Hide & Seek: Traffic Matrix Completion and Inference Using Hidden Information

Hide & Seek: Traffic Matrix Completion and Inference Using Hidden Information

Alessio Sacco * Flavio Esposito ‡ Guido Marchetto *
* Department of Control and Computer Engineering, Politecnico di Torino, Italy
‡ Department of Computer Science, Saint Louis University, USA

***Abstract*—Traffic matrices are used for many network management operations, from planning to repairing. Despite years of research on the topic, their estimation and inference on the Internet are still challenging and error-prone. For example, missing values are unavoidable due to flaws in the measurement systems and possible failure in data collection systems. It is thus helpful for many network operators to recover the missing data from the partial direct measurements. Some existing matrix completion methods do not fully consider network traffic behavior and hidden traffic characteristics, showing the inability to adapt to multiple scenarios. Others instead make assumptions on the matrix structure that may be invalid or impractical, curtailing the applicability. In this paper, we propose *Hide & Seek*, a novel matrix completion and prediction algorithm based on a combination of generative autoencoders and Hidden Markov Models. We demonstrate with an extensive experimental evaluation on real-world datasets how our algorithm can accurately reconstruct missing values while predicting their short-term evolution.**

***Index Terms*—traffic matrix, machine learning, inference**

I. Introduction

Traffic matrices (TM) are a critical input for many distributed system management tasks, including capacity planning, anomaly detection, and even business intelligence. For example, they can help with provider selection in routing with network resources provisioning, network debugging [1], or even with network anomaly detection and network security [2]. Despite their importance, obtaining a complete TM at any given time is a challenge. TM incompleteness may be caused by measurement impracticality [3], (voluntary) data amputations [4], or both.

After several years of research on this topic, operators still rely mostly on low-resolution measurements such as SNMP messages. However, even excellent measurement systems suffer from errors and missing data, so a TM is often required to be complete or reconstructed before it can be used in any application, or as an input of a machine learning model [4]. The majority of existing techniques to estimate a traffic matrix given a limited set of measurements rely on network inference and network tomography methods, *e.g.*, [5].

Moreover, although the literature has already addressed the problem from both spatial and temporal perspectives, giving birth to Machine Learning (ML)-based methods as in [6]–[8], a more general approach that can account simultaneously for both space and time is still missing, *i.e.*, a method that can both solve the matrix completion problem and predict future values of such matrix elements.

To this aim, we propose *Hide & Seek*, a novel solution that can complete the TM starting from *hidden information* while also predicting the future values of these missing entries. Our solution is based on an augmented Hidden Markov Model (HMM), where the traditionally employed Viterbi algorithm [9] is replaced with a more performant algorithm based on Adversarial AutoEncoder (AAE) [10]. In particular, the AAE-based encoding method is applied to complete the matrix, while the more general HMM keeps track of the evolution of the traffic data, so to predict the next value.

The key to our matrix completion/prediction approach is the ability to observe a sufficiently useful subset of the matrix entries, which we denote as *hidden information*. *The main advantage of our method is that we do not rely on any statistical assumptions about the rank of the traffic matrix*. Conversely, in our model, we convert the estimation problem into a process aiming to learn the hidden relationship between the partial traffic data, viewed as hidden information, and the missing traffic value.

In particular, our contribution in this paper is twofold. (i) We first define a novel learning-based predictor for traffic matrices that improves the traditional HMM. This new model leverages AAE to estimate missing information and infer their values in the near future. Then, (ii) we propose a traffic matrix completion algorithm that uses the aforementioned model. While we use the algorithm for traffic inference, the algorithm is applicable in a variety of contexts, within or outside network management, given the limited set of requirements and its learning-based nature.

In an extensive experimental evaluation on real-world Internet traces, namely collections from the Abilene and GEANT networks, we demonstrate how effective our Adversarial AutoEncoder is in finding the hidden relationship between the missing value and the observed traffic entries that are adjacent in the TM. Besides, the HMM can properly predict the evolution of these unknown entries.

The rest of the paper is organized as follows. In Section II we discuss the existing literature about the matrix completion problem. Section III describes the model of the system and presents the specific problems addressed by *Hide & Seek*. In Section IV we present the methods used in the solution, highlighting how we combined AAE with HMM and specifying how they are used in our algorithm. Results are then presented in Section V, and finally, we conclude our paper in Section VI.

II. Related Work

The problem of traffic matrix inference has been well-studied and has been addressed from different angles given the variety of fields where it finds applicability and the importance in networking [11]. Common methods for matrix completion to be developed are based on the incorporation of side information from different sources, such as total incoming bytes and number of customers [12]. For example, in [5] the proposed solution takes advantage of using multiple readily available data sources.

One of the most common approaches is the low-rank matrix completion, spanning a wide range of techniques, from norm minimization [13], to singular value thresholding [14], to alternating minimization [15], to name a few. What these approaches have in common is that they assume the whole matrix has a low rank, and pose an optimization to fit the entire matrix with a single rank-r model. At the same time, some studies have acknowledged some spatial and temporal properties in the TM. Based on the spatial traffic feature, in [16] TM is modeled as multi-Gaussian models and then used to estimate the missing data.

To recover the missing entries in traffic data, also spatio-temporal tensor completion methods have been studied in the literature [4], [17], [18]. For example, [17] introduces a tensor (a multidimensional array) to model a time series of pure spatial traffic matrices. The model takes the lower-dimensional latent structure of network traffic and hidden traffic characteristics into account to extract this latent structure of traffic via tensor factorization.

Different from these solutions, we propose a learning-based approach, combining statistical with ML features and taking advantage of the concept of spatial affinity inside a smaller submatrix.

III. Problem Definition

A traffic matrix, *i.e.*, a matrix reporting the traffic volumes between origin and destination in a network, has a potential utility for network capacity planning and management tasks. However, traffic matrices are often hard to measure directly in large operational IP networks, and it is often required to complete the matrix in the unknown cells.

In this paper, we analyze the problem of Traffic Matrix (TM) estimation from two different perspectives: matrix completion and future value prediction. Despite differing in the model resolution, both the first part, TM completion, and the second part, TM inference, start with a partially-observed traffic matrix. Since these two tasks share the same system model, we first describe the details of our considered model, and then we formally define the two problems addressed by *Hide & Seek*.

A. System Model

In our model, we consider a network with n nodes, and we denote with Ω the non-empty set of all sources and destinations in a network, where $|\Omega| = n$. Hence, at the time t, the resulting Traffic Matrix (TM) is an $n \times n$ square matrix, whose element

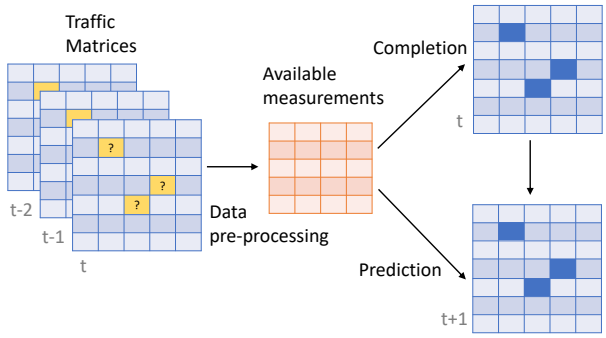

Fig. 1: System overview: we first identify the missing traffic matrix elements, then we extract the information required by our prediction model and finally, we restore the traffic matrix values and predict future traffic demands.

represents the number of bytes sent from node i to node j during the considered measurement interval.

We then consider the evolution of the sampling time, turning the TM into a 3-dimensional array, $Q \in \mathbb{R}^{n \times n \times m}$, where n is the cardinality of nodes, and there are m time intervals. Its entry $Q(i,j,t)$ denotes the traffic bytes of the origin-destination pair $i-j$ observed at time t, *i.e.*, in the measurement interval $[t-1,t)$, where $i = 1,\dots n$, $j = 1,2\dots n$, $t = 1,2\dots m$. Therefore, the entries $Q(i,j,:)$ represent the traffic bytes count variation along with the time for the Origin and Destination (OD) pair (i,j). We model the TM entries as continuous values, as we find this assumption reasonable for most volumes of traffic [19].

In addition, we use a binary matrix $Z \in \mathbb{R}^{n \times n \times m}$ to indicate whether entries of the TM Q are missing, defined as follows:

$$Z(i,j,t) = \begin{cases} 0 & \text{if } Q(i,j,t) \text{ is missing,} \\ 1 & \text{otherwise} \end{cases} \tag{1}$$

Consequently, the observed measurement matrix $R \in \mathbb{R}^{n \times n \times m}$, which denotes the set of information that is available, is obtained as:

$$R = Z \cdot Q, \tag{2}$$

where $\cdot$ represents the scalar product of two matrices, *i.e.*, $R(i,j,t) = Z(i,j,t) \cdot Q(i,j,t)$.

B. Solving the Matrix Completion and Inference Problems

Based on the previous traffic model, we define two different yet interconnected problems, *completion* and *prediction*, where the latter is also referred to as inference. The traffic matrix *completion* problem is defined as follows. Let R_t be the observed TM at time t and Q_t the actual TM. Let then D denote the set with all the entry points of R_t whose measurement is missing. In *matrix completion*, a mapping function F_1 must be found to constitute Q_t given as input R_t and the set D. Note that, conversely to other related studies [12], [17], in this paper, we do not limit this mapping function to be linear, and for this reason, we consider a model based on neural networks, as described in Section IV-C.

Then, we define the matrix *prediction* or matrix *inference* as follows. Given R_t as the the observed TM at time t and D as the set with all the entry points of R_t whose measurement is missing, in *matrix inference*, a mapping function F_2 must be found to constitute the actual TM at time $t+1$, Q_{t+1}, given as input R_t and D.

We summarize the main components of our architecture in Fig. 1. After extracting the information from the collected TM, we solve the two defined problems. It must be noted that, although the matrix prediction can potentially regard the entire TM, in this paper we limit our attention to matrix entries whose historical information is only partial. A large amount of literature has already addressed the problem of internet traffic prediction without considering missing data, achieving excellent results [20]–[22]. For this reason, we present a solution that is orthogonal to these traffic forecasting methods, and that can be used in conjunction with them for network planning optimizations.

IV. Predictive Model Design

In this section, we describe the model used to estimate the missing values within a traffic matrix. Such an estimator consists of an HMM model that dictates the evolution over time of traffic values and an autoencoder used to improve the performance of the HMM. *Part of our contribution in this paper is to improve traditional HMMs, and this is obtained by making use of adversarial autoencoder as an alternative method for the decoding problem.*

A. Hidden Markov Model Framework

Hidden Markov Models have received attention for traffic models given their ability to capture important traffic statistical characteristics with only a relatively small number of states [23], [24]. These studies have analyzed the efficacy of HMM in modeling the packet flow generated by an individual application or the aggregate traffic on a single channel. HMMs are built around the concept of *hidden* variable, x_i, and *observed* variable, y_i, the transition probability describing the dynamic behavior of the system, $p(x_{i+1} \mid x_i)$, and the emission probability describing how the system generates the observation based on the hidden variable, $p(y_i \mid x_i)$. To start the process, the model requires an initial state distribution, represented as $\pi(x_0)$. The key assumption in HMM is that the state evolves as a Markov process where the probability distribution of the current state only depends on the state of the previous epoch, *i.e.*, $p(x_i \,|\, x_{i-1}, ..., x_1) = p(x_i \,|\, x_{i-1})$. It has been shown how this first-order Markov process is sufficient for modeling temporal characteristics of the network channel [25], [26].

To describe the state evolution over time, a transition probability matrix (PM) is defined. Thus, PM is filled with all the transition probabilities $p(x_{i+1} \,|\, x_i), \forall x_i \in \chi$. Generally, PM, $\pi(x_0)$, and $p(y_i \,|\, x_i = j)$ are unknown, so we need to estimate them either using some parametric or data-driven approaches. Very often, each Hidden Markov Model can also be represented as $\lambda = (A, B, \pi)$, where A denotes the transition probability matrix, B refers to the emission probability matrix, and π is the vector containing the initial states probabilities. It should be noted that, despite the similarity, the PM matrix is completely different from the traffic matrix that we consider in our model, as the TM reports the number of bytes transmitted between a source and a destination, while the PM contains the transition probabilities among the states of HMM.

Moreover, HMMs are characterized by three basic problems: *training*, *likelihood*, and *decoding*. While *training* is common to other ML algorithms, in this paper we focus on the *decoding*, as *likelihood* is out of the scope (see [27] for further information). The *decoding* phase attempts to find the most likely hidden state sequence X given the observation sequence over time Y and the HMM model $\lambda = (A, B, \pi)$. The problem is usually solved by means of the *Viterbi* [9] algorithm for hidden state estimation, which uses a dynamic programming approach and a forward-backward method in order to maximize the likelihood of the whole generating state sequence. The problem of finding the most likely state sequence can be summarized as follows: given a sequence of observed values $(\widetilde{y}_0, \widetilde{y}_1, ..., \widetilde{y}_n)$, we would like to infer the corresponding hidden variable $\widetilde{x}_t$, *i.e.*,

$$\widetilde{x}_t \sim p(x_t \,|\, \widetilde{y}_t, ..., \widetilde{y}_0). \tag{3}$$

In H&S we design to replace the traditional Viterbi algorithm with an AAE, described in the following.

B. Learning with Adversarial Autoencoder

Adversarial AutoEncoder (AAE) has been firstly presented in [10] as a model that can turn an autoencoder into a generative model. Leveraging the more general approach of generative adversarial networks (GAN), AAE can perform variational inference by matching the aggregated posterior of the hidden code vector of the autoencoder with an arbitrary prior distribution.

In recent years GAN has been at the basis of a variety of alterations, giving rise to a large number of GAN-based models, such as CycleGAN [28], BiGAN [29], Super-Resolution GAN [30], to cite a few. This more general model is generally applied over bits of an image and is made up of two neural networks: a generator and a discriminator. The former accepts an input vector of randomly generated noise and produces an output "imitation" image that looks similar, if not identical, to the authentic image. The latter network attempts to determine if a given image is "authentic" or "fake". Similarly, in AAE, an autoencoder is trained with dual objectives—a traditional reconstruction error criterion and an adversarial training criterion. The presence of a discriminator and a latent space, along with a different training process, makes AAE different from the traditional autoencoder, *i.e.*, the well-known Variational autoencoder (VAE).

Generative models are often applied to capture rich distributions such as audio, images, or video, and to generate a synthetic version. Following the recent and mostly unexplored trend of applying these models in disparate domains [31],

[32], we consider this class of problems for Markovian environments. Specifically (as suggested in [32]), we use the autoencoder to map some of the available entries of the TM (observed state) to the missing values (hidden state), following the HMM modeling.

After a necessary training phase, the encoder learns to convert input data to an intermediate representation, while the decoder learns a deep generative model that maps this intermediate representation to the final output. In our scenario, the input data represents the traffic data coming with partial information, R, while the output data distribution is the complete traffic matrix with reconstructed values, Q. The generator of the adversarial network is also an encoder that helps training by drawing samples to create a posterior distribution that can fool the discriminator network into thinking that the hidden code comes from the true prior distribution.

C. H&S Procedure

We can now define the observed and hidden states in our specific model, which are then used to solve the matrix completion and inference tasks.

H&S states variables. For each missing value at time t we define a current evidence y_t as a square matrix with dimension $k \times k$. Such submatrix represents the elements surrounding the missing entry of the position (i, j) that we are interested in estimating. The hidden state value x_t, instead, is a single value that represents the missing entry of TM at position (i, j). Note that the number of observed states y_t is affected by the dimension of the submatrix, k, and is a crucial parameter to specify during the design of the model. While modeling a single hidden value is a classical procedure in HMM [9] models, there is a tradeoff in choosing the best matrix size k. A smaller size implies a simpler model but may yield an inadequate representation of the space of possible behaviors, accounting for insufficient spatial similarities. On the other hand, a large k leads to a more complex model with more parameters but may, in turn, lead to overfitting. In our validation, we used a 7×7 submatrix, as a result of a cross-validation study used to learn this critical parameter (herein omitted due to space limits).

Pre-processing. As a best practice, before using the traffic quantities in our model, they must be prepared. Data preparation involves using normalization or other standardization techniques to re-scale input and output variables prior to training the ML model. For this reason, we apply a standard normalization approach to scale input values into $0-1$ range, which makes the model more general and transferable over different networks. Moreover, to inform the model about missing values, we apply the concept of masking, which consists in marking the locations of the input space to be ignored using an identifiable value, e.g., -1. The AAE model, then, always expects the same number of inputs ($k \times k$) but can distinguish between measured values and missing ones. This procedure can be easily generalized when there exist multiple missing values to estimate.

HMM parameters. Another parameter to define is the number of possible values for hidden states, which affects the dimensions of the transition probability matrix (PM). This design is made more difficult since we are dealing with continuous values for the bytes of traffic, and we are also interested in predicting the future values where an entry is missing. For this reason, we decide to use an approach where the computation of the future value x_{t+1} is equivalent to estimating the difference with respect to the last hidden state x_t. As we have normalized values, this difference resides within the interval $[-0.9, 0.9]$. Besides, since the PM must be limited, we only consider 50 possible values inside this interval so that the prediction is accurate, but the problem is treatable. In other words, we compute the probability over a discrete set of possible evolutions rather than directly predicting future traffic. We referred to this set as E, and the value at time t is e_t. Hence, the PM reports the probabilities that the next hidden state is obtained by the current state added to the value in this set e_t, where considered traffic values have been opportunely normalized.

Matrix completion. In view of the foregoing, we are now able to describe the procedure dictating the H&S behavior and functionalities. The traffic matrix completion task is converted into the decoding problem of HMM, where the objective is to find the value of a missing entry (*hidden state*), given as input the adjacent submatrix (*observed matrix*). Using the aforementioned notation, the task becomes: given the observation y_t at time t, the traffic matrix is completed by finding the hidden x_t. Besides, using AAE to learn this mapping function, H&S is also able to deal with non-linear relationships, which allows generalizing our model over a broader area of traffic conditions.

Future traffic prediction. Along with the matrix completion at time t, it may be required to predict the future state at time $t + 1$. In this case, it is a task of computing posterior distribution over the future states, given evidence (i.e., current TM) and past evolution till now. We derive the predicted next hidden state as the one with the highest probability, according to:

$$\begin{aligned} e_{t+1} &= \arg\max_{e_{t+1}} p(e_{t+1} \mid \widetilde{y}_t, ..., \widetilde{y}_0), \\ \widetilde{x}_{t+1} &= \widetilde{x}_t + e_{t+1}. \end{aligned} \tag{4}$$

Since in our HMM model we consider the evolution of the hidden states in terms of difference to the previous step, we need first to determine this difference, e_{t+1}, and then add this value to the traffic value at time t, *i.e.*, x_t. In other words, given the sequence of observations at time t, we decode them into the hidden state using AAE, and we forecast the next hidden variable using the maximization of posterior probabilities.

V. Evaluation Results

In this section, we quantify the benefits brought by our *Hide & Seek* algorithm in the resolution of the traffic matrix completion problem.

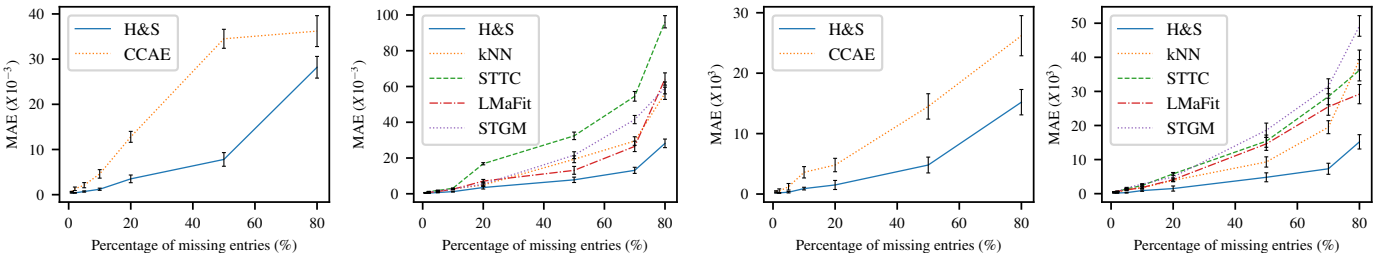

Fig. 2: **Abilene network.** (a) MAE error for autoencoder methods and (b) other benchmark solutions in completing the traffic matrix. **GEANT network.** (c) MAE error for autoencoders and (d) other benchmark solutions in traffic matrix completion.

A. Experimental Settings

Internet traffic traces. In our trace-driven evaluation we used two publicly available datasets. The first was the GEANT [33] dataset, consisting of traffic matrices built using full routing information of 23 routers, sampled Netflow data, and routing information of the European GEANT network, with a sampling interval of 15 minutes and duration of one week. The second set of traces was imported from the Abilene traffic matrix dataset [34], a backbone network consisting of 11 nodes of major cities in the USA. In this case, we used one week of traffic collected with a granularity of 10 minutes for a total of 1008 time intervals.

Benchmark algorithms. To demonstrate the effectiveness of our proposed H&S for matrix completion, we compare its performance with the following algorithms, opportunely adapted to our context. First, *CCAE* [35], an algorithm that transforms the recovery problem to images inpainting, a computer vision technique used to reconstruct missing segments in images. CCAE reconstructs the missing values using cascaded convolutional autoencoders, where matrices are regarded as "generalized" images. The second benchmark algorithm is the *Spatio-Temporal Tensor Completion* method, or *STTC* [17]. This method models network traffic as a tensor pattern, projecting tensors into a lower-dimensional latent space via tensor factorization, while preserving the multi-way nature of the network traffic data. Third, *LMaFit* [15], the Low-rank Matrix Fitting (LMaFit) algorithm. This predictor, one of the most commonly used methods for general matrix completion, solves a low-rank factorization model for matrix completion by applying a successive nonlinear over-relaxation. Forth, *STGM* [16], a recently developed algorithm in which the rows of a TM are clustered into K subgroups, and the TM is divided into many subparts with spatial similarities. The matrix is then completed by Bayesian inference, using the multi-Gaussian models obtained in the previous steps. Fifth, the classical k nearest neighbors (*kNN*), where we assume that the missing values of the TM are predicted by local interpolation of the targets associated to the nearest k neighbors [36]. As in our AAE case, we use a weighted average with $k = 7$. Sixth, *ConvLTSM*, a recent solution that integrates a Convolutional Neural Network (CNN) model and a Long Short-Term Memory (LSTM) network for spatiotemporal modeling and estimating the future network traffic [7].

B. Matrix Completion Performance on Random Loss Patterns

We now study the performance of our traffic matrix completion algorithm when varying the amount of known information. To this end, we hide data points independently at random, to evaluate the completion performance. Missing values range from 1 to 80 percent of the total entries.

Starting with the Abilene network dataset, we compare our solution against a similar approach, also based on autoencoders, as in the CCAE solution. Our AAE-based traffic matrix completion method outperforms the benchmark CCAE (Fig. 2a). This is because of the diverse machine learning model and how data are treated to fill the missing traffic matrix cells. In particular, our approach can handle a considerable percentage of missing entries, conversely to CCAE. We have experienced how the performance of CCAE largely depends on the position of the missing entries in the matrix, and its masking model poorly scales when the majority of the elements is unknown.

To validate this result, we then consider all other benchmark algorithms for matrix completion, reporting the results in Fig. 2b. The MAE error provided by H&S is the lowest among all the percentages of missing entries. Further, the error achieved is also minimal, considering the traffic values present in the matrices. This outcome is particularly important as it demonstrates how this technique can be used in real deployments to take network decisions even when the available information is incomplete.

To generalize these findings, we then perform the same set of experiments over the GEANT traffic data, comparing the ability of the two diverse autoencoders to complete the traffic matrix (Fig. 2c). While the error achieved for matrix completion is higher compared to the Abilene use case, due to the different order of magnitude of values themselves, our adversarial autoencoder method is still more effective than CCAE. Besides, H&S shows the same ability to handle a considerable amount of missing data. Similar considerations are valid when comparing our model against other related benchmarks, as seen in Fig. 2d. Other solutions, such as kNN and LMaFit, are able to provide a limited error either in one scenario or the other, but not consistently.

These results confirm our hypothesis that a GAN-based model can learn even when no guarantees of special properties between the cells of the traffic matrix hold.

VI. Conclusion

In this paper, we presented *Hide & Seek*, a method to efficiently achieve traffic matrix completion and inference, built on top of an HMM-based approach, but replacing the traditional encoding algorithm with an Adversarial AutoEncoder (AAE). We used our algorithm to estimate the missing entries in the traffic matrix and to predict their values in the short horizon. Our evaluation, performed over two publicly available real datasets, *i.e.*, Abilene and GEANT networks, validate the performance of our approach, highlighting the efficacy of AAE in computing the missing values starting from a limited set of information. Testing the generality of *Hide & Seek* to improve upon traditional HMMs over diverse traffic loads will be investigated as future work.

References

[1] M. Mardani and G. B. Giannakis, "Estimating traffic and anomaly maps via network tomography," *IEEE/ACM transactions on networking*, vol. 24, no. 3, pp. 1533–1547, 2015.

[2] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (IMC '05)*, 2005, pp. 331–344.

[3] V. Bharti, P. Kankar, L. Setia, G. Gürsun, A. Lakhina, and M. Crovella, "Inferring invisible traffic," in *Proceedings of the 6th International Conference*, 2010, pp. 1–12.

[4] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, 2009, pp. 267–278.

[5] Q. Zhao, Z. Ge, J. Wang, and J. Xu, "Robust traffic matrix estimation with imperfect information: Making use of multiple data sources," in *Proceedings of the joint international conference on Measurement and modeling of computer systems (SIGMETRICS '06)*, 2006, pp. 133–144.

[6] Z. Liu, Z. Wang, X. Yin, X. Shi, Y. Guo, and Y. Tian, "Traffic matrix prediction based on deep learning for dynamic traffic engineering," in *IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2019, pp. 1–7.

[7] P. Le Nguyen, Y. Ji *et al.*, "Deep convolutional lstm network-based traffic matrix prediction with partial information," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 261–269.

[8] F. Xiao, L. Chen, H. Zhu, R. Hong, and R. Wang, "Anomaly-tolerant network traffic estimation via noise-immune temporal matrix completion model," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1192–1204, 2019.

[9] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[10] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[11] G. Gürsun and M. Crovella, "On traffic matrix completion in the internet," in *Proceedings of the Internet Measurement Conference (IMC '12)*, 2012, pp. 399–412.

[12] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 161–174, 2002.

[13] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[14] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[15] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm," *Mathematical Programming Computation*, vol. 4, no. 4, pp. 333–361, 2012.

[16] H. Zhou, D. Zhang, and K. Xie, "Accurate traffic matrix completion based on multi-gaussian models," *Computer Communications*, vol. 102, pp. 165–176, 2017.

[17] H. Zhou, D. Zhang, K. Xie, and Y. Chen, "Spatio-temporal tensor completion for imputing missing internet traffic data," in *Proceedings of the 34th international performance computing and communications conference (IPCCC)*. IEEE, 2015, pp. 1–7.

[18] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices (extended version)," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, pp. 662–676, 2011.

[19] P. Tune and M. Roughan, "Spatiotemporal traffic matrix synthesis," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication (SIGCOMM '15)*, 2015, pp. 579–592.

[20] L. Nie, D. Jiang, S. Yu, and H. Song, "Network traffic prediction based on deep belief network in wireless mesh backbone networks," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–5.

[21] A. Sacco, F. Esposito, and G. Marchetto, "Rope: An architecture for adaptive data-driven routing prediction at the edge," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 986–999, 2020.

[22] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2017, pp. 2353–2358.

[23] K. Salamatian and S. Vaton, "Hidden markov modeling for network communication channels," *ACM SIGMETRICS Performance Evaluation Review*, vol. 29, no. 1, pp. 92–101, 2001.

[24] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, and B. Sinopoli, "Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction," in *Proceedings of the 2016 ACM SIGCOMM Conference on Data communication*, 2016, pp. 272–285.

[25] J. Liu, I. Matta, and M. Crovella, "End-to-end inference of loss nature in a hybrid wired/wireless environment," in *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003, pp. 1–9.

[26] S. Tao and R. Guérin, "On-line estimation of internet path performance: an application perspective," in *IEEE INFOCOM 2004-IEEE Conference on Computer Communications*, vol. 3. IEEE, 2004, pp. 1774–1785.

[27] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[28] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[29] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016.

[30] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[31] K.-Y. Chen, C.-P. Tsai, D.-R. Liu, H.-Y. Lee, and L.-s. Lee, "Completely unsupervised phoneme recognition by a generative adversarial network harmonized with iteratively refined hidden markov models," in *INTERSPEECH 2019 - Annual Conference of the International Speech Communication Association*, 2019, pp. 1856–1860.

[32] A. Sacco, F. Esposito, and G. Marchetto, "Restoring application traffic of latency-sensitive networked systems using adversarial autoencoders," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2521–2535, 2022.

[33] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 83–86, 2006.

[34] Y. Zhang, Z. Ge, A. Greenberg, and M. Roughan, "Network anomography," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (IMC '05)*, 2005, pp. 317–330.

[35] X. Wang, Y. Chen, W. Ruan, Q. Gao, G. Ying, and L. Dong, "Intelligent detection and recovery of missing electric load data based on cascaded convolutional autoencoders," *Scientific Programming*, vol. 2020, 2020.

[36] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.