-based models provide attention scores that can improve the interpretability of our prediction models. The attention scores allow us to interpret the results of the prediction models and visualize the relationships between the pixels used to generate the forecast. This subsection focuses on the interpretability of the channel vision transformer model based on the attention scores computed with Eq. (4) for the 4 input variables. Note that this model was chosen instead of the U-transformer model because (i) it obtains similar results to the U-transformer model and has a patch encoding of size 1; (ii) it uses a channelwise approach that allows us to determine the attention score for each task variable. We compute attention scores for each pixel in $\hat{I}^p(t)$ and the heads from the transformer encoder. The attention score is a value included in the range $[0, 1]$ that denotes the relevance of the information between two pixels. High attention is denoted by a score close to 1 (i.e., represented in red in Fig. 9) and implies a high correlation between these two pixels, while a small attention score close to 0 denotes uncorrelated pixels.

Table 5
Headway inpainting and forecasting results based on the weighted mean absolute percentage error (WMAPE) and root mean square error (RMSE).

Models (reference)	Inpainting		Forecasting	
	RMSE	WMAPE	RMSE	WMAPE
Naive	28.56	53.1	21.91	40.9
NN	9.14	32.5	11.0	27.5
CNN	8.64	28.2	10.91	23.5
CNN + NN (Ma et al., 2017)	9.12	29.9	10.89	24.2
Transformer (Vaswani et al., 2017)	7.70	28.6	10.99	26.9
VIT (Dosovitskiy et al., 2021)	8.98	18.1	11.06	29.4
U-net Bapaume et al. (2021)	**4.47**	**13.4**	**10.90**	**21.9**
U-Transformer (ours)	8.24	30.8	10.99	29.2
Channel Vision Transformer (ours)	6.26	18.3	10.93	24.3

Fig. 9 presents the attention scores computed for one pixel (i.e., in green) in $\hat{I}^p(t)$ in relation to the other pixels. Fig. 9 (a) shows the location of the target pixel (green pixel) in $\hat{I}^p(t)$. We limited the presentation to only one pixel and the head with high attention scores (meaningful information) or correlations between the different pixels in the selected images.

Fig. 9 (b) depicts the attention scores measured between the green pixel and all the pixels for four heads in the transformer encoder for the passenger load channel. Fig. 9 (c) and (d) show the same representation for the travel time and headway channels. The head is a part of the architecture that captures a specific context. The head is activated if the situation that it aims to detect is available in a image or between two pixels. This behavior is illustrated in Fig. 9 (b). In the presented images, the attention scores based on the load channel in Fig. 9 (b) show that the transformer model focused on the most recent train departures (e.g., Head0) and nearby stations on the line. Moreover, the attention scores indicate which trains, in particular, are correlated with the current pixel, such as Head0 or Head4. In Head0, we observe that some attention is spatially uncorrelated with the nearby train departures. Some pixels at the end of the line have significant attention scores. The model looks for information that is either temporally and spatially close to the green pixel or train departures that occur further along the line. In addition, some heads (e.g., Head1 or Head5 in Fig. 9 (b) did not show any significant attention scores, meaning that there were no correlations between the pixels and the image that activated the corresponding head. For the travel time channel shown in Fig. 9 (c), the focus is more on the pixel's position with respect to future train departures than on past journeys or past departures. This approach helps the model obtain information about the traffic profile (i.e., the number of missing pixels can be informative about the timestamp of the image). For the headway channel (see Fig. 9 (d)), heads 1 and 5 show that the attention was mainly focused on past train departures to obtain the context of the metro traffic image. With the attention scores, we can observe and highlight the information from the other pixels used to perform the forecasting. We also investigated other representation learning methods, such as latent space, to better explain the forecasting results.

6.3. Forecasting results for atypical operation scenarios

A public transport operator builds a transport plan based on the demand estimated by historical observation studies. Passenger load or headway prediction can be trivial when there is no variation from the planned schedule (timetables). However, the train schedule can be uncertain and disturbed by atypical events (i.e., events not foreseen by the operator) or change depending on the operational traffic management of the metro line. Thus, we selected several real atypical scenarios based on three criteria: (i) subsequent knowledge of the metro traffic, (ii) descriptive statistics computed based on the images, and (iii) latent space clustering. The goal was to evaluate the robustness of the framework and the proposed forecasting models for the whole test set and specific test sets, including those with disturbances.

The first criterion is based on external data such as event databases and calendar information. Next, we aim to identify situations based on knowledge that is not connected with the training data. Two specific cases differing from the nominal cases were selected: *strikes* and *lockdowns*. The Paris Metropolis experienced a long transport strike in December 2019. The *lockdown* dataset is represented by images obtained during March 2020, when the metro line was impacted by the COVID-19 crisis.

The second criterion is based on descriptive statistics computed based on the images. The idea is to identify atypical train trips or situations (e.g., longer travel time, missing stop point, high passenger load) within the images. We plotted the distribution of images according to several criteria and selected the tails of each distribution. Fig. 10 presents an example of the distribution of images according to the total travel time criterion. The distribution is similar to a Gaussian mixture distribution, with the main component representing the nominal case and the minor components representing atypical cases.

With this approach, two test sets were selected to represent atypical images. For the first test set, which is denoted as *delay*, we aimed to obtain images with longer train travel times. The criterion is the maximum absolute difference between the theoretical total travel time and the real total travel time for a train in the image. The tail of the distribution represents trains with total travel times longer than 10 min compared to the mean value denoted by the red line in Fig. 10. For the second test case, which is denoted as *high load*, we considered images with higher passenger loads than in the normal situation, where the criterion is the mean load based on all images. An image is considered atypical if the mean load exceeds 300 passengers.

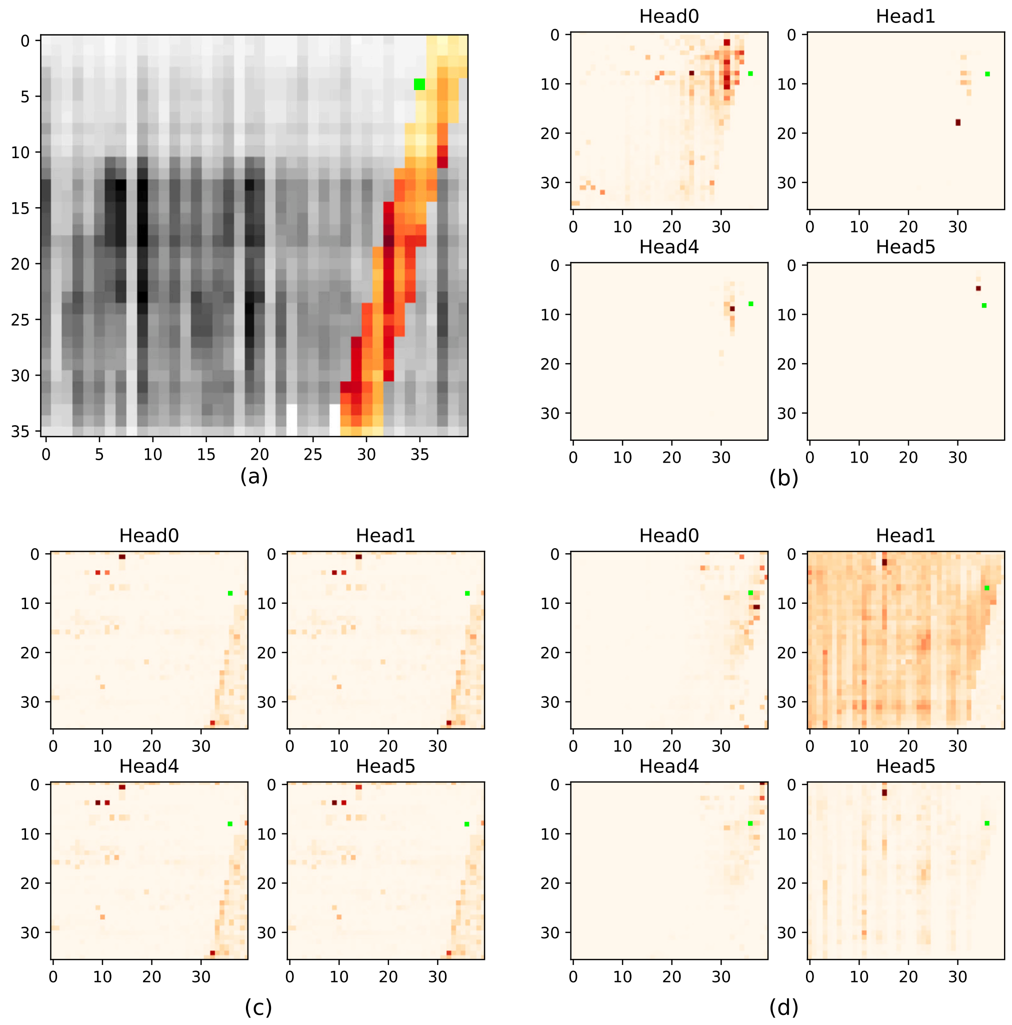


Fig. 9. Visualization of attention scores for a single pixel in 4 heads of the channel vision transformer model used for passenger load prediction: (a) Studied $I(t)$ with the target pixel; (b) attention scores from the passenger load channel; (c) attention scores from the travel time channel; (d) attention scores from the headway channel; [a deeper red color indicates a higher attention score.].

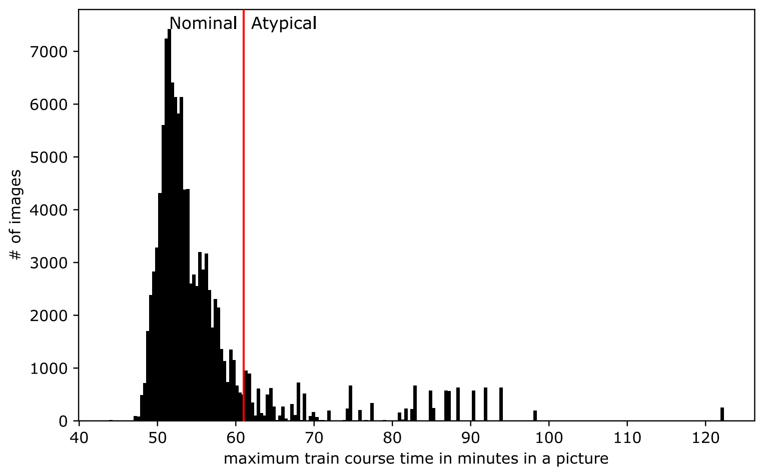


Fig. 10. Distribution of test set images according to the maximum total travel time of trains within the images. The red line represents a threshold between the atypical and nominal images.

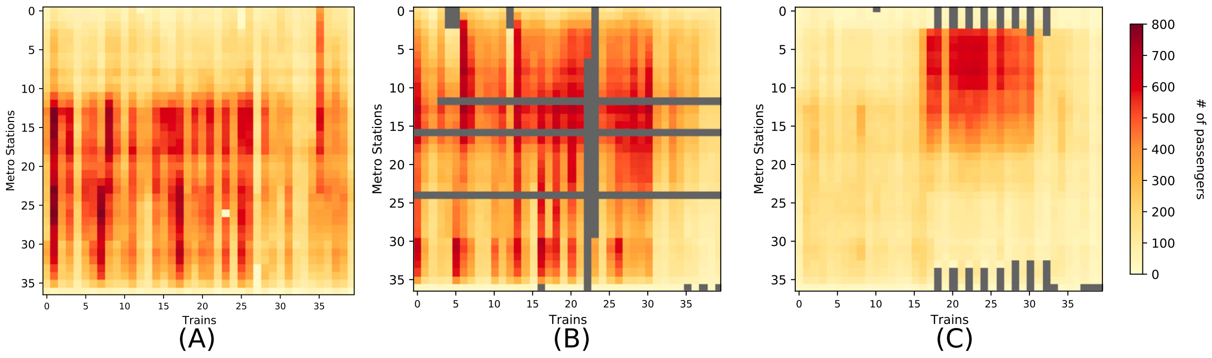


Fig. 11. Images in the selected atypical groups: (A) Cluster 3 represents the transfer of passengers from other lines to line 9; (B) cluster 12 represents disrupted situations and scenarios with closed stations; and (C) cluster 18 represents scenarios with sporting events, when high passenger loads move directly from the first station in the metro line.

Table 6
Passenger load forecasting results based on the weighted mean absolute percentage error (WMAPE) for the entire test set and the atypical situation test sets.

Models (reference)/WMAPE [%] for	Test set	Delay	High load	Strike	Lockdown	Transfer	Disruption	Event
Naive	19.9	27.3	20.1	38.2	21.1	28.9	**29.8**	22.3
NN	18.9	28.4	20.1	40.2	26.8	28.5	60.5	18.7
CNN	13.1	24.2	13.0	29.8	18.6	21.4	34.2	14.6
CNN + NN (Ma et al., 2017)	14.8	24.2	14.3	32.4	25.9	25.9	40.2	15.6
Transformer (Vaswani et al., 2017)	18.2	26.6	26.6	35.5	20.1	31.2	75.1	18.2
VIT-3 (Dosovitskiy et al., 2021)	12.3	20.3	12.1	31.1	18.2	20.9	52.8	13.1
U-net Bapaume et al. (2021)	12.3	19.3	11.6	**28.1**	16.8	19.5	30.4	13.9
U-Transformer (ours)	**11.4**	**19.1**	**11.2**	29.6	**13.7**	**18.4**	58.9	**12.5**
Channel Vision Transformer (ours)	12.1	19.9	12.0	30.0	14.2	19.5	51.6	13.1

Finally, the last criterion was derived from the latent space of the U-Transformer model for passenger load forecasting. A latent space (see Appendix A) represents a high-dimensional space where a machine learning model encodes the features of the input data. It is obtained from the bottleneck of the architecture, which is similar to an autoencoder. We apply a k-means clustering algorithm to partition the latent space into a reduced set of clusters, including some atypical groups of images (see the Appendix for more details). We chose 20 clusters according to operational considerations. The aim was to detect atypical clusters that were not extracted in an unsupervised manner using the first two criteria. We excluded the *lockdown* and *strike* clusters that were visible in this latent space. Finally, three clusters were selected, Cluster 3, Cluster 12, and Cluster 18, representing atypical situations from an operational point of view, accounting for approximately 10% of the data. Fig. 11 shows one image for each cluster. Cluster 3, which is denoted as *transfer*, corresponds to the transfer of passenger loads from other metro lines to the current line starting from the station Trocadéro (see Fig. 7). The passenger load increases each time the trains pass through a transfer station. Cluster 12, which is denoted as *disruption*, represents disrupted images or images with closed stations, while cluster 18, which is denoted as *events*, represents images with high loads at the beginning of the line (e.g., sporting events). Finally, the trained k-means algorithm was used with the images in the test set to extract atypical images from the selected clusters. In the following section, we analyze the images in Fig. 11. For image (a), the load increases significantly each time the train crosses a station linked to another metro line starting at station 11 (e.g., station Trocadéro, Alma or Havre–Caumartin). For the second image (b), we observe (i) three closed stations, as defined by the horizontal gray lines, and (ii) two atypical trains (numbers 22 and 23), the first of which stops earlier and the second of which starts from a station close to the end of the line. The last image (c) shows a specific group of trains used to manage high passenger loads moving from station 3 to the center of the line.

The latent space analysis is limited to the passenger load forecasting models because the equivalent latent space from the headway forecasting models does not highlight any relevant clusters except those related to the strike or lockdown periods. A statistical description is provided in Appendix B to summarize some characteristics of the atypical test set.

6.4. Results in atypical scenarios

Table 6 summarizes the forecasting results obtained from all the presented atypical groups and clusters. The prediction performance is represented with the WMAPE metric. The U-Transformer model presents the best results for most of the test cases except for two test sets, *disruption* and *strike*. For the *disruption* images, the naive approach obtains the best performance, with a WMAPE value of 29.8%, followed by U-net, whereas the transformers are outperformed, with WMAPE values higher than 50%. For the *strike* dataset, U-net presents the best performance, with a difference of 1% compared to the next two best models. These two traffic contexts have the worst results among all the atypical cases. They represent scenarios in which the traffic is highly impacted by external events such as closed stations, early stopping of trains, or traffic interruptions. Regarding the other atypical

Table 7
Headway forecasting results based on the weighted mean absolute percentage error (WMAPE) for the entire test set and the atypical situation datasets.

Models (reference)/WMAPE [%] for	Test set	Delay	High load	Strike	Lockdown	Transfer	Disruption	event
Naive	40.9	152	154	135	27	**18.6**	183	19.8
NN	27.5	63.9	61.2	57.6	22	25.7	79.3	20.2
CNN	23.5	61.9	58.5	55.0	21.0	21.4	**75.3**	15.6
CNN + NN (Ma et al., 2017)	24.2	59.9	57.6	53.1	19.0	23.0	76.7	17.3
Transformer (Vaswani et al., 2017)	26.9	70.7	69.3	63.1	21.0	21.0	95.4	18.8
VIT-3 (Dosovitskiy et al., 2021)	29.4	64.4	62.8	57.6	24.4	24.3	77.7	22.7
U-net Bapaume et al. (2021)	**21.9**	**59.4**	**57**	**52.**	**17.3**	18.8	76.0	**14.6**
U-Transformer (ours)	29.2	64.3	62.9	58.1	24.6	26.2	79.0	22.5
Channel Vision Transformer (ours)	24.3	61.4	59.8	56.8	19.5	19.6	75.6	15.9

scenarios, the *high load*, *lockdown*, and *event* sets are the best-predicted sets, with errors close to those obtained for the global test set. It is important to emphasize that two of these atypical cases present opposite traffic conditions, namely, *high load*, *lockdown*. The former includes images with overcrowded trains, whereas *lockdown* consists mainly of images of trains with light loads. Note that the prediction error does not increase proportionally with the number of passengers but depends more on the traffic context, similar to the previously mentioned *disruption* and *strike* sets. However, good forecasting results are obtained for atypical contexts such as the *event* images (i.e., sport events occurring close to several stations).

For the two remaining atypical situations, *delay* and *transfer*, the forecasting performance is worse than that for the global test set, with WMAPE values of 19.1% and 18.4% for the U-transformer model. For the *delay* set, the images are considered to contain perturbed traffic when regulations were employed to manage passenger flows. These situations are more difficult to forecast since they involve more complex traffic management, merging delays and high passenger loads. For the *transfer* images, the error can be explained by the metro line data: information on passenger transfer between two lines is not measured. For instance, none of the stations on metro line 9 provided data about passengers switching from one metro line to another for stations such as Trocadéro. Thus, a single-line image does not take passenger transfers from other lines into account. The results of these two sets inform us about two phenomena that can degrade the forecasting performance. Therefore, the performance in these specific cases can be used as performance criteria to evaluate models and their robustness. Adding other scenarios can enrich the image representations of the proposed atypical scenarios.

Regarding headway forecasting, we note two trends concerning the results in atypical situations (see Table 7). The models show poor predictive performance for unanticipated scenarios such as *delays*, *high loads*, *strikes*, and *disruptions*. For these scenarios, the transport operator performs real-time control maneuvers to manage trains and passenger flows. Within such contexts, forecasting the headway is equivalent to predicting the traffic management set up by the transport operator, which is a very challenging task. Therefore, the errors for all models are high, with WMAPE values greater than 50%. U-net still presents the best results, with WMAPE values of 59.4% for images with delayed trains, 57% for the high load test set, and 52% for the strike test set. For the remaining test cases, including *lockdown*, *transfer*, and *event*, the models present better results than the global performance. U-net presents the best results for *lockdown* and *event*, while the naive approach obtains a WMAPE of 18.6% for the *transfer* set. For these atypical scenarios related to adjusted supply for the *event* case or to a headway close to the theoretical transportation schedule such as in the *lockdown* and *transfer* cases, the results are even better than those of the global test case.

7. Conclusions

In this paper, we proposed a deep learning framework for the short-term prediction of train passenger loads and headway in an urban public transit network. The proposed model combines a vision transformer and inpainting approaches to forecast all the desired future train loads simultaneously (i.e., not recursively). Two new architectures (U-transformer and channel vision transformer) were proposed based on self-attention to reformulate and solve the forecasting task as an inpainting task. In addition, we used a real dataset covering 3 years of data for a Paris metro line to compare and evaluate the performance of the proposed models in a large number of metro traffic situations.

The benchmark results show the effectiveness of the proposed methods for passenger load forecasting. In addition, for headway forecasting, our previous U-net model is the most efficient approach. In addition, we identified 7 types of atypical situations in this study through statistical analysis and the latent space of the U-Transformer model. We evaluated the robustness of all the methods based on their forecasting performance in these atypical scenarios such as *lockdowns*, *strikes*, *high loads*, or *delays*.

The worst performance was obtained for highly disturbed traffic scenarios for load forecasting and images impacted by traffic management for headway forecasting. For load forecasting, our models present better results for most of the atypical scenarios. However, the U-net model still obtains the best performance in forecasting the headway in all atypical situations. The results show that the prediction challenge involves not only the model performance but also the integration of atypical situations in the model evaluation. The goal is to detect the weaknesses of the models in prediction tasks. Moreover, the results show that the most complex models, which are computationally expensive to train efficiently (i.e., memory usage increases with the number of heads, the projection dimension, and the patch size), do not always achieve better performance in challenging data situations.

The initial steps of labeling the images and evaluating the models are relevant to develop improved methods for effectively handling more challenging and underrepresented atypical situations in prediction tasks. With respect to the state-of-the-art methods for forecasting passenger flows and headway in public transport scenarios (see Tables 1 and 2), future work should focus on prediction in atypical operation situations, considering the transport network as a whole and emphasizing the interpretability of the forecasting models. Moreover, the proposed methodology can be extended in several directions. First, image representations can be used to identify atypical images over several years of data. Image clustering or labeling is a common task that can benefit from the computer vision toolbox. Then, it may be possible to develop a similar framework for anomaly detection (Wang et al., 2019) in transport networks or to improve the performance of the forecasting models in atypical cases by oversampling atypical images (i.e., avoiding underrepresentation of these scenarios in the training dataset). Another possible perspective for this work is to test our models based on public databases (public transport data without interval aggregation) in the framework of an open-source project that the 3 years of data used do not allow. Finally, we can exploit the sparsity of the images (i.e., many pixels in the image have *zero* values) to reduce the computational cost of the transformer.

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.

Data availability

The authors do not have permission to share data.

Acknowledgments

This research was not supported by any specific grants from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Latent space analysis

To explain the choice and results of the forecasting model, we explore the feature space that the model builds to represent all the input images. The U-Transformer architecture is similar to an autoencoder model, and the feature space, called the latent space, is represented by a bottleneck in the U-shaped architecture. We used t-distributed stochastic neighbor embedding (t-SNE) projection (Hinton and Roweis, 2002) to reduce the high-dimensional latent space to 2D to obtain an understandable visible representation. The t-SNE method was used only for visualization and has the advantage of maintaining the same distance between the 2D and the original high-dimensional space. Fig. A.12) presents the results of the t-SNE projection method for each year.

Fig. A.12 shows that the model identified the main component in the center of the figure and several small groups around the center. The main component represents the nominal case for transit images. The information for each year is shown in a different color. The surrounding groups represent atypical situations that do not match the nominal case, such as disruptions and strikes. The forecasting period included the COVID-19 crisis, which impacted mobility. We can observe 3 significant groups of images from 2020 and 2021 (highlighted by circles in Fig. A.12)). These groups represent the three French COVID-19 lockdowns. Although it is not shown in this visualization, time is the main characteristic of the latent space. Two images taken at the same time of day are close in the latent space. In contrast, an image from the morning and an image from the afternoon are distant in the latent space. Thus, as expected, the model forecasts loads according to calendar information, with a clear distinction between images of atypical situations and normal images.

Appendix B. Atypical case description

To provide some information about the atypical situations, we present statistical descriptions of the images according to 3 criteria: the load, the total travel time of the trains, and the number of missing pixels. Each criterion is represented by a table summarizing the following statistics: minimum, maximum, median, mean, and standard deviation. The first criterion (Table B.8) represents the load profile of the test set images. The second criterion (Table B.9) represents the travel times of the trains in different atypical situations. Finally, the number of missing pixels (Table B.10) provides information about the traffic context. The larger the number of missing pixels is, the larger the number of trains on the line, which represents the possible traffic context (e.g., peak hour, off-peak hour).

The first point to note is the maximum in Table B.8 and Table B.9. The fact that several test cases share the same value signifies that the same image can include several situations. In addition, we can see the impact of the COVID crisis on the passenger load. The passenger load decreases significantly to a mean of 95 ± 50 passengers per train compared to the passenger loads in the other atypical test cases, where the average is above 200 passengers.

For the total travel time of the trains, Table B.9 shows that images with high delays or high passenger loads tend to have longer travel times. Strike images have high standard deviations and the lowest mean or median travel times, which reflects the chaotic behavior of the traffic during these events. The *lockdown* test case presents the lowest standard deviation, reflecting the regularity of the trains during periods with small passenger loads.

Table B.10 shows that high load and strike images present higher mean and median numbers of missing pixels, with values of 499 ± 128 and 498 ± 290, respectively. High load situations occur mainly during peak hours. The strike situation defines an atypical context for the line. During strike events, many stations are closed (i.e., up to 15 stations in the worst scenario), introducing a large number of missing pixels. As indicated in Table B.10, some images do not present any forecasting pixels. These are images acquired at the end of the day when the trains had ceased running at the time of data acquisition. In the same way, the cluster 3 test case presents a low mean number of missing pixels.

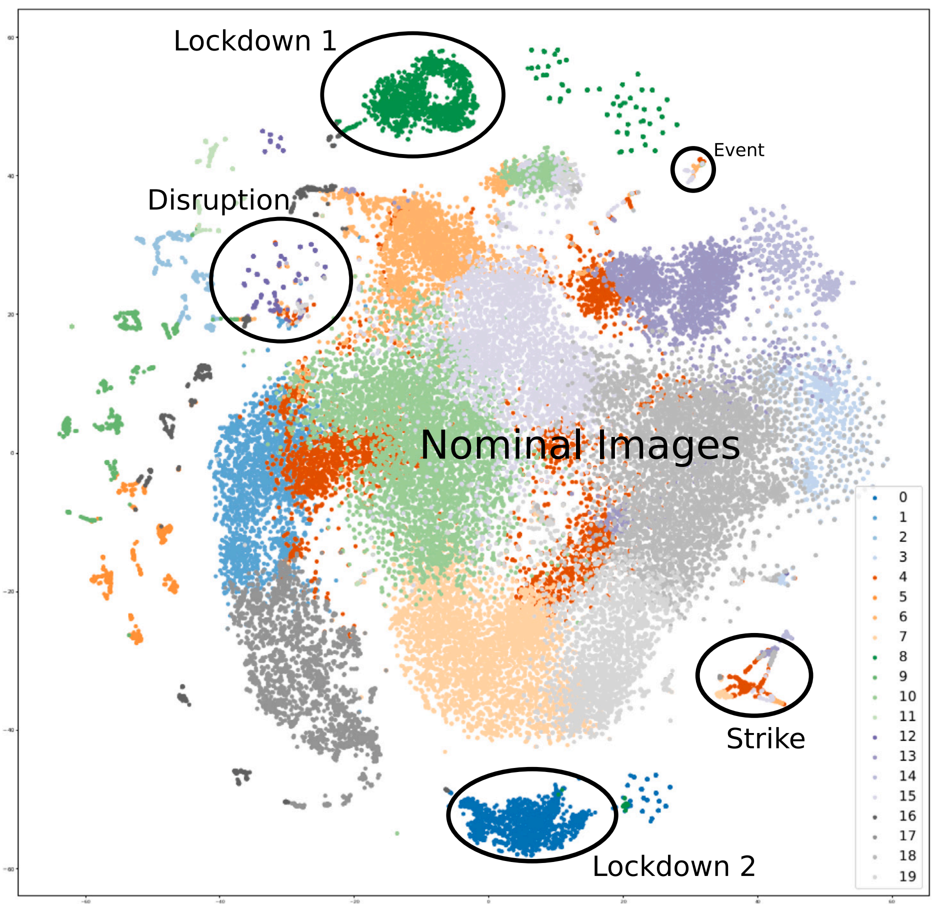


Fig. A.12. t-SNE representation of the latent space of the passenger load corresponding to the U-Transformer model. Different colors distinguish the 20 classes obtained from K-means clustering.

Table B.8
Statistical description of all datasets concerning the load per train within the image.

Atypical set	# Images	%	Min	Max	Median	Mean	Var
Test set	235200	100	1	961	134	162	116
High load	14,676	6.2	1	961	286	305	161
Delay	10,295	4.3	1	961	235	273	183
Strike	13,200	5.6	3	958	188	211	140
Lockdown	34,800	14.7	1	422	89	95	50
Transfer	12,635	5.6	2	958	119	139	94
Disruption	10,290	4.5	3	961	250	292	200
Event	11,230	5.0	2	958	160	180.3	109

Table B.9
Statistical description of the total travel time of the trains in the images in all datasets (in minutes).

Atypical set	# Images	%	Min	Max	Median	Mean	Var
Test set	235200	100	4.4	353	48.7	48.3	7.9
High load	14,676	6.2	4.6	353	52	52	11
Delay	10,295	4.3	5.3	353	50	54	19
Strike	13,200	5.6	4.8	353	47	38	22.6
Lockdown	34,800	14.7	4.6	74	47	47	4.6
Transfer	12,635	5.6	4.7	73	46.7	46.9	4.21
Disruption	10,290	4.5	4.7	94.0	49.8	50.4	8.8
Event	11,230	5.0	5.3	122.3	47.9	48.6	4.7

Table B.10
Statistical description of the number of missing pixels in the images in all datasets.

Atypical set	# Images	%	Min	Max	Median	Mean	Var
Test set	235200	100	0	1440	396	415	183
High load	14,676	6.2	132	1079	503	499	128
Delay	10,295	4.3	0	1273	382	406	206
Strike	13,200	5.6	0	1182	444	498	290
Lockdown	34,800	14.7	0	1067	493	494	156
Transfer	12,635	5.6	0	308	69	75	57
Disruption	10,290	4.5	0	606	150	183	132
Event	11,230	5.0	128	529	304	305	40

References

An, Q., Fu, X., Huang, D., Cheng, Q., Liu, Z., 2020. Analysis of adding-runs strategy for peak-hour regular bus services. Transp. Res. E 143, 102100, URL https://www.sciencedirect.com/science/article/pii/S1366554520307493.

Bapaume, T., Côme, E., Roos, J., Ameli, M., Oukhellou, L., 2021. Image inpainting and deep learning to forecast short-term train loads. IEEE Access 9, 98506–98522.

Beal, J., Kim, E., Tzeng, E., Park, D.H., Zhai, A., Kislyuk, D., 2020. Toward transformer-based object detection. arXiv preprint arXiv:2012.09958.

Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C., 2000. Image inpainting. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques. pp. 417–424.

Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., Wang, M., 2021. Swin-unet: Unet-like pure transformer for medical image segmentation. arXiv preprint arXiv:2105.05537.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers. In: European Conference on Computer Vision. Springer, pp. 213–229.

Chen, X., Cheng, Z., Jin, J.G., Trepanier, M., Sun, L., 2022a. Probabilistic forecasting of bus travel time with a Bayesian Gaussian mixture model. http://dx.doi.org/10.48550/ARXIV.2206.06915, arXiv URL https://arxiv.org/abs/2206.06915.

Chen, C., Liu, Y., Chen, L., Zhang, C., 2022b. Bidirectional spatial-temporal adaptive transformer for urban traffic flow forecasting. IEEE Trans. Neural Netw. Learn. Syst..

Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A.L., Zhou, Y., 2021. Transunet: Transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306.

Chen, L., Shi, P., Li, G., Qi, T., 2022c. Traffic flow prediction using multi-view graph convolution and masked attention mechanism. Comput. Commun. 194, 446–457, URL https://www.sciencedirect.com/science/article/pii/S0140366422003164.

Chen, L., Zhang, H., Xiao, J., Nie, L., Shao, J., Liu, W., Chua, T.-S., 2017. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5659–5667.

Colace, F., Santo, M.D., Lombardi, M., Pascale, F., Santaniello, D., Tucker, A., 2020. A multilevel graph approach for predicting bicycle usage in London area. In: Fourth International Congress on Information and Communication Technology. Springer, pp. 353–362.

Comi, A., Zhuk, M., Kovalyshyn, V., Hilevych, V., 2020. Investigating bus travel time and predictive models: a time series-based approach. Transp. Res. Procedia 45, 692–699, Transport Infrastructure and systems in a changing world. Towards a more sustainable, reliable and smarter mobility.TIS Roma 2019 Conference Proceedings, URL https://www.sciencedirect.com/science/article/pii/S2352146520301599.

Coulaud, R., Keribin, C., Stoltz, G., 2022. One-station-ahead forecasting of dwell time, arrival delay and passenger flows on trains equipped with automatic passenger counting (APC) device. In: WCRR 2022 - World Congress on Railway Research. p. xx.

Cui, Z., Henrickson, K., Ke, R., Wang, Y., 2020. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. IEEE Trans. Intell. Transp. Syst. 21 (11), 4883–4894.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. http://dx.doi.org/10.48550/ARXIV.1810.04805, arXiv URL https://arxiv.org/abs/1810.04805.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16 × 16 words: Transformers for image recognition at scale. arXiv:2010.11929.

Du, B., Peng, H., Wang, S., Bhuiyan, M.Z.A., Wang, L., Gong, Q., Liu, L., Li, J., 2019. Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction. IEEE Trans. Intell. Transp. Syst. 21 (3), 972–985.

Egu, O., Bonnel, P., 2021. Medium-term public transit route ridership forecasting: What, how and why? A case study in Lyon. Transp. Policy 105.

Gao, Y., Zhou, M., Metaxas, D.N., 2021. Utnet: a hybrid transformer architecture for medical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 61–71.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial networks. Adv. Neural Inf. Process. Syst. 3.

Hao, S., Lee, D.-H., Zhao, D., 2019. Sequence to sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system. Transp. Res. C 107, 287–300.

Heydenrijk-Ottens, L., Degeler, V., Luo, D., Van Oort, N., Van Lint, H., 2018. Supervised learning: Predicting passenger load in public transport. In: CASPT Conference on Advanced Systems in Public Transport and TransitData. pp. 30–32.

Hinton, G.E., Roweis, S., 2002. Stochastic neighbor embedding. Adv. Neural Inf. Process. Syst. 15.

Hochreiter, S., 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 6, 107–116.

Khan, S., Naseer, M., Hayat, M., Zamir, S.W., Khan, F.S., Shah, M., 2021. Transformers in vision: A survey. ACM Comput. Surv.

Kim, Y.J., Choi, S., Briceno, S., Mavris, D., 2016. A deep learning approach to flight delay prediction. In: 2016 IEEE/AIAA 35th Digital Avionics Systems Conference. DASC, IEEE, pp. 1–6.

Kingma, D., Ba, J., 2014. Adam: A method for stochastic optimization. In: International Conference on Learning Representations.

Lee, K., Rhee, W., 2022. DDP-GCN: Multi-graph convolutional network for spatiotemporal traffic forecasting. Transp. Res. C 134, 103466, URL https://www.sciencedirect.com/science/article/pii/S0968090X21004538.

Li, C., Bai, L., Liu, W., Yao, L., Waller, S.T., 2020. Graph neural network for robust public transit demand prediction. IEEE Trans. Intell. Transp. Syst.

Lin, C.-H., Yumer, E., Wang, O., Shechtman, E., Lucey, S., 2018. St-gan: Spatial transformer generative adversarial networks for image compositing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9455–9464.

Liu, Y., Dutta, S., Kong, A.W.-K., Yeo, C.K., 2022. An image inpainting approach to short-term load forecasting. IEEE Trans. Power Syst.

Liu, X., Gherbi, A., Li, W., Cheriet, M., 2019. Multi features and multi-time steps LSTM based methodology for bike sharing availability prediction. Procedia Comput. Sci. 155, 394–401, The 16th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2019), The 14th International Conference on Future Networks and Communications (FNC-2019), The 9th International Conference on Sustainable Energy Information Technology, URL https://www.sciencedirect.com/science/article/pii/S187705091930969X.

Liu, L., Hamilton, W., Long, G., Jiang, J., Larochelle, H., 2020. A universal representation transformer layer for few-shot image classification. arXiv preprint arXiv:2006.11702.

Liyanage, S., Abduljabbar, R., Dia, H., Tsai, P.-W., 2022. AI-based neural network models for bus passenger demand forecasting using smart card data. J. Urban Manag. URL https://www.sciencedirect.com/science/article/pii/S2226585622000280.

Ma, J., Chan, J., Rajasegarar, S., Leckie, C., 2022. Multi-attention graph neural networks for city-wide bus travel time estimation using limited data. Expert Syst. Appl. 202, 117057, URL https://www.sciencedirect.com/science/article/pii/S0957417422004717.

Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., Wang, Y., 2017. Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. Sensors 17, 818.

Marchev, Jr., A., Haralampiev, K., Lomev, B., 2022. Predicting travel times for on-demand public transport in sofia. IFAC-PapersOnLine 55 (11), 161–166, IFAC Workshop on Control for Smart Cities CSC 2022, URL https://www.sciencedirect.com/science/article/pii/S2405896322011569.

Meng, M., Toan, T.D., Wong, Y.D., Lam, S.H., 2022. Short-term travel-time prediction using support vector machine and nearest neighbor method. Transp. Res. Rec. 2676 (6), 353–365.

Pasini, K., Khouadjia, M., Same, A., Ganansia, F., Oukhellou, L., 2020. LSTM encoder-predictor for short-term train load forecasting. pp. 535–551. http://dx.doi.org/10.1007/978-3-030-46133-1_32.

Peng, H., Wang, H., Du, B., Bhuiyan, M.Z.A., Ma, H., Liu, J., Wang, L., Yang, Z., Du, L., Wang, S., et al., 2020. Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting. Inform. Sci. 521, 277–290.

Petersen, N.C., Rodrigues, F., Pereira, F.C., 2019. Multi-output bus travel time prediction with convolutional LSTM neural network. Expert Syst. Appl. 120, 426–435, URL https://www.sciencedirect.com/science/article/pii/S0957417418307486.

Petit, O., Thome, N., Rambour, C., Themyr, L., Collins, T., Soler, L., 2021. U-net transformer: Self and cross attention for medical image segmentation. In: International Workshop on Machine Learning in Medical Imaging. Springer, pp. 267–276.

Ricard, L., Desaulniers, G., Lodi, A., Rousseau, L.-M., 2022. Predicting the probability distribution of bus travel time to measure the reliability of public transport services. Transp. Res. C 138, 103619, URL https://www.sciencedirect.com/science/article/pii/S0968090X2200064X.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: Convolutional networks for biomedical image segmentation. arXiv:1505.04597.

Roos, J., Bonnevay, S., Gavin, G., 2016. Short-term urban rail passenger flow forecasting: A dynamic Bayesian network approach. In: 2016 15th IEEE International Conference on Machine Learning and Applications. ICMLA, pp. 1034–1039.

Serin, F., Alisan, Y., Kece, A., 2021. Hybrid time series forecasting methods for travel time prediction. Physica A 579, 126134, URL https://www.sciencedirect.com/science/article/pii/S0378437121004076.

Sheng, H., Cai, S., Liu, Y., Deng, B., Huang, J., Hua, X.-S., Zhao, M.-J., 2021. Improving 3d object detection with channel-wise transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2743–2752.

Toqué, F., Côme, E., El Mahrsi, M.K., Oukhellou, L., 2016. Forecasting dynamic public transport origin-destination matrices with long-short term memory recurrent neural networks. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems. ITSC, pp. 1071–1076.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. arXiv:1706.03762.

Wang, X., Fagette, A., Sartelet, P., Sun, L., 2019. A probabilistic tensor factorization approach to detect anomalies in spatiotemporal traffic activities. In: 2019 IEEE Intelligent Transportation Systems Conference. ITSC, pp. 1658–1663.

Wang, X., Guan, X., Cao, J., Zhang, N., Wu, H., 2020. Forecast network-wide traffic states for multiple steps ahead: A deep learning approach considering dynamic non-local spatial correlation and non-stationary temporal dependency. Transp. Res. C 119, 102763, URL https://www.sciencedirect.com/science/article/pii/S0968090X20306756.

Wang, D., Yang, Y., Ning, S., 2018. DeepSTCL: A deep spatio-temporal ConvLSTM for travel demand prediction. In: 2018 International Joint Conference on Neural Networks. IJCNN, pp. 1–8.

Wu, J., Wu, Q., Shen, J., Cai, C., 2020a. Towards attention-based convolutional long short-term memory for travel time prediction of bus journeys. Sensors 20 (12), 3354.

Wu, W., Xia, Y., Jin, W., 2020b. Predicting bus passenger flow and prioritizing influential factors using multi-source data: Scaled stacking gradient boosting decision trees. IEEE Trans. Intell. Transp. Syst. 22 (4), 2510–2523.

Yu, Y., Jiang, T., Gao, J., Guan, H., Li, D., Gao, S., Tang, E., Wang, W., Tang, P., Li, J., 2022. CapViT: Cross-context capsule vision transformers for land cover classification with airborne multispectral LiDAR data. Int. J. Appl. Earth Obs. Geoinf. 111, 102837, URL https://www.sciencedirect.com/science/article/pii/S1569843222000395.

Zhai, X., Kolesnikov, A., Houlsby, N., Beyer, L., 2022. Scaling vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12104–12113.

Zhao, Z., Chen, W., Wu, X., Chen, P., Liu, J., 2017. LSTM network: a deep learning approach for short-term traffic forecast. Iet Intell. Transp. Syst. 11, 68–75.

Zheng, Y., Wang, S., Dong, C., Li, W., Zheng, W., Yu, J., 2022. Urban road traffic flow prediction: A graph convolutional network embedded with wavelet decomposition and attention mechanism. Physica A 608, 128274, URL https://www.sciencedirect.com/science/article/pii/S0378437122008329.