Image inpainting and Deep Learning to forecast short-term train loads

Thomas Bapaume, Etienne Côme, Jérémy Roos, Mostafa Ameli, Latifa Oukhellou

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 20, 2021, accepted June 14, 2021, date of publication July 1, 2021, date of current version July 16, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3093987

Image Inpainting and Deep Learning to Forecast Short-Term Train Loads

THOMAS BAPAUME[1,2], ETIENNE CÔME[1], JÉRÉMY ROOS[2], MOSTAFA AMELI[1], AND LATIFA OUKHELLOU[1]

[1]COSYS-GRETTIA, Université Gustave Eiffel, 77454 Champs-sur-Marne, France
[2]Régie Autonome des Trasports Parisien (RATP), 75012 Paris, France

Corresponding author: Thomas Bapaume (thomas.bapaume@univ-eiffel.fr)

This work was supported by RATP.

ABSTRACT Developing an efficient short-term prediction framework for public transportation systems is of fundamental importance. This paper proposes a new image-processing-oriented methodology for the short-term prediction of train loads. First, we introduce a novel approach for representing the metro traffic by generating an image, exhibiting the spatial information of the trains running on a metro line while taking into account the irregular temporal sampling of the train loads. Second, we propose a prediction framework using deep learning methods. In particular, we build a U-net convolutional neural network, consisting of Inpainting and image-to-image translation mechanisms. We construct an image of the load predictions for different trains and stations. The framework performs a multi-step forecasting task for each station at any given time. The proposed prediction model is capable of making a global prediction for several departures on a whole metro line. Third, we benchmark our model against other prediction models using real load data collected over ten months on a Paris metro line. The comparison shows that the proposed framework is efficient compared to standard methods in image-processing prediction models. Finally, we evaluate the performance of the model in atypical operating situations (e.g., strike, incident). The results show that the performance of the model remains at acceptable levels of prediction errors in the event of metro traffic disruptions.

INDEX TERMS Short-term forecasting, U-net, deep learning, inpainting, image-to-image regression, public transport, train load, forecasting.

I. INTRODUCTION

Passenger flow prediction is vital for public transportation planners and operators. It allows them to anticipate the demand variations in order to optimize the service levels and operations in normal or disrupted situations [1]. The demand information is also crucial for travelers because having robust information (e.g., train loads) can help them optimize their travel plan, especially in dense urban areas with a congested transportation network. In this context, short-term forecasting of the train loads over an entire metro line is a relevant tool to foresee all flows within a transit network. The idea is to decompose the forecasting task of metro load for each train and station of one line. In addition, this decomposition allows us to consider multiple situations of a metro line and anticipate the future loads for several trains. In particular, we aim to introduce a robust prediction model to forecast the loads of all the following trains located within 10 to 20 minutes after a train's departure.

Practical prediction models have to take into account several implementation constraints in a large-scale urban transportation network. For instance, the structure of the prediction model should consider the characteristics of the data life cycle (e.g., data frequency, collection lag) while modeling the problem. In classical prediction models, the data collectors are based on regular time series and usually monitor particular locations of the urban area. This study is not limited to the representation and prediction of the spatio-temporal time series, since the data collectors, i.e. trains in the present study, are moving simultaneously in a univariate space along a metro line. The train load measurements are carried out during the train stops. Therefore, the data are

The associate editor coordinating the review of this manuscript and approving it for publication was Jiankang Zhang.

collected with irregular temporal sampling following the trains' movements.

In addition to the above-mentioned temporal and spatial constraints, we have to take into account technical constraints with respect to data accessibility and availability. Here, we built our model based on the data collected by the French Ile-de-France operator RATP (Régie Autonome des Transports Parisiens). For the purpose of short-term forecasting, we need to consider the availability of data to simulate the real state of prediction. In this case, train load data is not available on a short-term scale due to technical constraints (e.g., late data collection, several days of processing). So, the prediction model is built around train load correlated variables (e.g., metro tickets, waiting times) that are available in a limited time frame, less than 10 minutes. In other words, our prediction model infers the train load from correlated variables on short-term horizons without considering the short-term target dynamics.

In this context, the main challenge is to propose a formalization method to consider all the spatial, temporal, and the above-mentioned technical constraints without any assumptions for oversimplification or aggregation. The ultimate goal is to develop a framework to predict the train load for the next multiple departures for every station of a metro line, considering the following aspects:

- Univariate space: the predictor performs simultaneously on several metro stations.
- Irregular temporal sampling: the predictor is able to predict the train load for the next departures, while the train load can be potentially collected in different time steps.
- Data availability in real-time: the predictor considers the co-variables available for the prediction, excluding a recent history of the target variable.

This paper proposes a novel short-term image representation that encodes all the knowledge of a train departure in a pixel in order to simultaneously consider the spatial aspect of a transport network, the time constraints due to the transport schedule, and the data availability in real-time for a large transport infrastructure. Past and future data are included in this image, allowing us to perform the short-term prediction directly inside the image. Thus, we define the forecasting task as an image-to-image regression using image-oriented models. We validate the effectiveness of the proposed framework on a real test case of the Paris metro. Note that the efficiency of the model is directly related to the strength of the image processing models (Convolutional framework) used to extract local features of metro traffic.

The rest of the paper is organized as follows. Section II introduces the different regression and prediction methodologies reported in the state of the art and the contributions as well as the position of proposed methodology within the image processing and forecasting methods. Section III details the variables of the framework through an illustration of metro traffic. Section IV presents an image-like representation of the metro traffic. Then, section V introduces the image-oriented frameworks used to realize an image-to-image regression. Sections VI and VII present and discuss all the experiments and results carried out on real data collected on the Paris metro lines. Section VII section also analyzes the prediction results for train loads on test data set and some atypical situations. Finally, we provide concluding remarks and present future work in SectionVIII.

II. LITERATURE REVIEW AND CONTRIBUTIONS

A. LITERATURE REVIEW

In the literature, numerous studies have focused on predicting train loads, particularly in public transport. Most of these studies are based on time series prediction, in which the flows are aggregated according to a regular time sampling. They have been grouped here into four categories. The first three discuss the aspects previously introduced and existing methods to address them. The last part focuses on image processing frameworks to accomplish forecasting and the image processing methods used in this study.

1) UNIVARIATE SPACE

Most metro load forecasting studies employ autoregressive or deep learning methods to forecast a target variable for a single station [2]–[6]. Autoregressive methods (e.g., Auto-Regressive Integrated Mean Average, (ARIMA)) propose a linear response to a forecasting problem. [3] proposed a forecasting framework for passenger demand. The framework was set up through automated Box-Jenkins tools (based on the seminal study of [2]) tested on the Istanbul metro. Other regressors such as support vector machine are also applied to short-term prediction problems [7]. To deal with non-linear characteristics, [6] combined ARIMA with neural networks (NN) in order to manage both linear and non-linear variables in forecasting problems. [8] achieved better forecasting results by merging two models, Kalman filtering and the k-nearest neighbor (KNN).

Deep learning methods are also widely used in train load forecasting because of the capacities of neural network approaches to deal with non-linear characteristics. For example, [9] deployed an Artificial Neural Network (ANN) at the station level. They estimated the evolution of metro load in Naples based on passenger inflow data. [10] coupled ANN with time series analysis and forecasting methods to achieve more robust prediction. They combined a singular Spectral Analysis with ANN to forecast the total load of the Moscow metro. However, none of the above-mentioned studies take the spatial aspects of the data into account. In other words, they are limited only to predict the flow at a single station. They can only address multiple stations (a metro line) by considering one model per station, not one global model. This study proposes a global framework to address the spatial aspect of the whole metro line at once, wherein the short-term task is done on all elements of the network.

2) IRREGULAR TEMPORAL SAMPLING

In recent years, recurrent neural networks (RNN), in particular LSTM (introduced by [11]), have shown their efficiency in load prediction compared to standard neural networks by managing temporal dependencies [4], [5], [12], [13]. LSTM architectures are powerful in dealing with non-linear temporal dependencies in data. However, to address spatial dependencies, these models are usually combined with a spatial component to extract features from a network. Thus hybrid models [4], [14], [15] divided the task into a temporal brick, most of the time an LSTM, and a spatial brick. The spatial brick can take many forms. For instance, a two-dimensional network [4] can consider spatio-temporal correlation in the traffic network. For multi-lane short-term traffic prediction, [14] replaced a standard LSTM by a convolutional LSTM to address large transport networks. Attention mechanisms (popularized by [16]) can be used in one-dimensional convolutional neural networks (1D CNN) and Gated Recurrent Units (GRU) to improve the generalization capacity of the models [15]. These studies are therefore built on regular time sampling, ignoring the real transport schedule. They forecast aggregated outflows and inflows over a fixed time interval for stations or metro lines. Formalizing irregular temporal sampling of time series involves temporal variability, making it difficult to apply techniques that usually exploit the structural regularity of time series [12] such as previous LSTM or ARIMA frameworks. To resolve this constraint, [12] used an encoder-predictor to take into account the transport schedule in their LSTM train load predictor. In this paper, we consider the data's temporal irregularity as an optimal representation of public transport. Moreover, these methods always use dynamic targets as inputs, which is not always the case. That is why it is crucial to consider the availability of the data.

3) DATA AVAILABILITY IN REAL-TIME

In practice, the accessibility and availability of data influence data-driven prediction models. The level of access depends on the transport network considered. Older metro lines such as Paris may be more challenging to predict as there are constraints on data collection and on the short-term exploitation of models. Forecasting frameworks have to consider the availability constraints derived from data gathering (i.e., access delay, collection latency) and missing data. Many studies have been developed with spatio-temporal data from multiple sources and collection steps without considering the differences in data access. [17] proposed a specific model called DDP-GCN (Distance, Direction, and Positional relationship Graph Convolutional Network) in order to incorporate three spatial relationships into the prediction network for traffic forecasting. [18] used graph convolutions to capture spatial patterns and common convolution for temporal features with an attention-based spatio-temporal graph Convolutional Neural Network (CNN). [19] merged time and space into an image in order to forecast congestion and speed on the road by a simple CNN. [13] merged LSTM and CNN to compute spatial and exogenous dependencies at the same time for passenger demand forecasting. The Deep Spatio-temporal model (ConvLSTM) was used by [20] to capture temporal and spatial dependencies simultaneously. [21] performed passenger flow prediction on a single train line in Paris through spatio-temporal neighborhoods and dynamic Bayesian networks to deal with missing data. These models do not mention the delay or latency of data collected from all the sources equivalent to short-term data availability. By representing all the short-term information within a picture, data availability is encoded according to the dimensions and shape of the images generated.

4) IMAGE PROCESSING IN FORECASTING

Other related work involves using images as information carriers. Several prediction methods based on converting multivariate or univariate time series into images have been proposed in the literature. [19] proposed to construct a time-space image of the traffic road to predict all the future evolution of the traffic. [22] aimed to create an image from geospatial data to support traffic prediction. There are also more theoretical approaches such as [23] in which an effective method based on the image-to-image regression for cyclic data was proposed. We can also mention [24] where the authors tackled the forecasting task by Image Inpainting (FM2I). The aim was to decompose a time series in the form of an image in order to complete the image and extract predictions from it. Inpainting methods [25], [26] are used to reconstruct deteriorated images or images with missing parts. Moreover, images grant us access to all the tools developed in image processing, inpainting or the image-to-image translation [27] and feature visualization [28].

B. STATEMENT OF CONTRIBUTIONS

This paper presents a short-term load forecasting method based on a real-time image representation of the metro traffic. The image must keep the information of train movement and load. The contributions of this paper are as follows:

- We build an image to capture the metro traffic and keep the information of trains along one entire metro line. This image allows us to not only take all the above mentioned constraints into account but also produce a ''snapshot'' of the transit network from an operational point of view. Besides, this approach provides all the required information to forecast the train loads for the entire planning horizon without a recurrent process. All the forecasting values are computed all at once from the image. Therefore, the generated image illustrates a realistic description of the traffic conditions in the context of transit data analysis.
- The proposed image of the metro traffic addresses all the aspects introduced in section I. By encoding departure as a pixel, the image gives an effective representation of metro traffic. Each pixel carries the data related to a train

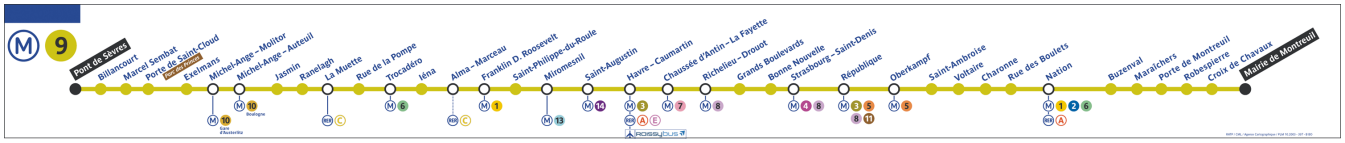

FIGURE 1. **Map of line 9 of the Paris Metro, ©RATP 2021.**

departure inside a network, thereby ensuring that there will be no aggregation over a time interval.

- This paper applies a deep learning model based on convolutional networks and a U-net architecture. The U-net architecture enables us to perform intelligent exploitation of all the available information of a train and its neighbor in order to make the prediction. The model relies on the capacity of convolution to extract metro traffic features from an image. Adopting such an approach allows us to respond to the different problem specificities listed above and to understand the contextual situation of a metro line.
- To validate the proposed framework, we implemented and evaluated it on real databases consisting of data from the Paris metro line nine run by the Parisian operator RATP over one year, 2019. This image-to-image model performs well on the cited test case and experimentally proves that the image-oriented approach is effective for load forecasting.
- We also challenge the proposed methodology over atypical situations, i.e. disruptions and strikes, to evaluate the image-to-image modeling performance. These experiments validate the robustness of the model and demonstrate the strength of monitoring metro traffic via the proposed image configuration from the operational point of view.

III. PROBLEM STATEMENT

In metro load forecasting, trains are the central element in order to build any prediction framework. They carry the passenger flows through all the stations. Note that at time t, multiple trains can be observed on a metro line. To track the position and load of all trains, we want to encode the traffic of trains as a "snapshot" at a given time t. This section introduces the formalization of a metro line state. The method is applied to metro line 9 in Paris.

A metro line is defined as a directed univariate chain of stations denoted by S. Let $s \in S$ denote the index of one station. Figure 1 illustrates the Paris metro line 9, which is used as a real test case in this study. Each element from S refers to a station of the metro map 1. As the space is ordered, station 1 is *Pont de Sèvres* and the last station 37 is *Mairie de Montreuil*. A complete journey of a train from the departure station to the terminus is called a "train course". We define an index set $C = \{1, 2, 3, \ldots, |C|\}$, representing the order of all the courses travelled by trains. The indexes $\{c_1, c_2\} \in C$ represent two random trains in the space S. The relationship between their indexes defines the order of the courses. For example, if $c_1 < c_2$, it means that train c_1 started its course before train c_2. For each stop in a course c at a station s, we have access to multiple information about the train load and other exogenous variables. Here, we define $y_{s,c}$ as the load measure at station s for course c at the departure time toward station $s+1$. Therefore, for each train, we can generate a sequence of loads $(y_{1,c}, \ldots, y_{|S|,c})$ associated to one train course (c) through the space S.

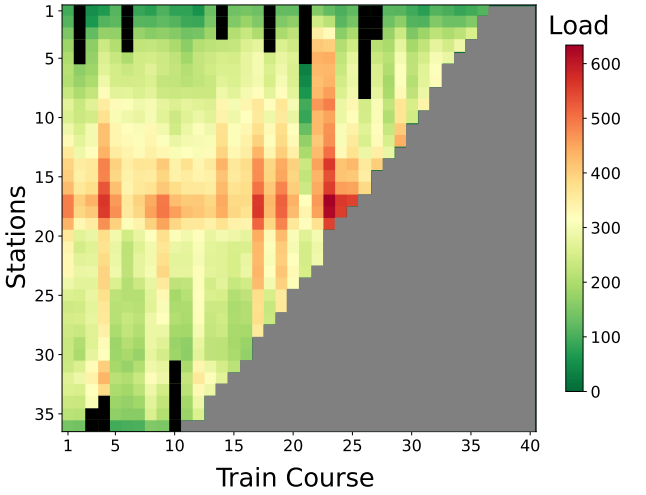

FIGURE 2. **An image sample of the train loads of a metro line leaving from the different stations at a given time. The black pixels represent the missing data, and the gray pixels denote the future load measurements of the trains. A column represents a train course at an instant t, and a row is a station of the metro line 9. The image denotes the real traffic data of the trains on a line.**

At this point, by gathering all the sequences with respect to all the trains moving in the network, we generate an image in order to represent the traffic (all sequences) within space S as shown on Figure 2. In this figure, on one axis, space S is represented by stations (one station for each row of the image) and, on the other axis, the travel dimension denoted by train courses C. In other words, each column corresponds to a trip made by a train in the image. Each pixel of the image represents an observation of the train load $y_{s,c}$ of train course c at station s. Thus no data are lost or aggregated; our image is a realistic time-dependent representation of a metro line. Some courses can have a special path, e.g., different origins and destinations from the first and last stations on the line, or they may skip some stations in their course. All the data corresponding to missing stations in a special path are collected as missing data (i.e., black pixels in Figure 2).

We can generate a snapshot of our traffic at any time t as an image of all previous loads in order to predict the subsequent course loads for all stations of S after time t. Not all the trains may have finished their courses at t, so the generated image contains pixels where no data are available because these pixels are related to the future. These pixels are our prediction objectives (colored in grey in Figure 2). The forecasting task in figure 2 is to compute the load value of all the grey pixels, which are future stops when the picture was generated.

In order to predict an image of all loads ($y_{s,c}$), we use a correlated image as input to the model based on the exogenous correlated vectors of variables $x_{s,c}$. The input image is generated in the same way as the load picture; however, it has multiple variables encoded in different channels. In this study, based on the availability of data on the metro line 9, $x_{s,c}$ is made up of three variables: station ridership, departure time, and travel time. Figure 3 presents an example for the three channels of our input images in this forecasting problem.

The prediction task of the metro load can be accomplished within the images of interest in which all the data of past load and future load are contained. Therefore, we build our methodology to process these images as input and output of the prediction framework. This means that we can define our prediction model in an image-oriented way to address load forecasting.

IV. IMAGE-BASED SHORT-TERM PREDICTION PROBLEM

This section presents the process of generating input and output images in detail. First, we present the components of the input and output pictures. We crop the images by defining a short-term range of the target variables in order to focus on the forecasting task. Second, we explain how missing data are managed in input images. Finally, we introduce two masks to map forecasting and past horizon for the prediction models.

A. PLACEMENT OF DATA IN IMAGE FORMAT

The idea developed in this article is to create from a univariate space S and a set of ordered sequences $c \in C$ an image of the input variables $x_{s,c}$ (the exogenous correlated variables) and an image of the target variable $y_{s,c}$ respectively noted I_x (1) and I_y (2). To encode all the variables in $x_{s,c}$, we use the channel concept for an image. Each variable is defined as a sub-image of I_x with the same width and height.

$$I_x = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \dots & x_{1,|C|} \\ x_{2,1} & x_{2,2} & x_{2,3} & \dots & x_{2,|C|} \\ \dots & \dots & \dots & \dots & \dots \\ x_{|S|,1} & x_{|S|,2} & x_{|S|,3} & \dots & x_{|S|,|C|} \end{bmatrix} \quad (1)$$

$$I_y = \begin{bmatrix} y_{1,1} & y_{1,2} & y_{1,3} & \dots & y_{|S|,|C|} \\ y_{2,1} & y_{2,2} & y_{2,3} & \dots & y_{|S|,|C|} \\ \dots & \dots & \dots & \dots & \dots \\ y_{|S|,1} & y_{|S|,2} & y_{|S|,3} & \dots & y_{|S|,|C|} \end{bmatrix} \quad (2)$$

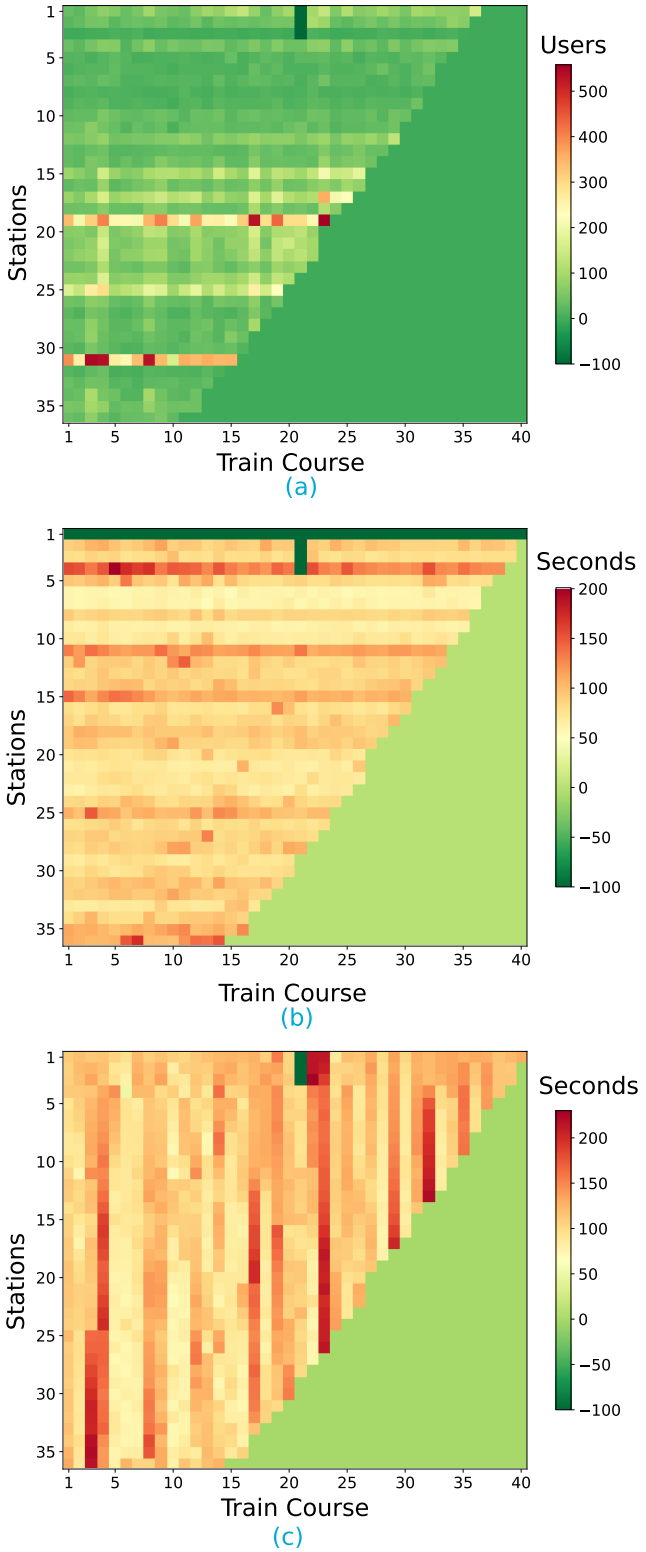

FIGURE 3. (a) Image of the number of users entering one station between two train departures, (b) Image of the travel time of trains between two consecutive stations, (c) Image of waiting time between two consecutive trains.

For short-term forecasting, only a reduced number of sequences is used as a period of interest. At a given time t,

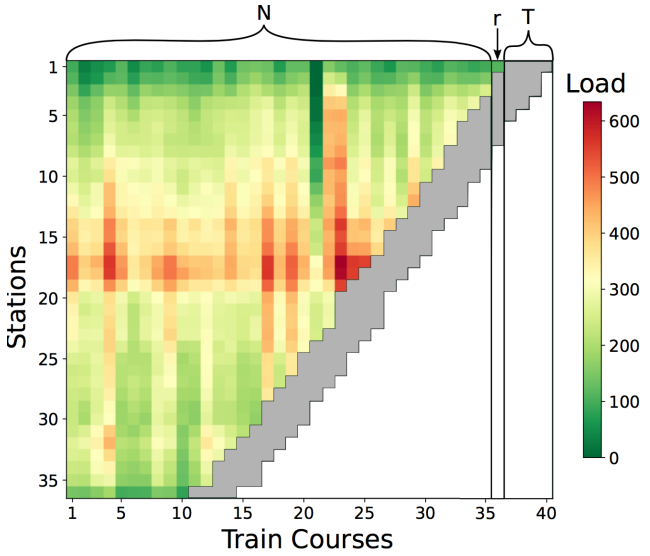

FIGURE 4. Image of train load at a given time t. The grey area defines the prediction objectives. The framed sequence refers to the last observed train course in the network. N represents the number of sequences to be considered as a period of interest for the inputs. T is the number of courses to be predicted at each station at time t.

we define C_t as a subset of all train courses (C):

$$C_t = \{r - N, \ldots, r, r + 1, \ldots, r + T\} \tag{3}$$

where r is the last train course at t that has at least one departure (see Figure 4). N denotes the number of sequences of our forecasting task. T is the number of future sequences we want to forecast. Consequently, the size of the subset C_t is constant $|C_t| = N + T + 1$ so as to keep a consistent and fixed width for all the images. Besides, the dimensions are fixed and determined by the height of the images. The height is set to the total number of stations in S. To apply deep learning methods, it is important to have a consistent and identical size for the inputs and output. Note that N must be large enough to have at least one completed train course inside the image in order to have data for all stations. The train course r is the sequence that is used to crop a subset of C at the instant t. It means that r defines the subset C_t based on the hyper-parameters N and T. Thus, we can redefine $I_x(t)$ by C_t, in (4), as the image generated at a time t from S and C_t where $C_t(a)$ represents the sequence of index a in the subset C_t.

$$I_x(t) = \begin{bmatrix} x_{1,C_t(1)} & x_{1,C_t(2)} & \ldots & x_{2,C_t(r)} & \emptyset & \ldots & \emptyset \\ x_{2,C_t(1)} & x_{2,C_t(2)} & \ldots & x_{2,C_t(r)} & \emptyset & \ldots & \emptyset \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ x_{|S|,C_t(1)} & x_{|S|,C_t(2)} & \ldots & \emptyset & \emptyset & \ldots & \emptyset \end{bmatrix} \tag{4}$$

Note that the columns before r in the image are not completed for a given time t. The objects observed in the image do not all have the same position in S and may not have finished their journey. The incomplete region of the image represents the future missing values at this time. These pixels are represented by the value $\emptyset$ in Equation (4). Depending on the different temporal or spatial contexts, the same pixel can be considered as future or past on two distinct images.

The multi-channel input image is defined by constant dimensions (height and width) and channels that encode all the image variables similar to standard RGB images (Red, Green, Blue). For an RGB color image, each of the three channels represents one of the colors. In our case, the dimensions of $I_x(t)$ and $I_y(t)$ of the image are defined by $|C_t|$ and the space S. In addition, the number of channels of the image $I_x(t)$ is equal to the number of different exogenous variables in $x_{s,c}$. In this study, $I_x(t)$ has 3 channels (see Figure 3) and $I_y(t)$ has a single channel of load.

Lastly, the temporal sampling is implemented by two mechanisms: (i) The position of the pixel related to its neighborhood. Every pixel has the same position in the image for regular or irregular temporal sampling at t; (ii) The times between pixels are recorded in one channel of the input image (Figure 3(b)). This coding by pixel position and integrating the inter-pixel times to a variable allows us to take into account both regular and irregular sampling.

Short-term predictions are equivalent to completing an image and translating it into an equivalent image of one or more target variables. Image construction makes it possible to deal with missing data in the same way as load forecasting by completing the missing part of the input image thanks to image processing models.

B. MISSING DATA

In image processing, a dead pixel is equivalent to missing data at a certain position in the image. In our case, there are two kinds of missing data: (i) the future horizon of the image; (ii) the past horizon. The first kind of pixels are by definition unknown and missing and represent the prediction objectives (i.e., grey pixels in the Figure 2). The second kind of dead pixels can be due to measurement errors caused by practical issues (i.e., black pixels in Figure 2) like the closure of a station or interrupted courses. In order to distinguish these two cases in the modeling, two different neutral elements are defined. The first one for the future horizon is returned as a null value for the predicted pixels. The second one for the past horizon is equal to a large negative load value which is not feasible. Recall that the load is the number of people inside a train, and the value is at least null or positive. The same conclusion can be drawn for ticket validation or train time travel. The choice of a negative value is made based on the experimental setting.

Since the proposed image contains both future and past data, we need to use masks to encode the information about both the past horizon and the position of the target pixels inside the image.

C. PREDICTION MASK AND HORIZON

All images $I_x(t)$ and $I_y(t)$ combine the past horizon, the set of known values $x_{s,c}$ at t, and the future horizon, the missing part of the image. To encode this information which is related to the observed image, we create two masks equivalent to the

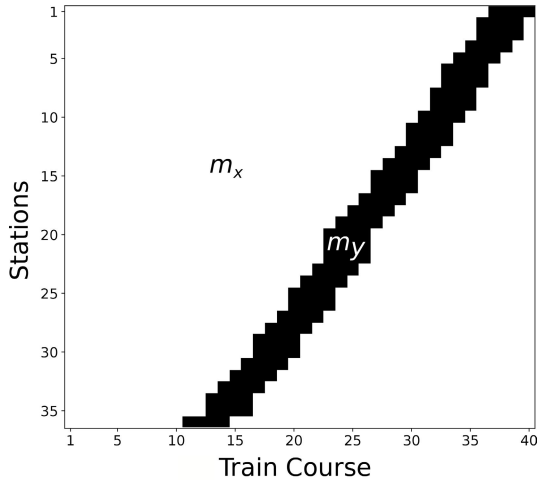

FIGURE 5. The prediction mask m_y refers to the forecasting horizon. m_x refers to the area of the equivalent mask.

future horizon m_y and the past horizon m_x. The masks are established according to (5) and (6), respectively.

$$m_x = m(x_{ij}, t) = \begin{cases} 1 & \text{if } x_{ij} \neq \emptyset; \\ 0 & \text{Otherwise.} \end{cases} \tag{5}$$

$$m_y = m(y_{ij}, t) = \begin{cases} 1 & \text{if } y_{ij} \text{ is a pixel to be predicted;} \\ 0 & \text{Otherwise.} \end{cases} \tag{6}$$

The mask of the past horizon can be different for each variable of the image. It reflects the knowledge of a variable at t, i.e., particularly its availability at the time when the image was generated. The size and the number of channels of m_x is equal to the equivalent picture $I_x(t)$. Each channel of m_x refers to the availability of the input variables $x_{s,c}$. However, m_x only focuses on past data whether or not the identified pixels have missing values.

The prediction horizon is not necessarily equal to the entire missing region of the image. It can be limited to a region pre-defined by a mask m_y based on hyper-parameter the T. Figure 5 presents a prediction mask used in the experimentation to extract forecast pixels (pixels in black) from $I_y(t)$. On the one hand, m_y and T make it possible to choose and focus on the short-term period of interest for prediction; on the other hand, m_y reduces the prediction error by privileging the pixels of the image for short term forecasting, i.e., past load pixels are neglected.

V. FORECASTING BY IMAGE PROCESSING

The prediction work is transformed into an image-to-image regression problem wherein the input image represents the exogenous data $x_{s,c}$ and the output image represents the equivalent target variable $y_{s,c}$. This section introduces image-oriented deep learning models. First, we define the forecasting task as an end-to-end image process. Second, we present the image processing mechanisms in order to perform the forecasting task. Third, we present the proposed forecasting framework based on the U-net architecture. Finally, we declare the loss function used in the learning phase of the models for the outputs.

A. IMAGE-ORIENTED FORECASTING MECHANISMS

There are two main tasks to be done by image processing in the framework. One task is to complete the partial input image. This task is similar to solving an inpainting problem, considering the future horizon as the missing region of the image. Besides, the unavailability of the past dynamics of the target variable related to $y_{s,c}$ limits the input of the prediction model to data consisting of exogenous data $I_x(t)$. If the load data can be used for a short-term task, we can increase the number of channels in $I_x(t)$ by adding the known load inside the picture. In this case, the prediction task can be denoted as an autoregressive task within the picture.

The other task of the model is to translate an image $I_x(t)$ into the target image $I_y(t)$ by a prediction function. The output image of this task is similar to figure 2, except that the predictor completes all the grey and black pixels, i.e., all the pixels are predicted. We define $f(.)$ as a prediction function to perform both tasks, inpainting, and translation. The general form of the prediction function f is presented in Equation (7).

$$\hat{I}_y(t) = f(I_x(t)) \tag{7}$$

Here, we focus on short-term prediction, so based on the definition of T and m_y, all the predicted pixels $\hat{I}_y(t)$ in equation (7) are not interesting. Both the future and past horizon are forecasted. For this reason, we deploy the prediction mask m_y in order to extract the target pixels corresponding to the period of interest in the future horizon:

$$\hat{I}_{pred}(t) = \hat{I}_y(t) \odot m_y \tag{8}$$

where $\odot$ is the Hadamard product. This equation excludes nonrelevant pixels of the reconstructed image, which vary according to the temporal and spatial context of the network. Figure 6 shows $\hat{I}_{pred}(t)$, which is the equivalent area (colored in grey) in Figure 4. It is produced from the product in Equation (8). For metro images, $\hat{I}_{pred}(t)$ takes a diagonal shape relative to the train front on the line.

B. IMAGE-PROCESSING-ORIENTED MODEL

The proposed framework to consider image-processing-oriented prediction is presented in Figure 7. The forecasting model aims to convert input pictures to output pictures. The inputs are the three channels of $I_x(t)$ and the output picture is the Total picture $\hat{I}_y(t)$ provided after the inpainting task. The final output is the Prediction Picture obtained by extracting $\hat{I}_{pred}(t)$ from $\hat{I}_y(t)$. In this subsection, we present the forecasting model in detail.

1) IMAGE-TO-IMAGE TRANSLATION

The mechanism to transform $I_x(t)$ to the train load image $I_y(t)$ is called image-to-image translation. In this translation process, an original image is translated to a new shaped picture or

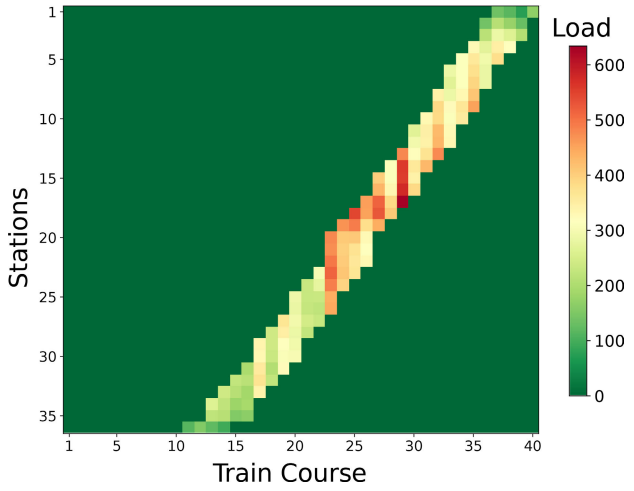

FIGURE 6. $\hat{I}_{pred}(t)$: **Final output, prediction image, of the next four departures for all stations of the metro line 9.**

an image with a new rendering (e.g., black and white pictures to colored images). In the literature, deep learning approaches are the main methods used in image-to-image translation. The most popular model is the Generative adversarial network (GAN) [29] consisting of the confrontation of a generator and a discriminator. The generator creates a new picture, and the discriminator validates whether the image is real or false. In this work, we used the pixel-to-pixel regression approach with GAN based on the study of [27]. Therefore, only the generative part of the GAN architecture is used in this study. GANs are introduced for natural image generation, in which the whole picture is important. However, in this study, we need to consider only some pixels of the picture as a regression goal. Moreover, this approach is a fair trade-off in terms of computation time and computational resources. GAN-based models are more computationally costly and complex frameworks than only the generator component. [27] used a U-net encoder-decoder as the generator of the model. In addition, we also find this kind of architecture in inpainting problems.

2) INPAINTING

The task of completing a missing area within an image is called inpainting. By extracting features from the image itself and understanding the context of the image, inpainting models aim to generate the most realistic complete image. Deep learning methods are the main fields of work to address inpainting methods. Many strategies can be applied to reconstruct an image such as skip connections [30] or attention mechanisms [25]. [26] showed the effectiveness of using an image history to reconstruct incompleted satellite images. Here, we follow the same approach as [26] to address the same goal regarding the similarity of traffic models defined for space S. Moreover, some models such as U-net or GAN can be transposed to the inpainting process because of their ability to extract features from images. This study focuses on the U-net-based model which is a less complex and time-consuming approach, and applies it to all metro lines of transit network.

C. U-NET ARCHITECTURE

The U-net architecture (Figure 8) was initially introduced in image segmentation work by [31]. The U-net model is a neural network consisting of a contracting path (left side of Figure 8) and an expansive path (right side of Figure 8). The contracting path consists of repeated convolution layers with max pooling. The expansive path consists of repeated up-sampling and convolution layers. Such architecture allows to extract relevant features at different scales which can be spread into the image generation part of the model (expansive path). In fact, each "up-convolution" [31] is linked to the equivalent convolution from the contracting part of the U-net. This mechanism is called skip connections and allow to propagate the features of the different convolutions and inputs to their equivalent up-sampling steps. One reason behind using such skip connections is that these direct path allow a more direct propagation of the gradient and therefore mitigate the vanishing gradient phenomena [32] observed in recurrent deep learning methods. Furthermore, in our (inpainting) case since a large part of the output is already known such paths are quite natural. Figure 8 shows that consecutive layers are designed symmetrically in order to keep the size of the input and final output identical. The output layer is a 1×1 convolution layer to map the features from the previous layer (in our case 64 feature map) into the desired image. In addition, this characteristic matches the way that our images have been previously defined.

In this study, we propose 3 different U-net architectures, detailed below:

- **U-net**: Based on the study of [31]
- Station-Train U-net (**ST U-net**): The goal is to integrate operational knowledge into the development of the model. The model is a U-net model that convolution and deconvolution phases are carried out in two steps using convolution (resp. deconvolution) filters operating on the trains (columns of the image) and the stations (rows of the image). The "station" filters of size $|C_t| \times 1$ consider all stations for the prediction. The "train" filters of size $1 \times |S|$ are focus only on the train course, which has at least one train load to be predicted.
- Partial Convolutional U-net (**Pconv U-net**): The model is similar to the U-net model, except that the convolution functions are replaced by partial convolutions and masks m_x to handle the irregularity of the pixels to be predicted which vary according to the temporal and spatial context of the trains.

We introduced two outputs for our framework, the Prediction image, $\hat{I}_{pred}(t)$, which has only the desired forecast pixels, and the Total image, $\hat{I}_y(t)$, which is the prediction of all pixels (see Figure 7). The idea of proceeding in two steps is to follow the structure of the image processing model of Inpainting and image-to-image Translation. It is worth

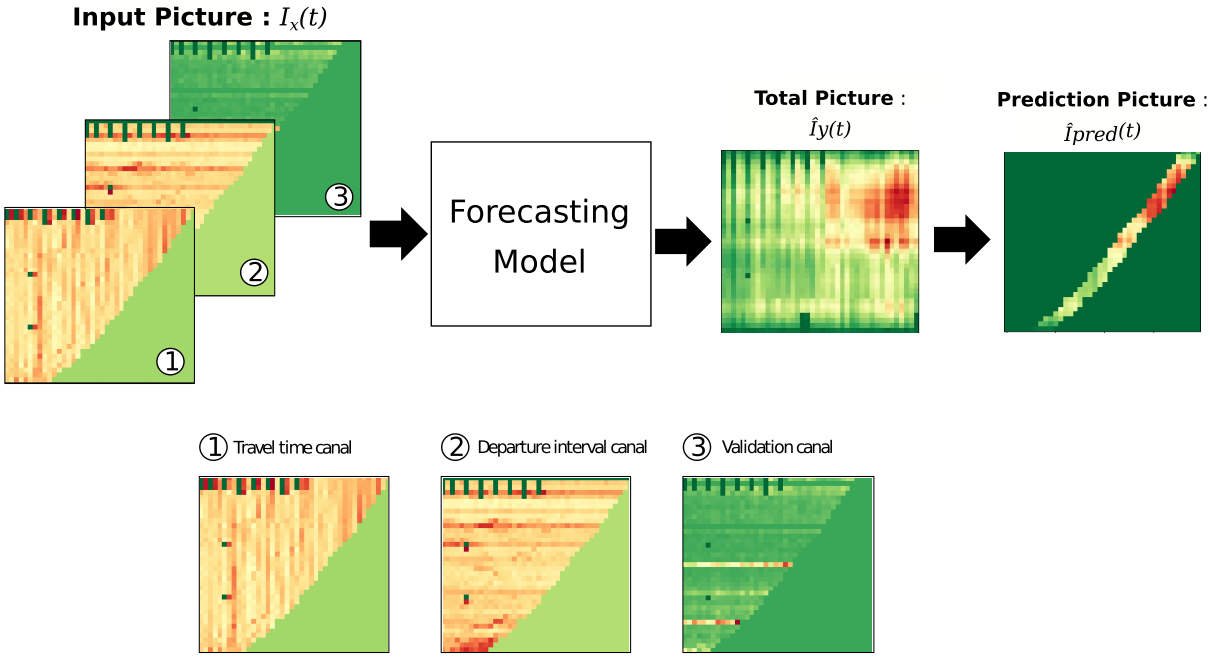

FIGURE 7. The formalization of the prediction problem using a three-channel image as the inputs and the two outputs, total image and prediction image.

mentioning that the proposed models can skip the first output to forecast $\hat{I}_{pred}(t)$ directly in order to avoid the cost of the whole image. However, this single output approach needs to use partial convolution layers to estimate the position of the predicted pixels inside the picture.

D. PARTIAL CONVOLUTION

The location of the pixels to be predicted in the image varies according to the spatial and temporal context. The same pixel can be considered to be predicted ($\in m_y$) or past ($\in m_x$) at two different times. The unknown region of the image is determined by the number of objects circulating in space S and their positions evolving in time. [33] proposed an approach based on partial convolution for image reconstruction with irregular holes inside input images such as our forecasting horizon. This approach aims to complete the image with our masks, m_y and m_x, in order to consider only the valid pixels (i.e pixels with $x_{c,s} \neq \emptyset$). The convolution step is replaced by the following equation [33]:

$$I' = \begin{cases} W^T(I \odot m_x)\dfrac{1}{\sum(m_x)} + b & \text{if } \sum(m_x) > 0 \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

where W denotes the convolution filters, b denotes the bias, and I' is the convolution result. The masks are updated according to equation (10), where we detect if at least one pixel is valid based on equation (5)

$$m' = \begin{cases} 1, & \text{if } \sum(m_x) > 0 \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

Depending on the required output mode, i.e. one prediction image or two outputs (Total and Prediction images), U-net architectures have to integrate partial convolution. For one output, the models use this partial approach. For two outputs, the two kinds of convolution layers can be selected in the construction of the model depending on the required application.

E. LEARNING LOSS FUNCTION

Equations (7) and (8) introduced the two possible outputs of the U-net architecture, the Total image, and the Prediction image. Depending on the output format, the model learning aims to minimize the following loss function:

$$L = \alpha \times L_{full}(I_y(t), \hat{I}_y(t)) + \beta \times L_{pred}(I_{pred}(t), \hat{I}_{pred}(t)) \quad (11)$$

$$\begin{cases} L_{total}(I, \hat{I}) = \dfrac{1}{|C||S|}\sum(I - \hat{I})^2 \\ L_{pred}(I, \hat{I_{pred}}) = \dfrac{1}{|C||T|}\sum(I - \hat{I}_{pred})^2 \end{cases} \quad (12)$$

- L_{total} denotes the cost function computed on the Total image,
- L_{pred} denotes the cost function computed on the Prediction image,
- α and β are hyper-parameters which make it possible to favor one of the outputs over the other one. For single output models, the parameter α is 0 because the Total image is not considered. In the case of a second output, these parameters allow us to penalize and make a trade-off between two outputs.

For the cost function, (11), we chose a Mean Square Error (MSE) per pixel (Equation (12)), where T represents the

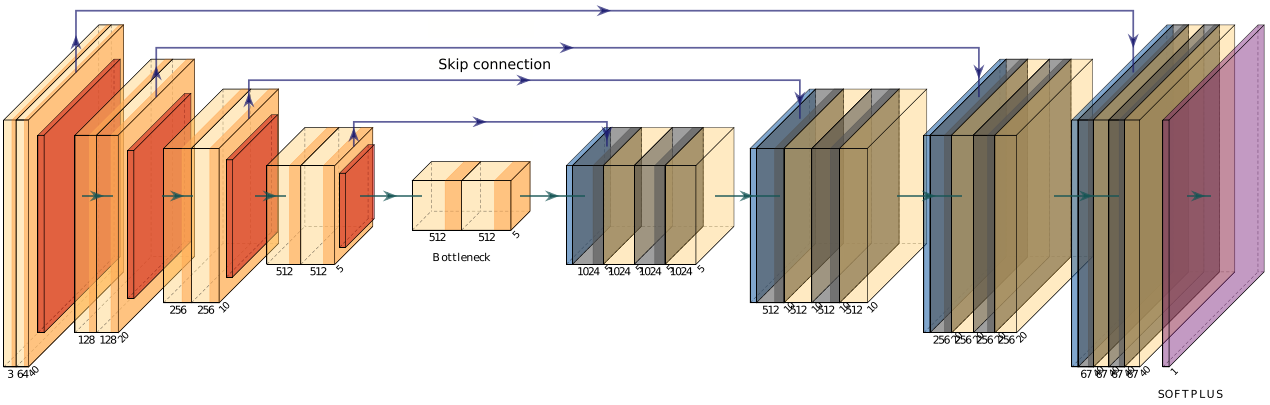

FIGURE 8. U-net architecture.

prediction horizon. This is a standard indicator for evaluating inpainting tasks even if it does not consider the global structure of the image [27], [34]. There are many other metrics such as Perceptual Loss, Style Loss, or Adversarial loss used in [25] that can be used; however, they are more appropriate to deal with natural images where we seek to judge the consistency between the generated image and the real image.

VI. EXPERIMENTATION

This section presents the real test case of this study to validate the proposed framework for train load prediction in detail. We first present the experimentation framework performed on line 9 of the Paris metro (Figure 1). Then, we introduce the list of prediction models, the values of the hyper-parameters, and the evaluation criteria.

A. DESCRIPTION OF THE EXPERIMENT

The experiment was conducted by RATP to predict the load of the next four train departures ($T = 4$) from each station of a Parisian metro line (line 9 as shown in Figure 1) in order to validate the framework. The load is defined as the number of passengers carried a train on departing from a station. This prediction has to take into account operational constraints related to the data availability. Loads are not available for a short-term observation window, and the modeling is only based on contextual data and variables available for the short-term and related to the departure of a train.

Our proposed prediction methodology requires the construction of input and output images of the model according to the data characteristics. The input images are constructed with the following size (see Figure 3):

- The width corresponds to the number of train sequences considered. The number of sequences was fixed at 40, taking into account the prediction horizon $T = 4$ (next four trains) and the maximum capacity of the line ($N = 35$).
- The height is the number of stations on the metro line 9. The last station is omitted because there is no departure.
- The number of channels is equal to the number of accessible variables collected for the experiment. In our case, there are three correlated variables, namely the aggregated ticket validation data between two successive departures for one station, the departure intervals which represent the waiting times between 2 successive departures at the same station, and the travel times determined by the times between 2 departures of a train between two successive stations for the same run

So the resulting image used as input for the prediction model has a dimension of $40 \times 36 \times 3$. The output image has a dimension of $40 \times 36 \times 1$. The channels of the output variables are reduced to the load data only. The picture visualization is scalable by increasing the dimensions of the image. For instance, by adding a new channel, we can add new data coming from the RATP network.

The images in the model dataset are generated every thirty seconds ($30s$). This delay is sufficient to be able to observe all train evolutions. The images were collected for a period of ten months from January 1st, 2019 to October 31st, 2019, during the operating hours of the Paris metro (5:30 am to 01:30 am), giving 2400 pictures per day.

B. PREDICTION MODELS

For the experimentation, we compared different U-net models built according to different architectures and according to the format of possible outputs with ANN and CNN models as a baseline. All these models have two versions depending on the output mode; either two outputs (Total image and Prediction image) using the prediction mask m_y (6) as in Figure 6 or a single output model where the Total picture (Figure 7) is skipped. The five models are as follows:

- Artificial Neural Network (**ANN**)
- Convolutional Neural Network (**CNN**) [19]
- Partial Convolutional Neural Network (**Pconv-CNN**).
- **U-net** defined in section IV
- Sequence-train U-net (**ST U-net**) defined in section IV
- Partial Convolutional U-net (**Pconv U-net**) defines in section IV

- **U-net with past load**. This model is similar to the U-net model with the difference that in addition to the three correlated variables, it uses a fourth one linked to the past dynamics of the load to be predicted. It is an ''optimistic'' model that cannot be implemented in real-time, as the load is not available in recent history. The performance of this model should therefore be logically better.

The proposed models are all composed of 3 convolutional layers completed with max-pooling layers. The first layer is composed of 128 filters of size 7×7, the second of 256 filters of size 5×5, and the third of 512 filters of size 3×3. The last layer of 512 filters of size 1×1 is used before the equivalent deconvolution part for the U-net or dense layers for the CNNs. All the models use the ''ReLU'' [35] activation functions except the model's output layer, which uses the ''softPlus'' function [35]. This function is selected based on the nature of the load, which is a positive value bounded between 0 and the maximum capacity of a train. For line 9, the maximum load is about 800 passengers.

The number of forecasting models that implement this image-to-image framework is limited in the literature. In addition, the image format is not compatible with many forecasting methods. Thus, auto-regressive models were not included in our comparisons because the load is unavailable, which does not allow the use of this type of approach (ARIMA and variants). Regression models built around ensemble learning models such as Random Forest or Gradient Boosting do not allow us to obtain an output equivalent to ours in a single model. These methods have one model addressed to each station and output. Lastly, we did not compare the proposed model with other types of deep learning architectures using recurrence mechanisms such as LSTM because all the sequences are already included within the image.

C. LEARNING AND SELECTION OF PARAMETERS

The mean square errors defined in equation (13) were chosen as the criterion to be minimized in our loss function (11). The ratio $\frac{\beta}{\alpha}$ in equation 11 was fixed at 1.5 by experience in order to focus on the Prediction picture. In addition, it is the ratio that presented the best result for the forecasting task in this test case. Beyond a certain value (2), we observed that some models did not converge to a solution with acceptable performances. This behavior is explained by the fact that the Prediction picture and the Total picture are correlated (Figure 7).

The dataset of images generated every 30 seconds runs from January to October 2019. We used the last two months to build the test dataset, which represents 20% of the data. The test base is not chosen randomly in order to keep an unknown temporal context with respect to the data. We want to avoid the models learning about the context of the test months and to avoid over-fitting on our train dataset. The optimization of the model weights is based on the Adam algorithm [36] with a learning rate of 1×10^4 and $\beta_1 = 0.9$. The learning is done over 10 epochs. The number of epochs for training is identical for all models in order to ensure consistent comparisons. In addition, the batch size is set of 32 random images to avoid memory overflow.

TABLE 1. Forecasting results of the total train loads.

Model	# parameters	MSE train	MSE test	WMAPE train [%]	WMAPE test [%]
ANN	11,523,440	1199	11043	11.9	18.4
CNN	30,057,752	**446**	1098	7.2	15.4
Pconv-CNN	32,990,552	1536	1276	14.0	16.5
U-net	8,032,516	712	**975**	**7.1**	**13.7**
ST U-net	13,917,441	2579	2073	18	21,8
Pconv U-net	14,676,007	2516	1338	16.7	16.9
U-net with past load	8,332,516	437	585	4.5	6.2

D. PERFORMANCE INDICATORS

The load prediction models are evaluated by comparing the prediction error on the training and test datasets. The performance metrics are the mean square error (MSE) in this study.

$$MSE = \frac{1}{|S|.|C_t|} \sum (I_y(t) - \hat{I}_y(t))^2. \tag{13}$$

and the Weighted Mean Average Percentage Error (WMAPE) in percent.

$$WMAPE = \frac{\sum |I_y(t) - \hat{I}_y(t)|}{\sum |I_y(t)|}. \tag{14}$$

VII. RESULTS

The results are presented in three parts. First, the benchmark results for the different architectures and outputs are discussed. Second, we analyse the best model according to spatio-temporal characteristics. Third, we observe the robustness of the model under atypical situations.

A. PERFORMANCE EVALUATION

The prediction performance will be provided in three different tables. The tables 1 and 2 refer to the prediction performance of the dual output models. The first table refers to the result for the Total picture (Figure 7). The second table focuses on the Prediction picture (Figure 7), which provides the next 4 station departures. Table 3 presents the results of models which forecast only the Prediction picture. In this case, no results are given for the U-net models without partial convolution because they do not converge to a relevant solution. Partial convolution disregards the knowledge of the prediction mask m_y. In all these tables, the last row refers to the best solution if the short-term load dynamics are used as inputs of the forecasting model.

The best result of the next departures is 13.1 % (U-net) for the models with two outputs and 15.5 % (Pconv U-net) for the model with one output. The availability of the data due to industrial constraints has a noticeable impact on the result. The performances of these models are impacted by the absence of load dynamics in the pictures. The best models with the past load have WMAPEs of 1.8% and 9.3%

TABLE 2. Forecasting results for the next 4 departures.

Model	# parameters	MSE train	MSE test	WMAPE train [%]	WMAPE test [%]
ANN	11,523,440	968	11520	10.8	16.4
CNN	30,057,752	28	174	6.5	14.4
Pconv-CNN	32,990,552	131	211	13.4	15.2
U-net	8,032,516	**19**	**136**	**5.1**	**13.1**
ST U-net	13,917,441	167	275	5.3	18
Pconv U-net	14,676,007	115	216	13.3	16.4
U-net with past load	8,332,516	6.3	4.3	2.2	1.8

TABLE 3. Forecasting results for prediction picture without mask.

Model	# parameters	MSE train	MSE test	WMAPE train [%]	WMAPE test [%]
ANN	11,523,440	–	–	–	–
CNN	30,057,752	282	399	**11.2**	20.2
Pconv-CNN	32,990,552	259	262	18.2	19.5
U-net	8,032,516	–	–	–	–
ST U-net	13,917,441	–	–	–	–
Pconv U-net	14,676,007	**149**	**174**	14.9	**15.5**
Pconv U-net with load	14,976,007	85	83	9.5	9.3

which is a better result than the previous models without load. Transforming metro traffic into an image seems to be a relevant forecasting solution for train load.

As expected, MSE values for the Prediction images (table 2) are significantly lower than those for the Total image (table 1). The use of the prediction mask m_y removes the prediction noise from all pixels not selected by the mask. This tendency is much less visible when we look at the WMAPE criterion. The improvement is less than 1% between table 1 and 2. Only the ST U-net model presents an improvement of about 4% but it has the worst forecasting result compared with baseline models. The U-net and Pconv U-net models provide better results and have less time consuming architectures (three times fewer parameters to learn compared to ANN and CNN models). Note that the MSE error for ANN explodes in the test case (more than ten times higher than other models). This model over-fits and cannot forecast a new metro context and images well. In other words, the convolution layers are needed to forecast and process the input images.

Table 3 presents the results of the models that use partial convolution. The other models cannot converge to a relevant solution (all the pixels are set at 0). This behavior is explained by the fact that applying mask the m_y simplifies the forecasting task. The model has to deal with the irregularity of the Prediction image's shape. Partial convolution is interesting by dealing with the detection within the picture of the pixels to be forecasted.

The last model, "U-net with past load," represents the prediction performance without considering the constraint imposed by the availability of load data. This model uses the past load dynamics to perform the prediction on the future horizon, as the loads are not available in a short-term window from a real-time perspective. It represents the best possible result. As expected, knowledge of the train load dynamics results in WMAPE performances lower than 5%, which outperforms all the models without load knowledge. Thus, We compared to the work [21] conducted with the same transport operator on a similar problem and data (note that the metro line test case is different). We manage to improve the performance while taking into account the specificities of short-term prediction, in particular by including the irregular temporal sampling.

B. SPATIO-TEMPORAL ANALYSIS OF PREDICTION RESULTS

This section presents the result of the best model, U-net, based on the test dataset. Here, we analyze the prediction performance from a temporal and spatial point of view through the evolution of the error per time interval and station. Figure 9 illustrates these evolutions estimated on the test dataset. The Root mean square error (RMSE) metric is used to present the result, as it is more understandable than the MSE criterion and the U-net model. The image format makes it very easy to split pixels by station or by train. Note that The focus of our analysis is on the Prediction picture (Figure 7).

Figure 9 a depicts the evolution of load error through the metro line 1. It can be seen that the error increases from station 1 to station 14. Then the error decreases slightly to station 32, named Nation, which is a hub of line 9 (colored in red). A hub is a station with high transit and multiple connections with other transportation modes in the network, e.g. RER station in the Paris metro. This feature of the station can be explained by the evolution of the train load as it travels along the line, which is similar to the figures (e.g., the evolution of the load is the same along the line).

The lowest error corresponds to stations with low train load and passenger inflow, i.e. under 2% of the total load and a median load per train of fewer than 100 people (14 users for the last station). The WMAPE criterion was not chosen because the same representation does not provide any information about the impact of the network due to the metric weighting except for stations with low train load. However, the topology and hub of the metro line can impact the error. At the end of the line, it can be seen that the prediction error increases following the stop at station 31. The position of a hub such as Nation at the end of the line impacts the prediction of this station and the following station. In addition, stations with multiple network connections (see the metro plan 1) do not have the same behavior as Nation in the center of the line.

In Figure 9b shows the error for one time interval from the beginning of the metro operating schedule at 05:30 to the end of the day at 01:30. The RMSE error is smaller for the early (5:30-6:30) and last (23:30-01:30) hours of operation than during the day, due to low load volume. These time intervals represent less than 2% of the total load of the line. In comparison, the peak hours of the Paris metro are from 07:30 to 09:30 in the morning and 16:30 to 19:30 in the evening. They represent respectively 13% and 28% of all the load and 12% and 30% of all incoming users for line 9. As mentioned before, if the station has a low train load

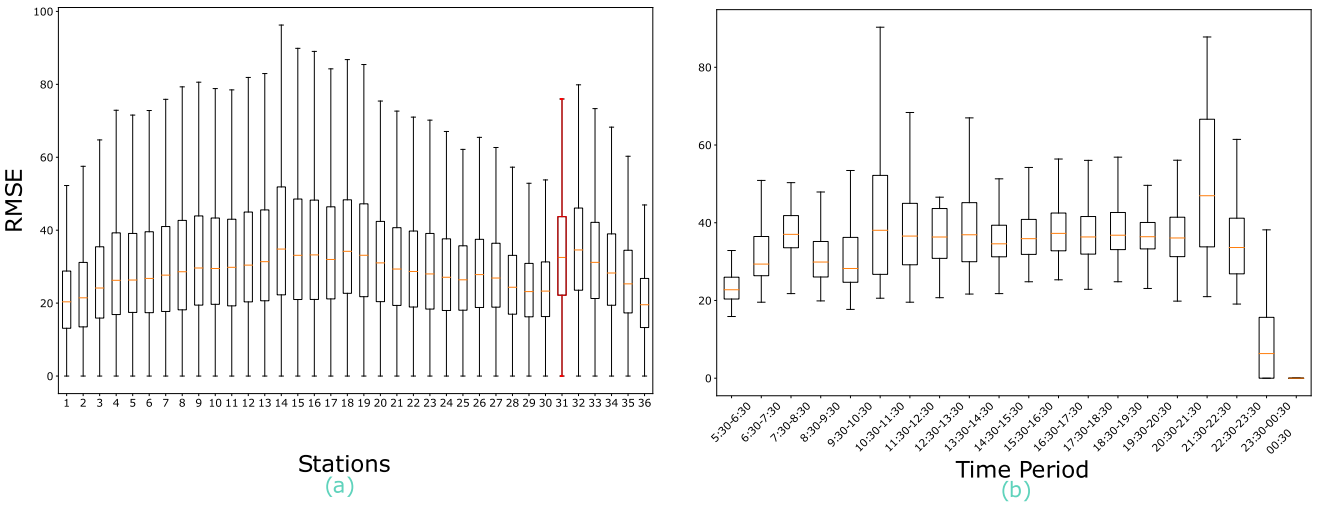

FIGURE 9. (a) Boxplot of the RMSE per station. Boxplots in red represent stations with an RER connection. (b) Boxplot of the RMSE per hour from the beginning of operation (5:30 am) to the end of service of a line (01:30 am). Each boxplot shows the error of all the images in the test dataset.

and passenger inflow, the prediction error is expected to be low. We expect the error to increase for those high-volume contexts. However, the result is unexpected because the time intervals with higher median errors and high error variances are not the peak hours, when the load is correlated to dense traffic and high load volume, but the times with higher errors represent transitions between off-peak hours and peak hours where each represents 3% of the load. Our model has difficulties handling the transition between two distinct load contexts from a high volume context to a low transit one.

Based on the results, we can conclude that the proposed framework is impacted in the short term by the train loads. A higher load implies a higher forecasting error. The following section examines the model efficiency in greater depth and observes the response of the model for special cases such as disruptions or strikes.

C. PREDICTION RESULTS FOR ATYPICAL CASES

Our predictions are influenced by the network topology and station context (closed, busy, etc.), the train load, and the transitions between different load time contexts. From an application point of view, the short-term prediction of train loads is important in the management and understanding of atypical events for both operators and users. It is relevant to observe the response of the prediction model to perturbed traffic situations. We illustrate the performances obtained through two examples, a disrupted situation and a strike. Besides, in practice, forecasting abnormal situations in the network is more interesting for transport operators because operators and users are already familiar with normal daily traffic. These special cases can be seen as the achievement of short-term forecasting. Image processing with masks is a possible solution to deal with disruptions and an abnormal network typology. The first case of disruption is interesting for an operator such as RATP since the normal cases can be easily learned by load history and knowledge of the line, whereas the main challenge in short-term forecasting is to predict abnormal cases well. The second case of a strike situation shows the interest of using masks to help the model under a special train circulation context.

1) DISRUPTED SITUATION

The atypical situation happened on October 11th, 2019, between 8:00 am and 9:30 am during the morning peak period (the peak hours of the Paris metro are between 7:30 am and 9:30 am). Figure 10(a) shows a Marey graph of the circulation of the different train courses over time with the information about train loads denoted by the size of points. We generates two images for this period in which each color (red or blue) corresponds to a given time for the prediction task. The red dots denote the moment of disruption (at 8:30) and the blue dots represent the results of prediction at the end of the disruption (at 9:00).The graph shows some train courses beginning in the middle of the line and significant time intervals between two trains. In blue, we observe the effect of the disruption of the line with high loads for all the stations impacted.

Figure 10b shows that the model provides lower predictions than the actual values for the disrupted situation. This behavior should be avoided in short-term forecasting for the operator to warrant a consistent passenger information. The overloads in highly disrupted situations remain difficult to predict. Nevertheless, the U-net model makes it possible to forecast from local features. We note that the undisturbed stations at the end of the line perform well, as shown in the two prediction images in Figure 10(c) and Figure 10(d). Only the region affected by the disruption shows a noticeable degree of error. It is respectively 18% and 21% in WMAPE (see Figures 10c and 10d). For a complex operating context

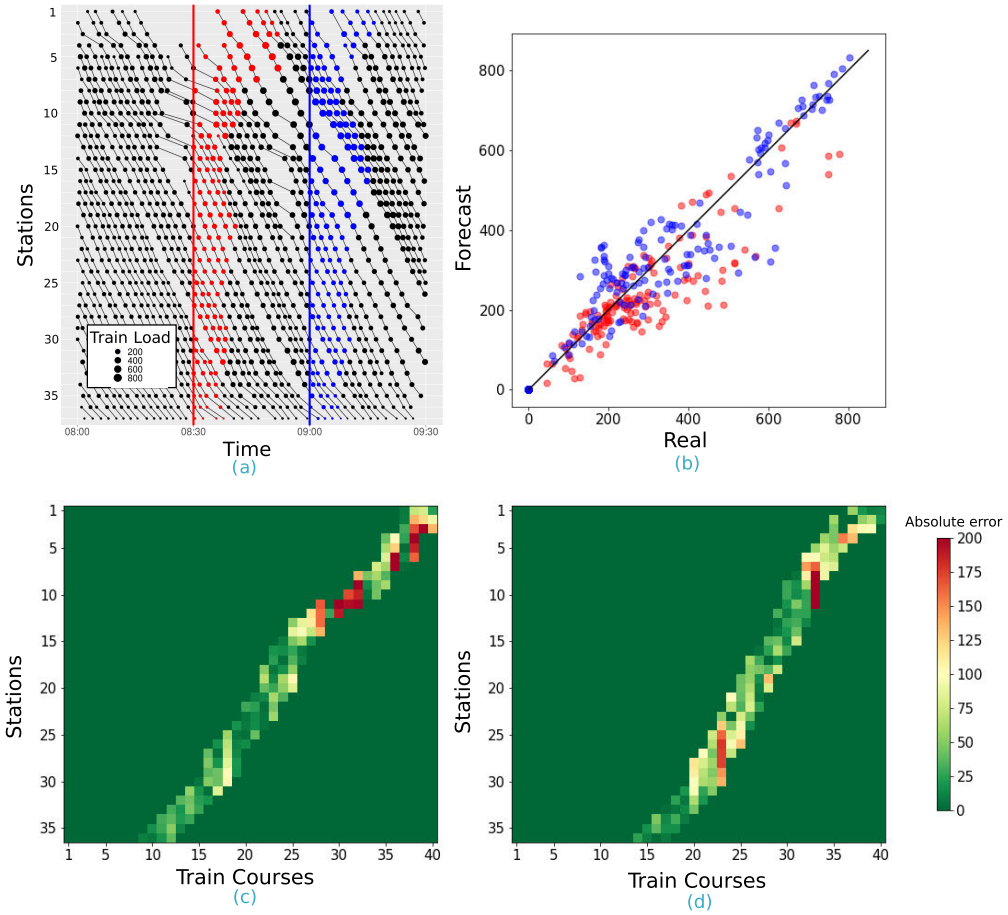

FIGURE 10. Results for the disrupted situation on line 9. (a) Scatterplot of the prediction results of two images according to the observed (real) values for an atypical situation. The points that corresponded to the beginning of the disruption are in red, and those at the end of the disruption are blue. (b) Marey plot showing the evolution of metro traffic between 7:00 pm and 8:30 pm. The red/blue circles represent the next four departures to be predicted for two images generated during the atypical situation. (c) and (d) represent the images of the absolute prediction error images for the red and blue points, respectively.

such as this one, such a prediction error remains acceptable. Moreover, the shape of the image shown on Figure 10c (i.e. a shifted diagonal), is a clue that an atypical situation has occurred. These results of traffic disruptions show that this image processing forecasting model can efficiently predict abnormal cases without having any context or information about the events. The proposed framework seems to be robust for unpredicted abnormal situations.

2) STRIKE SITUATION

The second scenario concerns the situation in which the metro line was on strike. 17 station were closed, namely stations 7, 12, and 15 to 30, strongly impacting the traffic and the transport schedule throughout the day. Note that the case study is part of the test dataset, and no similar case in the traffic status is recorded in the train dataset.

The data set corresponds to the traffic data of September 13, 2019. The line was split into two sub-lines separated by the closed stations between stations 14 (Franklin Roosevelt) and 31 (Nation). That is why the predicted images Figure 11 c and d show two uncorrelated areas of train courses. The hole between two train courses represents a train travelling on the other subline.

The predictions made in this highly disturbed and model-unknown situation (no strikes resulting in partial line operation are present in the training dataset) are encouraging, although the WMAPE prediction error is 58% and 34% respectively for figures 11c and 11d.

This case shows the interest of the mask m_y during the prediction task. It makes it possible to consider an abnormal topological context by focusing only on the pixels to be forecasted while keeping a global vision of the line. For instance, in the total image, many pixels with no real train courses (e.g., closed stations 7 and 12) are predicted, but the mask does not take them into account. The idea of providing two output images namely, the total image, and the prediction image, shows the strength of the model with respect to the spatial context. The masked output helps the model deal with a strike by considering only valid prediction pixels. The model may

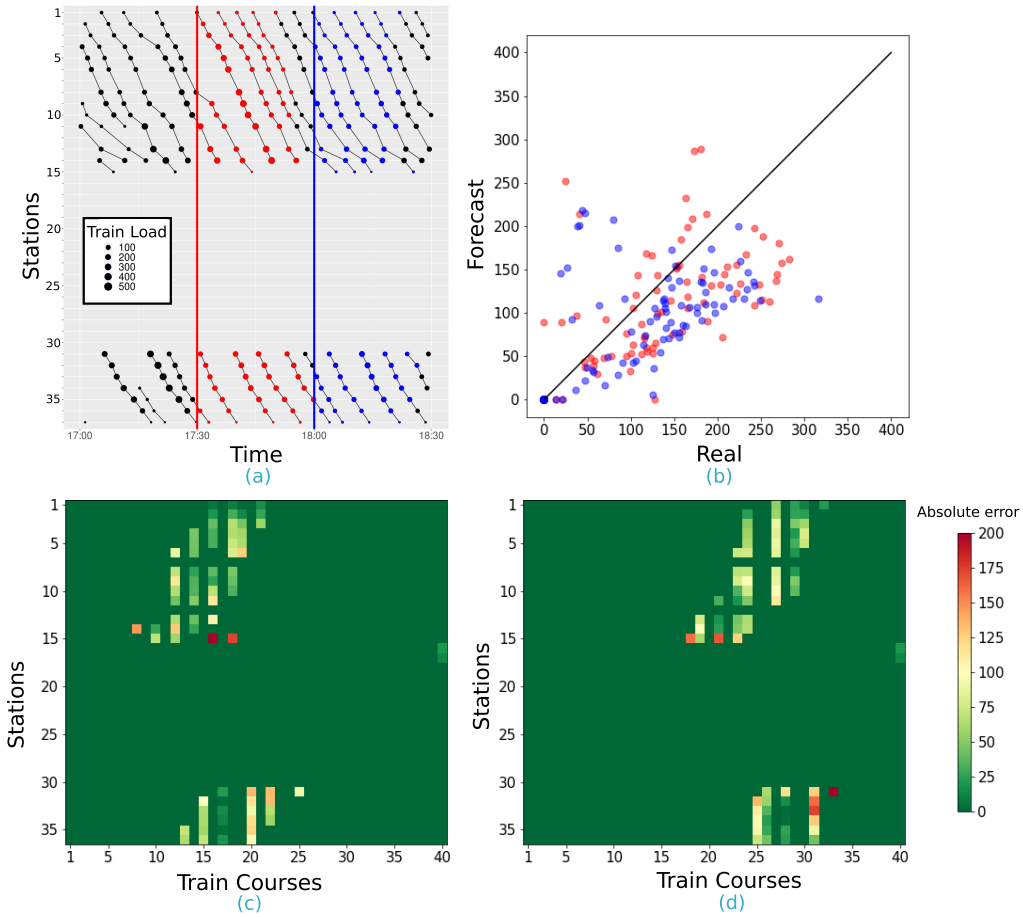

FIGURE 11. Results for the strike situation on line 9. (a) Scatterplot of load predictions versus actually observed loads in a strike situation (b) Marey graph showing the evolution of the metro traffic between 17h and 18h30. Blue and red represent the next four departures to be predicted for two generated images. (c) and (d) represent the images of the absolute error of prediction respectively for the red and blue dots.

mispredict some pixels but they are ignored by the model thanks to the prediction mask (m_y).

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have introduced for the first time an inpainting image-oriented framework in order to achieve short-term train load forecasting. The model uses a metro traffic image-oriented representation generated in real-time. The traffic images thus generated take into account three aspects of a transit system's characteristics at the same time: univariate spatial space (a metro line), irregular temporal sampling depending on the timetables, and the unavailability of short-term past dynamics of the target loads. The proposed model is based on a U-net architecture that aims to reconstruct an image of train loads from an input image of correlated variables for several time steps. The U-net solves an image-to-image regression taking account of multiple image processing mechanisms such as inpainting and image-to-image translation.

Several variants of prediction models are proposed and compared based on their prediction performances on real data collected during ten months on a metro line. The results obtained showed the relevance of the image-based traffic modeling approach both in the representation of the data and the consideration of spatio-temporal features. In terms of prediction performance, the proposed approach proves to be very efficient in addition to its ability to treat the complete line and provide predictions for several time steps. In the results section, we saw that the topology of the metro line impacts the model. In addition, while peak hours do not affect prediction, the transition from a high volume context to a low load volume does. Finally, atypical cases such as disruptions or a strike have shown the strength of the model thanks to the image representation and the use of the mask.

The authors intend to focus on atypical situations of metro traffic, which are a challenge for a public transport operator. The first step is to use a more complex framework (GAN - Transformer) to extract some contextual features from the pictures and not only focus on the forecasting task. Another clue can be to integrate data from Covid crises, which significantly impact transport and load forecasting. The idea is to build prediction models capable of learning rare and atypical operating situations in order to predict them better.

REFERENCES

[1] O. Egu and P. Bonnel, "Medium-term public transit route ridership forecasting: What, how and why? A case study in lyon," *Transp. Policy*, vol. 105, pp. 124–133, May 2021.

[2] G. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. Toronto, ON, Canada: Holden-Day, 1976.

[3] S. Anvari, S. Tuna, M. Canci, and M. Turkay, "Automated Box-Jenkins forecasting tool with an application for passenger demand in urban rail systems," *J. Adv. Transp.*, vol. 50, no. 1, pp. 25–49, Jan. 2016.

[4] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transp. Syst.*, vol. 11, no. 2, pp. 68–75, Mar. 2017.

[5] F. Toqué, E. Côme, M. K. El Mahrsi, and L. Oukhellou, "Forecasting dynamic public transport origin-destination matrices with long-short term memory recurrent neural networks," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 1071–1076.

[6] J. He and B. Si, "The application of ARIMA-RBF model in urban rail traffic volume forecast," in *Proc. 2nd Int. Conf. Comput. Sci. Electron. Eng. (ICCSEE)*, 2013, pp. 1662–1665, doi: 10.2991/iccsee.2013.416.

[7] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 871–882, Jun. 2013.

[8] S. Liang, M. Ma, S. He, and H. Zhang, "Short-term passenger flow prediction in urban public transport: Kalman filtering combined K-nearest neighbor approach," *IEEE Access*, vol. 7, pp. 120937–120949, 2019.

[9] M. Gallo, G. De Luca, L. D'Acierno, and M. Botte, "Artificial neural networks for forecasting passenger flows on metro lines," *Sensors*, vol. 19, pp. 1–14, Aug. 2019.

[10] V. V. Ivanov and E. S. Osetrov, "Forecasting daily passenger traffic volumes in the Moscow metro," *Phys. Particles Nuclei Lett.*, vol. 15, no. 1, pp. 107–120, Jan. 2018.

[11] S. Hochreiter and J. J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] K. Pasini, M. Khouadjia, A. Samé, F. Ganansia, and L. Oukhellou, "LSTM encoder-predictor for short-term train load forecasting," in *Proc. ECML/PKDD*, 2019, pp. 535–551.

[13] J. Ke, H. Zheng, H. Yang, and X. Chen, "Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach," *Transp. Res. C, Emerg. Technol.*, vol. 85, pp. 591–608, Dec. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X17302899

[14] Y. Ma, Z. Zhang, and A. Ihler, "Multi-lane short-term traffic forecasting with convolutional LSTM network," *IEEE Access*, vol. 8, pp. 34629–34643, 2020.

[15] S. Du, T. Li, X. Gong, and S.-J. Horng, "A hybrid method for traffic flow forecasting using multimodal deep learning," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, p. 85, 2020.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[17] K. Lee and W. Rhee, "Graph convolutional modules for traffic forecasting," 2019, *arXiv:1905.12256*. [Online]. Available: https://arxiv.org/abs/1905.12256

[18] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI*, 2019, pp. 922–929.

[19] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, Apr. 2017.

[20] D. Wang, Y. Yang, and S. Ning, "DeepSTCL: A deep spatio-temporal ConvLSTM for travel demand prediction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[21] J. Roos, S. Bonnevay, and G. Gavin, "Short-term urban rail passenger flow forecasting: A dynamic Bayesian network approach," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2016, pp. 1034–1039.

[22] W. Jiang and L. Zhang, "Geospatial data to images: A deep-learning framework for traffic forecasting," *Tsinghua Sci. Technol.*, vol. 24, no. 1, pp. 52–64, Feb. 2019.

[23] N. Cohen, S. Sood, Z. Zeng, T. Balch, and M. Veloso, "Visual forecasting of time series with image-to-image regression," 2020, *arXiv:2011.09052*. [Online]. Available: http://arxiv.org/abs/2011.09052

[24] N. Maaroufi, M. Najib, and M. Bakhouya, "Predicting the future is like completing a painting!" 2020, *arXiv:2011.04750*. [Online]. Available: https://arxiv.org/abs/2011.04750

[25] C. Xie, S. Liu, C. Li, M.-M. Cheng, W. Zuo, X. Liu, S. Wen, and E. Ding, "Image inpainting with learnable bidirectional attention maps," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8857–8866.

[26] Q. Zhang, Q. Yuan, C. Zeng, X. Li, and Y. Wei, "Missing data reconstruction in remote sensing image with a unified spatial–temporal–spectral deep convolutional neural network," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 8, pp. 4274–4288, Aug. 2018.

[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1125–1134.

[28] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, pp. 818–833, Nov. 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 3, Jun. 2014, pp. 1–9.

[30] Z. Yan, X. Li, M. Li, W. Zuo, and S. Shan, "Shift-Net: Image inpainting via deep feature rearrangement," 2018, *arXiv:1801.09392*. [Online]. Available: https://arxiv.org/abs/1801.09392

[31] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," 2015, *arXiv:1505.04597*. [Online]. Available: https://arxiv.org/abs/1505.04597

[32] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 6, no. 2, pp. 107–116, Apr. 1998.

[33] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," 2018, *arXiv:1804.07723*. [Online]. Available: http://arxiv.org/abs/1804.07723

[34] Z. Wang and A. Bovik, "Modern image quality assessment," in *Modern Image Quality Assessment*. Williston, ND, USA: Morgan & Claypool Publishers, 2006, p. 156.

[35] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, in Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., Fort Lauderdale, FL, USA, vol. 15, Apr. 2011, pp. 315–323. [Online]. Available: http://proceedings.mlr.press/v15/glorot11a.html

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, Dec. 2014, pp. 1–15.

THOMAS BAPAUME received the M.S. degree in datascience and artificial intelligence from the ESIEE Engineering School, Gustave Eiffel University, where he is currently pursuing the Ph.D. degree in computer science with the collaboration of Régie Autonome des Transports Parisiens (RATP) with a CIFRE programme.

His current research interests include the issue of the short-term forecasting of train load. His work focuses on the atypical situations observed on public transport systems.

ETIENNE CÔME received the master's and Ph.D. degrees from the Université de Technologie de Compiègne, Compiègne, France, in 2005 and 2009, respectively. He is a Researcher with the Engineering of Surface Transportation Networks and Advanced Computing (GRETTIA) Laboratory, Development and Networks (University Gustave Eiffel), Marne-la-Vallée, Paris. His research interests include random graph models, mixture models, and more generally probabilistic graphical models and their use to solve transportation problems.

JÉRÉMY ROOS received the Ph.D. degree from the University of Lyon, in 2018. In his thesis, he developed a Bayesian network approach to forecast the short-term passenger flows of the Île-de-France urban public transport network. He currently works at Autonomous Parisian Transportation Administration (RATP), where he manages the data science activities related to passenger mobility. His research interests include machine learning and probabilistic and statistical modeling applied to public transportation problems.

MOSTAFA AMELI received the B.Sc. and M.Sc. degrees (Hons.) in industrial engineering from the University of Tehran, Tehran, Iran, in 2014 and 2016, respectively, the second M.Sc. degree in mechanical engineering from the Arts et Métiers ParisTech, Paris, France, in 2016, and joined the Horizon 2020 ERC Project, MAGnUM, in 2016, and received the joint Ph.D. degree from the University Paris-est, The French Institute of Science and Technology devoted to Transport, Planning, and Networks (IFSTTAR), Paris, and the University of Lyon, Lyon, France, in 2019. He is currently a Research Officer (eq. Assistant Professor) in applied mathematics, computer science, and transportation science at the Transportation Engineering and Computer Science Laboratory (GRETTIA), University Gustave Eiffel, Paris. His research interest includes the intersection of operations research and computer science, especially with applications in transportation management systems.

LATIFA OUKHELLOU received the Ph.D. degree from Paris-Sud University, in 1997, and the Habilitation degree from Paris-Est University, in 2010. She is currently the Research Director of Université Gustave Eiffel, France, and was formerly an Assistant Professor with the University of Paris-Est Créteil. She created the Data and Mobility Group within the GRETTIA Laboratory (Transportation Engineering and Computer Science Laboratory). She has published over 120 articles in international scientific journals and conference proceedings. She is involved in several research projects in the field of intelligent transportation systems or urban computing for smart cities. Her research interests include data mining, machine learning, and information fusion applied to diagnosis problems and to spatio-temporal data mining for identifying driving behavior, analyzing urban mobility or monitoring energy, and water smart grids.