ResCue: Inferring Fine-Grained Traffic Matrices via Distributed Deep Residual Networks

Lorenzo Pappone†, Cristian Zilli*, Alessio Sacco*, Flavio Esposito†
† *Department of Computer Science*, Saint Louis University, USA
* *Department of Control and Computer Engineering*, Politecnico di Torino, Italy

***Abstract*— Network measurement and telemetry techniques are central to the management of modern computer networks. Internet traffic matrix estimation is a popular technique employed for network management and telemetry to reconstruct missing information. Existing approaches use statistical methods, which often make impractical assumptions about the structure of the Internet traffic matrix. Data-driven methods, instead, heavily rely on the assumption of full knowledge of network topology data, that may be unavailable or impractical to collect. In this work, we propose *ResCue*, a deep residual networks technique to infer fine-grained Internet network traffic starting from spatial coarse-grained measurements. To address scenarios with network visibility constraints, we design a federated learning approach for fine-grained traffic prediction with partial network knowledge. Our evaluation across real-world traffic data shows that our proposed approach outperforms existing interpolation techniques and that our federated learning design achieves similar accuracy with respect to its centralized counterpart while requiring only partial knowledge of the network.**

***Index Terms*—traffic prediction, super resolution, deep learning, federated learning**

I. Introduction

Optimizing network traffic efficiency often requires sophisticated analytics and timely problem-solving, especially after readily achievable improvements have been exhausted. The complexity of modern networks poses challenges for effective data collection, particularly when it comes to identifying, monitoring, and debugging issues in hidden network segments. As the Internet becomes an increasingly distributed and federated infrastructure for web services, visibility into its traffic characteristics is hindered by several factors: from business intelligence to prohibitive computational cost and storage requirements associated with collecting Internet traffic data at scale [1], [2], [3].

Even after years of extensive research in this area, network operators and DevOps continue to depend on fine-grained metrics for their applications and troubleshooting. Moreover, even the most accurate network measurement systems are prone to inaccuracies or data amputations. Consequently, it's often necessary to refine the Traffic Matrix (TM) of interest to a more complete state before it can be an input to various debugging applications, or serve as complete training data for data mining algorithms, including those based on Deep Learning for network management automation or semi-automation tasks [4].

Recent works highlighted the efficacy of Machine Learning (ML), in general, and Neural Network (NN) architectures, in particular, in the estimation of traffic matrices. These include approaches leveraging Deep Neural Networks (DNNs) [5], Recurrent Neural Networks (RNNs) [6], and Convolutional Neural Networks (CNNs) [7], to reconstruct missing cells in TMs [8], [9], [10]. While these solutions yield promising results, they still assume access to fine-grained measurements, which might not always be available [11], [12]. In addition, partial visibility is often inherently restricted due to architectural, privacy, or regulatory constraints. These scenarios typically involve multiple entities or network segments that cannot or should not share raw traffic data [13].

In this work, we introduce *ResCue*, a novel algorithmic approach that leverages *Deep Residual Networks* for inferring real-time network monitoring and telemetry, focusing on the fine-grained prediction of Internet traffic matrices. Unlike traditional methods, which rely on direct and high-frequency measurements, our method uses aggregate, coarse-grained data to estimate high-resolution traffic matrices with minimized inference error. Additionally, to further address scenarios marked by very limited visibility, we propose a distributed approach based on the Federated Learning (FL) procedure [14]. This extension allows us to train *ResCue* across multiple network partitions, each possessing only partial visibility of the global traffic patterns.

Somehow surprisingly, our findings demonstrate that even in the absence of specific data proximity features — commonly utilized in classical super-resolution methods for computer vision — TMs retain essential characteristics that allow for accurate inference. Tests on real-world datasets, i.e., GÉANT [15] and Meta [16] validate the effectiveness of our approach for network traffic prediction compared to other super-resolution techniques. Moreover, we demonstrate its effectiveness when distributing the learning process among multiple clients with partial or no network visibility, making the solution more scalable and efficient.

The rest of the paper is organized as follows. Section II discusses the related work and in Section III we formulate the traffic matrix super-resolution problem, as an extension of the traffic matrix inference problem. In Section IV we detail our model and the FL approach. In Section V we describe the datasets and our data preparation methodology, and in Section VI we discuss our evaluation results, showing the benefits both in a centralized and federated fashion. Finally, in Section VII we present our conclusion.

II. Related Work

The problem of traffic matrix (TM) estimation has received considerable attention within the field of computer networking, and it has been approached from various perspectives due to its wide-ranging applicability for web and other Internet services [17], [18].

A range of methods for predicting such matrices is based on the incorporation of side information from different sources, such as total incoming bytes and a number of customers [19]. Approaches rooted in network tomography theory have traditionally been proposed as solutions: these methods involve, for example, describing the relationship between link loads and the end-to-end traffic flows in the network, and generally require topological information and heuristically defined constraints to function [20], [21]. Tomographic models have thus often been paired with algorithms (e.g., Compressive Sensing [22]) in order to overcome the obstacles the former entails. In the same vein, data-driven solutions have also been explored as a way to estimate end-to-end flows from link load measurements [23] or from more granular (in time) information [24]. An alternative approach comprises reconstructing TMs starting from partial traffic information, exploiting either spatial or temporal patterns. To predict future traffic values, Autoregressive Integrated Moving Average (ARIMA)-based approaches have been proposed [25], [26]. However, they struggle when capturing the nonlinear complexities of network traffic. To solve this issue, recent Machine Learning (ML) and Deep Learning (DL) techniques have shown great promise in tackling these nonlinear issues. The authors in [5] demonstrated the superior capacity of DL models to unearth the nonlinear characteristics of network traffic, as opposed to ARIMA models. Following this line of work, various approaches make use of Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNNs) to reconstruct TMs either spatially or temporally [6], [27], [28], [29], [9]. Differently from these solutions, we attempt to solve this problem from a new angle that answers to this research question: *Can we infer fine-grained measurements starting from coarse-grained ones with partial network knowledge?* The fine-grained prediction task finds similarities in the computer vision fields, where often it is the case to learn complex mapping functions that upgrade low-resolution images to high-resolution images [30], [31], [32], [33]. Within the networking field, Zhang et al. [34] leverage a GAN-based super-resolution technique to infer mobile network traffic from coarse-grained measurements. Unlike ours, their approach relies on spatial information specific to cellular traffic and assumes full visibility of the network data, often unrealistic in many practical scenarios, e.g., cloud environments with data isolation policies, multi-tenant datacenter networks, or ISP networks with multiple administrative domains. In conclusion, our work is the first attempt to address TM prediction from a fine-grained perspective while operating on aggregated coarse-grained network traffic measures with partial network visibility.

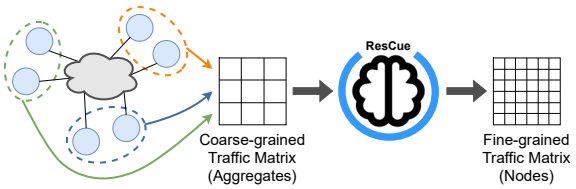


Fig. 1: Overview of *ResCue*. Our solution takes traffic matrices built with aggregate flows from geographically clustered network nodes and super-resolves them into fine-granularity traffic matrices indexed by single network nodes.

III. Problem Definition

A traffic matrix is a data structure used in network engineering that represents the volume of traffic between different points in a network. Specifically, it quantifies the amount of data being sent from each origin node to each destination node within a given timeframe. The matrix is often used for a variety of network planning, optimization, and management tasks, including capacity planning, load balancing, and fault diagnosis. We define a TM as a 2-dimensional array $M \in \mathbb{R}^N \times \mathbb{R}^N$, where N is the number of nodes in the network. We denote with $M(i, j; t)$ the traffic from node i to j, averaged over the time interval $[t, t+\Delta t)$, defined as the aggregation granularity.

With this setting, at a given interval $[t, t + \Delta t]$, each element of a matrix $M(i, j)$ indicates the volume of data, in bytes, transmitted from node i to node j within the specified measurement interval.

A. Traffic Matrix Inference

With ResCue, we aim to infer network traffic data at a high spatial resolution (i.e., fine-grained), using as a starting point measurements collected at a lower spatial resolution (i.e., coarse-grained) in the network. We represent this problem in Fig. 1, highlighting how the aggregated matrix can combine information from different regions/operators. Specifically, spatially fine-grained data refers to traffic measurements between individual nodes or endpoints in the network. This might represent, for example, rack-to-rack traffic flows in a datacenter or the traffic between specific hosts or points of presence in a wide-area network (WAN). Spatially coarse-grained (low-resolution) data represents aggregated traffic measurements over larger network areas or clusters - i.e., pod-level traffic in a datacenter or regional traffic in a WAN, where multiple individual nodes are grouped together.

To formally define the traffic matrix inference problem we address in this paper, let M^{LR} denote the low-resolution (LR) traffic matrix, and M^{HR} the high-resolution (HR) traffic matrix. The goal is to recover an approximation $\widehat{M}^{HR}$ of the ground truth HR traffic matrix, starting from its LR version M^{LR}:

$$\widehat{M}^{HR} = F(M^{LR}; \theta), \tag{1}$$

where F is the super-resolution model and θ represents its parameter vector. The recovered HR traffic matrix $\widehat{M}^{HR}$ is also denoted as *super-resolved traffic matrix* M^{SR}. Our

traffic matrix estimation problem is modeled by the following *optimization*:

$$\widehat{\theta} = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\widehat{M}^{HR}, M^{HR}) + \lambda\Phi(\theta), \tag{2}$$

where $\mathcal{L}(\widehat{M}^{HR}, M^{HR})$ is the loss function between the predicted HR traffic matrix $\widehat{M}^{HR}$ and the ground truth traffic matrix M^{HR}, $\Phi(\theta)$ is a regularization term and λ is the trade-off parameter. We solve this problem using a deep learning approach, and we adopt the mean absolute error (MAE) loss function to measure the quality of our super-resolution traffic matrix inference. Unlike other metrics, such as Mean Squared Error (MSE), MAE is less sensitive to outliers and therefore offers a more robust evaluation of our model's performance in generating fine-grained network traffic predictions.

B. Inferring Traffic With Partial Network Visibility

Deep learning models typically require comprehensive network data to achieve high accuracy and generalizability. In ideal scenarios, they are trained on complete network traffic matrices, which is not always feasible or realistic in large-scale, complex network environments. In many practical scenarios, different parts of the network may be managed by separate entities, each with only partial visibility of the overall network traffic. Our work addresses this challenge by integrating our model with a federated deep learning approach. In our framework, each client represents a portion of the network with limited visibility, possessing only a subset of the complete traffic matrix.

We distribute non-overlapping portions of the fine-grained traffic matrix across different clients, each representing a segment of the network. These fine-grained sub-matrices serve as the ground truth for each client. Each client then creates its own coarse-grained traffic matrices corresponding to its fine-grained sub-matrix (see Section V-A), which are used as inputs to predict the fine-grained ones. Our aim is to reconstruct the original fine-grained matrix by utilizing the model parameters learned by these distributed agents.

The fine-grained matrix is split into smaller sub-matrices, each of which is $\frac{m}{\sqrt{N}} \times \frac{m}{\sqrt{N}}$ (assuming N is a perfect square for simplicity), where m is the total number of nodes in the fine-grained traffic matrix. Let $\sqrt{N} = k$, then each client n with $1 \leq n \leq N$ would receive a portion of the fine-grained traffic matrix defined as

$$M_{(i,j)} = M\left(\left[\frac{(i-1)m}{k} + 1, \frac{is}{k}\right], \left[\frac{(j-1)m}{k} + 1, \frac{jm}{k}\right]\right), \tag{3}$$

where $n = (i-1) \times k + j$.

The model is then distributed among the involved N clients, and over T rounds of aggregations, a global model is derived and used to predict the full fine-grained data. Throughout this process, each client learns to infer traffic patterns based only on its own portion of the network, contributing to the overall global model, as per the procedure detailed in Section IV-B.

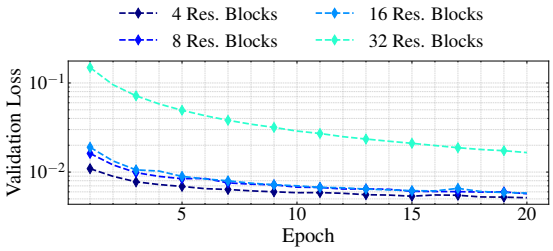


Fig. 2: Impact of the number of residual blocks on the learning procedure.

IV. Traffic Matrix Estimation with Deep Residual Networks

In the field of machine learning, Deep Residual Networks (ResNets) have emerged as a groundbreaking innovation, particularly designed to alleviate the vanishing and exploding gradient issues plaguing deep networks [30], [35], [31], [36]. Central to the ResNet paradigm is the concept of "residual learning," wherein each layer learns not the direct output mapping but rather the residual (or difference) between the input and the desired output. Mathematically, if $H(x)$ represents the ideal mapping, ResNets aim to approximate $F(x) = H(x) - x$, subsequently recovering $H(x)$ through the sum $F(x) + x$. A residual block consists of multiple convolutional layers interjected by batch normalization and ReLU activation functions.

We studied the impact of the number of residual blocks on the training procedure, reporting results in Fig. 2. Contrary to other studies, e.g., the findings in [30], a deeper architecture does not necessarily yield better results for our dataset. In fact, increasing the number of residual blocks tends to induce significant overfitting, especially when dealing with sparse traffic matrices. The final output is obtained by adding this block's output to its input, followed by an activation function. The residual learning mechanism thus equips ResNets with the capability to form much deeper networks, promoting easier training and more robust feature representations, thereby setting new benchmarks across a wide range of machine learning tasks. Inspired by [36], we propose a Deep Residual Network architecture that infers fine-grained network traffic volume from coarse-grained aggregates.

A. Model Architecture

Motivated by results of Fig. 2, we design our core model with $N_b = 4$ residual blocks. Each residual block comprises of two convolutional layers separated by a Parametric ReLU [37] activation function. Each convolutional layer is equipped with small 3×3 kernels and 64 feature maps, followed by a batch normalization layer. The merged feature map then undergoes up-sampling to match the desired scale. The up-sampling is realized through a sub-pixel convolution layer [38].

B. Federated Training

Finally, we leverage a distributed learning technique to train our super-resolution mode even in scenarios of partial network visibility, where complete knowledge of the entire network is unavailable, and only non-overlapping segments of

the network are accessible. In this setup, we aim to evaluate how effectively the distributed training approach allows insight into network traffic from other clients, even in the absence of comprehensive global knowledge of the network. By limiting visibility to only a sub-portion of the entire network, each federated client will have to monitor, aggregate, and process only a limited amount of data, leading to a reduction in the costs of communication and storage of the network traffic information collected.

We formulate our distributed learning problem as a three-phase procedure: Initialization, Aggregation, and Update. During Initialization, each client receives a pretrained global model ω_t from the aggregator. The clients then train this model using their local data D_k, which consists of coarse-grained (i.e., training set) and fine-grained (i.e., ground truths) measurements. During the Aggregation phase, clients send their local gradients to an aggregator, either a central entity or one client serving as aggregator. The average loss over the client's local dataset, for each client k, is given by:

$$\min_{x \in \mathbb{R}^d} F_k(x) = \frac{1}{D_k} \sum_{i \in D_k} E_{z_i \sim D_k} f(x; z_i) + \lambda h(x), \quad (4)$$

where $f(\cdot;\cdot)$ is the local loss function, λ is a regularization term, $E_{z_i \sim D_k}$ stands for the analytical expected local loss function, and $h(x)$ is a regularizer function. In the Update phase, the aggregator uses the Federated Averaging (FedAvg) algorithm [39] to update the global model ω_{t+1} for the next iteration as follows:

$$\omega_{t+1} \leftarrow \omega_t + \frac{1}{N} \sum_{n=1}^{N} F_n^{t+1} \quad (5)$$

After training, we use the final global model to predict the original fine-grained matrices starting from the original coarse-grained ones, effectively super-resolving the entire network traffic matrix.

V. Data Processing and Methodology

To validate effectiveness of ResCue, we evaluate our model over both datacenter and WAN traffic data. We consider the GÉANT network dataset [15] and the Meta datacenter traffic data [16]. GÉANT dataset includes $10,772$ traffic matrices constructed using Interior Gateway Protocol (IGP), NetFlow data aggregated from all edge links, and Border Gateway Protocol (BGP) from the GÉANT network. Meta datacenter production traffic includes hundreds of thousands of 10-Gbps nodes. The dataset is obtained through a combination of Meta-wide monitoring systems and per-host packet header traces, and collected over a 24-hour span from three different clusters—Frontend, Database, and Hadoop—located in Facebook's Altoona Data Center. In this work, we only consider intra-cluster traffic.

A. Coarse-grained Traffic Matrix Generation

In this work, we consider coarse-grained measurements, i.e., spatially aggregated traffic volume, to build a training set for the super-resolution model. While in datacenter architectures, we naturally have access to hierarchical traffic data - i.e., from pod-level (coarse-grained) to rack-level (fine-grained) measurements, where a pod consists of multiple racks (typically 10-48 racks per pod, depending on the specific design), this is not the case for GÉANT network dataset. To generate the datasets for experiments, we had to perform pre-processing, which we detail herein.

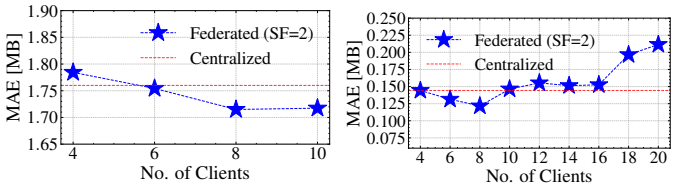
WAN Traffic. To adapt our inference problem to (GÉANT) WAN dataset, we employ a clustering procedure where geographically proximate nodes are grouped together, simulating a coarse view of the network. In particular, we use K-means clustering to group network nodes, where the number of clusters K is determined by our desired scale factor s, where $K = N/s$ and N is the total number of nodes. Once the clusters are formed, we aggregate the traffic between clusters by summing the traffic between all pairs of nodes belonging to different clusters to generate LR matrices. Formally, we create the coarse-grained matrix M_c by aggregating traffic volumes between clusters: $M_c(i,j) = \sum_{n \in C_i, m \in C_j} M_f(n,m)$, where C_i and C_j represent the sets of nodes in clusters i and j respectively.

Datacenter Traffic. Given the presence of multi-level traffic data, we simply employ a hierarchical approach that leverages the natural rack and pod structure. We first create the fine-grained traffic matrix M_f by aggregating rack-to-rack traffic volume using a time window of 10 seconds. To create the coarse-grained matrix, we select N racks per pod and aggregate their traffic in order to preserve a fixed upsampling factor for our super-resolution model. This results in a pod-level matrix $M_c^{(1)}$ where $T_c^{(1)}(i,j) = \sum_{r \in R_i, s \in R_j} M_f(r,s)$, with R_i and R_j representing the sets of N selected racks in pods i and j respectively. For even coarser representations, we further aggregate traffic at the pod level. Given a scale factor k, we combine traffic from k adjacent pods: $M_c^{(k)}(m,n) = \sum_{i \in P_m, j \in P_n} M_c^{(1)}(i,j)$, where P_m and P_n are sets of k adjacent pods. This process yields a series of coarse-grained matrices $M_c^{(k)}$ of size $|P|/k \times |P|/k$, where $|P|$ is the total number of pods. The resulting matrices serve as inputs for our super-resolution model, aiming to reconstruct finer-grained rack-level traffic from the corresponding aggregated representations.

Data Normalization. For both datasets, we apply min-max normalization to both granularities and generate input-output pairs for each time step t, splitting them into training, validation, and test sets (70:15:15 ratio). Each traffic matrix is further subjected to quantile clipping and square root normalization operations to remove outliers and normalize the value distribution.

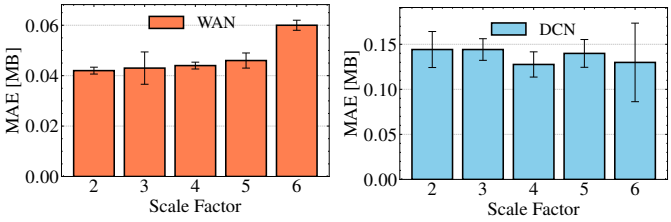
VI. Evaluation Results

Next, we describe the experimental settings and discuss the performance of our proposed approach for fine-grained traffic matrix prediction.



(a) GÉANT Network Traffic (b) Meta Datacenter Traffic

Fig. 3: Performance in terms of MAE (MB) of ResCue varying the number of clients of the federated settings with scale factor 2, compared with the fully centralized model.



(a) GÉANT Network Traffic (b) Meta Datacenter Traffic

Fig. 4: Evaluation of ResCue (centralized) in terms of MAE, varying the scale factor. Each scale factor corresponds to a different resolution of the coarse-grained traffic matrices. Lower values indicate better performance. (a) Performance over Meta datacenter network production traffic. (b) Performance over GÉANT network traffic (i.e. WAN).

A. Matrix Inference Baselines

In computer vision, super-resolution has been successfully used for tasks such as enhancing medical images, improving satellite imagery, and upscaling low-quality video content. Despite its widespread use in these areas, the application of super-resolution techniques to networking data, particularly traffic matrices, remains largely unexplored. Network traffic data presents unique challenges due to its temporal nature, sparsity, and complex patterns that differ significantly from image data. We compared our model against popular super-resolution architectures from the field of computer vision [40], [33], [30]. Very Deep Super-Resolution (VDSR) [30] employs a very deep convolutional network, inspired by VGG-net, with a final model comprising 20 weight layers with cascading small filters. Super-Resolution Convolutional Neural Network (SRCNN) [33] is a benchmark deep learning architecture used for super-resolution, consisting of three convolutional layers. We further consider Bicubic interpolation [40], a classical spatial interpolation technique often used in image processing, and a recent interpolation method for fine-grained traffic estimation [24]. Specifically, the latter reconstructs end-to-end traffic matrix in finer time granularity from sampled traffic traces, using a combination of fractal interpolation, cubic spline interpolation, and the weighted geometric average algorithm. For simplicity, we refer to it as Fractal. Although this work relies on time granularity, we adapt their interpolation technique to our spatial case.

B. Evaluation Metrics

In this work, we evaluate our approach leveraging the Mean Absolute Error (MAE) as the main error metric. As a commonly used metric for regression problems, MAE measures the average magnitude of the errors between the predicted and actual values, without considering their direction. However, in this work we consider a weighted version of MAE, defined as follow:

$$\text{WMAE} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i |y_i - \hat{y}_i|$$

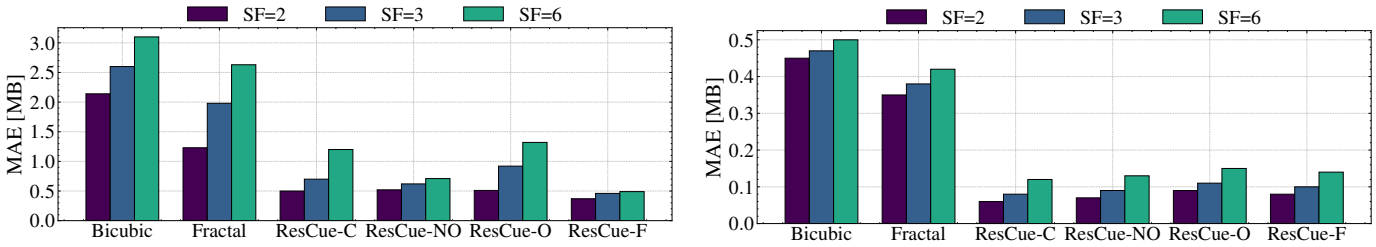
Where y_i and $\hat{y}_i$ are the actual and predicted values, respectively, w_i are the weights associated with each observation (with higher weights assigned to peak traffic volumes), and n is the number of samples. Given the sparse nature of the traffic matrix data, we aim to measure the inference accuracy by penalizing models that make mistakes over peak traffic volumes. In the rest of the paper, we will use MAE to denote the weighted-MAE unless indicated otherwise.

C. Experimental settings

In our work, we evaluate our approach by following a centralized and distributed setting for the training process. For the centralized training, we leverage Adam optimizer, with an initial learning rate set to 10^{-3}, and a learning rate scheduler to halve it to 5×10^{-4} after 10^5 minibatch updates. The model is trained with a minibatch size of 16 and a L1 loss function for optimization. Further, we train our model for 50 epochs. We deploy two different optimizers for federated training: each client uses an Adam optimizer with a learning rate of 0.001 locally, whereas the aggregator uses a Stochastic Gradient Descent (SGD) procedure with a learning rate of 1.0 to update the global model. Moreover, our federated training process leverages a minibatch of size 16 and a $L1$ loss function. We run each federated training for 100 rounds. Experiments are conducted on a server equipped with 4 Nvidia Tesla T4 GPUs.

D. Impact of the Number of Clients

We start evaluating the impact of the number of clients during the training in our federated setting, measuring the performance of the model on unseen fine-grained traffic matrices across both WAN and datacenter network traffic (Fig. 3). Before the training, each client receives a unique, non-overlapping portion of the original fine-grained traffic matrix that corresponds to the ground truth data (Section V-A). Subsequently, clients generate their own coarse-grained versions of the fine-grained sub-matrix. We fixed the scale factor at 2 for each client. The number of submatrices is proportional to the number of clients of the federated setting. Since the GÉANT network has only 22 hosts, in order to distribute one unique sub-matrix to each client, we do no consider more than 10 clients, as the coarse-grained data generation produces 3×3-size images. A smaller size for the training set would not be significant for our evaluation results.



(a) GÉANT Network Traffic.

(b) Meta DCN Traffic.

Fig. 5: Performance comparison of methods in terms of MAE across different scale factors (SF), over (a) GÉANT network and (b) Meta DCN datasets. ResCue variants demonstrate superior performance compared to traditional (Bicubic) and fractal-based approaches.

The results for WAN traffic dataset show a clear decreasing trend, where error values tend to lower while increasing the number of clients. The federated process effectively creates a composite model that encapsulates the knowledge gained by each client about its specific portion of the network. Client's traffic matrices have a smaller size than the fine-grained one, which is determined from the number of clients itself, resulting in an advantage in terms of pattern discovery for the super-resolution model.

We further evaluate ResCue over the Meta datacenter traffic and vary the number of clients within a wider range (i.e., 4 to 20 clients), given the significantly larger number of nodes (i.e., racks) available. Somewhat surprisingly, the aggregated model outperforms its centralized counterpart when less than 8 clients participate in the federated learning process. The performance slightly begins to degrade with the increase of clients participation. Within this specific network, traffic is often concentrated between specific areas (i.e., pods), while other communication paths see little to no traffic. Indeed, as we increase the number of clients beyond 8, each sub-matrix becomes smaller and potentially sparser. This increased sparsity means that each client has access to fewer non-zero entries, which represent actual traffic patterns. Consequently, the clients struggle to learn meaningful patterns from their increasingly limited and sparse data subsets.

E. Impact of Scale Factors

Figures 4b and 4a report the performance in terms of MAE of our proposed super-resolution model across different scale factors, for both datasets. For datacenter networks (Fig. 4b), we observe that the model maintains a stable performance, with MAE values ranging between approximately 0.12 and 0.15 MB. As specified in Section V-A, to generate coarser view of the network traffic (i.e., scale factors greater than 2), adjacent pods are clustered together. Interestingly, the model resilience suggests that it can effectively infer rack-level traffic not only from individual pod-level aggregates but also from clustered pod data with larger scale factors, indicating that the model captures underlying traffic patterns that persist even as the level of aggregation changes.

For wide area networks (Fig. 4a), instead, we observe a different trend. The model's performance gradually decreases as the scale factor increases, indicated by the rising MAE values. The best performance is achieved at a scale factor of 2, with an MAE of about 0.042 MB. The performance remains relatively stable for scale factors 3 and 4, with MAE values around $0.044-0.045$ MB. However, there's a noticeable degradation in performance for scale factors 5 and 6, with MAE values reaching approximately 0.047 MB and 0.06 MB, respectively. The increasing error range at higher scale factors indicates growing uncertainty in predictions as the level of clustering increases. This trend also reflects the increasing challenge of accurately predicting fine-grained traffic patterns from more heavily clustered data.

F. Fine-Grained Traffic Inference Baselines and Knowledge-Sharing Analysis

We then measure how ResCue can effectively reconstruct complete fine-grained traffic measurements without relying on shared information between clients within the federated learning process (see Section III-B). We compare our approach with "overlapping" scenarios, where clients can share information about each other sub-matrices. This approach translates to distributing portions of the same fine-grained traffic matrix with different degrees of overlap. With this in mind, we consider the following ResCue variants. ResCue-C, the centralized version, processes all network data in a centralized way. ResCue-NO implements federated learning with non-overlapping sub-matrices, where each client works independently on its unique network portion. ResCue-O introduces a 50% overlap between adjacent sub-matrices, such that clients possess up to 50% of the network knowledge of the remaining clients. ResCue-F provides clients with full network knowledge, where the full traffic matrix is shared as ground truth to all participating clients.

Fig. 5 shows a comparison between ResCue variants and the baselines for both DCN and WAN datasets. Across all methods, we see a consistent trend of increasing MAE as the scaling factor increases from 2 to 6, reflecting the increasing difficulty of the super-resolution task at higher scales. However, the ResCue variants consistently outperform the

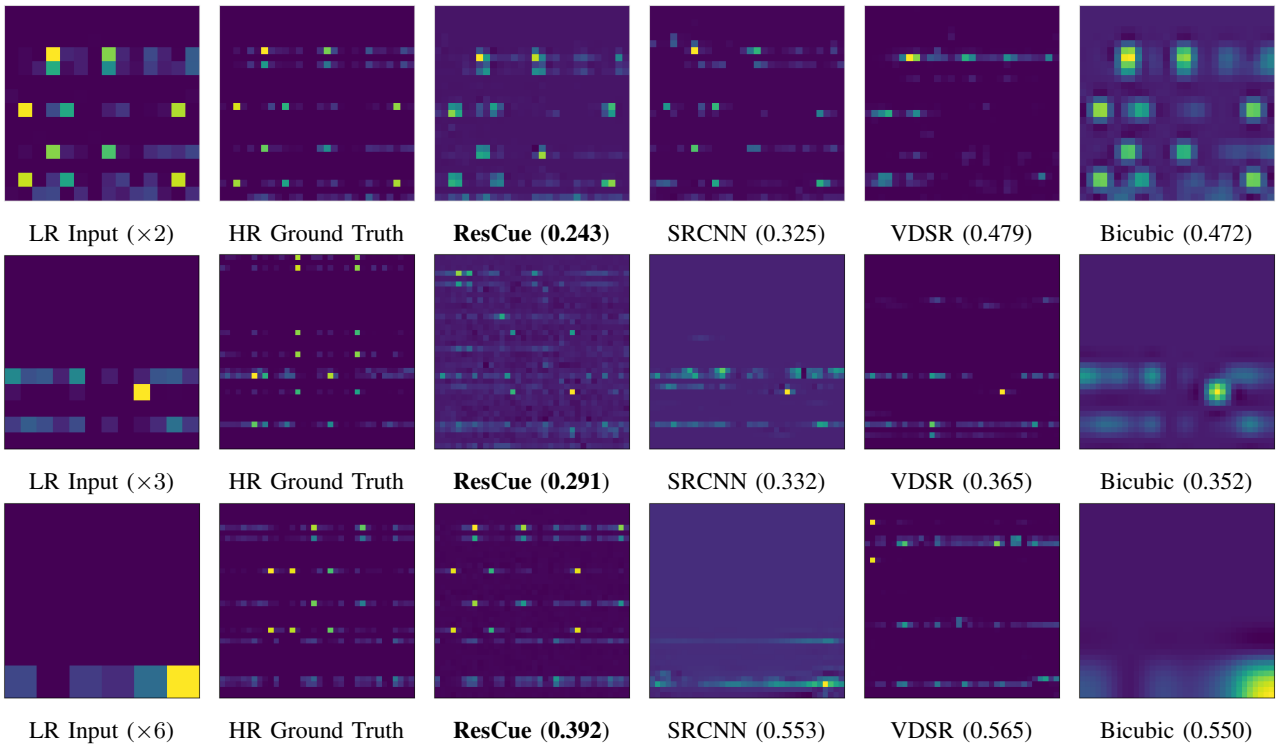


Fig. 6: Visual comparison between ResCue, state-of-art super-resolution models and the bicubic baseline, with different scale factors ($\times 2$, $\times 3$ and $\times 6$). The leftmost column shows input coarse-grained traffic matrices, while the second column presents the corresponding fine-grained ground truths. We report the performance in terms of MAE for each model. The lowest error is highlighted in bold.

traditional Bicubic and Fractal methods across all scaling factors, demonstrating the effectiveness of our approach. As expected, among the ResCue variants, ResCue-C and ResCue-F generally show the lowest MAE, suggesting that the full network knowledge positively impacts performance. Somewhat surprisingly, ResCue-F (Full knowledge) only marginally outperformed ResCue-O, showing a 5% lower MAE, suggesting that the 50% overlap in ResCue-O captures most of the benefits of information sharing without the privacy concerns of full data exchange. More interestingly, ResCue-NO (i.e., non-overlapping sub-matrices) achieves comparable results to ResCue-F and ResCue-O performance, and consistently low error values across scale factors. This promising result proves the effectiveness of our proposed federated approach in predicting fine-grained measurements from coarse-grained ones, with only partial visibility of the full network. Notably, the performance gap between traditional and ResCue methods widens as the scaling factor increases, highlighting the robustness of ResCue to more challenging upscaling tasks. Even with a scale factor of 6, representing a dramatic reduction in the number of clusters and consequently a highly abstracted view of the network, our model maintained impressive prediction accuracy.

TABLE I: Evaluation of TM reconstruction accuracy between *ResCue* (centralized), *ResCue-NO* (No-Overlap) and super-resolution baselines in terms of MAE, on a test set of Meta datacenter traffic [16] with different scale factors.

Scale	Bicubic	VDSR	SRCNN	ResCue	ResCue-NO
$\times 2$	0.553	1.41	0.32	**0.15**	**0.14**
$\times 3$	1.29	1.181	0.46	**0.164**	**0.192**
$\times 4$	3.51	2.375	0.62	**0.19**	**0.21**
$\times 5$	3.121	2.15	0.712	**0.32**	**0.37**
$\times 6$	4.19	3.11	1.52	**0.514**	**0.56**

G. Comparison Against Super-Resolution Architectures

We then compare our approach against the super-resolution models from the field of computer vision, i.e., SRCNN, VDSR, and Bicubic, reporting in Table I the MAE for different scale factors. While these models have shown remarkable results in image enhancement tasks, our experiments reveal that they fall short when applied directly to network traffic matrix inference. Our ResCue model consistently outperformed both SRCNN and VDSR across various metrics and scaling factors. For instance, at a scale factor of 3, ResCue achieved a 40% lower Mean Absolute Error (MAE) compared to SRCNN and a 25% lower MAE than VDSR. This performance gap widened further at higher scale factors, with ResCue showing a 60%

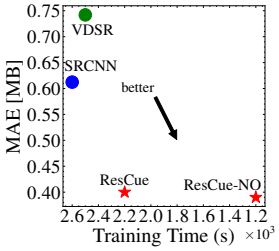


Fig. 7: MAE vs. training time trade-off.

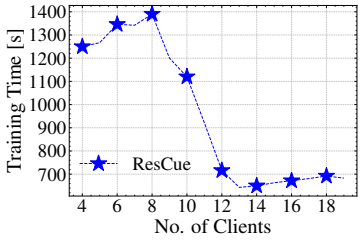


Fig. 8: Training time vs. Number of clients

and 45% reduction in MAE compared to SRCNN and VDSR, respectively, at a scale factor of 6. These results underscore that a super-resolution method cannot be directly applied to network traffic inference tasks and achieve optimal results. SRCNN and VDSR, despite their sophisticated convolutional architectures, struggle to capture the intricate interdependencies present in network traffic data. Their reliance on local convolutional operations, which work well for images, proves less effective in modeling the global patterns and characteristics of fine-grained network traffic. Instead, our approach consistently outperforms the other methods and shows robustness against coarser versions of the training sets (i.e., larger scale factors).

Furthermore, Fig. 6 reports a visual comparison between state-of-art super-resolution models and ResCue global model after the federated training on Meta datacenter traffic data. Our approach infers fine-grained traffic patterns with higher accuracy compared to the baselines and shows robustness across different scale factors.

H. Balancing Inference Accuracy and Training Efficiency

Efficient training is particularly critical in federated learning scenarios where data is distributed across multiple clients, and rapid model updates are essential for maintaining up-to-date performance. This is especially true in modern network environments, where the volume of data available for training models has become a performance bottleneck. In this analysis, we evaluate the trade-off between the inference accuracy of our distributed approach and the efficiency in terms of training time.

Fig. 7 illustrates the trade-off between Mean Absolute Error (MAE) and training time for various models. ResCue-NO, our federated learning approach with non-overlapping sub-matrices, achieves the best balance between accuracy and efficiency. With a MAE of approximately 0.39 MB and a training time of around 1000 seconds, it significantly outperforms traditional computer vision models like SRCNN and VDSR in both aspects. The centralized ResCue model shows slightly better accuracy of about 0.4 MB, but requires nearly twice the training time (around 2000 seconds). This comparison reveals that ResCue-NO sacrifices only a minimal amount of accuracy - 2.5% in terms of MAE - for a nearly 50% reduction in training time compared to its centralized counterpart.

In Fig. 8, we provide an analysis of how the number of clients affects the training time in our federated setup. In this analysis, we fixed the number of rounds at 20 and the scale factor to 2. We leverage the Meta datacenter traffic as it contains the larger number of points per data sample (i.e., $108 \times 108 = 11664$ values per training sample and $216 \times 216 = 23328$ per ground truth sample). Interestingly, as the number of clients increases from 4 to 8, there's a slight increase in training time, peaking at 8 clients with about 1400 seconds. Beyond this point, the training time gradually decreases as more clients are added, reaching around 700 seconds with 18 clients. This trend is due to the initial communication overhead of coordinating more clients, followed by the benefits of splitting the original fine-grained matrix into unique, non-overlapped smaller portions. The peak at 8 clients likely represents the point where coordination costs are balanced by the advantages of distributed computation. As the number of clients increases further, the benefits of parallelization and traffic matrix splitting outweigh the coordination overhead, resulting in reduced overall training time. While the centralized ResCue model achieves slightly lower MAE, the federated ResCue-NO offers significantly reduced training time, especially as the number of clients increases. This makes our federated model more scalable and adaptable to large networks.

VII. Conclusion

In this work, we propose a super-resolution model to infer network traffic volumes with fine granularity. We present the design of ResCue, a super-resolution-based Deep Residual Network architecture to predict fine-grained traffic from the coarser representation of the network. To address partial network visibility in multi-domain network scenarios, where complete network knowledge is often infeasible, we leverage a federated learning procedure. Results on real-world datasets validate that our proposed method achieves a 55% reduction in terms of weighted Mean Absolute Error (MAE) compared to state-of-art super-resolution architectures and a 87% reduction compared to existing interpolation and fine-grained traffic inference approaches. Furthermore, our distributed learning design achieves comparable performance to its centralized counterpart while reducing training time up to 80%.

VIII. Acknowledgment

This work has been supported by the National Science Foundation (NSF) Awards #1908574, #2201536, and by a Comcast Innovation Award.

References

[1] 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.

[2] A. Sacco, F. Esposito, and G. Marchetto, "Resource inference for sustainable and responsive task offloading in challenged edge networks," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 3, pp. 1114–1127, 2021.

[3] 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.

[4] 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.

[5] D. Jiang, Z. Zhao, Z. Xu, C. Yao, and H. Xu, "How to reconstruct end-to-end traffic based on time-frequency analysis and artificial neural network," *AEU-International Journal of Electronics and Communications*, vol. 68, no. 10, pp. 915–925, 2014.

[6] F. Qian, G. Hu, and J. Xie, "A recurrent neural network approach to traffic matrix tracking using partial measurements," in *2008 3rd IEEE Conference on Industrial Electronics and Applications*. IEEE, 2008, pp. 1640–1643.

[7] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[8] R. A. Memon, S. Qazi, and B. M. Khan, "Design and implementation of a robust convolutional neural network-based traffic matrix estimator for cloud networks," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–11, 2021.

[9] A. Sacco, F. Esposito, and G. Marchetto, "Completing and predicting internet traffic matrices using adversarial autoencoders and hidden markov models," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 2244–2258, 2023.

[10] 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.

[11] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. Braverman, "One Sketch to Rule Them All: Rethinking Network Flow Monitoring with UnivMon," in *Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM '16)*, 2016, pp. 101–114.

[12] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: Scaling Flow Management for High-Performance Networks," in *Proceedings of the 2011 ACM SIGCOMM Conference (SIGCOMM '11)*, 2011, pp. 254–265.

[13] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 75–86, 2010.

[14] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106775, 2021.

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

[16] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren, "Inside the social network's (datacenter) network," in *Proceedings of the ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15)*, 2015, pp. 123–137.

[17] C. Zilli, D. Monaco, A. Sacco, O. Okafor, G. Marchetto, F. Esposito *et al.*, "Inferring Visibility of Internet Traffic Matrices Using eXplainable AI," in *IEEE/IFIP Network Operations and Management Symposium (NOMS '24)*. IEEE/IFIP, 2024.

[18] 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.

[19] 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.

[20] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale ip traffic matrices from link loads," *ACM SIGMETRICS Performance Evaluation Review*, vol. 31, no. 1, pp. 206–217, 2003.

[21] Y. Zhang, M. Roughan, C. Lund, and D. L. Donoho, "Estimating point-to-point and point-to-multipoint traffic matrices: An information-theoretic approach," *IEEE/ACM Transactions on networking*, vol. 13, no. 5, pp. 947–960, 2005.

[22] L. Nie, D. Jiang, and L. Guo, "End-to-end network traffic reconstruction via network tomography based on compressive sensing," *Journal of Network and Systems Management*, vol. 23, pp. 709–730, 2015.

[23] D. Jiang, X. Zhengzheng, N. Laisen, and L. Jindi, "An approximate approach to end-to-end traffic in communication networks," *Chinese Journal of Electronics*, vol. 21, no. 4, pp. 705–710, 2012.

[24] D. Jiang, L. Huo, and Y. Li, "Fine-Granularity Inference and Estimations to Network Traffic for SDN," *PloS one*, vol. 13, no. 5, p. e0194302, 2018.

[25] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot, "Traffic matrices: balancing measurements, inference and modeling," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2005, pp. 362–373.

[26] W. Liu, A. Hong, L. Ou, W. Ding, and G. Zhang, "Prediction and correction of traffic matrix in an ip backbone network," in *2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*. IEEE, 2014, pp. 1–9.

[27] A. Azzouni and G. Pujolle, "Neutm: A neural network-based framework for traffic matrix prediction in sdn," in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2018, pp. 1–5.

[28] J. Zhao, H. Qu, J. Zhao, and D. Jiang, "Towards traffic matrix prediction with lstm recurrent neural networks," *Electronics Letters*, vol. 54, no. 9, pp. 566–568, 2018.

[29] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018, pp. 187–193.

[30] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1646–1654.

[31] 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.

[32] K. I. Kim and Y. Kwon, "Single-image super-resolution using sparse regression and natural image prior," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 6, pp. 1127–1133, 2010.

[33] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[34] C. Zhang, X. Ouyang, and P. Patras, "ZipNet-GAN: Inferring fine-grained mobile traffic patterns via a generative adversarial neural network," in *CoNEXT 2017 - Proceedings of the 2017 13th International Conference on emerging Networking EXperiments and Technologies*, vol. 17, 2017, pp. 363–375.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2017, pp. 1132–1140.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[38] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 1874–1883.

[39] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[40] R. E. Carlson and F. N. Fritsch, "Monotone piecewise bicubic interpolation," *SIAM journal on numerical analysis*, vol. 22, no. 2, pp. 386–400, 1985.