A Trust and Explainable Federated Deep Learning Framework in Zero Touch B5G Networks

Sabra Ben Saad*, Bouziane Brik‡, Adlen Ksentini* , *Senior IEEE,*

Abstract—**The emergent Zero touch Service and Management (ZSM) paradigm aims to automate the orchestration and management of running network slices, in Beyond 5G networks (B5G), with an unprecedented level of scalability. To achieve this vision, ZSM calls for a large usage of advanced deep learning algorithms, in order to dynamically build efficient decisions. In this context, Federated deep Learning (FL) proved their efficiency in not only building collaborative deep learning models, among several network slices, but also ensuring the privacy and isolation of such network slices. Indeed, FL-based solutions give "machine-centric" decisions about running network slices and their performance, which will be then executed/applied by managers, i.e., slice manager staff/module. However, FL-enabled solutions do not provide any details about why and how such decisions were made, and thus such decisions cannot be properly trusted/understood by slice managers. To alleviate this issue, we leverage eXplainable Artificial Intelligence (XAI) paradigm that aims to improve the transparency of black-box FL decision-making process. In particular, XAI helps to explain the FL-based decisions to make them interpretable/trustable by network slices managers. In this paper, we design a novel XAI-powered framework to explain FL-based decisions. We first build a deep learning model in federated way, to predict key performance indicators (KPI) of network slices. Our FL-based KPI prediction is useful for the configuration and the management of network slice lifecycle, especially for the Service Level Agreement (SLA) violation and the network slice re-configuration. Then, we develop several XAI models on the top of our FL-based model, such as SHapley Additive exPlanations (SHAP), Local Interpretable Model-agnostic Explanations (LIME), RuleFit, and Partial Dependence Plot (PDP), to enhance the level of trust, credibility (in the local data/models), transparency, and explanation of the FL-based decisions, while adhering the data privacy, to different B5G network stakeholders, such as slice managers. Experiments results show the efficiency of our XAI-powered framework, to explain FL-based decisions related to latency KPI predictions.**

Index Terms—**Zero Touch Management (ZSM), 5G and Beyond, Federated Deep Learning, Explainable Artificial Intelligence.**

I. Introduction

Beyond 5G networks (B5G), or so-called "6G", are considered as key enabler of a wide range of new pervasive services related to different vertical industries, including eHealth, automotive, energy, and manufacturing [1]. This is possible by the support of massive number of coexisting network slices with different requirements and functionality. Indeed, thanks to network softwarization paradigm, network slicing consists of building isolated logical networks on top of a common physical resources, while meeting the needed requirements by network slices, in terms of latency, bandwidth, and reliability [2] [3]. However, such new services and paradigms introduce more issues on the management and orchestration of the massive network slices, during their life-cycles, that traditional network infrastructures fail to cope with. ZSM emerges as a promising paradigm, towards providing zero-touch orchestration and management of running network slices at massive scales in B5G networks. It designs a new autonomic orchestration and management framework, that heavily enabling distribution of functions together with data-driven techniques. To achieve this vision, ZSM calls for a large usage of advanced deep learning algorithms, in order to dynamically build efficient decisions at different levels, including radio resource allocation, energy-efficiency management, computing/memory resource provisioning, etc. In this context, FL proved their efficiency in building collaborative deep learning models, among several network slices. In particular, FL enables each network slice manager to build a local learning model using its proper data, and send it to a central entity, e.g. the central orchestrator and management framework, to be aggregated with the other network slices' local models. Hence, FL allows running network slices to create deep learning models, without sharing their data, and thus ensuring the privacy and isolation of such network slices. Indeed, FL-based solutions give "machine-centric" decisions about running network slices and their performance, which will be then executed/applied by managers, i.e., slice manager staff/module. However, FL-enabled solutions do not provide any details about why and how such decisions were made, and thus such decisions cannot be properly trusted/understood by slice managers. It is a very critical decision, because the FL model is collaboratively trained in disturbed nodes, and without access and verification to their local dataset. In other words, the main issue of FL-based mechanisms are the black-box decisions, whose internal functioning of the FL model is not understood and hidden. To alleviate this issue, we leverage XAI paradigm that aims to improve the transparency of black-box FL decision-making process. In particular, XAI helps to explain the FL-based decisions to make them interpretable/trustable by network slices managers. In this paper, we design a novel XAI-powered framework to explain FL-based decisions. Our framework studies the use of linear and non-linear XAI techniques, to identify the most informative features and investigate their impact on the final FL model predictions. Therefore, we first build a deep learning model in federated way, to predict KPI of network slices. Specifically, our FL-based model predicts the

*Eurecom, Sophia Antipolis, France. ‡University of Bourgogne, France. This work has been partially supported by the European Union's H2020 MonB5G (grant no. 871780) project.

latency KPI of network slices, in order to help in anticipating any violation of SLA, related to the latency KPI. Then, we develop several XAI models on the top of our FL-based model, such as SHAP, LIME, RuleFit, and PDP to enhance the level of trustiness (in the local data/model and the FL global model), transparency, and explanation of the FL-based decisions, to different B5G network stakeholders, such as slice managers. Noting that our framework leverages a real dataset about latency KPI that, we generate using OpenAirInterface platform. This paper is organized as follows. Section II gives a review of related work. Section III describes the design and specification of our proposed XAI-powered FL framework. Section IV presents the performance evaluation of our proposed XAI-powered FL framework. Finally, section V concludes the paper.

II. Related work

In this section, we present the few works, that addressed the explainability of FL-based models. In [5], the authors proposed a novel scheme to interpret FL-based models. They leveraged shapley value to compromise the data-privacy protection and model explainability in FL. This scheme enables each FL participant to get feature importance for its own features (data), and a unified feature importance for the features of the other participants. In work [6], the author proposed an explainable horizontal federated learning approach, leveraging integrated gradients explainability method. The authors compared between centralized and federated models, and applied integrated gradients to show the score of each feature related to each prediction for both models. We observe that few studies have been proposed to deal with the explanation issue of FL-based models. These studies addressed this issue either in general way, i.e. whatever the use case [5], or focusing on a specific and simple use case, such as Taxi travel time prediction [6], which is very simple and different from B5G-enabled use cases. In this work, we designed a novel framework that leverages RuleFit, LIME, SHAPE and PDP as XAI approaches, to explain and interpret an FL-based model of latency prediction. We note that our framework enables not only to deduce the most relevant features conducting to each FL-based latency prediction, but also providing both local and global explanations related to latency predictions of running network slices.

III. Proposed architecture of the XAI-Empowered FL Framework

In this section, we present our XAI-empowered FL framework for the B5G networks. First, we give an overview about the

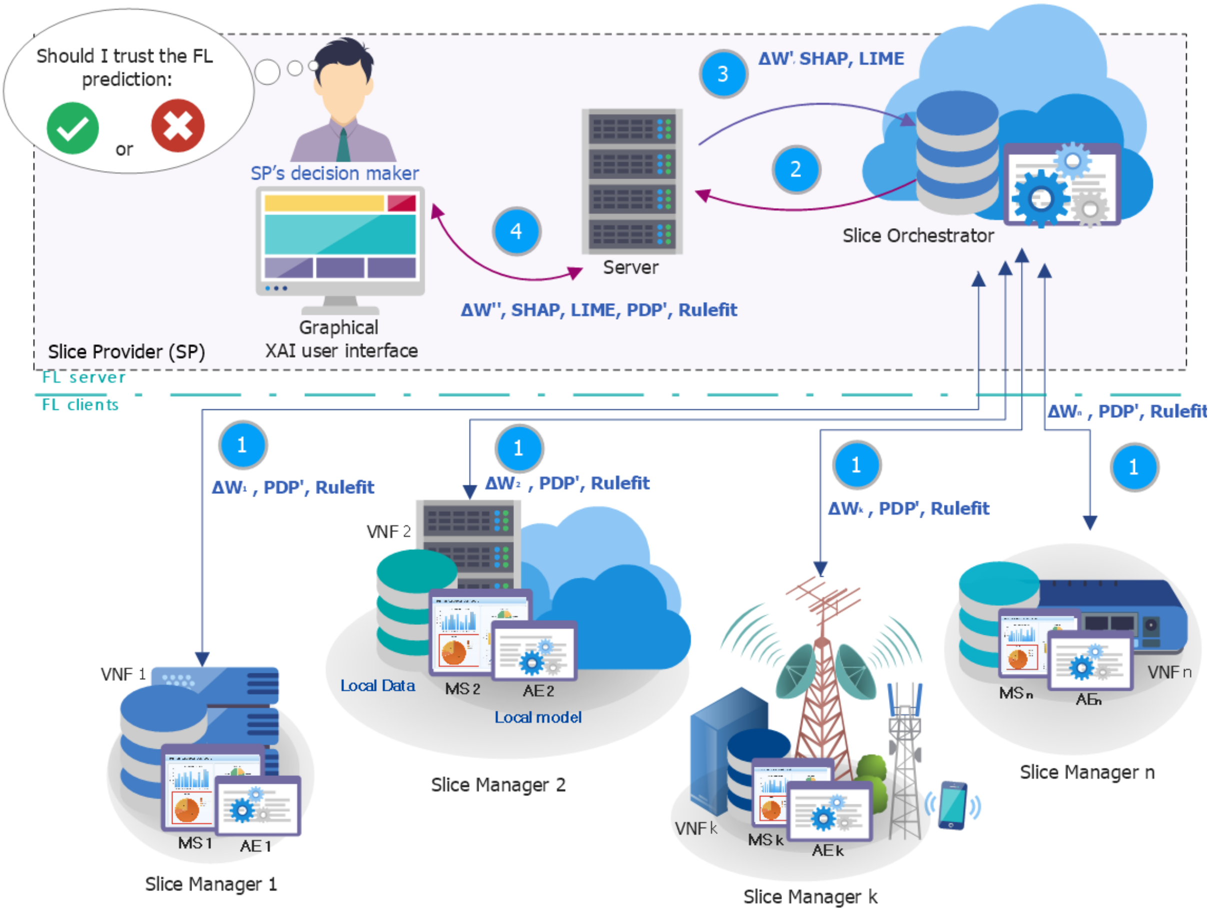


Fig. 1: The proposed trust B5G Architecture of our XAI-Empowered FL Framework.

architecture of our proposed FL framework. Then, we describe our deep neural architecture we build to predict the latency related to the B5G networks, and our XAI-FL approaches we applied to interpret and explain the outputs of our deep learning model, trained on local data in disturbed nodes, during the FL process.

A. *System overview Architecture*

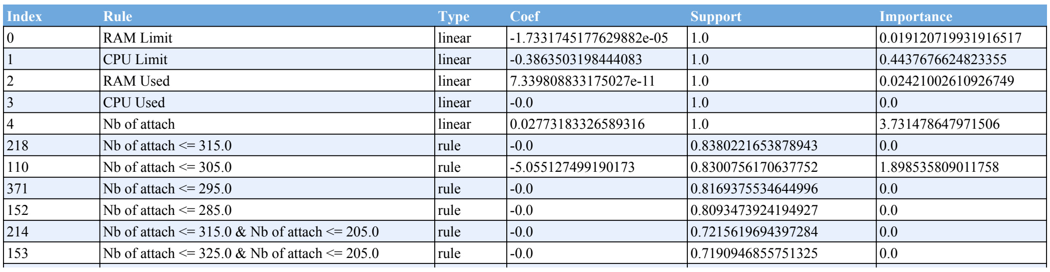
Fig.1 gives an overview about the architecture related to our FL framework. Our architecture presents different 5GB parties, including the slice provider, the slice managers, and inter-slice manager (orchestrator), that collaboratively train a deep learning model, in federated way. First of all, the slice orchestrator initiates the federated learning process to build a learning model about latency KPI prediction. It defines the learning parameters such as, learning rate, neural architecture (number of layers/neurons), activation functions, and neural weights optimizer, in addition to the needed data. The orchestrator then sends such information to the involved slice managers in the FL process (step 1 in Fig. 1). After that, the slice managers start to build their local learning models, using their own data. Once done, the slice managers send their local models to the slice orchestrator for aggregation Δw_i, in order to generate a global model (steps 1 and 2 in Fig. 1). The latter will be deployed at the slice managers level, to predict the latency KPI in real time (steps 3 and 1 in Fig. 1). In addition, we apply four different XAI approaches, to generate local, global, and feature importance-based explanations, related to the latency predictions. Thus, our framework allows the slice provider's decision makers to not only monitoring the latency KPI of running network slices, but also how and why predictions are made, through a graphical XAI user interface (step 4 in Fig 1). It is worth noting that our study is mainly based on a realistic B5G dataset, that we generate using OpenAirInterface, implementing the network slicing concept, called EARCD (Eurecom AMF Resource Consumption Dataset). Our dataset provides the response time (latency) of the Access Mobility Function (AMF), running as virtual network functions (VNF) inside the network slices, to handle user attach requests, while considering various parameters, including available and consumed CPU and RAM memory resources. Therefore, we consider ten running network slices and we generate a sub-dataset for each network slice of 2813 samples (rows). Each sub-dataset comprises five input features (CPU capacity, RAM capacity, used CPU, used RAM, and number of attach request), and one output feature, which corresponds to latency in terms of average duration to process users attachments, by the network slices' AMFs. Noting that we also leverage 5G UE emulator, my5G-RANTester[1], to emulate multiple number of users that send a high number of attach request packets, to the ten network slices (AMF functions). Furthermore, each slice manager implements a neural network of one input layer of five neurons (our five input features), four connected hidden layers of 20 neuron nodes, and one output layer of one neuron (latency prediction). The activation function in all layers is the rectified linear activation function.

B. *Explainable FL-based models for B5G networks*

Several ML/DL explainable approaches were proposed to show the features impact, on the target labels. In order to interpret "Black-Box" of our FL-based model, we apply model agnostic approaches [4], which aim to understand the inner working of learning models. The model agnostic methods can be divided into two main categories: local model-agnostic and global model-agnostic. In what follow, we present the model-agnostic approaches, that we apply to explain our FL model.

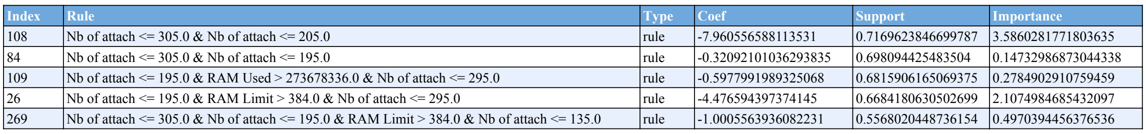
1) Global Model-Agnostic Methods: The Global methods aim interpret learning model working in general way. We apply Partial Dependence Plots (PDP) method [4].

a) Partial Dependence Plot (PDP): PDP method shows which input features of the dataset will highly impact the predictions of the learning model. In particular, PDP shows



Index	Rule	Type	Coef	Support	Importance
0	RAM Limit	linear	-1.7331745177629882e-05	1.0	0.019120719931916517
1	CPU Limit	linear	-0.3863503198444083	1.0	0.4437676624823355
2	RAM Used	linear	7.339808833175027e-11	1.0	0.02421002610926749
3	CPU Used	linear	-0.0	1.0	0.0
4	Nb of attach	linear	0.02773183326589316	1.0	3.731478647971506
218	Nb of attach <= 315.0	rule	-0.0	0.8380221653878943	0.0
110	Nb of attach <= 305.0	rule	-5.055127499190173	0.8300756170637752	1.898535809011758
371	Nb of attach <= 295.0	rule	-0.0	0.8169375534644996	0.0
152	Nb of attach <= 285.0	rule	-0.0	0.8093473924194927	0.0
214	Nb of attach <= 315.0 & Nb of attach <= 205.0	rule	-0.0	0.7215619694397284	0.0
153	Nb of attach <= 325.0 & Nb of attach <= 205.0	rule	-0.0	0.7190946855751325	0.0

Fig. 2: Rulefit results: Top 15 Rows of the Un-filtered Rules Features.



Index	Rule	Type	Coef	Support	Importance
108	Nb of attach <= 305.0 & Nb of attach <= 205.0	rule	-7.960556588113531	0.7169623846699787	3.5860281771803635
84	Nb of attach <= 305.0 & Nb of attach <= 195.0	rule	-0.32092101036293835	0.698094425483504	0.14732986873044338
109	Nb of attach <= 195.0 & RAM Used > 273678336.0 & Nb of attach <= 295.0	rule	-0.5977991989325068	0.6815906165069375	0.2784902910759459
26	Nb of attach <= 195.0 & RAM Limit > 384.0 & Nb of attach <= 295.0	rule	-4.476594397374145	0.6684180630502699	2.1074984685432097
269	Nb of attach <= 305.0 & Nb of attach <= 195.0 & RAM Limit > 384.0 & Nb of attach <= 135.0	rule	-1.0005563936082231	0.5568020448736154	0.4970394456376536

Fig. 3: Rulefit results: Top 5 Rows of the Filtered Rules.

[1] https://github.com/my5G/my5G-RANTester

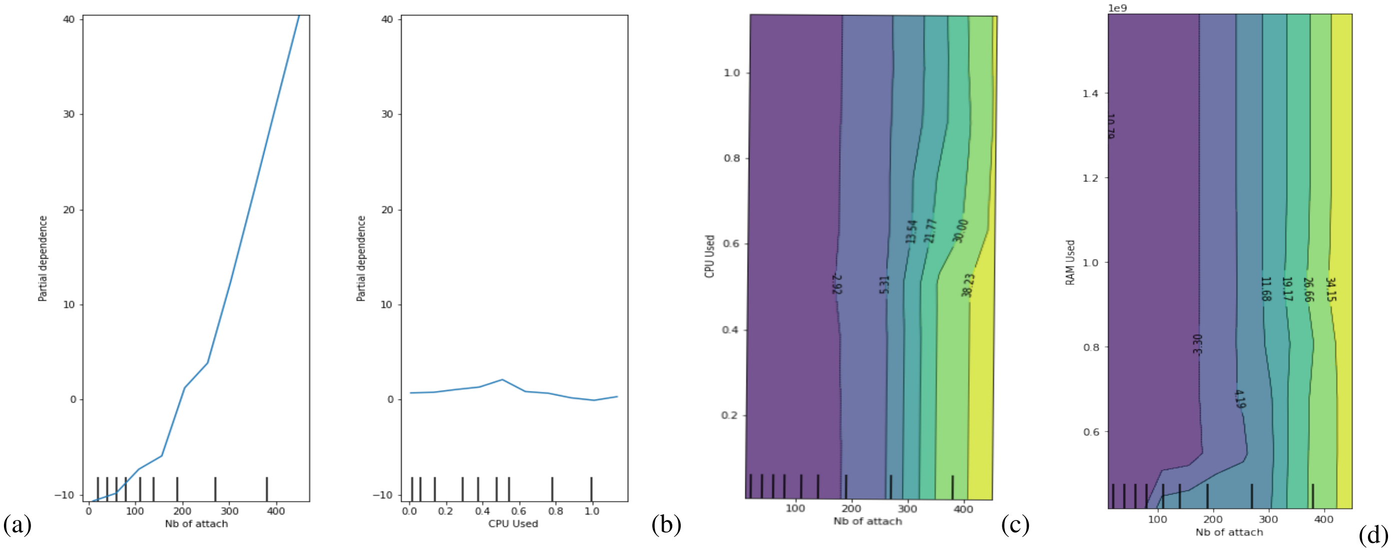


Fig. 4: Partial Dependence Plots on EARCD dataset.

the linear relationship between features and target label (s). It means that if one features goes up, the target label will go up (or down) too. Thus, A PDP enables to determine whether the relationship type between the target label and each feature is linear, monotonic or more complex. The PDP plots visualize the average partial relationship between the predicted target and one or more features. In our case, the FL clients will only share a partial information of PDP, which are the variation percentage, in order to save their data privacy.

2) Local Model-Agnostic Methods: The local interpretation methods explain individual predictions. Therefore, the predictions of a single instance is described as the sum of feature effects. In this context, we apply three main methods: SHAP, LIME, and RuleFit.

a) SHapley Additive exPlanations (SHAP): It is based on game theoretically optimal Shapley values, to interpret the outputs of any machine-learning models [4]. SHAP consists to calculate the contribution/impact of each feature to the final ML/DL model output. Specifically, The SHAP approach calculates shapley values from coalitional game theory, where a player, in a coalition, corresponds to a feature value of the data instance. Shapley values show how to fairly distribute the "output" (prediction) among the features. Noting that a player can be a set of features, such as pixels of an image, or one feature value, such as for tabular data. In our study, we apply SHAP method by calculating the shapley values of each individual prediction, where each individual feature of our dataset corresponds to a player in the coalition.

b) Local Interpretable Model-agnostic Explanations (LIME): LIME method consists to train interpretable models to approximate the predictions of the ML/DL model. In particular, LIME creates a new dataset containing perturbed samples and their predictions of the studied ML/DL model. Then, an interpretable model is trained using the new dataset, which is weighted by the proximity of the sampled instances to the instance of interest. However, the new interpretable model will approximate accurately the ML/DL model predictions locally (local fidelity), and cannot provide a good global approximation. In our case, we first create a perturbed data instances of our EARCD dataset, then we build an interpretable model of our local/individual predictions of the latency KPI.

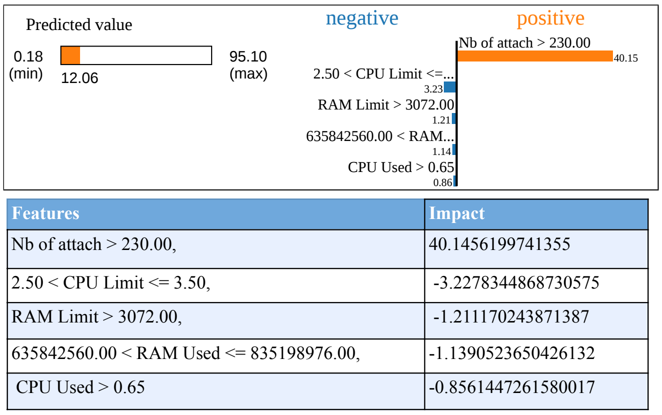
c) Rulefit Method: The RuleFit algorithm is also another explainable solution to understand the relationship between features in the form of decision rules [4]. RuleFit builds a sparse linear model, leveraging both the dataset features and a set of new features that generated from the interactions between the original dataset features (decision rules). These new features are automatically built from decision trees, where each tree path is transformed to a decision rule. In our study, we apply RuleFit to generate hundreds of new decision rules. We then apply additional filtering/combining to determine the most important/informative rules. Therefore, such new rules will help slice provider's decision makers to better interpret and understand our FL-based model predictions.

IV. Performance and Evaluation

In this section, we validate our XAI-empowered framework through an experimental study. We implemented the proposed XAI framework in Python, leveraging tensorflow library as well as XAI libraries, including SHAP, and PDPBox. Following sections, we present the obtained results of the different XAI-techniques used.

A. Performance evaluation of Global Model-Agnostic Method

Fig. 4 shows the obtain partial dependence plots trained on our EARCD dataset. The left plot in the Fig. 4 depicts the effect of the number of attach requests on the latency; we can clearly see a linear relationship among them (Straight line), when the number of attach requests is superior to 200. Similarly, we



Features	Impact
Nb of attach > 230.00,	40.1456199741355
2.50 < CPU Limit <= 3.50,	-3.2278344868730575
RAM Limit > 3072.00,	-1.211170243871387
635842560.00 < RAM Used <= 835198976.00,	-1.1390523650426132
CPU Used > 0.65	-0.8561447261580017

Fig. 5: Lime results displaying the predicted label and the top five features impact.

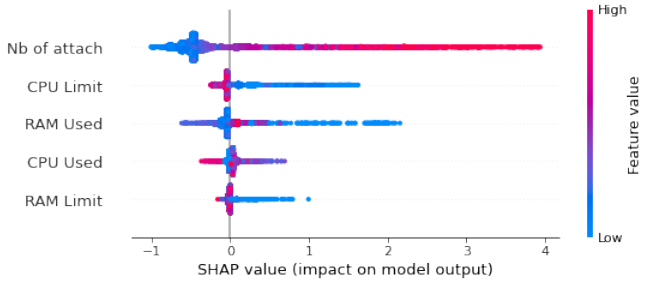


Fig. 6: SHAP value (impact on model output)

analyze the effect of the used CPU on the latency (Fig.4-b) as well as on the number of attach requests (Fig.4-c), to show the interaction between these two features. We clearly see an interaction (different stripes and colors from purple to yellow) between the “Number of attach” and “CPU Used” features. Furthermore, the purple/blue color reflects the low dependence, while the green/yellow presents the high dependence of the features. For a number of attach less than 300 requests, the latency is nearly independent (=2.92%) of the CPU used, shown in purple and blue; whereas for values greater than 300 there is a strong dependence (30.00%) on CPU used (shown in green, and yellow color). Similarly, there is also a strong dependence between the ”number of attach” feature and ”RAM used”, ”CPU Limit”, and ”RAM Limit” features, when their values are greater than ”300”, ”200”, and ”300”, respectively.

B. *Performance evaluation of Local Model-Agnostic Method*

Fig. 6 and Fig. 7 show the most informative features in an orderly manner. For a particular observation, each input feature has either a positive or a negative contribution to the final latency prediction (Fig. 6). Moreover, the blue color reflects the low value of the feature, while the Red color presents the high value of the features. Based on the obtained plots, we can notice that the most important feature is the number of received attach request. Also, the more number of attach is high, the more the latency is high. It is reasonable explanation because the processing duration of the received attach requests increases as the number of received requests increases as well. After that, the second important feature is the ”CPU Limit” feature. Indeed, the AMF functions with a high number of CPUs will generate a less latency, as compared with AMFs with low number of CPUs. Hence, the more resources allocated to AMFs are high, the more attach request treatment is faster, so the latency of attach requests decreases. Similarly, using the RuleFit method, Fig. 8 shows similar results of feature importance, which confirms the obtained results in Fig. 6 and Fig. 7 for the SHAP method. In Fig. 8, the highest scoring features corresponds to the following features: (1) number of attach: corresponds to the number of Attach Requests sent to the AMF; (2) CPU Limit: corresponds to the Number of CPU allocated to AMF; (3) RAM Used: corresponds to the memory used during AMF processing; (4) CPU Used: corresponds to the CPU used during the AMF processing; (5) RAM Limit: corresponds to the memory allocated to the AMF. Additionally, RuleFit method enables to generate new rules about dataset features and their combinations. Such rules will help to show the importance of the features for the model predictions.

Fig. 2 illustrates some rules of the RuleFit method through a table containing five main columns: ”Rule” which is either an existing feature or a rule formula (combining more features). ”Type”, which is either linear for the existing features, or rule for a new generated rule. ”Coef” which is the coefficient of the rules and features. ”Support” presents what proportion of the dataset supports the corresponding rule/feature. ”Importance”, which reflects the importance of each rule on the predictions. Noting that the feature importance is deduced from the weights of the regression model. Besides, in Fig. 3, we eliminate the insignificant features based on several criteria, such as (1) eliminate the rules with 0.00 coefficient since they are not significant; (2) sort the rules based on their support value; and (3) eliminate the leanest rules to keep more important rules. As results, all the most significant rules involves the ”Nb of attach” feature, which confirms the previous results (Fig. 6, Fig. 7, and Fig. 8). Therefore, we notice that the most significant rule combinations also mostly contain these features. As an example, the first rules above work as follow: **IF** ”Nb of attach” $\leq$ 305 and (**&**) ”Nb of attach” $\leq$ 205, **THEN** we have 3.58 as a feature importance for the KPI predictions. The second rule is very similar to the first one. Therefore, it might make

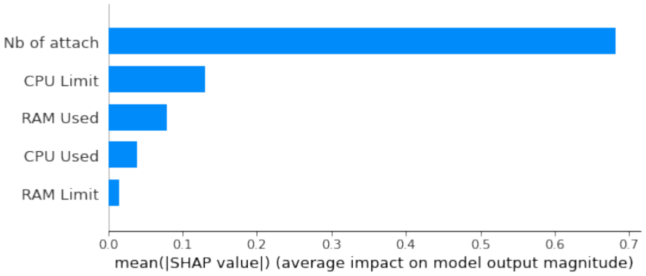


Fig. 7: SHAP value (average impact on model output)

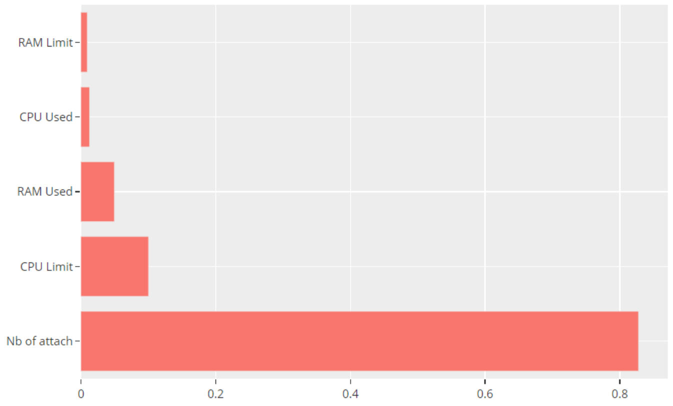


Fig. 8: Feature Importance using Rulefit on EARCD Dataset.

sense to combine or eliminate some of the similar rules for a more interpretable set of rules. Ideally, we need to apply some filtered methods to group the similar rules and create a leaner model, which would cover all the interaction effects, which would ultimately help the ML analyzer to understand the working mechanism of our FL trained model. Fig. 5 depicts the positive/negative impact of the features on our FL model predictions (latency KPI), using the LIME method. Using a test instance, the value of the predicted label, which is the latency, is equal to 12.06 seconds. It is presented by a bar on the left, and depicted by the given vector. We can see the colors blue and orange, depicting negative and positive associations, respectively. As we observe, the "number of attach" feature has a higher and positive impact (sign (+)) on the predicted latency, and when the number of attach is more than 230 requests, the higher latency we obtain. In addition, we also see that the "CPU Limit" feature has a lower/negative impact on the predicted latency (sign (–) in Fig. 5). In general, we can deduce that our XAI-empowered framework provides a good performance in explaining our FL-based model, when predicting the latency KPI. It mainly helps to interpret and understand the local and global functioning of our FL-based model by showing the most informative features conducting to each prediction. Therefore, our XAI-based framework will help decision makers of slice providers, to not only detecting latency SLA-related violations for each running network slice, but also to determine the main reasons that conduct to such SLA violations, and hence studying how to anticipate them and use the results for the most advantageous network slice re-configuration (i.e, the resource allocated for the slice: RAM, CPU,..). Additionally, according to the FL concept and the characteristics of the XAI algorithms, the latter can be classified into 2 categories: (1) XAI models run in the FL client side, which are PDP and Rulefit (because these algorithms are related to the local dataset and the local model (model fit)), and (2) XAI models run in the FL server side, which are LIME and SHAP (because these algorithms are related to the label prediction). However, for PDP, the FL clients will only share the plots that show the percentage of feature impact on the target label, in order to save their data privacy. Moreover, the reasons behind the usage of different XAI algorithms, that present similar results, is about increasing confidence in the outcomes of XAI algorithms.

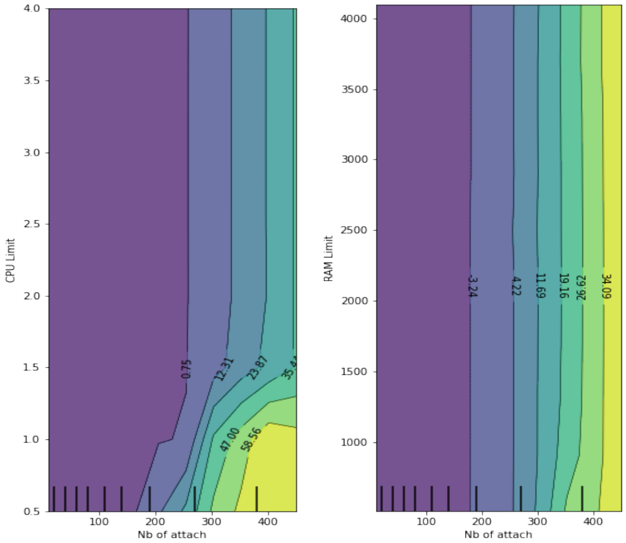


Fig. 9: Partial Dependence Plots on EARCD dataset.

V. Conclusion

In this paper, we designed a novel XAI-powered FL-enabled framework that enables not only the prediction of the latency KPI, but also the interpretation of critical predictions made by the FL-based model. First, we built a new DNN model to predict the KPI (latency) of network slices, in federated way. Then, we developed multiple XAI models, i.e., RuleFit, LIME, SHAP, and PDP, on top of our federated DNN architecture, to enable more trust, credibility (in the local data, the local model and the FL global model), transparency, and explainability of the predictions made by our FL-based framework to different B5G ML users, such as the slice provider. The in-depth experiments results on Latency predictions, showed the efficiency and explainiblity of our proposed FL framework. This makes it a promising FL framework and explainable Deep Learning FL Framework for its users.

References

[1] M. Isaksson et al, "Secure Federated Learning in 5G Mobile Networks", IEEE GLOBECOM, 2020.

[2] Sabra Ben Saad et al, "A Trust architecture for the SLA management in 5G networks", IEEE ICC, 2021.

[3] Sabra Ben Saad et al, "An end-to-end trusted architecture for network slicing in 5G and beyond networks", wiley journal, 2021.

[4] Christoph Molnar, "Interpretable Machine Learning A Guide for Making Black Box Models Explainable", Chapter 6, 2022-03-29.

[5] Guan Wang, Digital and Smart Analytics, "Interpret Federated Learning with Shapley Values", FML Workshop, Hong Kong, 2019.

[6] Jelena Fiosina, "Explainable federated learning for taxi travel time prediction", VEHITS, 2021.