This is the author's accepted version of the article. The final version published by IEEE is Pablo Fernández Pérez, Iñaki Bravo, Anirudth Kamath, Claudio Fiandrino and Joerg Widmer, "CHRONOPROF: Profiling Time Series Forecasters and Classifiers in Mobile Networks with Explainable AI," *26th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Forth Worth, Texas, USA, 2025, pp. TBD, doi: TBD.

CHRONOPROF: Profiling Time Series Forecasters and Classifiers in Mobile Networks with Explainable AI

Pablo Fernández Pérez*†, Iñaki Bravo*†, Anirudh Kamath*, Claudio Fiandrino* and Joerg Widmer*
*IMDEA Networks Institute, Spain, †Universidad Carlos III de Madrid, Spain
Email: {name.surname}@imdea.org

***Abstract*—The next-generation of mobile networks will increasingly rely on Artificial Intelligence (AI)/Machine Learning (ML) for effective network automation, resource orchestration and management. This translates into performing classification and regression tasks on time series data. Unfortunately, the existing AI/ML models are inherently complex and hard to interpret, which hinders their deployment in production networks. Further, the vast majority of the existing EXplainable Artificial Intelligence (XAI) techniques are either primarily conceived for computer vision and natural language processing and thus fail to provide useful insights.**

In this paper, we take the research on XAI for time series classification and regression tasks one step further proposing CHRONOPROF, a new tool that builds on legacy XAI techniques. By creating a linearized version of the original model for different observations, CHRONOPROF provides insights about the dynamic changes in the model decision-making process across observations and is agnostic to the influence of feature magnitude, which is a key limitation of legacy explainers. Thus, CHRONOPROF highlights the real influence of model parameters on the output. Our extensive evaluation with real-world mobile traffic traces shows that CHRONOPROF is able to measure the feature importance, especially in classification tasks where linearized explanations across observations show high consistency.

I. INTRODUCTION

The fifth-generation (5G) mobile networks is now a commercial reality and is transforming the landscape of the mobile network ecosystem. The growing availability for higher and faster access to mobile services has contributed to increase the demand for mobile traffic which is growing at a staggering pace.

Fine-grained analysis of traffic patterns at the individual level of single Base Station (BS) or at the city scale is key for multiple data-driven tasks such as effective resource management, monitoring and debugging. These data-driven tasks hinge on traffic forecasting and classification. Traffic forecasting at city scale makes diverse optimizations possible, such as network deployment planning [1], and mobility management [2], network slicing [3] and resource allocation [4].

Time series analysis is a well-known research area in the AI community. Recently, Deep Neural Networks (DNN) models like DLinear [5], PatchTST [6], TSMixer [7], and MultiRocket [8] were shown to improve significantly model accuracy over well-known techniques like Long-Short Term Memory (LSTM) or AutoRegressive Integrated Moving Average (ARIMA), in a variety of datasets for both forecasting and classification. In the context of mobile networks, LSTM are still by far the most popular technique for univariate next-step time series forecasting [9], [10] and classification [11], often utilized preceded by convolutional blocks [12]. However, for forecasting, transformer-based models like PatchTST learn better long-range dependencies than LSTM. Also with respect to classification, convolutional kernel-based models like MultiRocket outperform the state-of-the art performers, based on the intuition that linear classifiers with feature transformation and selected pooling operators instead of LSTM work better for classification purposes.

The fil-rouge that interconnects the above applications of time series forecasting and classification is that the underlying AI/ML models are closed-boxes. The principles governing models like DLinear, PatchTST, TSMixer, MultiRocket, and also LSTM are opaque. The inherent lack of transparency is a major roadblock for these models to be actually used in production-grade networks because it makes tasks like debugging and troubleshooting daunting for network managers. Additionally, these models proved to be vulnerable to adversarial attacks [13] as perturbations to the original model inputs can be crafted to be imperceptible to the human eye, but sufficient to worsen the model accuracy [14]. The above cases exemplify the need for explainability for these models to be actually deployed in production networks. Auric [15] is a counter-example that underlines the pressing need for explainability. AT&T developed Auric based on Decision Trees (DTs) [16] for automatic parameter configuration of newly deployed BSs. The distinct feature of DTs is that they are inherently explainable, which allows them to gain the trust of the operator. Unfortunately, DTs are not suited for complex time series analysis tasks like forecasting and classification.

The overarching objective of EXplainable Artificial Intelligence (XAI) is precisely to shed light on the opaque behavior of complex AI/ML models like PatchTST, DLinear, MultiRocket with logical and human-understandable explanations. Unfortunately, XAI techniques have not been conceived for time series originally [17], but for computer vision and Natural Language Processing (NLP). The driving forces were the inherent characteristics of the data. Images, videos, and language are high-dimensional data that humans easily understand, unlike time series where pattern identification is less obvious. Moreover, the surge of interest in computer vision-based applications (such as medical imaging) has generated interest in synthesizing explanations for such models. Prominent XAI techniques like Local Interpretable Model-agnostic Explanations (LIME) [18],

SHapely Additive exPlanations (SHAP) [19], LayeR-wise back-Propagation (LRP) [20], DeepLIFT [21] have been adapted for time series. Further, techniques like AIChronoLens [22] were conceived specifically for time series. The fil-rouge that interconnects the above techniques is that they fail to provide explanations that go beyond single observations of the time series, and fail to capture the temporal dynamics of how models adapt to the input variation over time.

In this paper, we take the research on XAI one step further by enhancing the quality of explanations for multi-variate time series forecasting and classification for mobile networks. For this, we propose and design CHRONOPROF, a new XAI technique that addresses the main shortcomings of the state-of-the art and provides useful insights to understand the logic of AI/ML models in action. We envision that CHRONOPROF is one step further to lower the barrier for the adoption of these models in production-grade networks. Likewise many other second-generation of XAI techniques like [22], [23], [24], CHRONOPROF is built on top of the legacy XAI technique SHAP and is specifically tailored to time series. SHAP provides insights into the model operation by computing the contribution of each input feature with a *game-theory* approach. While this approach works well with categorical and numerical features usually available in several applications (e.g., age, education, and worked hours for forecasting annual income), in time series analysis, the features are array-based (i.e., instances of past traffic load to predict future traffic). In such a setting, SHAP is limited as it is unable to isolate the effect of each element in the input feature from its magnitude.

At a glance, CHRONOPROF obtains the linearized representation of the AI model at each observation. These linear representations are valuable since recent advances in the AI community prove that linear models are well-suited to capturing periodic patterns in time series [25]. With CHRONOPROF, we can track the evolution of these linear representations over time, providing insights into how the linear weights adapt to improve accuracy across long periods. This adaptation explains how models leverage non-linear operations to capture patterns that extend beyond the length of input sequences, such as the weekday/weekend differences observed in network traffic.

We perform an extensive evaluation of the strengths of CHRONOPROF with real-world mobile traffic data for both applications, *i.e.*, forecasting and classification. Specifically, we focus on relevant use cases, forecasting/classifying traffic loads in an urban area and forecasting the number of connected users to a BS. For the former, we use a measurement dataset collected in a production 4G network serving a major metropolitan region in Europe with minute-level traffic information. For the latter, we use a measurement dataset collected at production BSs with millisecond-level traffic information [26]. We demonstrate that CHRONOPROF identifies behaviors in both applications that cannot be spotted otherwise.

The key contributions ("C") and findings ("F") of our study are summarized as follows:

C1. We propose CHRONOPROF, a new explainer tailored to time series that generates explanations for both classification and regression tasks.

C2. Our extensive evaluation of CHRONOPROF with real-world datasets and several classification and forecasting models highlight that CHRONOPROF's explanations make it possible to spot model behavior that would not be possible otherwise.

F1. We show the need for time series explainers to adopt a more dynamic approach than simply focusing on determining explanations based on individual observations only, in order to prioritize the characterization of the temporal changes in feature importance.

F2. We recognize the significant value of explainability methods capable of dimensionality reduction in the context of time series analysis since models usually generate high dimensional predictions like in multivariate multihorizon forecasting tasks. Techniques that can concisely generate explanations for multiple prediction values simultaneously or distill an entire lookback window into a single value have proven particularly useful.

F3. Our research shows that CHRONOPROF, when used together with SHAP, provides valuable insights into model errors for both forecasting and classification tasks,offering a more complete understanding of model behavior and potential areas for improvement.

We plan to release the artifacts of our study publicly upon acceptance of this paper.

II. BACKGROUND AND MOTIVATION

A. Background on Time Series

A multivariate time series is a sequence of data vectors denoted as $\mathcal{X}_\mathcal{T} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_T}\}$, collected at equally spaced time points $t = \{1, 2, \ldots, T\}$. Each data vector $\mathbf{x_t} \in \mathbb{R}^\mathbf{n}$ represents an observation or measurement at time step t for the n variates.

Time Series Classification: Problem Formulation. In time series classification, the objective is to assign a label that identifies uniquely a class to time series sequences. Formally, the problem definition is as follows: Given a time series subsequence $X_t = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_L}\} \subset \mathcal{X}_\mathcal{T}$ where $\mathbf{x_i}$ is a vector of n features, and L is the length of the subsequence, the goal is to predict a class label $y \in \{1, 2, \ldots, C\}$, where C is the number of possible classes. The classification function f can be represented as: $f : X_t \rightarrow \{1, 2, \ldots, C\}$ such that $\hat{y} = f(X_t)$. A set of labeled training examples $(X_t^{(i)}, y^{(i)})_{i=1}^N$ is used to capture the behaviour of f, via minimizing an appropriate loss function, such as *Cross-Entropy* loss. Upon successful training, the model can accurately classify unseen time series sequences.

Time Series Forecasting: Problem Formulation. The objective of time series forecasting is to predict a set of future values $Y_t = \{\mathbf{y_{t+1}}, \mathbf{y_{t+2}} \ldots, \mathbf{y_{t+h}}\}$, having observed a sequence of past values or lookback $X_t = \{\mathbf{x_{t-L+1}}, \mathbf{x_{t-L+2}}, \ldots, \mathbf{x_t}\} \subset \mathcal{X}_\mathcal{T}$, where $\mathcal{X}_\mathcal{T}$ denotes the whole sequence of values of the time series of length T. Typically, in the vast majority of

multi-variate time series forecasting applications, the variables of $\mathbf{x_t}$ and $\mathbf{y_t}$ are of the same scale, unlike classification tasks where $\mathbf{y_t}$ are categorical labels. The regression function f can be represented as $f : X_t \rightarrow \{\hat{\mathbf{y}}_{t+1}, \hat{\mathbf{y}}_{t+2}, \ldots, \hat{\mathbf{y}}_{t+h}\}$, such that $\hat{Y}_t = f(X_t)$, where $\hat{Y}_t$ denotes the predictions. The forecasting model is trained by minimizing a loss function between the predictions and the real values, $\mathcal{L}(\hat{Y}, Y)$, such as the *Mean Squared Error* or the *Mean Absolute Error*.

B. Background on XAI

XAI: an Instant Primer. There has been a growing interest in promoting trustworthiness in AI over the last few years [27]. Interpretability and explainability are key to trustworthiness. The former contextualizes the model outputs in relation to its design, while the latter provides customized knowledge describing how and why a model comes to achieve a given output. *Intrinsic* or *transparent* XAI techniques foster interpretability, while *post-hoc* XAI techniques are only applicable after the training process and concern explainability [28]. CHRONOPROF positions itself as *post-hoc* technique.

XAI for Time Series. Although XAI was natively conceived and tailored for computer vision and NLP, there exists some applicability to time series [17], especially in the context of time series classification [29], [30]. The most widely-adopted XAI techniques LIME [18], SHAP [19], LRP [20], DeepLIFT [21], can be applied to time series. A wave of second-generation techniques builds on the above techniques, and are often tailored to uni-variate time series [22]. Unfortunately, [29], [30], [22] are only capable of providing explanations for single observations of the time series at a time without explaining the temporal dynamics of models in action. Moreover, they are all implicitly influenced by the magnitude of the input features. Next, we overview the first-generation or legacy XAI techniques. CHRONOPROF follows in the category of second-generation techniques as it uses SHAP underneath.

First-Generation of XAI Techniques. There exists model-agnostic and model-specific techniques. SHAP [19], and LIME [18] belong to the first category and provide explanations by perturbing the inputs of the models to determine how relevant the features are for the prediction. These techniques differ in the way they compute the relevance scores. Conversely, LRP [20] is model-specific and evaluates which neurons are relevant to a prediction given the input data, making it thus possible to the connection "which part of the input data influences the prediction the most".

C. Motivation

DNN models lack transparency in their decision-making processes. Their complex structure often obscures the underlying logic that drives their predictions. In contrast, linear models have a set of weights that directly reflect the importance of features in the prediction, as the prediction is obtained through a linear combination of the weights and features. When applying out-of-the box legacy XAI explainers to time series models, issues arise.

In this work, we go beyond the above insights and expose other limitations that are specific to SHAP. We focus on the problem of classification of applications from coarse traffic volumes. We use a dataset with samples at a hour granularity (more details on the dataset in Section IV-A), train a model for time series classification, TSMixer with a lookback window of 24 hours (1 day), and use SHAP.

Figure 1 portrays on top the SHAP values computed for a representative set of applications, and on the bottom the actual traffic for the corresponding application. We observe the following aspects from the extensive analysis:

- The actual value of the feature is implicit in the SHAP value, making it difficult to compare changes in feature importance across different time steps. For example, in regression tasks, if an element x_{t-1} of window X_t undergoes a sudden change with respect to the previous window X_{t-1}, there will also be a sudden change in the SHAP value of that element. This occurs not because of an actual change in the weight that the model assigns to that value in the lookback, but simply due to the variation in the feature value itself.
- When using the mean of SHAP values as a global explainer to assess feature importance, we observe zero values across all features when the time series is zero-scaled. This occurs due to the nature of time series forecasting and the use of sliding windows, where each value appears in every feature at some point. As a result, the mean of the SHAP values converges toward the mean of the time series, rendering SHAP values uninformative in a global context.
- There is no straightforward method to reduce the dimensionality of SHAP explanations. Each input value has a corresponding SHAP value, causing the complexity of the explanations to increase with the complexity of the input and output. For multivariate forecasting tasks, the dimensionality of SHAP values is $\mathbb{R}^{B \times C \times L \times H}$, where B, C, L, H are the Batch, Channel or Variate, Length of the sequence and Horizon dimensions. In this setting, the sheer amount of values may obscure the derived explanations.

Overall, the above insights call for a new explainer able to isolate the influence of the feature values and focus only on the decision making process of the models. This is precisely the gap that CHRONOPROF fills.

III. CHRONOPROF

In light of the motivation presented in Section II-C, this Section presents CHRONOPROF, a new explainer for time series. We first delve into its design principles (Section III-A) and then present its architectural design (Section III-B).

A. Overview and Design Principles

CHRONOPROF leverages the interpretability of linear models to explain more complex DNN models. It does so by generating a set of virtual weights, one for each feature, that

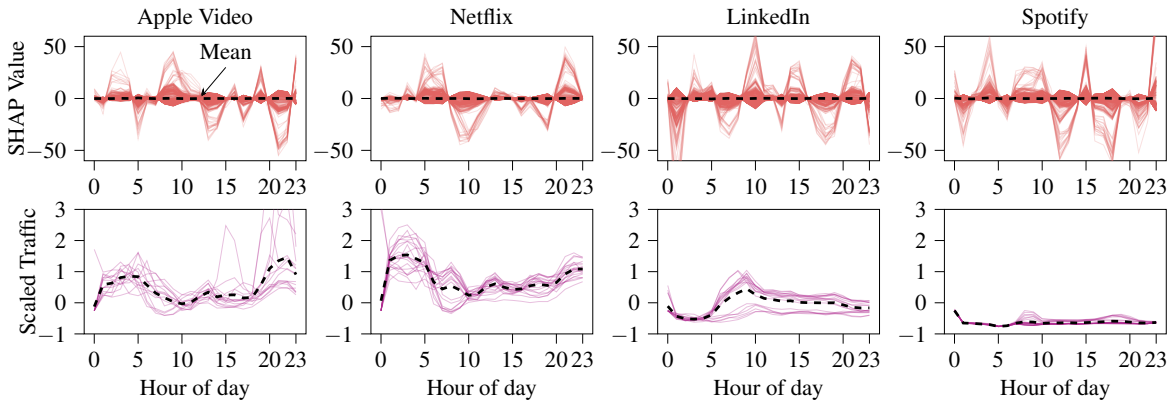


Fig. 1. SHAP values and input sequences from a network traffic classification task. The dotted black line corresponds to the mean.

are derived from the SHAP values. These virtual weights function analogously to the coefficients in a linear model, providing a straightforward interpretation of the model's behavior for a specific input. Unlike LIME's approach, which often generates problematic "fake" neighbors that are out-of-distribution and create unstable explanations (particularly for time series data with temporal dependencies) [31], the virtual weights method provides unique and nearly exact weights for each instance, with only minimal numerical error when computing SHAP values. This ensures temporal coherence alongside consecutive time steps, which is highly useful for explainability purposes. This approach generalizes the output y for a particular observation in a linear form:

$$y = \sum_{i=1}^{N} (v_i \cdot x_i) + b, \tag{1}$$

where v_i and x_i are the weight and values corresponding to feature i, N is the total number of features considered by the model and b is a bias term.

Fig. 2 outlines the high-level design of ChronoProf. In a nutshell, ChronoProf extracts from SHAP relevance scores (L_n) that defines the contribution of each element of the input sequence X_t to the output (module ❶ in the figure), then it compute the virtual weights (VW - module ❷ in the figure), and defines explanations via the analysis of the virtual weights via the profiler (module ❸).

We design ChronoProf with the following *design principles* (*DP*) in mind:

• DP_1: *XAI Specificity.* We restrict ChronoProf to utilize SHAP rather than other existing XAI techniques. The underlying reason is that SHAP indicates the quantitative and exact contribution of each feature to the output, a property that is indispensable to generate ChronoProf explanations since it maintains temporal coherence.

• DP_2: *Model Agnostic.* ChronoProf is applicable to any AI/ML model since it only requires SHAP which is also model agnostic. This property makes ChronoProf a highly generalized solution for enhancing model interpretability.

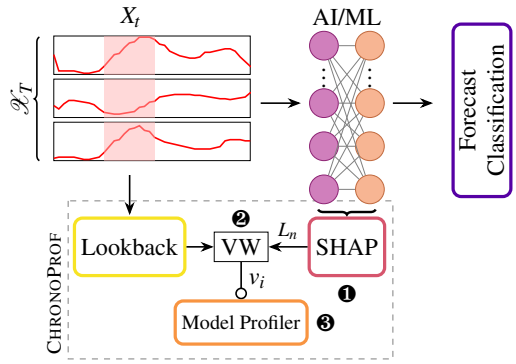


Fig. 2. ChronoProf's architecture

B. ChronoProf's Architectural Design

Relevance Scores from SHAP (❶). By taking into account that each prediction $\hat{y}$ depends on the past, i.e., on the lookback or input sequence X_t, then the legacy XAI techniques like SHAP provide relevance scores L_n to each element of the the input sequence $\mathbf{x_i} \in X_t$, $\forall i = \{1, 2, \ldots, L\}$.

SHAP is a XAI framework based on Shapley values from game theory. Shapley values measure the contribution (positive or negative) of each feature in the input to predict the output. Note that a negative contribution means a negative deviation from the expected value of the model, which is the empirical mean of the predictions over a specific set of observations. Shapley values are defined as follows: Given a set of players (features if extrapolated to ML problems) $N = \{1, 2, \ldots, n\}$ and a characteristic function (or ML model) $f : 2^N \to \mathbb{R}$, the Shapley value $L_i(f)$ for player i is defined as:

$$L(f) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} \left(v(S \cup \{i\}) - v(S) \right), \tag{2}$$

where $S \subseteq N \setminus \{i\}$ denotes all subsets of N that do not contain player i, $|S|$ is the number of players in subset S, n is the total number of players, $f(S)$ is the value of the subset S.

Computation of the Virtual Weights (❷). By definition, the sum of SHAP values for a given instance, when added to the expected value of the model, precisely equals the model's

output for that instance. This fundamental property can be expressed mathematically as:

$$f(X_t) = E[f(X)] + \sum_{i=1}^{N} L_i. \quad (3)$$

The above equation is similar to (1), offering an intuitive framework for interpretation. $E[f(X)]$, corresponds to the bias term b in the linear equation. Each SHAP value L_i in the sum is analogous to the product $v_i \cdot x_i$ in the linear model, representing the contribution of feature i to the model's output. Consequently, the virtual weights are calculated from the SHAP values and the input values of the features as $v_i = L_i/x_i$.

This very simple formulation makes it possible to isolate the implicit influence of the input values x_i and focus on the local linear approximation function that generates the predictions. The approach enables a finer separation between the contribution of the feature's magnitude from its importance, by shifting the focus from the input values to the model's sensitivity to each feature change. Note that in the particular case of a linear model, the virtual weights generated by CHRONOPROF remain constant across all observations. This is because these virtual weights are identical to the actual weights of the linear model.

Synthesis of Explanations with Profiler (❸). Analyzing the evolution of virtual weights v_i across consecutive time steps offers valuable insights into model behavior and enhances explainability. This comparative approach unveils dynamic patterns in the model's decision-making process, which is particularly beneficial in time series analysis. The analysis primarily focuses on the models and the underlying physical phenomena rather than the apparent characteristics of the time series itself. This distinction is crucial because time series with different properties – such as stationarity, periodicity, or trends – can potentially be generated by an invariable linear function. This insight highlights the power of examining virtual weights in understanding the deeper mechanics of the model and the system it represents.

• *In time series forecasting*, tracking changes in virtual weights can reveal distinct regions or states within the data. Significant shifts in these weights may indicate regime changes, seasonal transitions, or other temporal patterns that influence the model's predictions. This information can help identify critical points where the model's focus shifts from one set of features to another, providing a better understanding of how the model adapts to evolving time series dynamics. One way to measure these changes is by computing the rolling variance of a set of s samples at each time t:

$$Var(\mathbf{V_t}) = \mathbb{E}[(V_t - \mathbb{E}[V_t])^2], \quad (4)$$

$$\mathbf{V_t} = \{\mathbf{v_{t-s+1}}, \mathbf{v_{t-s+1}}, \ldots, \mathbf{v_t}\}, \forall t \subset T. \quad (5)$$

• *For time series classification tasks*, this analysis of virtual weights can lead to a simplified representation of the complex Deep Learning (DL) model. By identifying clusters in virtual weights, it becomes possible to distill the DL model's decision criteria into a more interpretable form. This simplification could potentially transform a sophisticated DL classifier into a more transparent, rule-based system or a simpler linear model that captures the key decision boundaries without sacrificing significant performance. Therefore, this approach bridges the gap between the high performance of complex DL models and the interpretability of simpler models.

IV. ANALYSIS OF CHRONOPROF'S EXPLANATIONS

A. Evaluation Settings

Datasets. For our validation, we rely on the following different datasets (divided according to the task: regression "R," or both classification and regression "CR"):

• D_1: (CR). The first dataset in our study consists of traffic volume measurements from a production 4G network serving a major European metropolitan area, recorded over three months. These measurements are distinguished by various applications, providing a rich view of network usage patterns. For the forecasting task, we utilize fine-grained data with a 10-minute granularity. The classification task, in contrast, employs data at an hourly granularity, with each sequence representing a full day of activity (24 values per sequence) and labeled for each application.

• D_2: (R). The second dataset contains the estimated number of active users currently connected to a production BS [26]. The dataset was collected with an LTE passive monitoring tool decoding the unencrypted information of the control channel. The dataset contains information at the level of 1 ms about the temporary user ID currently associated with the user, *i.e.*, the Radio Network Temporary Identifier (RNTI), and scheduling information, from which we estimate the number of active users at a 10 minute granularity.

Models. In our study, we selected a diverse set of models to ensure comprehensive coverage of current State-of-The-Art (SoTA) approaches in both forecasting and classification tasks.

• *For the forecasting task*, we use three distinct models, including a transformer-based architecture, namely PatchTST [6], a Multilayer Perceptron (MLP) based model TSMixer [7], and a third linear model DLinear [5]. This selection allows us to compare the performance and interpretability of attention-based mechanisms against traditional feed-forward networks. We also ensured the inclusion of models capable of performing both univariate (channel-independent) like DLinear and PatchTST and multi-variate operations (cross-channel) like TSMixer.

• *For the classification task*, we also use PatchTST and TSMixer. Furthermore, we include two additional models as baselines. We create a single linear layer model called "Linear". Akin to DLinear for forecasting, Linear makes it possible to quantify the performance gains or shortfalls achieved by more sophisticated architectures that use non-linearities. Then, we use MultiRocket [32]. MultiRocket provides state-of-the-art accuracy results while taking less time to train than most other time series classification models. It exploits the extraction of useful properties by applying a fixed set of convolutional transformations to the input series and the array of first

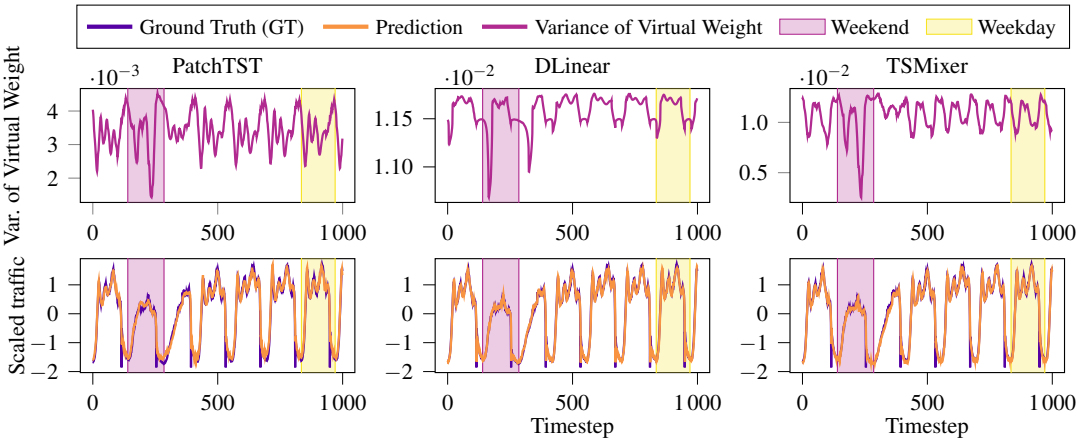


Fig. 3. Analysis of different model states using the rolling variance of the virtual weights.

order differences. For each transformation, four statistically significant features are retrieved: proportion of positive values, mean of positive values, mean of indices of positive values, and longest stretch of positive values.

B. Explanations

The analysis of virtual weights differs significantly between forecasting and classification tasks due to the inherent nature of the data and objectives in each case. In forecasting tasks, we typically work with sliding windows over a continuous time series. This approach results in overlapping data points appearing in multiple windows, albeit at different positions within each window. For instance, a data point might appear at the end of one window and gradually move towards the beginning in subsequent windows. This temporal continuity allows us to explore the model's consistency over time and focus on the evolution of virtual weights rather than their absolute values. By examining how these weights change as data points shift positions, we can gain insights into the model's adaptability and its sensitivity to the temporal structure of the data. Conversely, classification tasks usually involve distinct, non-overlapping sequences, each uniquely associated with a specific class or category. In this scenario, consecutive observations in the dataset do not necessarily share any temporal relationship. Each sequence stands alone as a representative of its class, making it less meaningful to observe changes in virtual weights between consecutive observations in the dataset. Instead, the analysis in classification tasks tends to focus on how the virtual weights distribute across different classes, helping to identify the distinguishing characteristics that the model uses to differentiate between categories. This fundamental difference in data structure and task objective necessitates distinct approaches to interpreting virtual weights in forecasting versus classification scenarios.

Explanations for Forecasting Tasks. A key advantage of virtual weights over SHAP values is their ability to generate predictions. This property allows us to compare predictions at certain times with predictions for the same data points using virtual weights from a different time step. By doing so, we can not only identify the difference in virtual weights but quantify their impact on the prediction. Figure 4 demonstrates this concept. It can be seen how PatchTST, a highly nonlinear transformer-based model is able to adapt much better than DLinear when using sequences for the same time of the day on different days. This fact is illustrated by comparing the prediction using the virtual weights of that time step and the time step from a day before. The predictions utilizing the previous day's lagged virtual weights exhibit a noticeable offset from the ground truth, whereas the predictions based on current weights align much more closely with actual values. This distinction is not observed in the DLinear model, where its linear architecture limits its ability to adapt to specific sequences. The contrast highlights PatchTST's superior capacity to fine-tune its predictions based on the most recent temporal context, a capability that the simpler linear structure of DLinear cannot match. Note how the most significant differences in virtual weights correspond closely with the points where the sequences show the greatest divergence (lookback 30 to 50).

Another useful application of our tool in forecasting tasks is observing how virtual weights change over time by applying (5). Patterns in the rolling variance of virtual weights may indicate states or periods where the time series have different statistical properties directly related to it predictability. Figure 3 illustrates this concept for dataset D_1 where there are notable differences in the patterns during the weekdays and weekends.

In forecasting, when time series exhibit periodic patterns, it is very useful to understand the periods in which the model underperforms when compared to other similar periods. Figure 5 (central portion of the figure) shows two input lookback sequences from dataset D_2. Both input sequences are from the same time of the day, on two consecutive days. We use PatchTST for the predictions. The input sequences are very similar (their Euclidean distance is 0.5). The prediction of the current day saw a higher error as compared to the following day. The nearly identical sets of virtual weights (Fig. 5-left) can provide major insight to explain this behaviour when used in conjunction with their respective SHAP values (Fig. 5-right). Using the cumulative SHAP values of the input lookback

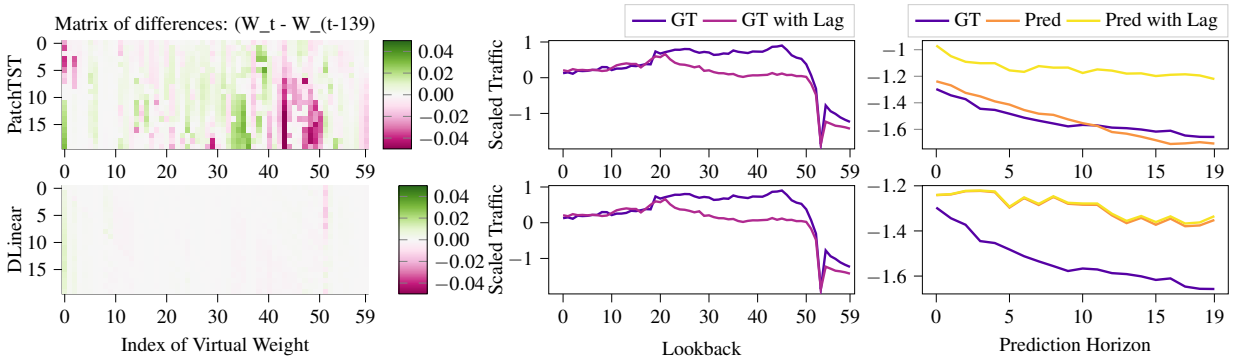


Fig. 4. Analysis of model predictions using two distinct sets of virtual weights: one derived from the current time step and another from the day before (lag of 139 time steps). The heatmaps are the differences between the virtual weights for each element in the horizon Y_t (y-axis) and each element of the lookback X_t (x-axis). The second plot are the two input sequences X_t and $X_{(t-139)}$. The third plot are the predictions $\hat{Y}_t = v_t \cdot X_t + b$ and $\hat{Y}_t = v_{(t-139)} \cdot X_t + b$.

sequence, which showcases the prediction so far seen till that point, it can be observed that the cumulative predictions start to diverge early on in the lookback window. As the SHAP values are linearly related to the virtual weights, the stable and identical linear virtual weights between the two lookback patterns show that this divergence is due to the drop in number of UEs early on the lookback window on the current day as compared to the next day. The scaled number of UEs is nearly identical for both input sequences from lookback point 25 onwards, and for most offset input sequence pairs this is where the cumulative SHAP scores would start to converge again. However, the cumulative SHAP values for the current day do not increase enough due to both input sequences reaching near-zero values for these lookback points, causing the product of the virtual weight v_i and the lookback value x_i to be a small number, which in turn does not thus not contributes enough to the prediction. This type of error is directly related to the standard scaling applied to the dataset where there are near-zero values in the lookback sequence and can be mitigated with a different scaling technique.

Explanations for Classification Tasks. In classification tasks, virtual weights can be understood as the importance given by the model to each feature to characterize a specific class. Despite the highly non-linear nature of most DL models used in our experiments, we can demonstrate through virtual weights that they exhibit a nearly linear behavior. This fact is illustrated in Fig. 6. Although there are multiple profiles for the virtual weights, it can be asserted that most of them belong to a single large cluster. This property enables users to explain the model as if it were a simpler linear model. This linearization through virtual weights enhances the interpretability of the model's decision-making process.

Fig. 6 showcases the virtual weights for PatchTST, focusing on two specific classes: Twitch and YouTube. From the graph, we can assert that this model considers hours 11, 13, 16 and 21 as the most relevant ones to differentiate Twitch from all other applications. Furthermore, from the sign of the virtual weights, we conclude that the model mostly favors sequences with high traffic at 13 and 21 and lower traffic at 11 and 16 for the prediction of this class. From a human perspective, one would expect that a recreational application such as Twitch would yield higher traffic at leisure hours and lower at working ones. This information could be leveraged to differentiate this application from all others with different use purposes. This is exactly the observed behaviour in the virtual weights analysis. When looking at how the model differentiates between YouTube and Twitch, we have to look at the features with higher difference in virtual weights between the two classes, i.e., hours 11, 13, 15, and 21. From those, we can observe that the model would strongly penalize the classification score of Twitch for sequences with high traffic at 11 and 15, while priming those sequences with high traffic at 13 and 21. Indeed, YouTube's traffic remains much more constant throughout the day due to its less recreational component vis-a-vis Twitch.

Figure 7 shows the distribution of virtual weights for two features and how they are clustered around specific values well apart one for each other, which enhances the confidence of analyzing feature importance as as a reliable source of information. This phenomenon is consistent across all tested models and it is not observed in SHAP values that are consistently overlapping. This makes it possible to explore the different decision boundaries stated by each model and understand their differences in the decision making process.

C. Use case: Understanding Model Errors in Classification

This section shows how virtual weights and SHAP values together can provide deeper insights into mis-classification errors in time series classification tasks. We focus on a case study using the transformer-based PatchTST model, examining an instance where the model incorrectly classified a Spotify traffic trace as Apple Music. This error is particularly interesting due to the potential similarities in traffic patterns between these popular music streaming platforms. Figure 8 illustrates both a correctly (Fig. 8(a)) and incorrectly (Fig. 8(b)) classified sequence for Spotify. Each subfigure contains three plots. The left one displays the sequence to be classified, alongside the mean and error bars for all Spotify and Apple Music sequences. The central one depicts the virtual weights

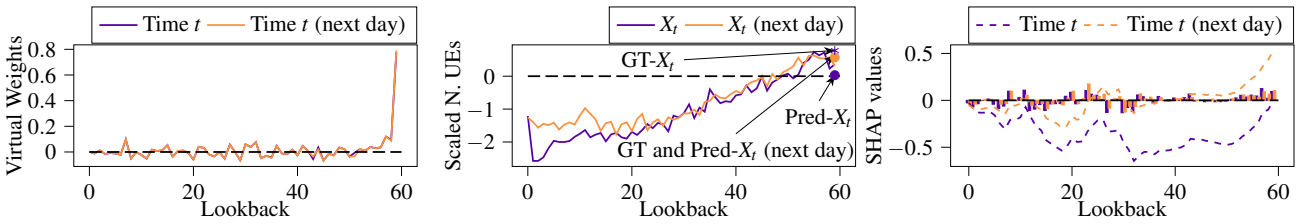


Fig. 5. Explanation of a time series forecasting error using the virtual weights and SHAP

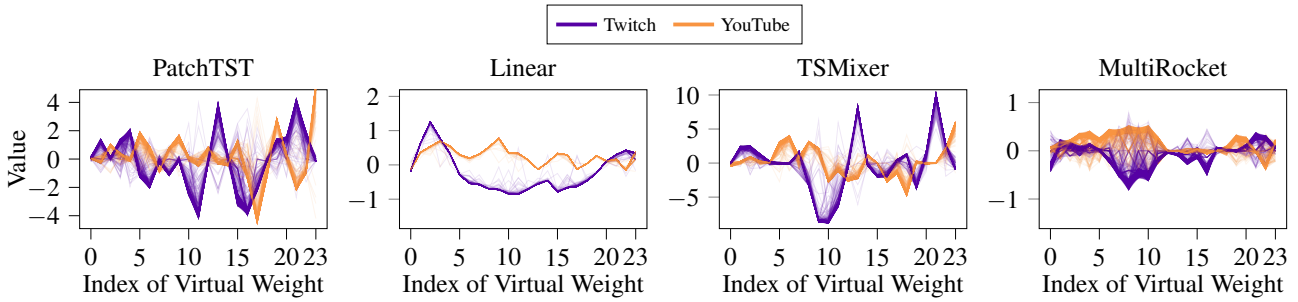


Fig. 6. Visual comparison of virtual weights of different models applied to classification

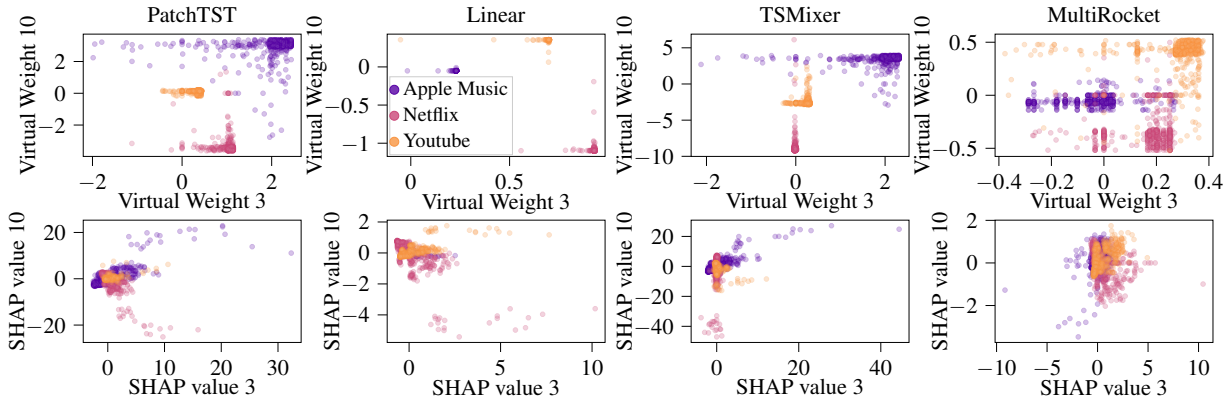
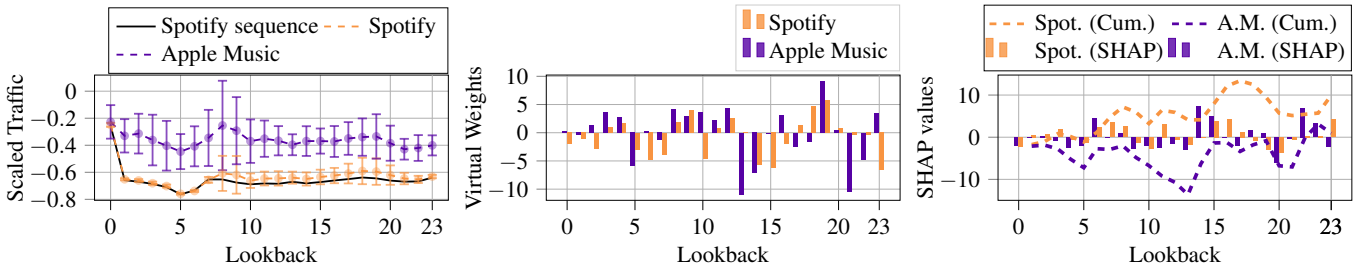
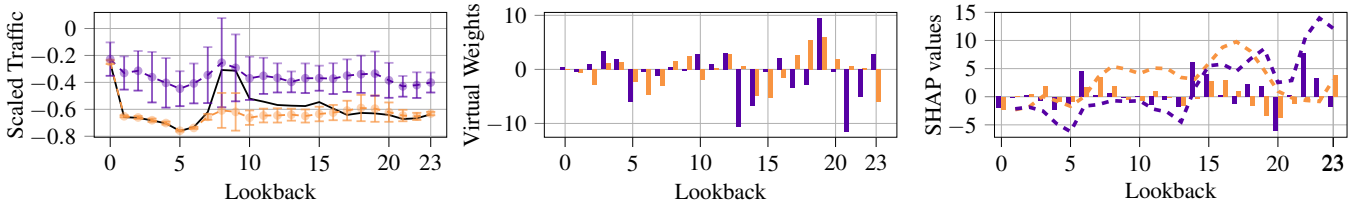


Fig. 7. Showcasing the benefit of ChronoProf for the dataset D_1 utilized for classification. The plot focuses on the contribution of two items of the lookback highlighting the differences between SHAP and the virtual weights.

assigned to each element in the window. The right one shows the SHAP values and their cumulative sum. As stated throughout the paper, virtual weights allow us to understand which features are more relevant for the prediction of a given class according to the model. It is worth noting that this use case belongs to a multi-class classification problem. Therefore, some virtual weights may have high absolute values in apparently irrelevant places. These points may be helpful to differentiate between all 23 classes and not just the two selected for the example. Here, the most significant hours of the day include 5 AM, 10 AM, 1 PM, and 9 PM as they are the ones with a bigger weight difference between the two classes. In the misclassified sequence (8b), only hours from 7 to 10 AM notably deviate from the Spotify mean. However, during these hours, the likelihood of being classified as Spotify remains similar to that of the correctly classified observation. This phenomenon is evidenced by the cumulative SHAP values, which represent the confidence of a prediction based on current and past data points. At hour 23, the final cumulative SHAP value for each class corresponds to the model's prediction score for that class. The one with the highest score becomes the predicted label for the given instance. By observing the cumulative SHAP values plots from 7 to 10 AM, we can assert that the misclassification occurs not because the deviation from the mean causes a lower classification score for Spotify, but because the score of Apple Music does not decrease sufficiently to prevent it from being the predicted class. This fact is visually evident when comparing the cumulative SHAP values at hour 1 PM between the misclassified and correctly classified cases. In the misclassified instance, the cumulative SHAP values hovers near the value -5, whereas in the correct classification, it reaches approximately -12. We can



(a) Correct classification



(b) Incorrect classification

Fig. 8. Classification use case comparing correct and incorrect classifications of a Spotify sequence (confused with Apple Music in the incorrect case)

understand this phenomenon by observing the virtual weights. At 1 PM and 9 PM, there are highly negative virtual weights for Apple Video. Since the traffic in the sequence is negative at these hours, it significantly increases the classification score for this application. Therefore, for the classification to be accurate, it is necessary to compensate for this with the other weights in the sequence. Since traffic is higher than expected during hours from 7 to 10 AM this compensation is not enough and the sequence is incorrectly classified.

Summary. The misclassification can be attributed to a complex interplay of factors. The unexpectedly high traffic values observed between 7 and 10 AM plays a crucial role. While these values do contribute negatively to the classification score for Apple Music, their impact is not substantial enough to decisively rule out this class. Subsequently, the highly negative virtual weights around 1 PM and 9 PM boost the Apple Music classification score as these negative weights are multiplied with the negative sequence values leading to a positive uptick, ultimately resulting in it surpassing Spotify's prediction score. This combination of factors— the insufficient reduction in score during morning hours, and the increase in the cumulative SHAP values due to afternoon and evening patterns—ultimately leads to the erroneous classification.

V. Related Work

XAI For Mobile Networks. Future 6G networks embrace the vision for native, explainable network intelligence. as the lack of explainability may lead to poor AI/ML model design. All the areas where AI is applied to mobile networking tasks can benefit from explainability. These include the physical and MAC layer design, and localization [33]. One of the shortcomings of the existing XAI tools is the lack of deep relation between input data and the explanations [22]. Our work goes beyond AIChronoLens because ChronoProf makes it possible to capture changes in the way the models respond over time, while AIChronoLens limits its scope of applicability to single observations at a time. Moreover AIChronoLens is designed to detect point of interest in the time series that could be related to errors or anomalies and do not measure the feature importance.

Applications of Forecasting. The recent years have witnessed a surge of interest in applying Deep Neural Networks for forecasting as they entail higher quality predictions than other approaches like statistical models [34]. The prediction of future traffic volumes forms the cornerstone of several applications that include scheduling of pilot signals for channel estimation [9], user throughput [10] and to infer Physical Resource Block (PRB) utilization [35]. While all the above works rely on simple LSTM models, the works [36], [37] are more complex ML architectures proposed with the objective of better exploiting temporal characteristics of the inputs.

Applications of Classification. Traffic classification on time series data has received comparatively less attention vis-a-vis forecasting [38]. Despite this, it has found applicability in a number of relevant use cases. Among the others, in indoor-outdoor detection using low power consumption IoT sensors to infer a user's environment [12] like streaming, video conferencing, Voice over IP (VoIP) and gaming [39].

VI. Conclusions

In this paper, we tackled the problem of explaining time series AI/ML models. We propose ChronoProf, a new explainer that builds on SHAP to generate explanations for both classification and regression tasks. Unlike legacy explainers that provide insights on individual observations, ChronoProf makes it possible to capture changes in the way the models

respond over time. The key intuition is to isolate the implicit influence of the input sequences to focus on the local linear approximation function that generates the predictions. It does so by defining virtual weights whose analysis allows to synthesize explanations. Our extensive evaluation of CHRONOPROF with real-world mobile traffic traces shows the benefits of the tool over legacy explainers like SHAP. For classification, CHRONOPROF allows to easily understand the contribution of each feature and, for forecasting, it allows to track changes in dynamic patterns over time. This capability helps identify and explain why errors occur in predictions, making it easier to understand and improve model performance.

ACKNOWLEDGMENTS

This work is partially supported by bRAIN project PID2021-128250NB-I00 funded by MCIN/ AEI /10.13039/501100011033/ and the European Union ERDF "A way of making Europe"; by Spanish Ministry of Economic Affairs and Digital Transformation, European Union NextGeneration-EU projects MAP-6G TSI-063000-2021-63, and RISC-6G TSI-063000-2021-59; P. Fernandéz received funding from "Programa Investigo" (grant 2022-C23.I01.P03.S0020-0000038) funded by EU-NextGenerationEU and SEPE/PRTR. C. Fiandrino is a Ramón y Cajal awardee (RYC2022-036375-I), funded by MCIU/AEI/10.13039/501100011033 and the ESF+.

REFERENCES

[1] P. D. Francesco *et al.*, "Assembling and using a cellular dataset for mobile network analysis and planning," *IEEE Trans. on Big Data*, vol. 4, no. 4, pp. 614–620, 2018.

[2] S. Zhao *et al.*, "Cellular network traffic prediction incorporating handover: A graph convolutional approach," in *Proc. of IEEE SECON*, 2020, pp. 1–9.

[3] D. Bega *et al.*, "DeepCog: Optimizing resource provisioning in network slicing with AI-based capacity forecasting," *IEEE JSAC*, vol. 38, no. 2, pp. 361–376, 2020.

[4] L. Chen *et al.*, "Data-driven C-RAN optimization exploiting traffic and mobility dynamics of mobile users," *IEEE Trans. on Mobile Computing*, vol. 20, no. 5, pp. 1773–1788, 2021.

[5] A. Zeng *et al.*, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[6] Y. Nie *et al.*, "A time series is worth 64 words: Long-term forecasting with transformers," in *Proc. of ICLR*, 2023.

[7] S.-A. Chen *et al.*, "TSMixer: An all-MLP architecture for time series forecasting," *arXiv preprint arXiv:2303.06053*, 2023.

[8] C. W. Tan *et al.*, "Multirocket: multiple pooling operators and transformations for fast and effective time series classification," *Data Mining and Knowledge Discovery*, vol. 36, no. 5, pp. 1623–1646, 2022.

[9] C. Fiandrino *et al.*, "Traffic-driven sounding reference signal resource allocation in (beyond) 5G networks," in *Proc. of IEEE SECON*, 2021, pp. 1–9.

[10] J. Lee *et al.*, "PERCEIVE: Deep learning-based cellular uplink prediction using real-time scheduling patterns," in *Proc. ACM MobiSys*, 2020, p. 377–390.

[11] A. Scalingi *et al.*, "A framework for wireless technology classification using crowdsensing platforms," in *Proc. of IEEE INFOCOM*, 2023, pp. 1–10.

[12] S. Bakirtzis *et al.*, "Deep-learning-based multivariate time-series classification for indoor/outdoor detection," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 24 529–24 540, 2022.

[13] W. Huang *et al.*, "Adversarial attack against LSTM-based DDoS intrusion detection system," in *Proc. of IEEE ICTAI*, 2020, pp. 686–693.

[14] D. Adesina *et al.*, "Adversarial machine learning in wireless communications using RF data: A review," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 77–100, 2023.

[15] A. Mahimkar *et al.*, "Auric: Using data-driven recommendation to automatically generate cellular configuration," in *Proc. of the ACM SIGCOMM*, 2021, p. 807–820.

[16] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nature machine intelligence*, vol. 2, no. 1, pp. 56–67, 2020.

[17] T. Rojat *et al.*, "Explainable artificial intelligence (XAI) on timeseries data: A survey," 2021.

[18] M. T. Ribeiro *et al.*, ""Why Should I Trust You?": Explaining the predictions of any classifier," in *Proc. of ACM SIGKDD*, 2016, p. 1135–1144.

[19] S. M. Lundberg *et al.*, "A unified approach to interpreting model predictions," in *Proc. of NIPS*, 2017, pp. 4768–4777.

[20] G. Montavon *et al.*, *Layer-Wise Relevance Propagation: An Overview*. Springer International Publishing, 2019, pp. 193–209.

[21] A. Shrikumar *et al.*, "Learning important features through propagating activation differences," in *Proc of ICMLR*, vol. 70, Aug 2017, pp. 3145–3153.

[22] C. Fiandrino *et al.*, "AIChronoLens: advancing explainability for time series AI forecasting in mobile networks," in *Proc. of IEEE INFOCOM*, 2024.

[23] J. Tritscher *et al.*, "Generative inpainting for Shapley-value-based anomaly explanation," in *World Conference on Explainable Artificial Intelligence*. Springer, 2024, pp. 230–243.

[24] N. O. Breuer *et al.*, "Cage: Causality-aware shapley value for global explanations," in *World Conference on Explainable Artificial Intelligence*. Springer, 2024, pp. 143–162.

[25] Z. Li *et al.*, "Revisiting long-term time series forecasting: An investigation on linear mapping," *arXiv preprint arXiv:2305.10721*, 2023.

[26] P. F. Fernández Pérez *et al.*, "Characterizing and modeling mobile networks user traffic at millisecond level," in *Proc. of ACM WiNTECH*, 2023, p. 64–71.

[27] S. E. Middleton *et al.*, "Trust, regulation, and human-in-the-loop AI: Within the european region," *Commun. ACM*, vol. 65, no. 4, p. 64–68, mar 2022.

[28] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery." *Queue*, vol. 16, no. 3, p. 31–57, jun 2018.

[29] A. Abanda *et al.*, "Ad-hoc explanation for time series classification," *Knowledge-Based Systems*, vol. 252, p. 109366, 2022.

[30] R. Mochaourab *et al.*, "Post hoc explainability for time series classification: Toward a signal processing perspective," *IEEE Signal Processing Magazine*, vol. 39, no. 4, pp. 119–129, 2022.

[31] H. Meng *et al.*, "Segal time series classification—stable explanations using a generative model and an adaptive weighting method for lime," *Neural Networks*, vol. 176, p. 106345, 2024.

[32] C. W. Tan *et al.*, "Multirocket: multiple pooling operators and transformations for fast and effective time series classification," *Data Mining and Knowledge Discovery*, vol. 36, no. 5, pp. 1623–1646, 2022.

[33] U. Challita *et al.*, "When machine learning meets wireless cellular networks: Deployment, challenges, and applications," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 12–18, 2020.

[34] S. P. Sone *et al.*, "Wireless traffic usage forecasting using real enterprise network data: Analysis and methods," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 777–797, 2020.

[35] Y. Xu *et al.*, "Wireless traffic prediction with scalable gaussian process: Framework, algorithms, and verification," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1291–1306, 2019.

[36] F. Li *et al.*, "A meta-learning based framework for cell-level mobile network traffic prediction," *IEEE Trans. on Wireless Communications*, vol. 22, no. 6, pp. 4264–4280, 2023.

[37] H. Nan *et al.*, "MSTL-GLTP: A global–local decomposition and prediction framework for wireless traffic," *IEEE Internet of Things Journal*, vol. 10, no. 6, pp. 5024–5034, 2023.

[38] S. Banerjee *et al.*, "Frequency-aware time series forecasting, anomaly detection, classification and granger causality," in *Proc. of COMSNET)*, 2022, pp. 217–221.

[39] M. Khadmaoui-Bichouna *et al.*, "5G RAN service classification using long short term memory neural network," in *Proc. of IEEE IWCMC*, 2024, pp. 467–472.