Federated Transfer Component Analysis Towards Effective VNF Profiling

Xunzheng Zhang, Shadi Moazzeni, Juan Marcelo Parra-Ullauri, Reza Nejabati, and Dimitra Simeonidou
High Performance Networks Research Group, Smart Internet Lab, Faculty of Engineering,
University of Bristol, Bristol, Clifton BS8 1UB, UK
Email: {xunzheng.zhang, shadi.moazzeni, jm.parraullauri, Reza.Nejabati, Dimitra.Simeonidou}@bristol.ac.uk

***Abstract*—The increasing concerns of knowledge transfer and data privacy challenge the traditional gather-and-analyse paradigm in networks. Specifically, the intelligent orchestration of Virtual Network Functions (VNFs) requires understanding and profiling the resource consumption. However, profiling all kinds of VNFs is time-consuming. It is important to consider transferring the well-profiled VNF knowledge to other lack-profiled VNF types while keeping data private. To this end, this paper proposes a Federated Transfer Component Analysis (FTCA) method between the source and target VNFs. FTCA first trains Generative Adversarial Networks (GANs) based on the source VNF profiling data, and the trained GANs model is sent to the target VNF domain. Then, FTCA realizes federated domain adaptation by using the generated source VNF data and less target VNF profiling data, while keeping the raw data locally. Experiments show that the proposed FTCA can effectively predict the required resources for the target VNF. Specifically, the RMSE index of the regression model decreases by 38.5% and the R-squared metric advances up to 68.6%.**

***Index Terms*—Federated Transfer Learning, VNF Profiling, GANs, Domain Adaptation, Network Function Virtualization**

I. Introduction

In Virtual Network Functions (VNFs), network functions are decoupled from proprietary hardware and implemented as software that can run on standard network infrastructure, such as firewalls, load balancers, and routers [1]. This allows for more flexibility and agility in deploying and scaling network services. Especially for the next generation of networks (6G), orchestrators in the Network Function Virtualization (NFV) architecture should automatically deploy, configure, and scale different VNFs [2]. In this case, the needed resources are allocated and interconnected to meet the requirements of specific network services. This progress needs VNF profiling.

VNF profiling [3] refers to the process of analyzing and characterizing the resource requirements and performance characteristics of VNFs. Profiling involves gathering insights and data about how a specific VNF utilizes computing, storage, and network resources, as well as its overall performance in a virtualization environment. Considering the pros, VNF profiling is proposed as a solution to model the observed trends. It provides a model to accurately describe how the specific VNF reacts under a certain amount of workload and specific resource configurations. VNF profiling is dedicated to providing efficient resource allocation and performance plans, to ensure service performance requirements in the network [4].

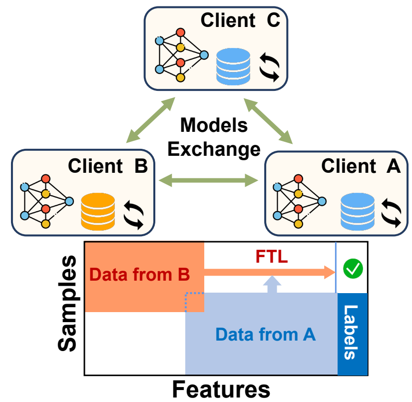


Fig. 1. Federated transfer learning with the decentralized architecture.

However, profiling all possible resource configurations for each VNF takes time, and collecting the profiling data sets from all different VNF types is costly and difficult. Many possible configurations lead to a multiplicative growth rate of the profiling test time [5]. In this case, if we can transfer the already profiled VNF (source VNF) resource configuration knowledge to another less-profiled VNF (target VNF) type and predict its needed resources, the target VNF may save more profiling time and effort. This motivation will make the profiling of new VNFs easier, and quickly get the needed resources for the target VNF type. Furthermore, different VNF types are typically considered distinct entities. Each type of VNF represents a specific network function. These VNF types cannot directly exchange their profiling data because of data privacy concerns. Therefore, this paper focuses on how to transfer the profiling knowledge from the source VNF to the target VNF, while keeping the raw profiling data locally.

To deal with the above-mentioned problems, decentralized federated learning (FL) [6] architecture is considered. As shown in Fig. 1, each VNF type is considered as one client. Firstly, the already profiled source VNF trains a data synthetic GANs [7]–[9] model based on its enough profiling data set. Then this model (or its parameters) will be sent to the target VNF client to generate the synthetic data which is similar to the source VNF profiling data. Secondly, transfer learning (TL) theories, or domain adaptations [10] are applied at the

This work was supported by the China Scholarship Council and the Future Open Network Research Challenge (FONRC) call, the project Realizing Enabling Architectures and Solutions for Open Networks (REASON), UK.

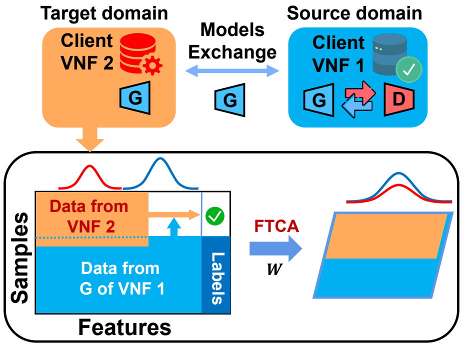


Fig. 2. FTCA for VNF profiling. **G** is the data generator model. **D** is the discriminator. **W** is the mapping matrix. After FTCA, target domain data and 'source domain' data have similar distributions.

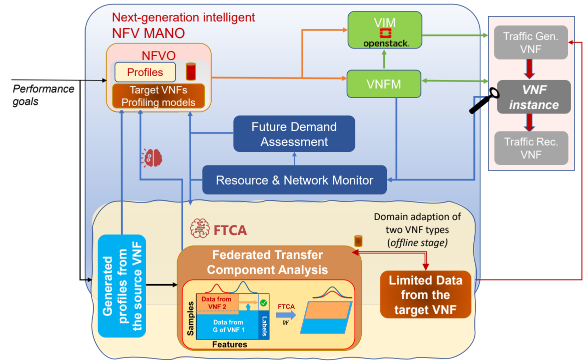


Fig. 3. Intelligent NFV MANO with FTCA VNF profiling interaction.

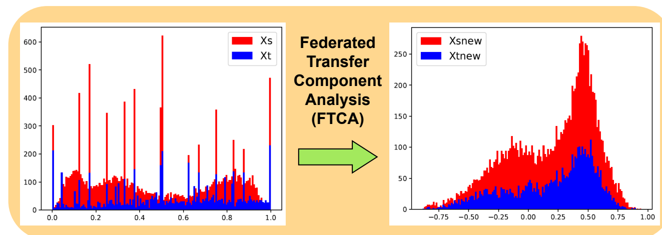


Fig. 4. Feature distribution after federated domain adaptation FTCA.

target VNF domain (Fig. 2). After the kernel matrix operation in the target VNF, the generated source VNF data and less target VNF profiling data are mapped into a new space where they have similar distributions. Then in this new space, the regression machine learning (ML) model can be trained to predict the labels (resource configurations of VNFs like CPU cores, memory, etc) that have been profiled in the source VNF but not yet profiled in the target VNF. Combining two learning methods, federated transfer component analysis (FTCA) is proposed in this paper as a novel method under federated transfer learning (FTL).

The main contributions of this paper can be summarized as follows:

(1) A novel FTCA method is proposed considering the profiling knowledge transfer among different VNF types. Unlike the model-based FTL methods [11]–[13], FTCA directly adopts the feature-based transfer component analysis (TCA) [10] in a federated manner, with generated source domain data from GANs (Fig. 2). FTCA not only controls the complexity of TCA but also keeps the raw profiling data locally. The target VNF can make domain adaptation based on its local data and the generated source VNF data, then predict the needed resource configuration.

(2) The work involves multiple VNF profiling knowledge transfer tasks which include SNORT Inline, SNORT Passive and virtual firewall (vFW) VNF types. Multiple regression models are trained after FTCA to predict the resource configurations (CPU cores, memory and link capacity) of the target VNF. Experiment results show the effectiveness of the FTCA.

The remainder of this paper is organized as follows. Section II describes the related work. Section III details the FTCA method. The experiments and analysis are given in Section IV. The conclusions and future work are drawn in Section V.

II. BACKGROUND AND RELATED WORK

A. Intelligent VNF Profiling

Regarding the contribution to the field of intelligent VNF profiling, our *Novel Autonomous Profiling (NAP)* method introduces an autonomous offline method for profiling VNFs [3]. More precisely, utilizing our NAP method, the weighted randomly Autonomous Profiler selects a configuration of resources and requests the orchestrator to assign them to the VNF under profiling. It then finds the maximum traffic rate that the VNF can handle while meeting the Key Performance Indicators (KPIs) and records this performance data. The offline profiling process and profiling data set generation continue until the profiling time is completed. Moreover, in [4], we introduced a framework for analyzing VNFs, focusing on their resource characteristics, their VNF-level service performance, as well as the discovery of resource-performance correlations. Fig. 3 shows the interaction between the intelligent NFV MANO (Network Functions Virtualization Management and Orchestration), the traffic generator, network and computing resources monitoring tools, the VNF under profiling, our proposed FTCA method.

B. Federated Transfer Learning

Considering the data collection process of training and test sets are not in the same working place, FTL aims to improve the generalization ability of ML models, with data privacy concerns. There are primarily two methods of FTL in existing research. FL first, then TL like [6], [11]. Another is TL first, followed by FL [12], [13]. However, these works mostly emphasize parameter-based TL (pre-training and fine-tuning the model). This paper directly focuses on feature-based TL (TCA) [10] which aims to reduce the difference between the target and generated source domain, but in a federated manner. Thus, to the best of our knowledge, this is the first work combining TCA in TL with decentralized FL as FTCA. Fig. 4 shows the data distribution change in our work. The blue and red histograms represent the target VNF domain and the

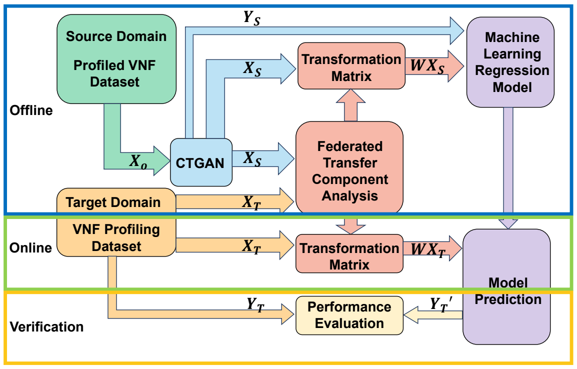


Fig. 5. The detailed FTCA process in VNF profiling.

generated source VNF domain. After FTCA, the distributions are more similar.

III. FEDERATED TRANSFER COMPONENT ANALYSIS

A. GANs Model Delivery

In GANs training, two neural networks compete in a two-player min-max game to simultaneously train a generator G and a discriminator D. The goal of the generator $G(\boldsymbol{z};\theta_g)$ is to learn a distribution $p_g(\boldsymbol{z})$ over data $\boldsymbol{x}$, by mapping input noise $\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})$ to real samples $\boldsymbol{x}$. $p_{\boldsymbol{z}}(\boldsymbol{z})$ is usually an easy-to-sample distribution like the Gaussian distribution with (0,1). Meanwhile, the discriminator $D(\boldsymbol{x},\theta_d)$ is trained to discriminate between the real samples $\boldsymbol{x}$ and generated samples $G(\boldsymbol{z})$. The value function can be described as:

$$\begin{aligned}\min_G \max_D V(D,G) =& \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] \\ &+ \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))]\end{aligned} \tag{1}$$

In general, D strives to discriminate the real samples, simultaneously training G to make D cannot discriminate between the raw data and the generated data. This training process will continue until the generated data successfully cheats the discriminator. Some typical GANs models have been proposed to generate tabular data like TableGAN [9] and CTGAN [8]. TableGAN uses GANs to synthesize fake tables that are statistically similar to the original table yet do not incur raw data leakage. CTGAN designs a conditional generator and training-by-sampling to deal with the imbalanced discrete columns, with fully connected neural networks for the generator. Here we choose the CTGAN as the data synthetic model.

In the proposed FTCA, the source domain first trains a CTGAN model by using the already profiled VNF tabular data. After training, the generator can produce similar synthetic VNF profiling data. Then the source domain VNF will send this well-trained generator model (or generator parameters) to the target domain VNF, as shown in Fig. 2. This means that the target VNF can do the domain adaptation based on the generated data, without substantial raw data leakage.

B. Federated Transfer Component Analysis

FTCA considers generator models from the isolated source client in FL and does TCA at the target client. After generating the 'source data' at the target VNF, TCA transforms the features into a new feature space by a matrix, where the difference between source and target data is smaller. The Maximum Mean Discrepancy (MMD) [10] is used as an indicator to evaluate the distance between the source and target domain. It describes the kernel mapping distance in a Reproducing Kernel Hilbert Space (RKHS). The detailed expression of MMD is:

$$\mathrm{MMD}(\mathbf{X}_s,\mathbf{X}_t) = \left\| \frac{1}{n_s}\sum_{i=1}^{n_s}\phi(\mathbf{x}_{S_i}) - \frac{1}{n_t}\sum_{i=1}^{n_t}\phi(\mathbf{x}_{T_i}) \right\|_{\mathcal{H}}^2 \tag{2}$$

The generated source domain data set that join the FTCA is set as $\mathbf{X}_s = \{\mathbf{x}_{S_i}\} = \{\mathbf{x}_1,...,\mathbf{x}_{n_s}\}$. The number of generated source samples n_s can be controlled by the target VNF client. The target domain data set is set as $\mathbf{X}_t = \{\mathbf{x}_{T_i}\} = \{\mathbf{x}_1,...,\mathbf{x}_{n_t}\}$. $\mathbf{X}_s \in \mathcal{D}_{\mathcal{S}}$ and $\mathbf{X}_t \in \mathcal{D}_{\mathcal{T}}$. We assume that the conditional probability $P(\mathbf{Y}_s|\mathbf{X}_s) \approx P(\mathbf{Y}_t|\mathbf{X}_t)$. $\mathbf{Y}_s$ and $\mathbf{Y}_t$ are resource configuration labels of VNFs, $\mathbf{Y}_s$ has been profiled in the source VNF but $\mathbf{Y}_t$ has not been profiled in the target VNF. $S(s)$ and $T(t)$ represent the source domain and target domain respectively, ϕ function is the feature mapping corresponding the kernel map $\mathcal{K}(\mathbf{x}_i,\mathbf{x}_j) =< \phi(\mathbf{x}_i),\phi(\mathbf{x}_j) >$. A semi-defined adaptation matrix $\mathbf{W}$ can also denote the $\mathrm{MMD}(\mathbf{X}_s,\mathbf{X}_t)$ feature transformation as follows:

$$\left\| \frac{1}{n_s}\sum_{i=1}^{n_s}\mathbf{W}^{\mathrm{T}}\mathbf{x}_{S_i} - \frac{1}{n_t}\sum_{i=1}^{n_t}\mathbf{W}^{\mathrm{T}}\mathbf{x}_{T_i} \right\|_{\mathcal{H}}^2 \tag{3}$$

According to the property of trace in matrix, (2) and (3) can be separately simplified as $\mathrm{MMD}(\mathbf{X}_s,\mathbf{X}_t) = \mathrm{tr}(\mathbf{KL})$ and $\mathrm{MMD}(\mathbf{X}_s,\mathbf{X}_t) = \mathrm{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right)$. Here $\mathbf{X} = [\mathbf{X}_s,\mathbf{X}_t]$. $\mathbf{K}$ is Gram matrices defined on the generated source domain, target domain, and cross-domain data in the embedded spaces. tr is the trace of a matrix. The $\mathbf{K}$ and $\mathbf{L}$ are shown below:

$$\begin{aligned}\mathbf{K} &= \begin{bmatrix} <\phi(\mathbf{x}_s),\phi(\mathbf{x}_s)> & <\phi(\mathbf{x}_s),\phi(\mathbf{x}_t)> \\ <\phi(\mathbf{x}_t),\phi(\mathbf{x}_s)> & <\phi(\mathbf{x}_t),\phi(\mathbf{x}_t)> \end{bmatrix} \\ &= \begin{bmatrix} K_{s,s} & K_{s,t} \\ K_{t,s} & K_{t,t} \end{bmatrix} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}\end{aligned} \tag{4}$$

$$\mathbf{L} = \begin{bmatrix} \frac{1}{n_s^2}\mathbf{1}\mathbf{1}^{\mathrm{T}} & \frac{-1}{n_s n_t}\mathbf{1}\mathbf{1}^{\mathrm{T}} \\ \frac{-1}{n_s n_t}\mathbf{1}\mathbf{1}^{\mathrm{T}} & \frac{1}{n_t^2}\mathbf{1}\mathbf{1}^{\mathrm{T}} \end{bmatrix} \tag{5}$$

where $L_{ij} = \frac{1}{n_s^2}$ if $\mathbf{x}_i,\mathbf{x}_j \in \mathbf{X}_s$, $L_{ij} = \frac{1}{n_t^2}$ if $\mathbf{x}_i,\mathbf{x}_j \in \mathbf{X}_t$, otherwise, $L_{ij} = -\frac{1}{n_s n_t}$. Here $\mathbf{1}$ means the column vector with all values of 1. To address this computationally complex problem of calculating high dimensional $\mathbf{K}$ (a semi-definite program), Principal Component Analysis (PCA) is used [10] to get new representations of $\mathbf{X}_s$ and $\mathbf{X}_t$. Then the above m (m $\ll n_s + n_t$) leading eigenvectors can be selected to construct low-dimensional representations. In FTCA for the VNF profiling scenario, features of the VNF profiling data set are not so many, so we directly set the number of features as m (which also satisfies m $\ll n_s + n_t$), to avoid discarding potentially useful information. Then the final minimize MMD optimization problem can be written as:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \lambda \left\|\mathbf{W}\right\|_F^2 \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{W} = \mathbf{I}_m \end{aligned} \tag{6}$$

where $\lambda > 0$ is the hyper-parameter to control the complexity of $\mathbf{W}$, the regularization term is $\lambda \left\|\mathbf{W}\right\|_F^2$. $\mathbf{I}_m \in \mathbb{R}^{(m\times m)}$ is the identity matrix and will be simplified as $\mathbf{I}$. Considering the constraint as maximum projection variance in PCA, we also need to keep the original data properties of $\mathbf{X} = [\mathbf{X}_s, \mathbf{X}_t]$ after mapping $\mathbf{W}$ in FTCA, so the maximum data variance is included as the form of the scatter matrix $\mathbf{S} = \left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\right)\mathbf{H}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\right)^{\mathrm{T}}$. The centering matrix $\mathbf{H}$ is calculated as $\mathbf{H} = \mathbf{I}_{(n_s+n_t)} - \frac{1}{n_s+n_t}\mathbf{1}\mathbf{1}^{\mathrm{T}}$. Here $\mathbf{I}_{(n_s+n_t)} \in \mathbb{R}^{(n_s+n_t)\times(n_s+n_t)}$ denotes identity matrix, $\mathbf{1} \in \mathbb{R}^{(n_s+n_t)}$ is the column vector with all 1's, and $\mathbf{H}^n = \mathbf{H}$.

Lagrange multipliers are used to solve constrained optimization problems like (6):

$$\begin{aligned} L(\mathbf{W}, \mathbf{\Phi}) = & \operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\left(\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}} + \lambda\mathbf{I}\right)\mathbf{W}\right) \\ & - \operatorname{tr}\left(\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\mathbf{\Phi}\right) \end{aligned} \tag{7}$$

Let $\frac{\partial L}{\partial \mathbf{W}} = 0$, the Lagrange function (7) will have:

$$\left(\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}} + \lambda\mathbf{I}\right)\mathbf{W} = \mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{\Phi} \tag{8}$$

Let $\mathbf{A} = \left(\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}$, the (8) will become $\mathbf{A}^{-1}\mathbf{W} = \mathbf{W}\mathbf{\Phi}$. Here it is clear that the solution of mapping $\mathbf{W}$ are the eigenvectors of $\mathbf{A}$ or $\mathbf{A}^{-1}$, the reciprocal of $\mathbf{\Phi}$ is the eigenvalues of $\mathbf{A}$. By using the above formulas, we can determine the transformation matrix $\mathbf{W}$ for domain adaptation. This matrix makes two distributions close to each other. In the new space constructed by $\mathbf{W}$, new data can be obtained and used to address classification or regression problems. The step-by-step FTCA is outlined in Algorithm 1, Fig. 5 also gives the detailed FTCA process in VNF profiling.

There are two obvious advantages of FTCA. First, the number of generated source samples n_s can be controlled by the generator at the target VNF domain. The target FTL client can generate a proper number of source samples to make adaptations and avoid the heavy computation complexity on the above matrices. Second, by transferring the profiled VNF knowledge, the time to profile a function-related VNF can be significantly decreased rather than 45 hours for one VNF in [5]. Future network orchestrations require self-generation of data, self-training of models and automatic performance improvement. FTCA may be an ideal approach for generating synthetic data and transferring knowledge without privacy concerns.

IV. EXPERIMENTS AND ANALYSIS

A. Federated Transfer Tasks and Comparison Methods

Our previous work [3] has provided autonomous profiling data sets of three types of different VNFs: SNORT Inline mode (I mode), SNORT Passive mode (P mode) and Virtual Firewall mode (V mode), the optimum resource configuration is found to meet the pre-defined demands. Fig. 6 depicts our setup with a SNORT instance, showing the connection between the profiled VNF, the traffic generator and server end-point machines (iPerf client and iPerf server) [3]. The recorded VNF performance data set has eight features and one output. The input variables are CPU utilization (CPUUTP), Memory utilization (MEMUTP), Network latency (RTT), VNF maximum input rate (MIR) and Packet loss (In_RX, Out_Tx). The output variables are one of the VNF resource configurations like CPU cores (CPU), Memory (MEM_MB) and Link Capacity (LINK_Mbps). Our goal is to use FTCA to predict resource configurations of less-profiled target VNFs in which CPU, MEM or LINK is unknown. The number of collected profiling data samples in I mode (1112), P mode (896) and V mode (775) are decreasing, so we illustrate the federated transfer missions as I mode to P mode (I2P), I mode to V mode (I2V), and P mode to V mode (P2V). In order to compare the advantages of the proposed FTCA, several regression methods are used in the new feature space after mapping to get the prediction results: Polynomial Regression (Poly), Support Vector Regression (SVR), Random Forest Regression (RF) and Multilayer Perceptron (MLP). The evaluation metrics

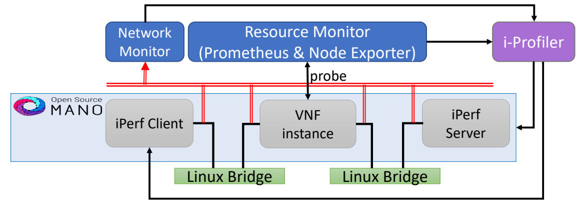


Fig. 6. Experiment setup [3].

Algorithm 1: FTCA Algorithm for VNF profiling

Input: The source VNF profiling data $\mathbf{X}_o$. The target VNF profiling data $\mathbf{X}_t$.

Output: The target VNF resource configuration labels estimated by FTCA.

1 Collect the source VNF data $\mathbf{X}_o$ and train the CTGAN model.
2 Send the well-trained CTGAN.pkl model to the target VNF entity.
3 Collect the target VNF data $\mathbf{X}_t$ and generate source VNF data $\mathbf{X}_o^{'}$ ($\mathbf{X}_o^{'} = \{\mathbf{X}_s, \mathbf{Y}_s\}$).
4 Select the VNF resource labels as $\mathbf{Y}_s$ that exist in $\mathbf{X}_o^{'}$ but missing in $\mathbf{X}_t$. Normalize data sets $\mathbf{X}_s$ and $\mathbf{X}_t$.
5 The matrix of $\mathbf{L}$ and $\mathbf{H}$ are constructed by (3) to (6) according to the datasets $\{\mathbf{X}_s, \mathbf{X}_t\}$.
6 The solution mapping matrix $\mathbf{W}$ is calculated by the Lagrange multiplier according to (7) and (8).
7 Train regression models with the data set $\{\mathbf{W}\mathbf{X}_s, \mathbf{Y}_s\}$.
8 Normalize the new profiling data $\mathbf{X}_t$, estimate the $\mathbf{Y}_t$ by the $\mathbf{W}\mathbf{X}_t$ on the trained regression models.

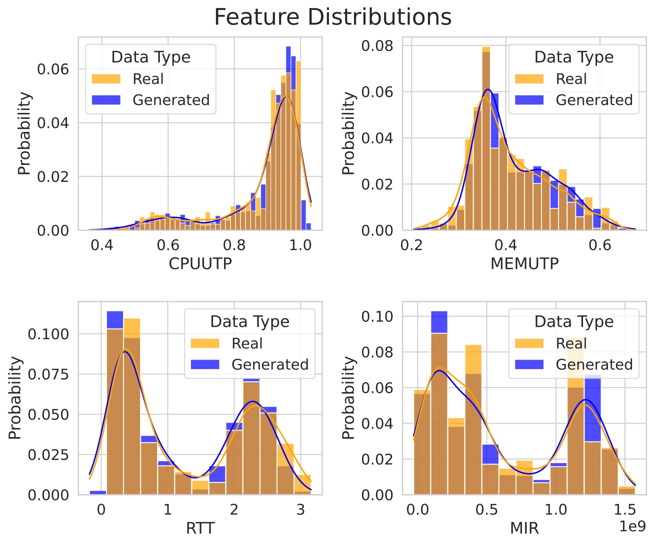
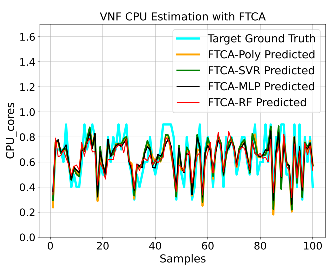
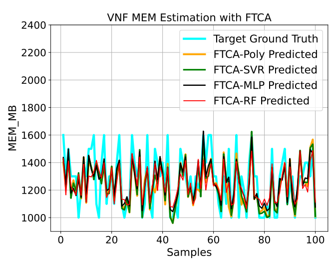


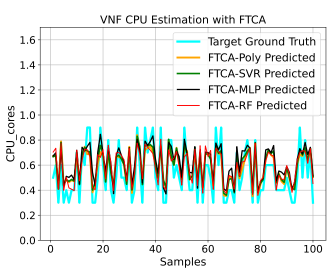
Fig. 7. Comparison between the illustrated features from generated data and real raw data.



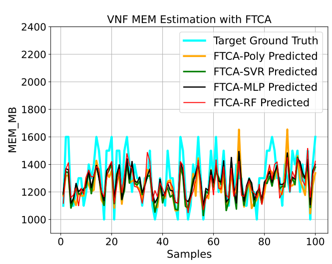
(a) CPU cores



(b) Memory



(c) CPU cores



(d) Memory

Fig. 8. The comparison of target VNF CPU and memory resources predictions on FTL missions I2P ((a),(b)) and P2V ((c),(d)).

TABLE I
PERFORMANCE EVALUATION OF TARGET VNF CPU RESOURCE CONFIGURATION.

Task	Method	Original MAE	Original RMSE	Original R2	FTCA Optimization MAE	FTCA Optimization RMSE	FTCA Optimization R2
I2P	Poly	0.219	0.273	0.222	**0.133**	**0.168**	**0.707**
	SVR	0.174	0.201	0.579	**0.137**	**0.173**	**0.688**
	RF	0.158	0.195	0.604	**0.151**	**0.187**	**0.635**
	MLP	0.169	0.198	0.523	**0.129**	**0.158**	**0.737**
I2V	Poly	0.356	0.410	-0.560	**0.278**	**0.315**	**0.077**
	SVR	0.292	0.326	0.013	**0.270**	**0.303**	**0.150**
	RF	0.289	0.328	0.005	**0.287**	**0.323**	**0.031**
	MLP	0.308	0.345	-0.105	**0.254**	**0.281**	**0.265**
P2V	Poly	0.319	0.356	-0.173	**0.180**	**0.213**	**0.579**
	SVR	0.221	0.246	0.439	**0.172**	**0.192**	**0.657**
	RF	0.206	0.229	0.512	**0.178**	**0.197**	**0.638**
	MLP	0.197	0.221	0.546	**0.183**	**0.209**	**0.596**

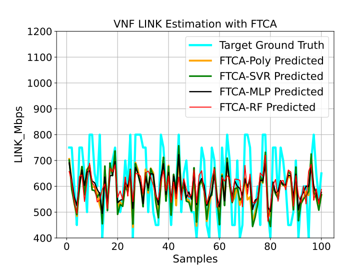
TABLE II
PERFORMANCE EVALUATION OF TARGET VNF MEMORY RESOURCE CONFIGURATION.

Task	Method	Original MAE	Original RMSE	Original R2	FTCA Optimization MAE	FTCA Optimization RMSE	FTCA Optimization R2
I2P	Poly	0.527	0.506	-0.176	0.178	0.220	0.535
	SVR	0.182	0.227	0.502	0.179	0.221	0.529
	RF	0.149	0.209	0.575	0.189	0.228	0.497
	MLP	0.194	0.231	0.484	0.169	0.210	0.570
I2V	Poly	0.157	0.208	0.572	0.198	0.244	0.407
	SVR	0.178	0.227	0.487	0.205	0.252	0.370
	RF	0.184	0.238	0.440	0.190	0.239	0.431
	MLP	0.203	0.251	0.373	0.201	0.244	0.406
P2V	Poly	0.345	0.455	-1.058	0.208	0.251	0.375
	SVR	0.202	0.270	0.277	0.203	0.245	0.400
	RF	0.153	0.206	0.580	0.212	0.254	0.358
	MLP	0.171	0.213	0.529	0.213	0.253	0.362

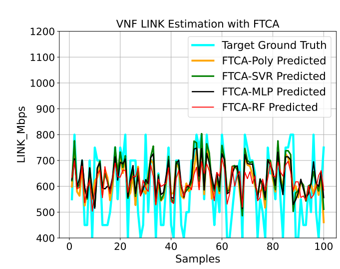
are the Mean Absolute Error (MAE), the Root Mean Square Error (RMSE) and R-squared (R2). CTGAN [8] and YData Synthetic Tool [1] are selected to train on the source VNF and send the corresponding generator model to the target VNF.

B. Comparison Performance on VNF Profiling

Fig. 7 illustrates the distribution of certain features in the real data (yellow) and data generated by CTGAN (blue), after training on the source VNF profiling data set. The synthetic data effectively approximates the distribution of the original data. The received CTGAN model will synthesize the source domain data (2000 samples) at the target VNF for FTCA. There are three target VNF resources: CPU cores, memory, and link capacity on three transfer tasks I2P, I2V and P2V, as shown in Tables I, II and III. Original means that the regression models are trained on raw source data and directly tested on the target data. The FTCA optimization makes the models



(a) Link Capacity



(b) Link Capacity

Fig. 9. A comparison of target VNF link capacity resources predictions on FTL mission I2P (a) and P2V (b).

[1]YData Synthetic: https://github.com/ydataai/ydata-synthetic

TABLE III
PERFORMANCE EVALUATION OF TARGET VNF LINK CAPACITY RESOURCE CONFIGURATION.

Task	Method	Original			FTCA Optimization		
		MAE	RMSE	R2	MAE	RMSE	R2
I2P	Poly	0.175	0.215	0.535	0.207	0.254	0.344
	SVR	0.207	0.252	0.358	0.207	0.242	0.410
	RF	0.184	0.223	0.495	0.221	0.265	0.289
	MLP	0.178	0.217	0.523	0.211	0.253	0.352
I2V	Poly	0.754	0.964	-8.163	0.206	0.252	0.375
	SVR	0.126	0.185	0.661	0.203	0.248	0.394
	RF	0.070	0.130	0.832	0.222	0.265	0.308
	MLP	0.189	0.264	0.315	0.212	0.256	0.354
P2V	Poly	0.297	0.365	-0.315	0.216	0.263	0.315
	SVR	0.213	0.244	0.413	0.204	0.248	0.391
	RF	0.220	0.244	0.412	0.233	0.278	0.240
	MLP	0.302	0.329	-0.068	0.204	0.241	0.425

train on generated source data and test on the target data, after mapping $\mathbf{W}$ (Fig. 4). It is clear in Table I that the RMSE error can be reduced by 38.5% and the R2 metric advances up to 68.6%.

The target VNF CPU prediction results of FTL task I2P and P2V are shown in Fig. 8 (a) and (c), where the cyan line is the real needed CPU cores on the target VNF after profiling, the same for memory predictions in Fig. 8 (b) and (d). The experiment results show that the proposed FTCA can effectively help the regression models predict the target VNF CPU cores and memory.

However, when predicting the link capacity, the FTL results may not be very good (Fig. 9 and Table III). This negative transfer phenomenon is caused by the information gain (IG) of features or labels in the source VNF profiling data set. From the hierarchical clustering correlation heat-map [14] of the raw SNORT Inline VNF data set (Fig. 10), we can see that most of the IG exists in the deep colour features like CPU and MIR. In contrast, link capacity (LINK) does not have enough IG in the data set. In this case, it is important to consider what kind of conditions we need federated transfer or not transfer [15].

V. Conclusion and Future Work

In this paper, a novel feature-based federated transfer learning method is proposed as federated transfer component analysis (FTCA). It transfers knowledge among different VNF profiling entities, based on GANs data in federated learning with transfer learning among profiling datasets. First, FTCA is a feasible approach for improving tabular data estimation, without substantial data leakage. Second, by transferring the profiled VNF knowledge, faster new VNFs deployment can be expected as service function chaining (SFC). FTCA also has reference value for the future 6G network orchestration. The next step will focus on problems like non-independently and identically (non-iid) features in data for FTL, multiple input-output regression models for SFC, and encrypted CTGAN models.

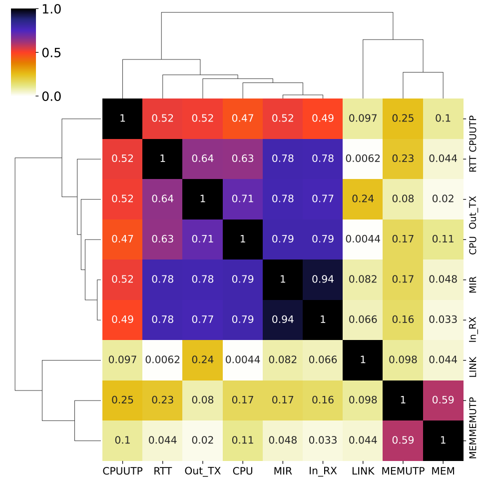


Fig. 10. The hierarchical clustering heat-map from SNORT Inline VNF profiling data set.

References

[1] K. Kaur, V. Mangat, and K. Kumar, "A review on virtualized infrastructure managers with management and orchestration features in nfv architecture," *Computer Networks*, vol. 217, p. 109281, 2022.

[2] Y. Yue, X. Tang, W. Yang, X. Zhang, Z. Zhang, C. Gao, and L. Xu, "Delay-aware and resource-efficient vnf placement in 6g non-terrestrial networks," in *2023 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2023, pp. 1–6.

[3] S. Moazzeni, P. Jaisudthi, A. Bravalheri, N. Uniyal, X. Vasilakos, R. Nejabati, and D. Simeonidou, "A novel autonomous profiling method for the next-generation nfv orchestrators," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 642–655, 2020.

[4] N. Ferdosian, S. Moazzeni, P. Jaisudthi, Y. Ren, H. Agrawal, D. Simeonidou, and R. Nejabati, "Autonomous intelligent vnf profiling for future intelligent network orchestration," *IEEE Transactions on Machine Learning in Communications and Networking*, 2023.

[5] S. Van Rossem, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester, "Profile-based resource allocation for virtualized network functions," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1374–1388, 2019.

[6] I. Kevin, K. Wang, X. Zhou, W. Liang, Z. Yan, and J. She, "Federated transfer learning based cross-domain prediction for smart manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4088–4096, 2021.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[8] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional gan," *Advances in neural information processing systems*, vol. 32, 2019.

[9] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data synthesis based on generative adversarial networks," *arXiv preprint arXiv:1806.03384*, 2018.

[10] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE transactions on neural networks*, vol. 22, no. 2, pp. 199–210, 2010.

[11] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[12] Y. Cheng, J. Lu, D. Niyato, B. Lyu, J. Kang, and S. Zhu, "Federated transfer learning with client selection for intrusion detection in mobile edge computing," *IEEE Communications Letters*, vol. 26, no. 3, pp. 552–556, 2022.

[13] K. Zhang, X. Liu, X. Xie, J. Zhang, B. Niu, and K. Li, "A cross-domain federated learning framework for wireless human sensing," *IEEE Network*, vol. 36, no. 5, pp. 122–128, 2022.

[14] X. Zhang, A. Mavromatics, A. Vafeas, R. Nejabati, and D. Simeonidou, "Federated feature selection for horizontal federated learning in iot networks," *IEEE Internet of Things Journal*, 2023.
[15] W. Zhang, L. Deng, L. Zhang, and D. Wu, "A survey on negative transfer," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 2, pp. 305–329, 2022.