IFIP Networking 2024 1571019173
2024 IFIP Networking Conference (IFIP Networking)

Navigating Explainable Privacy in Federated Learning

Chamara Sandeepa, Thulitha Senevirathna, Bartlomiej Siniarski, Shen Wang, Madhusanka Liyanage
School of Computer Science, University College Dublin, Dublin, Ireland.
Email: {abeysinghe.sandeepa, thulitha.senevirathna}@ucdconnect.ie
{bartlomiej.siniarski, shen.wang, madhusanka}@ucd.ie

***Abstract*—With the dawn of distributed Artificial Intelligence (AI) accelerated with the upcoming Beyond 5G (B5G)/6G networks, Federated Learning (FL) is emerging as an innovative approach to performing distributed learning in a privacy-preserved manner. Numerous techniques are available for fine-tuning AI-based parameters in FL. Depending on factors such as specifications, use cases, and limitations, this tuning method can vary among different FL processes, which is essential to consider when designing and developing FL-based systems. Therefore, in this research, we conduct an investigation on the variation and trade-offs of such AI-based parameters, with metrics including user diversity analysis using t-SNE-based projections, and we also examine AI parameter performance from a Differential Privacy (DP) perspective. In addition, we expand our study to Explainable Artificial Intelligence (XAI)–based tuning and use SHAP and LIME methods to decipher distributed model complexity. We assess the interpretability of FL models by leveraging parameters like consistency, currentness, and compacity, which provide insight into their effectiveness and resilience in practical settings.**

***Index Terms*—Federated Learning, XAI, Privacy, Tuning parameters, B5G/6G**

I. Introduction

Federated Learning is a distributed approach to ML that prioritises privacy preservation in decentralised data environments. Unlike traditional centralised models, where data is collected and sent to a central server for processing, FL enables model training directly on individual devices or local servers while keeping sensitive data localised. This distributed process allows participants, such as smartphones, IoT devices, or edge servers, to collaboratively learn a shared model without sharing raw data. By aggregating the model updates instead of raw data, FL mitigates privacy risks associated with centralised data sharing, making it particularly valuable for industries like healthcare, finance, and other sectors dealing with sensitive information.

The fine-tuning of parameters in FL is an important consideration for security and privacy due to its varying properties and diverse set of options. For example, the interpretability and success of a model training may depend on the type of ML model used in the training process. It may also be a key factor that can affect the privacy of clients since if the model is too simple and interpretable, a third-party eavesdropper may be able to infer properties or reconstruct data from the trained models [1]. However, if the model is highly complex, it may be difficult to understand and inspect, making it easier for an adversarial client to inject malicious updates like poisoning or backdoors, which can be hidden inside complex parameters in the model. Thus, in this research, we observe which aspects of fine-tuning in FL can affect the security and privacy of clients while understanding its effect on explainability.

The distributed process of FL can be offered in a service-oriented architecture named FL as a Service (FLaaS), which is introduced in [2]. To simulate such a federated service, we present a simplified representation of this architecture. We use two metrics on this architecture to evaluate the following properties in FL:

- **Accuracy trade-off with privacy** – to identify the trade-off of a privacy solution compared to the utility or the accuracy of the FL models. Local DP is used as the privacy preservation technique.
- **User diversity** – this metric quantifies the diversity among the clients participating in the FL process. This is to observe how client models vary with the changes in FL configurations and training processes. As the quantifying metric, we use cosine similarity.

The application of specific parameters has already been investigated in FL. For example, work in [3] analyse parameter variations within client DP. However, they do not provide further investigations on the effect of DP on FL under varying forms of data, model architectures, and time considerations. Authors in [4] already provide comprehensive analysis on Federated aggregation algorithms; thus, we do not consider them. Investigations on applying XAI on FL have already been started in research works like [5]. However, none of the related works provided an analysis of how the XAI parameters can be tuned and their implications on FL systems, which is considered in our work.

The rest of the paper is arranged as follows: Section II provides an overview of the system architecture.

978-3-903176-63-8/24/$31.00 ©2024 IEEE

Tuning of AI parameters is discussed in Section III. Next, in Section IV, tuning of XAI parameters under the same test environment is presented. The paper finally concludes in Section V with key findings.

II. System Architecture

During the implementation of an FL-based system in a real-world application, many options are available for fine-tuning parameters and making system configurations. Fig. 1 provides an overview of the available tuning components in the designed FL process.

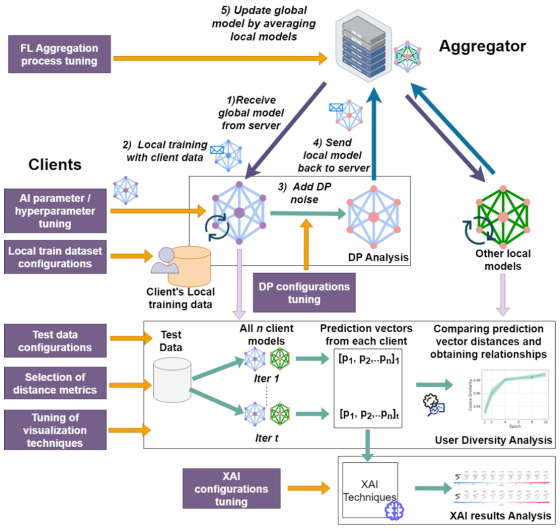


Fig. 1: Overview of the FL architecture, metrics derivation process, and possible options for parameter tuning.

For example, when considering the server-side aggregation, several algorithms, such as Federated Averaging (FedAvg) or Federated Stochastic Variance Reduced Gradient (FSVRG) [4], can provide different aggregation results. The number of FL rounds is also another parameter that the server can consider achieving an expected level of global accuracy.

For individual clients, depending on the client's availability of the dataset amounts, the local models trained can vary in accuracy. They may also have Non-Independent and Identically Distributed data (non-IID), which can lead to biases in the local models, resulting in significant variations in the distance metrics in user diversity. The client model architectures and parameters can also be configured based on the requirement, which can impact the model's accuracy and convergence. Furthermore, their hyperparameters, such as learning rate, activation functions, dropout rates, local model optimisation function, and local training epochs, are some other components that each individual client can tune. However, having unique or varying configurations for these local hyperparameter settings can also change the convergence outcome of the global model.

III. Tuning AI parameters with fixed FL setting

The tuning of FL-based systems can be performed by varying a selected independent parameter and observing its impact over a dependent parameter while keeping other configurations constant. We first discuss on how a selected AI parameters can be tuned under a specific configuration in an FL environment.

A. Experiment Configuration

Tuning AI parameters in the local models is an option for obtaining optimal trade-offs among utility, privacy, and explainability. Therefore, for AI parameter analysis, we used two experiments: the privacy utility trade-off with DP and user diversity analysis. For DP-based experiments, we selected two such FL client-specific parameters for testing and tuning. They are: 1) the architecture of the AI model used for the local clients and 2) a number of local training rounds for the clients. In the user diversity, we analyse how t-SNE-based projections of client model output vary with 1) global iterations and 2) DP-based privacy. For the experiments, we considered two datasets: 1) MNIST, which is a set of images of handwritten digits from 0-9, and 2) NSL-KDD, a network intrusion detection dataset with different attack types. For FL simulations, we used the framework Flower [6], where 10 clients are defined. All simulations were run 10 global aggregation rounds. In the experiments, we set each client has a majority 90% of data belonging to a single class, where other 10% is added with randomly selected label. We use 10,000 data records per each client. The experiments were performed using a computing instance with an Intel Xeon 2.20 GHz CPU and 26GB RAM.

B. Discussion of Results

1) Differential privacy variation: In privacy-based analysis, first, an experiment was performed on the Neural Network (NN) architecture, where three NN models are compared for the NSL-KDD dataset. The initial NN is called NN-1, which consists of a hidden layer with 512 units followed by a dropout layer and the output layer. We define the next model as NN-2, which consists of an extra layer with 256 hidden units. Another model is defined as NN-3, which further adds a hidden layer with 128 units. The results of the experiment are shown in Fig. 2a. From the observations, the initial round for more complex NN has relatively lower accuracy, which gets compensated over the next rounds. Therefore, we can consider that more complex NN architectures tend to require more rounds for convergence.

The second scenario is the analysis on the effect of local rounds on Fig. 2c shows accuracy results for a number of local rounds used for training by a client. In this case, we select 1, 5, and 10 local epochs of

local training for an individual client before sending the locally trained model to the aggregator. Then, changes in the overall accuracy of the global models over FL rounds were considered. When using higher local rounds, the accuracy is higher even at the initial round. This may be the reason that the model will be a better fit for the local data with more local training rounds.

Comparison of accuracy by applying DP is next experimented with a selected Epsilon privacy value of 1 for the DP. This value is kept constant for all DP-based experiments. By varying the NN architecture, we observe a considerable change in the accuracy for more complex NN architectures, which take more global iterations to converge to a threshold accuracy of around 90% for NSL-KDD. This could result in due to the accumulation of more noise with the presence of a larger number of NN layers. When considering the local training rounds, fewer rounds per client also have lesser initial global model accuracy, similar to the scenario without privacy but amplified with DP-based noise. Fig. 2b and 2d provide the results with DP.

Next, experiments regarding the CPU time taken for running the 10 FL rounds were conducted. Results are presented in Fig. 3. It shows that with a more complex NN architecture, the time taken to process 10 rounds is higher, but it may be increasing linearly by a small quantity with an extra layer. However, with an increase in local rounds, compared with 1 local round, a significant increase in time can be observed for 10 local rounds. The use of DP can result in much higher aggregation times compared with no DP due to intermediate steps occurring in the DP process.

The aggregation of the MNIST dataset was also considered without and with DP, which is shown in Fig. 4. This dataset suffers much higher utility loss over FL rounds compared to any scenario of NSL-KDD in Fig. 2. Therefore, the nature of the dataset can affect the results when utilising DP.

2) User diversity metrics variation: The user diversity here is defined as the t-SNE projections analysis with the observation of clusters with different classes belonging to a particular group of clients. Each client has its own cluster, which comprises points that represent test data points belonging to that client. Higher diversity among users signifies a clear separation of classes, which can further imply the potential vulnerability of having privacy leakages due to highly distinguishable data in the clients.

The t-SNE embeddings can be visualised by tuning a parameter called perplexity [7], which balances the attention between local and global aspects of the data. Two values of perplexity were considered, 10 and 100, where perplexity of 10 can highlight local properties, while 100 provides more global properties in the dataset. As shown in Fig. 5, user diversity is clearer for MNIST with global features at higher perplexity in early rounds. However, FL models may tend to get combined, leading to lower clarity in class clusters and resulting in lesser diversity for users. Thus, it can signify an increase in privacy since an adversary may not be able to isolate a particular cluster.

Considering the NSL-KDD dataset, we observe a similar property, where for NN-1 configuration, the perplexity values are shown in Fig. 6. Here, only two classes are present for either DoS traffic or benign data, where multiple clients can be available in the same cluster. Higher perplexity can provide better visualisations of the global class-specific properties.

When applying DP, the perplexity can be observed to be changed further; here, unlike the initial scenario where clusters are clearly separated in the first round, we can observe the scattered t-SNE cluster configurations with noise injection. The results are shown in Fig. 7. This also confirms that the addition of DP-based noise can increase the privacy levels of individual clients since clusters now have less user diversity, making it difficult for an attacker to isolate a particular client. However, as previously shown, the model utility gets heavily lower with this level of DP for MNIST.

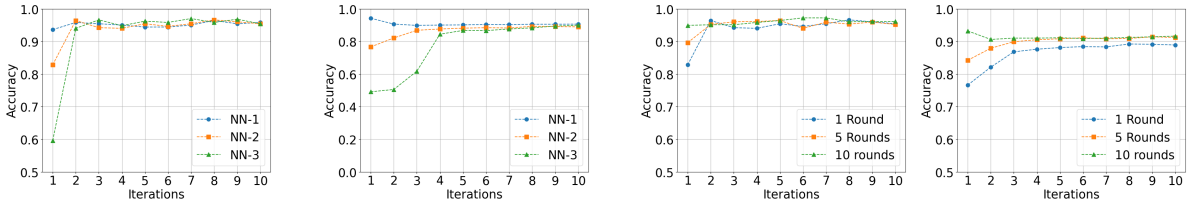
Next, we show the variation in L2 distance metrics and cosine similarity metrics among clients observed with the DP application. The variations of the metrics are shown in Fig. 8. Without privacy, cosine similarity can be observed to increase gradually after the first aggregation round, making models less diverse over the training round. However, this is not evident with the application of DP, since it makes all model updates noisy, causing uniform similarity across many FL rounds.

IV. Tuning XAI Parameters with Same FL Environment Setting

XAI hyperparameters are controllable parameter values used in fitting and generating explanations. Tuning these parameters is a use-case-specific task, with each method having its own set. However, different hyperparameters can have different combined effects. Matching hyperparameter values with use case knowledge can narrow the parameter range needed for testing and can improve the accountability of the system with a minimum sacrifice of system efficiency. Thus, the knowledge of the behaviour of hyperparameter tuning in distributed environments is highly important. Therefore, we carried out experiments to test the effect of several hyperparameters in XAI for the same distributed environment as in Section III. The results and observations are discussed below.

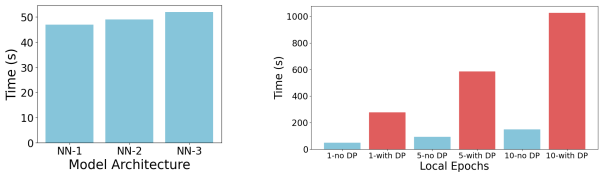
A. Experiment Details

In this experiment, we vary the explainer type when generating explanations from the global model under



(a) NN architecture, no DP (b) NN architecture with DP (c) Local rounds, no DP (d) Local rounds with DP

Fig. 2: Accuracy variation of FL rounds without vs. with DP for NN architectures and local rounds for NSL-KDD.



(a) NN architecture (b) Local rounds without/with DP

Fig. 3: Time taken for local epochs with vs. without DP

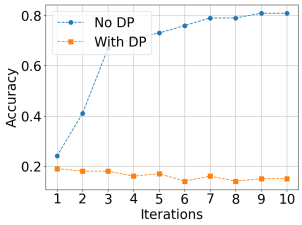


Fig. 4: Aggregation for MNIST with vs. without DP.

9 iterations of the federated learning system introduced in the previous section. The model converges to above 95% accuracy in the first 4 aggregation rounds. However, we left it for five more rounds to ensure the model is not at any local minimum. The main hyperparameter we changed in this experiment was the XAI method itself. Currently, many XAI methods have been introduced that can be used for a wide range of model types. Model-agnostic XAI methods can be used for any model type despite their internal architecture.

However, for a single AI distributed model, there are several possible model-agnostic XAI methods available. Therefore, one of the main questions would be, "Which model-agnostic XAI method is the best for our model?". Thus, we consider three popular XAI methods, namely, LIME, KernelSHAP, and SamplingSHAP. All these methods can generate local explanations based on the local neighborhood of a data point by having only the query access to the global/client models in a distributed environment. However, the explanation generation techniques are different for each method. KernelSHAP calculates the relevance of each feature [8] using a unique weighted linear regression. SamplingSHAP is an adaptation of the technique suggested in [9] and computes SHAP values under the premise of feature independence. LIME [10] explains instances by locally approximating a surrogate model to explain the neighborhood.

For a given XAI method, there are several parameters that can be changed to optimise the performance and outcome of the explanations. To showcase this, we use the $num\_samples$ parameter in LIME to check how the accountability of the explanations is varied based on it. The four accountability metrics are currentness, which measures how timely explanations can be generated for a data sample; explainer score (only for LIME), which measures how well-fit the surrogate model used for generating explanations is; consistency, which measures how consistent the explanations are across different explainers, and compacity, which measures how much of the model output is explained by a fixed number of features (in this case, 5 features).

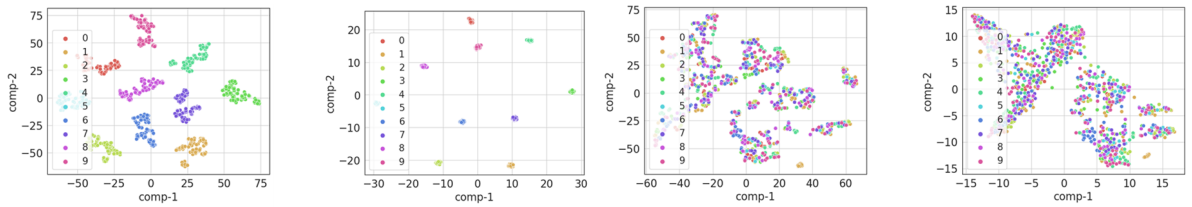
B. Observation and Discussion

Variation of accountability metrics with a number of perturbed samples for different rounds of global model aggregation is shown in Fig. 9. Here, except for the R2 score of the Ridge regressor in LIME, all the rest of the metrics show an upward trend. The compacity of all the models has a linear relation to the number of samples whent the perturbations are over 4000. Currentness, which is the time to generate explanations, has a linearly increasing relationship with the number of samples due to the increase in computation time.

We further analyse the variation of the following metrics over the training iterations.

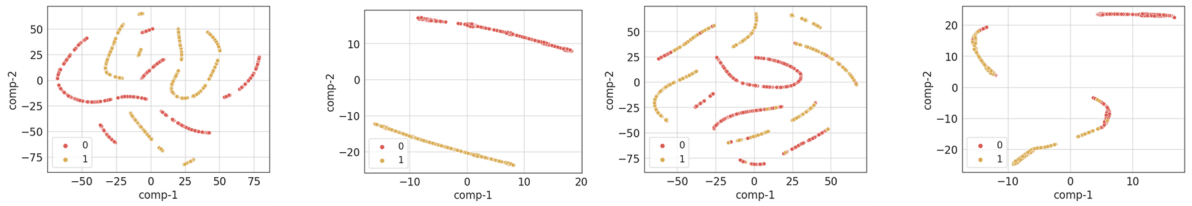
1) Consistency metric: The average consistency variation was observed across the iterations, as shown in Fig. 10. Apart from the drop in consistency across the aggregation iterations, the average consistency of the explanations for the model shows an increase at iteration 4. But with further breaking down the average into consistency between each explainer, we can see that consistency involving LIME (Fig. 10c Fig. 10d) is showing the at iteration 4 (Fig. 10b). This means that LIME explanations show a higher distance away from the other two explanations (kernelSHAP and sampling-SHAP). The consistency of kernel and sampling SHAP explanations shows a slight increase when the model is aggregated over time.

2) Currentness metric: When analysing the currentness, we observed the patterns as shown in Fig. 11. The



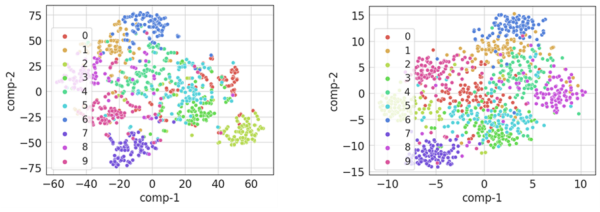
(a) round 1, perplexity 10 (b) round 1, perplexity 100 (c) round 10, perplexity 10 (d) round 10, perplexity 100

Fig. 5: t-SNE clusters for different configurations of perplexities over multiple rounds for MNIST



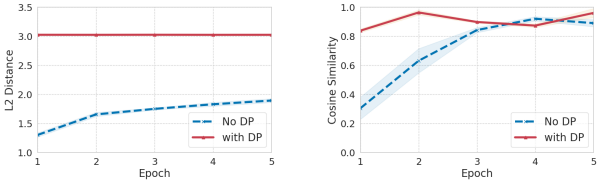
(a) round 1, perplexity 10 (b) round 1, perplexity 100 (c) round 10, perplexity 10 (d) round 10, perplexity 100

Fig. 6: t-SNE clusters for different configurations of perplexities over multiple rounds for NSL-KDD



(a) round 1, perplexity 10 (b) round 1, perplexity 100

Fig. 7: t-SNE clusters for MNIST data with DP



(a) L2 distance (b) Cosine similarity

Fig. 8: L2 distance and cosine similarity variations without vs. with DP

currentness represented in Fig. 11 was taken in seconds required to calculate explanations for 20 data points. It is important to note that currentness is highly dependent on the available CPU and memory allocations. Therefore, the results can depend on the computing instance used. One of the distinct facts we can observe here is that the sampling SHAP takes the longest computation time in the range of 35 to 50 seconds for 20 data points, while LIME takes around 3.5 seconds. Even with a higher number of perturbed samples, LIME performs much faster.

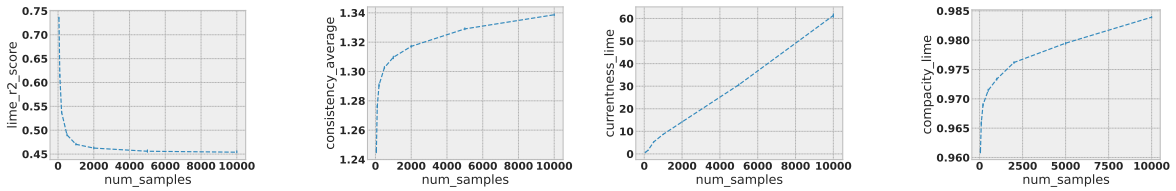
3) Compacity metric: Compacity is a measure of the proportion of the model explained through the top features given by an explanation. A higher portion of the model explained by a fixed number of features would be the desired outcome from a good explainer. This can be understood in another way, too. By keeping the portion of the model explained fixed, compacity can be understood as the number of features required to explain a model. But for a better representation of compacity, we use the first definition of keeping the number of features fixed. The average compacity shown in Fig. 12 is calculated by taking the average of similarity of the explanation by only 5 features vs the full explanation given by all the features for all the test data points. This similarity reduces for lime and kernel SHAP as the model goes through global aggregations. For sampling SHAP, the compacity varies rapidly in the beginning but seems to stabilise with iterations.

V. Conclusion

In this research, we examined the accountability and performance metrics of FL models, with a focus on aspects including DP, model complexity, and user diversity. Metrics such as compacity, consistency, and currentness were analysed using explainers like LIME, kernelSHAP, and samplingSHAP, revealing trends with increasing perturbed samples. Training iterations influenced metrics differently, with notable changes observed in consistency at iteration 4 for LIME. It was found that DP implementation affected convergence in complex neural network architectures (NN-1, NN-2, NN-3), impacting accuracy and computational time. DP also reduced user diversity, enhancing privacy but compromising model utility, as evidenced by t-SNE embeddings and distance metrics.

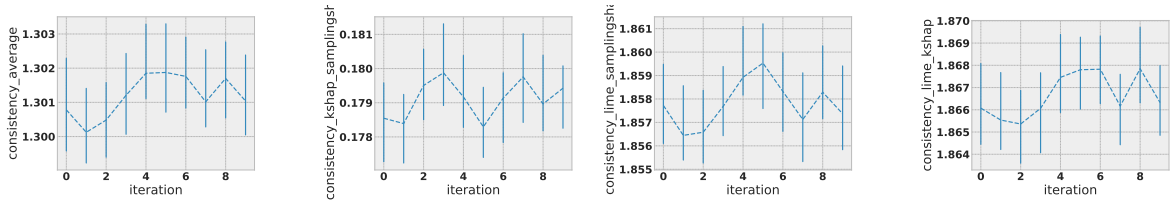
Acknowledgment

This work is partly supported by the European Union under the SPATIAL project (Grant ID. 101021808) and



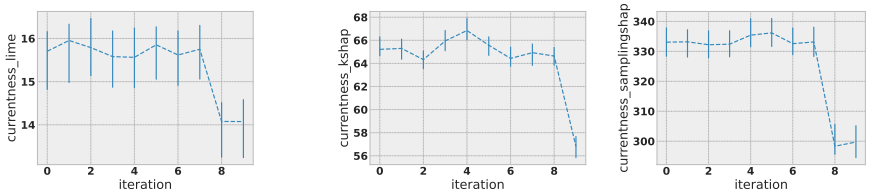
(a) LIME R2 score (b) LIME consistency (c) LIME currentness (d) LIME compacity

Fig. 9: Variation of the LIME score, consistency with kernel and sampling SHAP, currentness, and compacity with a number of samples used for fitting lime.



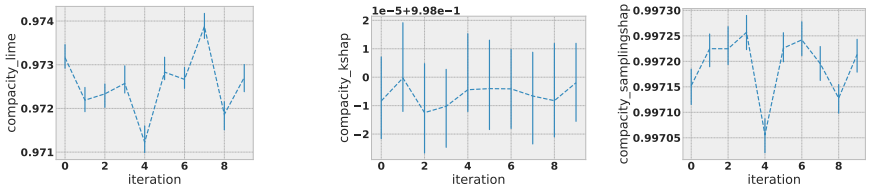
(a) All three explainers (b) Between kernelSHAP and samplingSHAP (c) Consistency between LIME and samplingSHAP (d) Consistency between LIME and kernelSHAP

Fig. 10: Consistency variation of the explainers against global model aggregation iterations.



(a) LIME (b) Kernel SHAP (c) Sampling SHAP

Fig. 11: Currentness variation of the explainers against global model aggregation iterations.



(a) LIME (b) Kernel SHAP (c) Sampling SHAP

Fig. 12: Compacity variation of the explainers against global model aggregation iterations.

by Science Foundation Ireland under CONNECT phase 2 (Grant no. 13/RC/2077_P2) projects.

REFERENCES

[1] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, 2019.

[2] N. Kourtellis, K. Katevas, and D. Perino, "Flaas: Federated learning as a service," in *Proceedings of the 1st workshop on distributed machine learning*, 2020, pp. 7–13.

[3] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "Performance analysis and optimization in privacy-preserving federated learning," *arXiv preprint arXiv:2003.00229*, 2020.

[4] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proceedings of the second workshop on distributed infrastructures for deep learning*, 2018, pp. 1–8.

[5] J. L. C. Bárcena, P. Ducange, F. Marcelloni, G. Nardini, A. Noferi *et al.*, "Enabling federated learning of explainable ai models within beyond-5g/6g networks," *Computer Communications*, vol. 210, pp. 356–375, 2023.

[6] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[7] M. Wattenberg, F. Viégas, and I. Johnson, "How to use t-sne effectively," *Distill*, vol. 1, no. 10, p. e2, 2016.

[8] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[9] E. Strumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *The Journal of Machine Learning Research*, vol. 11, pp. 1–18, 2010.

[10] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.