This is the author's accepted version of the article. The final version published by IEEE is Claudio Fiandrino, Eloy Pérez Gómez, Pablo Fernández Pérez, Hossein Mohammadalizadeh, Marco Fiore and Joerg Widmer, "AIChronoLens: Advancing Explainability for Time Series AI Forecasting in Mobile Networks," *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications*, Vancouver, Canada, 2024, pp. TBD, doi: TBD.

AIChronoLens: Advancing Explainability for Time Series AI Forecasting in Mobile Networks

Claudio Fiandrino, Eloy Pérez Gómez, Pablo Fernández Pérez, Hossein Mohammadalizadeh,
Marco Fiore and Joerg Widmer
IMDEA Networks Institute, Madrid, Spain
Email: {name.surname}@imdea.org

***Abstract*—Next-generation mobile networks will increasingly rely on the ability to forecast traffic patterns for resource management. Usually, this translates into forecasting diverse objectives like traffic load, bandwidth, or channel spectrum utilization, measured over time. Among the other techniques, Long-Short Term Memory (LSTM) proved very successful for this task. Unfortunately, the inherent complexity of these models makes them hard to interpret and, thus, hampers their deployment in production networks. To make the problem worsen, EXplainable Artificial Intelligence (XAI) techniques, which are primarily conceived for computer vision and natural language processing, fail to provide useful insights: they are blind to the temporal characteristics of the input and only work well with highly rich semantic data like images or text. In this paper, we take the research on XAI for time series forecasting one step further proposing AIChronoLens, a new tool that links legacy XAI explanations with the temporal properties of the input. In such a way, AIChronoLens makes it possible to dive deep into the model behavior and spot, among other aspects, the hidden cause of errors. Extensive evaluations with real-world mobile traffic traces pinpoint model behaviors that would not be possible to spot otherwise and model performance can increase by 32 %.**

I. Introduction

The advent of fifth-generation (5G) mobile networks has considerably changed 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 and is expected to reach 329 EB/month in 2028 [1]. By the end of 2023, the worldwide average monthly smartphone usage is expected to surpass 20 GB/month.

The capability to analyze and forecast mobile traffic volumes at the individual level of single Base Station (BS) or at the city scale has become key for operators to properly perform resource management. Traffic forecasting makes diverse optimizations possible, such as network deployment planning [2], routing [3], and mobility management [4], resource allocation [5] and network slicing [6], and to reduce the energy consumption footprint [7]. In the context of individual BSs, forecasting traffic volumes has found applicability in anomaly event detection [8], scalable scheduling of pilot signals to improve the quality of channel estimation [9], uplink single-user throughput [10], grant scheduling [11] or buffer status reports [12], and to infer Physical Resource Block (PRB) utilization [13]. Although several techniques have been utilized for forecasting like Deep Reinforcement Learning (DRL) [14] or simply Gaussian Processes [13], LSTM are by far the most popular technique for individual time series forecasting [8], [9], [10], [12], often outperforming other methods [15].

The logic governing LSTM is not easily understandable by humans, which creates an inherent lack of explainability of the models and hampers their use in production networks. Indeed, without a proper understanding of the logic governing LSTM models, network managers are understandably reluctant to blindly trust their output. Moreover, network engineers remain skeptical of the opaque internal operation of LSTMs that make tasks like troubleshooting daunting and that create new surfaces for adversarial attacks [16]: indeed, it has been shown how perturbations to the original input (*e.g.*, added load or jamming) can be crafted to be imperceptible to humans, but sufficient to worsen the accuracy of a predictor [17]. These examples show how LSTM explainability is mandatory if those models are to be deployed in production-grade networks. As an interesting counterexample, decision trees [18] have been used in practical scenarios by AT&T for automatic parameter configuration of newly deployed BSs [19]: as explained by the authors of that study, a key element that allowed those models to gain the trust of the operator was their inherent interpretability. Unfortunately, decision trees operate on discrete output variables and are thus very cumbersome to use for mobile traffic forecasting.

In this context, the fundamental objective of EXplainable Artificial Intelligence (XAI) is precisely to provide logical and human-understandable explanations for the black-box behavior of neural networks like LSTMs. Historically, XAI techniques have been conceived and tailored for computer vision and Natural Language Processing (NLP), and not for time series [20]. This is mainly attributed to data characteristics (high-dimensional data like images and video are more intuitive to be explained than time series for which pattern identification is more complex) and the surge of interest for computer vision-based applications (medical imagining or security built on object detection and recognition, and are very popular which has drawn the attention for embedding interpretations; in contrast, mobile traffic forecasting is not as popular). Prominent XAI techniques like LayeR-wise backPropagation (LRP) [21], SHapely Additive exPlanations (SHAP) [22], Local Interpretable Model-agnostic Explanations (LIME) [23], DeepLIFT [24] have been adapted for time series. However, as we show in Section II, they fail to provide useful explanations from a fundamental perspective that goes beyond the simple understanding of input relevance. For example, they are not capable to reveal the hidden causes of model errors that are

specific to both *(i)* the model's inner logic, and *(ii)* the currently observed input.

In this paper, we tackle precisely the problem of enhancing the quality of explanations in the context of time series forecasting for mobile networks. Our far-reaching objective is to lower the barrier for LSTM adoption in production networks. For this, we propose and design AICHRONOLENS, a new tool that addresses the main shortcomings of legacy XAI techniques and provide means to better comprehend LSTM models in action. In essence, AICHRONOLENS resolves the ambiguity of legacy XAI techniques in assigning the same relevance scores to highly diverse input sequences by exploring the Pearson correlation between relevance scores and an enriched expressiveness of the input sequence. We do so by applying an imaging technique called Gramian Angular Field (GAF) [25] that turns an input time series sequence into a 2D representation, making it possible to capture pairwise relationships like local maxima/minima within the input sequence and their spatial distance. Positive or negative correlations between relevance scores and the GAF imply that higher or lower importance is given to relevant samples like local maxima or minima. AICHRONOLENS exploits such added expressiveness to characterize the model behavior. AICHRONOLENS should be used offline for model inspection to synthesize tailored explanations on model behavior that can be next used at online inference times for monitoring purposes.

We perform an extensive evaluation of the strengths of AICHRONOLENS with real-world mobile traffic data for two relevant use cases, *i.e.*, forecasting of traffic load and 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. We demonstrate that AICHRONOLENS spots model behaviors that cannot be identified otherwise.

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

C1. We design AICHRONOLENS, a new tool that addresses the inherent shortcomings of prominent XAI tools when applied to Artificial Intelligence (AI) models for time series forecasting by harnessing liner relationship between relevance scores of XAI tools and temporal characteristics of the input sequences.

C2. We perform an extensive evaluation of AICHRONOLENS with real-world datasets and several LSTM models to demonstrate that it provides highly detailed explanations regarding model behavior that are useful at the time of verifying model robustness and monitoring.

C3. For the sake of reproducibility and to further stimulate the research in the field, we release the artifacts of our study (the trained LSTM models and the code of AICHRONOLENS) at: https://git2.networks.imdea.org/wng/aichronolens.

F1. We find that, unlike legacy XAI tools, AICHRONOLENS is capable to pinpoint differences in hyperparameter settings at training times of different models applied to the same test data. For example, higher learning rates translate into stronger correlations between the relevance scores and the time series inputs while lower learning rates exhibit weak or non-linear correlation.

F2. We find that the correlation coefficients obtained as outcomes of AICHRONOLENS show geometrical properties that can be related with the model errors. Further, we show the root causes of this issue, *i.e.*, poor model design or data inherently hard to predict.

F3. We find that AICHRONOLENS can be used to refine the training and thereby improve model performance.

II. BACKGROUND AND MOTIVATION

A. Background

Time Series Forecasting: Problem Formulation. The objective of Machine Learning (ML) models that tackle the problem of time series traffic forecasting like LSTM is to predict the future value at time $t+1$, having observed a sequence of past values. Values can be traffic volumes, number of users or PRBs measured over time. Formally, let $\mathcal{X}_{\mathcal{T}} = \{x_1, x_2, \ldots, x_T\}$ be the whole sequence of values at time $t = \{1, 2, \ldots, T\}$. Let X_t be the set of historical n past values at time t: $X_t = \{x_{t-n+1}, x_{t-n+2}, \ldots, x_t\}$. n is known as *history* or *input sequence*, with $n \ll T$. Then, the forecast $\hat{x}_{t+1}$ at time $t+1$ is:

$$\hat{x}_{t+1} = F(X_t). \tag{1}$$

F is a generic prediction function and the ML model design phase is all about defining a proper F for the problem under analysis. F is trained by evaluating at each iteration a loss function $Z_\theta(x_{t+1}, \hat{x}_{t+1})$ and updating the parameters θ (*e.g.*, the weights) to fulfill a specific objective, *e.g.*, minimizing the Mean Absolute Error (MAE) or Mean Square Error (MSE).

Primer on XAI. Promoting trustworthiness in AI has experienced a surge of interest over the last few years [26], [27], also in the context of mobile networks [28], [29]. While *interpretability* focuses on contextualizing the model outputs about its design, *explainability* goes beyond and provides customized knowledge that describes how and why a model comes to achieve a given output [30]. *Intrinsic* or *transparent* XAI techniques foster interpretability, while *post-hoc* XAI techniques apply after training and concern explainability [31]. AICHRONOLENS synthesizes *post-hoc* explanations.

XAI for Time Series. Although XAI was conceived and tailored for computer vision and NLP, there exists some applicability to time series [20], especially in the context of time series classification [32], [33]. The techniques that apply to forecasting are often tailored to multi-variate time series [34]. Many mobile network problems instead require tools for univariate time series, which we present next.

XAI Techniques. There exists model-agnostic and model-specific techniques. SHAP [22], LIME [23] and Eli5 [35] 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 [21] is model-specific. LRP provides explanations by evaluating which neurons are relevant to a prediction given the input data, making it thus possible to highlight which part of the input data influences the prediction the most.

We now provide the reader with the necessary background on the XAI techniques used in the rest of the paper:

• *LRP* assigns a score to all the inputs of a predictor and this score indicates the extent of their contribution to the predictor. The relevance scores are computed by tracking back from the output the individual activation of each neuron and its weight in subsequent layers of the model. LRP follows a conservation principle for which the total amount of relevance distributed in layer p remains unaltered in layer q. When the backpropagation reaches the input layer, the relevance score is distributed to the input sequence.

• *SHAP* provides feature-based explanations by approximating the Shapley values of a prediction. These are obtained by examining the effect of removing one feature at a time under the combination of the presence/absence of all other features. SHAP generates global and local explanations in the form of log-odds, which can be turned into a probability distribution with the `softmax` operation.

LRP appears to be more suitable for the specific case of time series prediction, as it provides high quality fidelity in the explanations vis-a-vis the SHAP and LIME counterparts [36]. Nevertheless, for the sake of completeness, we will use both LRP and SHAP.

B. Motivation

We now elaborate on the need for AICHRONOLENS by showing the limitations of LRP and SHAP techniques. For the specific problem of time series classification, a recent work [37] shows that different methods lead to different post-hoc explanations for the same model on the same dataset. In this work, we go further and show that when applied to univariate time series the *same* XAI method provides ambiguous explanations with no relation to the input sequence.

For this purpose, we train a model composed of an LSTM layer with 200 neurons followed by a fully connected output layer with a single hidden unit. The model applies to a dataset that contains traffic volumes from a production network with a 3 minute granularity where 28 541 samples are used for training and 7 121 for testing. §IV-A provides more details on the datasets and all the other models trained for validation purposes from which we derived the model currently under consideration. We apply both LRP and SHAP on the test set. Fig. 1 portrays an example of an input sequence of 20 samples, the forecast and the LRP scores. Next, we perform an extensive clustering analysis utilizing Dynamic Time Warping (DTW) Barycenter Averaging (DBA) [38] and Soft-DTW [39] k-means. For each technique, we run DBA and Soft-DTW for several cluster sizes (*i.e.*, $\kappa = [3 : 10]$) and compute the silhouette score to identify the optimal number of clusters [40]. Fig. 2 portrays an example of the obtained results for LRP where the

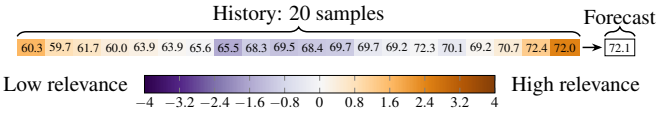


Fig. 1. Example of LRP scores for an input sequence of traffic load

optimal number of clusters is $\kappa = 4$ for both DBA and Soft-DTW. It takes nearly 16 hours to execute on an Intel® Core™ i9-11900K processor operating at 3.5 GHz and equipped with an Nvidia RTX 3090 GPU. On the top of the figure, we show the LRP scores and, on the bottom, the corresponding input sequence that produced a prediction explained by the generated scores. From Fig. 2 we observe that, for each cluster, there is no unique relationship that bonds the LRP scores with input sequences. We verify that the same behavior holds for SHAP too. The lack of such a relationship suggests that the XAI techniques are either not effectively capturing the salient characteristic of the model or that the model itself is not adequate for the job.

III. AICHRONOLENS

In light of the motivation presented in Section II-B, this Section presents AICHRONOLENS, a new technique that enhances the depth of explanations of legacy explainability tools. We first delve into its design principles (Section III-A) and next present its architectural design (Section III-B).

A. Overview and Design Principles

Fig. 3 outlines the high-level design of AICHRONOLENS. In a nutshell, AICHRONOLENS extracts through XAI techniques like SHAP or LRP relevance score (L_n) that defines the contribution of each element of the input sequence X_t to the forecast $\hat{x}_{t+1}$ (module ❶ in the figure). To resolve the ambiguity highlighted in §II-B, AICHRONOLENS uses an imaging technique, the GAF, on X_t to reveal patterns within the input sequence (module ❷). Next, AICHRONOLENS probes for linear correlation between with the Pearson's coefficient between the relevance scores L_n and the GAF representation $G_{n\times n}$ (❸ in the figure). We specifically probe for linear correlation to understand whether the model provides higher or lower importance to relevant samples in the input sequence like local maxima or minima. Finally, the "Analyzer" module monitors when this relation holds true (alignment between relevance scores and input sequences as series of correlation vectors R_n) or not and exploits transitions between the two cases as base information to synthesize more profound explanations (❹ in the figure).

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

• DP_1: *XAI Generality.* We allow for any of the existing XAI tools to be plugged into AICHRONOLENS, which makes AICHRONOLENS highly general. At the same time, this allows to compare the explanations that the different XAI tools provide when applied to the same trained LSTM model on the same dataset.

• DP_2: *LSTM Specificity.* While being general regarding to the pluggable XAI techniques, in this paper we restrain the

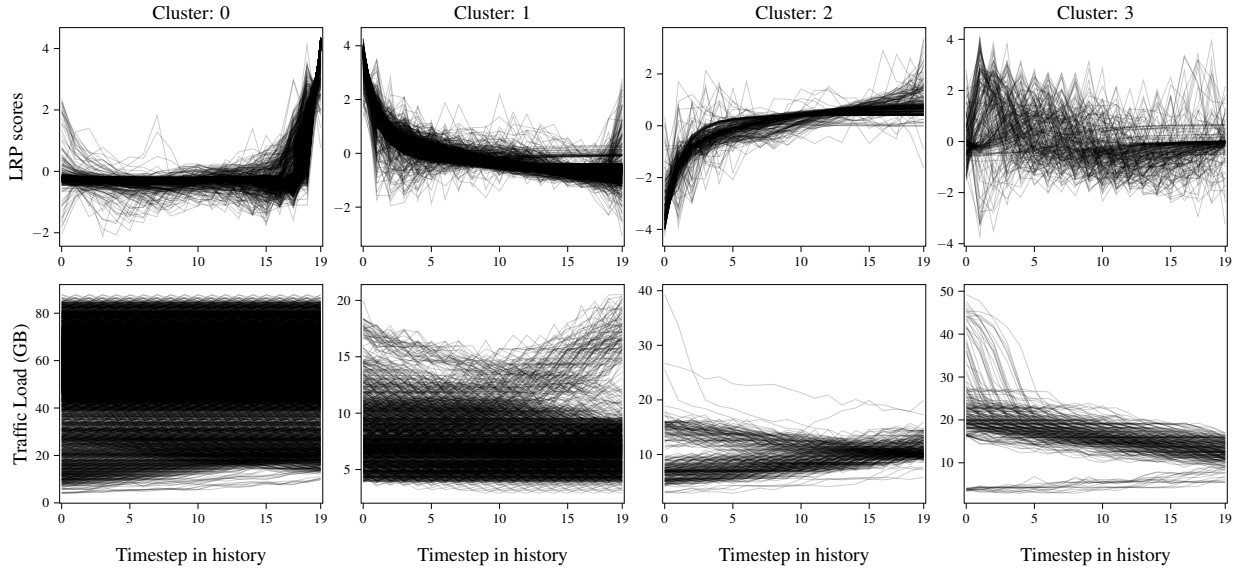


Fig. 2. Main shortcoming of prominent XAI methods: explanation profiles can originate from highly diverse input sequences

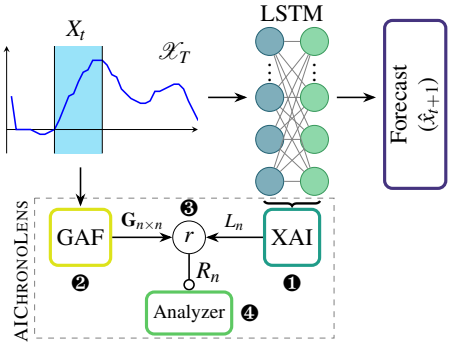


Fig. 3. AIChronoLens architecture

scope of action to LSTM models. We leave to future work the adaptation of AIChronoLens for models dealing with spatio-temporal inputs.

B. AIChronoLens *Design*

As introduced above, AIChronoLens consists of four main modules: module ❶ exploits an XAI technique for each prediction, module ❷ applies the GAF operation, module ❸ computes the Pearson's correlation coefficient on both relevance scores and GAF matrix, and module ❹ synthesizes explanations from the obtained correlation coefficients.

Relevance Scores from XAI (❶). In computer vision, XAI techniques indicate the relevance of each pixel of an image at each point in time t. In analogy, by taking into account that each prediction $\hat{x}_{t+1}$ depends on the past, or input sequence X_t, then XAI techniques provide relevance scores L_n to each element of the the input sequence $x_i \in X_t$, $\forall i = 1, 2, \ldots, n$.

According to DP_1, we now show how to obtain the relevance scores L_n by considering the two most prominent XAI techniques for each category of methods, *i.e.*, LRP (backpropagation) and SHAP (perturbation).

• LRP computes the relevance scores L_n by tracking back from the output the individual activation a_i of each neuron i and its contribution to neuron j with weight $w_{i,j}$ in subsequent layers of the Neural Networks (NN) p and q. Formally:

$$L_{i\leftarrow j}^{(q)} = L_j^{(p)} \sum_{i,j} \frac{a_i \cdot w_{i,j}}{\sum_k a_k \cdot w_{k,j}}. \tag{2}$$

Following a conservation principle for which the total amount of relevance distributed in layer p remains unaltered in layer q, when the backpropagation reaches the input layer, LRP distributes the total relevance to the input, *i.e.*, X_t in our case.

• SHAP computes relevance scores by determining the average contribution of each element of the input sequence across all possible permutations of the elements' values. To do so, SHAP relies on Shapely values. Formally $\forall i = 1, 2, \ldots, n$, $l_i \in L_n$ is computed as:

$$l_i(f) = \frac{1}{(n-1)!} \sum_{k=1}^{n-1} \sum_{\substack{X_s \subseteq X_t \\ |s|=k}} \left[\binom{n-1}{k} \right]^{-1} \cdot \left(f(X_t) - f(X_s) \right), \tag{3}$$

where $s = n - 1$ is a subset of the n features of the input sequence X_t, $f(X_s)$ is the model prediction with X_s, where $X_s = X_t \; x_i\}$, and $f(X_t)$ is the prediction with all the features, *i.e.*, $\hat{x}_{t+1}$.

Imaging via GAF (❷). Given the inherent ability of ML in dealing with images, there exists several attempts of transforming time series into images [41]. Recurrence Plots (RP), GAF, and Markov Transition Field (MTF) are popular imaging techniques for time series [25]. In a nutshell, all of them turn a time series of length m into an image of $m \times m$ pixels. The difference between these techniques lies in how they define the image. RP compute the Euclidean distance for each value $j \in m$ of the time series. RP are not capable to deal with time series of variable length and different scales, and can not effectively represent upward and downward trends [42]. GAF represents a time series using polar rather than Cartesian coordinates by constructing a Gram matrix where each element is the cosine of the sum of 2 angles.

Finally, MTF constructs a Markov matrix of quantile bins on the time series values and encodes into a quasi-Gramian matrix the dynamic transition probability of each element $j \in m$. Although the MTF technique preserves temporal dependencies like GAF, it does not allow reconstructing the original time series as GAF does. The difference lies in the fact that GAF operates on time series values directly, while MTF operates on transition probabilities of quantiles. Hence, we use GAF for AICHRONOLENS.

To obtain a GAF, the original elements of time series $x_i \in X_t$, with $i = 1, \ldots, n$, undergo a set of transformations. First, we rescale them in the range $[-1, 1]$:

$$\widetilde{x}_i = \frac{(x_i - \max(X_t)) + (x_i - \min(X_t))}{(\max(X_t) - \min(X_t))}. \tag{4}$$

Next, we represent $\widetilde{X}_n$ in polar coordinates by encoding the value as the angular cosine and the time step as the radius:

$$\begin{cases} \phi = \arccos(\widetilde{x}_i), -1 \leq \widetilde{x}_i \leq 1, \widetilde{x}_i \in \widetilde{X}; \\ r = \dfrac{i}{Y}, i \in \mathbb{N}. \end{cases} \tag{5}$$

In this equation, Y is a factor that regularizes the span of the system of polar coordinates. With time increase, the values of the time series shift between angular positions, while the radius increases at a steady rate. This method of visualizing the time series brings two important properties. First, it is bijective, as $\cos(\phi)$ is monotonic when $\phi \in [0, \pi]$ which makes it possible to recover the original time series. Second, it preserves absolute temporal relations, since, as opposed to Cartesian coordinates, the corresponding area from time step i to j does not only depends on $|i - j|$, but it also depends on the absolute values of the time series in the time steps i and j. Armed with such representation, we can define the GAF as $\mathbf{G}_{n\times n}$ for each $t \in T$:

$$\mathbf{G}_{n\times n} = \begin{bmatrix} \cos(\phi_1 + \phi_1) & \cdots & \cos(\phi_1 + \phi_n) \\ \cos(\phi_2 + \phi_1) & \cdots & \cos(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ \cos(\phi_n + \phi_1) & \cdots & \cos(\phi_n + \phi_n) \end{bmatrix}. \tag{6}$$

By defining the inner product as follows:

$$\langle v, z \rangle = v \cdot z - \sqrt{1 - v^2} \cdot \sqrt{1 - z^2}. \tag{7}$$

$\mathbf{G}_{n\times n}$, that is a Gram matrix [43], can be rewritten as:

$$\mathbf{G}_{n\times n} = \begin{bmatrix} \langle \widetilde{x}_1, \widetilde{x}_1 \rangle & \cdots & \langle \widetilde{x}_1, \widetilde{x}_n \rangle \\ \langle \widetilde{x}_2, \widetilde{x}_1 \rangle & \cdots & \langle \widetilde{x}_2, \widetilde{x}_n \rangle \\ \vdots & \ddots & \vdots \\ \langle \widetilde{x}_n, \widetilde{x}_1 \rangle & \cdots & \langle \widetilde{x}_n, \widetilde{x}_n \rangle \end{bmatrix}. \tag{8}$$

A Gram matrix [43] of a set of vectors $\boldsymbol{v_1}, ..., \boldsymbol{v_n}$ in an inner product space is the Hermitian matrix of the inner product (a matrix $\boldsymbol{B}$ is Hermitian if and only if each element fulfills $b_{ij} = \overline{b_{ji}}$, that is, the matrix is equal to its own conjugate transpose). The GAF representation provides several features. First, it preserves temporal dependency, because the time increases as we move from top left to bottom right. It contains temporal correlations, since $\mathbf{G}_{(i,j||i-j|=t)}$ corresponds to the relative correlations of the directions that lie in the time step t. The main diagonal of $\mathbf{G}_{n\times n}$ is a special case containing the original time series values. In a GAF, high values (close to 1) are those where local maxima or minima in the original time series correlates either with themselves or other maxima or minima respectively. The values close to 0 are the result of a correlation between local maxima or minima with points of intermediate values in the original time series. Finally, negative values (close to -1) originate from the correlation between a point with local maxima or minima with another point in the original time series with a local minima or maxima respectively.

Defining Correlations (❸). Armed with relevance scores L_n and $\mathbf{G}_{n\times n}$, we seek correlation between these two quantities. In essence, we aim to understand if there is a linear relation between the relevance scores (*i.e.*, L_n) and elements of the input time series (*i.e.*, $\mathbf{G}_{n\times n}$). Specifically, by construction, each row of $\mathbf{G}_{n\times n}$ characterizes inner relationships between samples of the input time series. Denote the ith row of this matrix as G_i, a $1 \times n$ vector defined in (8). Given that also L_n is a $1 \times n$ vector, we can compute the Pearson's correlation coefficient between these two quantities. Specifically:

$$R_n = \frac{\text{cov}(G, L)}{\sigma_G \sigma_L} = \begin{bmatrix} \rho_0 \\ \rho_1 \\ \vdots \\ \rho_n \end{bmatrix}. \tag{9}$$

where $\sigma_\cdot$ is the standard deviation and $\text{cov}(\cdot, \cdot)$ is the covariance. To simplify the notation, call ρ_i the correlation vector between G_i and L_n. By repeating the process for each timestep $t = 1, \ldots, T$, we obtain a correlation matrix $\mathbf{C}$ with dimensions $n \times T$ where R_n is only one column:

$$\mathbf{C} = \begin{bmatrix} \rho_{1,1} & \rho_{1,2} & \cdots & \rho_{1,T} \\ \rho_{2,1} & \rho_{2,2} & \cdots & \rho_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n,1} & \rho_{n,2} & \cdots & \rho_{n,T} \end{bmatrix}_{n\times T}. \tag{10}$$

Analyzing Correlations (❹). At the heart of AICHRONOLENS, the "Analyzer" module exploits $\mathbf{C}$ to synthesize explanations. For the sake of illustration, let us consider the case of observing $W = 3$ timesteps (*i.e.*, t, $t+1$, and $t+2$) of correlation vectors with a history of $n = 6$ samples. With the increase in time, a correlation coefficient ages and vanishes once the presence in the history of the sample of the time series that contributed to its generation is over. To observe the evolution of the Pearson's coefficients of each sample over time, we create a new matrix $\mathbf{S}$ by storing all the secondary diagonals of length T in rows:

$$\mathbf{C}_{6\times 3} = \begin{bmatrix} \rho_{1,1} & \rho_{1,2} & \rho_{1,3} \\ \rho_{2,1} & \rho_{2,2} & \rho_{2,3} \\ \rho_{3,1} & \rho_{3,2} & \rho_{3,3} \\ \rho_{4,1} & \rho_{4,2} & \rho_{4,3} \\ \rho_{5,1} & \rho_{5,2} & \rho_{5,3} \\ \rho_{6,1} & \rho_{6,2} & \rho_{6,3} \end{bmatrix}, \mathbf{S}_{4\times 3} = \begin{bmatrix} \rho_{3,1} & \rho_{2,2} & \rho_{1,3} \\ \rho_{4,1} & \rho_{3,2} & \rho_{2,3} \\ \rho_{5,1} & \rho_{4,2} & \rho_{3,3} \\ \rho_{6,1} & \rho_{5,2} & \rho_{4,3} \end{bmatrix}. \tag{11}$$

$t \quad t+1 \quad t+2$

For practical use, $\mathbf{S}^T$ is more convenient than $\mathbf{S}$.

All the explanations of AICHRONOLENS rely on the analysis of $\mathbf{C}$ or $\mathbf{S}^T$ over windows $w \leq T$. In w, depending on their value, either positive or negative, the correlation values generate

TABLE I
CONFIGURATION OF THE MODELS TRAINED FOR D_1

MODEL ID	NEURONS	LEARNING RATE	MAE
A	200	0.0001	0.96
B	100	0.0001	0.99
C	50	0.0001	1.09
A_A	200	0.001	0.67
B_B	100	0.001	0.68
C_C	50	0.001	0.95

triangle shapes. For instance, from left to right, bottom to top: $\rho_{6,1}$, $\rho_{6,2}$, $\rho_{6,3}$, $\rho_{5,2}$, $\rho_{5,3}$, and $\rho_{4,3}$ would form a triangle of negative correlation if all the coefficients are in the range $[-1, 0]$. A triangle represents the trend of the prediction given the time series input. We are interested in how these triangles transition one to another: smooth and non-smooth transitions indicate respectively that the model catches well changes or not the trend. Errors usually occurs in the presence of non-smooth transitions.

To summarize, AICHRONOLENS links relevance scores with temporal characteristics of the input sequences in a unique manner. The output of the tool are correlation coefficients that, if observed over time, generate patterns (series of triangles of positive or negative values) that can be geometrically interpreted and spot different causes of errors. In §IV we will show two techniques for pattern recognition that uniquely identifies different causes of errors.

IV. DISTILLING EXPLANATIONS WITH AICHRONOLENS

In this Section, we first describe the datasets and models used to validate AICHRONOLENS (§IV-A). Next, we empirically evaluate AICHRONOLENS's explanations, showing the cause of model errors and to optimize model performance (§IV-B).

A. Dataset and Models

Datasets. For our validation, we rely on two different datasets:

• D_1: The first dataset contains measurements of traffic volumes recorded in a production 4G network that serves a large metropolitan region in Europe. The dataset provides fine-grained information at 3 minute granularity about the traffic volumes at each BS. The dataset covers 3 months.

• D_2: The second dataset contains the estimated number of active users currently connected to a production BS [44]. The dataset has been recorded using a popular LTE passive monitoring tool that decodes unencrypted information that the BSs exchange with the associated users. The dataset contains information at millisecond level about the temporary user ID currently associated with the user, *i.e.*, the Radio Network Temporary Identifier (RNTI), and scheduling information. We use the methodology proposed in [45] to estimate the number of active users every 6 minutes.

LSTM Models. Our objective is to highlight the benefits of AICHRONOLENS under different perspectives. Hence, we train different LSTM models for the two datasets. The two models share the same LSTM architectural design, with one unidirectional LSTM layer followed by an output layer configured with one neuron and a linear activation function for one-step prediction. Both use a sequence of past 20 samples to predict the next one, and are trained with Adam optimizer using MAE as loss function. Specifically, for D_1, we train 6 different models by intentionally varying the number of neurons in the LSTM and the learning rate; we also introduce regularization by randomly discarding some neurons at each iteration using a dropout layer before the output. Table I summarizes the details of the D_1 models. They allow analyzing how AICHRONOLENS captures variations in the hyperparameters. In contrast, for D_2, we train a single optimized model based on extensive prior testing. The LSTM layer features 25 neurons with a `tanh` activation function. Finally, we use standard 80:20 train-test split ratios.

B. Explanations and Optimizing Model Development

In this subsection, we showcase the breadth of the explanations generated by AICHRONOLENS, across all the models trained for D_1 and D_2. Our main results are explanations that go beyond the simple attribution of relevance scores to the input sequence. In summary, our results, derived from the quantitative analysis performed on the test set of both datasets are the following:

• R_1: In cases where LRP and SHAP produce relevance scores that are very similar over time and thus not informative, the correlation vectors that are the output of AICHRONOLENS clearly pinpoint the temporal characteristics that stimulate the model. These are samples entering or leaving in the input sequence whose values are either local maxima and local minima or very close to the local maxima or minima. The absence of such samples entering or leaving the input sequence turns strong positive or negative correlation values into weak correlation values. The observed behavior holds true in general (*i.e.*, in the analyzed test sets of both datasets).

• R_2: AICHRONOLENS can spot errors that are due to poor model design and errors that are specific to data inherently hard to predict. The different types of errors can be spotted by analyzing the shape of the correlation matrix $\mathbf{C}$.

• R_3: Higher learning rates (*i.e.*, models A_A, B_B, and C_C) produce weaker correlations with respect to models featuring smaller learning rates (*i.e.*, A, B, C). This behavior occurs regardless of the number of neurons.

Finding R_1. Fig. 4 demonstrates qualitatively R_1. In the figure, we show in a combined fashion from top to bottom the input sequence (*e.g.*, timesteps), the output of the XAI techniques (SHAP in this case), the GAF of the corresponding input sequence, and, finally, the correlation vectors. While the SHAP scores are highly similar, the correlation vectors vary considerably. Specifically, we can appreciate in window 20 (used to predict timestep 21) almost no correlation at all. This is due to the fact that SHAP assigns high relevance to samples in the input sequence closest in time to the next prediction (see history $15 - 19$ in the bottom) while these samples are not particularly relevant from the input sequence perspective (the GAF highlights the corresponding values with dark colors). In contrast, in timestep 22 a new local minimum enters: the alignment between SHAP and GAF triggers a

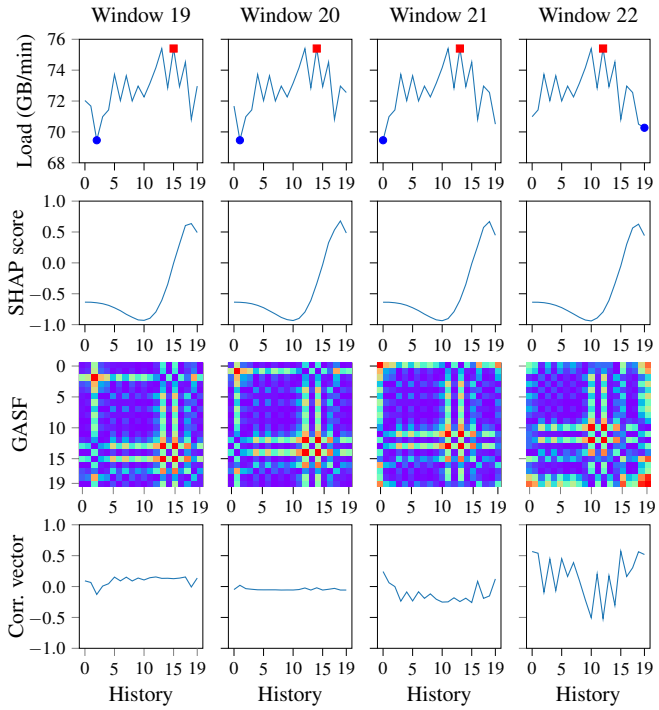


Fig. 4. A detailed look at AICHRONOLENS. Red squares and blue dots represent the local maxima and local minima respectively.

significant modification to the correlation vector if compared with the correlation vector of the previous timesteps. In contrast, if left alone, SHAP would not capture such a change. We will see next why being blind to such changes is detrimental to model performance.

Finding R_2. We prove quantitatively that AICHRONOLENS can detect different categories "E" of errors:

- E_1: is attributed to poor model design (shown for D_1),
- E_2: is specific to the dataset when using an optimized model (shown for D_2).

Analysis of E_1. Next, we will show that by tracing the root cause of the errors, it is possible to identify weaknesses due to poor model design that are not captured by coarse evaluation metrics like MAE or MSE. Finally, we will show that an informed model re-design can address such a shortcoming.

We perform a complete analysis over the AICHRONOLENS output $\mathbf{C}$ computed on the test set for all the trained models with both SHAP and LRP techniques. We observe that in the presence of trend changes in the time series, correlation vectors exhibit triangles with negative correlation followed by triangles with positive correlation. We find that the shape of the triangles varies. Well-formed, sharply outlined triangles like those in Fig. 5(a) (top) indicate that in the corresponding part of the time series, the model does not make significant mistakes (see Fig. 5(a) (bottom)). We define these triangles as *sharp*. In contrast, noisy *non-sharp* triangles like that of Fig. 5(b) (top) lead to high errors (see Fig. 5(b) (bottom)) in the presence of abrupt falls where the model is not able to accurately predict when the decrease stops in the actual data. This behavior is systematically observed throughout all the decreasing slopes present in the test set.

We now show a pattern recognition technique that identifies sharp and non-sharp triangles. Numerically, we identify the transition between triangles by computing the difference of the median correlation scores between two consecutive correlation vectors G_t and G_{t+1} in timesteps t and $t+1$. Upon finding the column that interrupts the triangle, we set a window w of observation centered in such column. For example, $W = 6$ indicates that we observe 3 preceding and succeeding columns forming a matrix $C_{n\times w}$. To spot sharpness, we perform the following operation on each element $c_{i,j}$ the matrix:

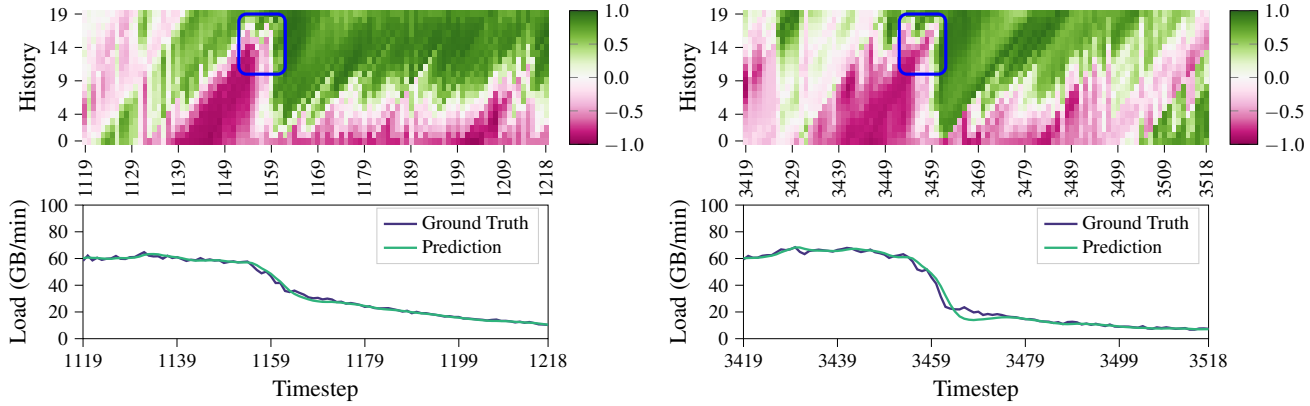
$$\overline{c}_{i,j} = \begin{cases} -1 & \text{if } -0.9 \leq c_{i,j} \leq 0 \\ 1 & \text{if } 0 \leq c_{i,j} \leq 0.9. \end{cases} \tag{12}$$

On the resulting $\overline{C}_{n\times w}$, we compute the number h of positive and negative values per each secondary diagonal of length w and store it into an array. By construction values of h are in the range $[-w : w]$. Next, we compute a sharpness score σ on the resulting array as follows:

$$\sigma = 1 - \frac{\sum_{i=1}^{n-(w-1)} h_i}{|h_i| \cdot (w+1)}. \tag{13}$$

For $0 < \sigma < 1$, the higher the value of σ, the higher the degree of non-sharpness. For $-1 < \sigma < 0$, the lower the value of σ, the higher the degree of sharpness. By relating the sharpness score and the error only in the presence of sever load decrease, we observe that as the sharpness score increases, the error does too (see Fig. 6(a)). By taking a close look to the errors in the entire test set (see Fig. 6(b)), we observe that the highest absolute errors (5-8 GB/min) occur in the correspondence of moderate to low loads that are all connected with abrupt falls. Fig. 6(c) reveals that the model underestimates significantly the ground truth in the presence of severe load decrease (bottom-left in the plot). A careful analysis of the train set reveals a lack of training samples exactly in the proximity of traffic volumes for which the model is often mistaken (see Fig. 6(d)).

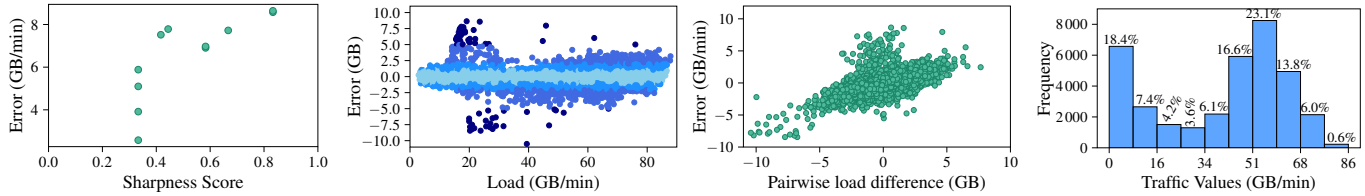
Ultimately, AICHRONOLENS allows appreciating that the model does not generalize in the presence of abrupt load decrease, as it has not observed such trends in the training set. In this way, our tool points to the solution to the model shortcoming, *i.e.*, data augmentation. Specifically, we copy from the train set a number of samples that represent 3 days (overall the train set was about 8 weeks) and append it to the end of the train set. Next, in the presence of falls, we carefully remove samples with the objective of including those abrupt load decrease that were missing. We then train a new model, starting from model A_A settings, with the augmented training dataset. The new model differentiates from A_A only by the presence of a sigmoid activation function before the output layer. Fig. 7 outlines that the new optimized model outperforms the baseline model A_A by reducing not only the tails (errors of high magnitude - errors are computed as $x_{t+1} - \hat{x}_{t+1}$ -, which is especially clear on the right inside of the plot for underestimation errors), but also the frequency of errors with small magnitude and that the error bell is centered around zero. These are all indicators of the poor training process of model A_A vis-a-vis the optimized counterpart. Overall, by only



(a) Sharp triangle. Top matrix $\mathbf{C}$ and bottom model errors.

(b) Non sharp triangle. Top matrix $\mathbf{C}$ and bottom model errors.

Fig. 5. Relating triangle sharpness of AICHRONOLENS output $\mathbf{C}$ with model errors



(a) Linking errors and sharpness score

(b) Errors as a function of load

(c) Errors as a function of load changes

(d) Train set

Fig. 6. Root cause analysis of model errors

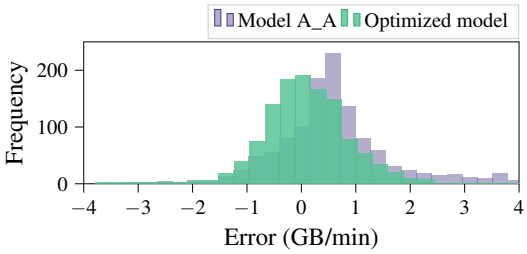


Fig. 7. Error of baseline and optimized models after AICHRONOLENS diagnosis

considering windows around the abrupt load decrease, model A_A would lead to a MAE $= 0.921$ (which is higher than the overall MAE computed over the entire test set, see Table I) while the optimized model would lead to MAE $= 0.619$ which is an improvement of 32 %. When applied to the entire test set, the optimized model achieved MAE $= 0.69$, only a 2 % decrease in accuracy with respect to model A_A.

Analysis of E_2. Unfortunately, even after addressing the category of errors E_1, there may be model errors due to the characteristics of the data itself. We now show that AICHRONOLENS can identify those too by analyzing the output $\mathbf{S}$ for D_2.

Fig. 8 shows qualitatively that there exists consecutive errors with high magnitude that change of sign (*e.g.*, first positive then negative or viceversa). AICHRONOLENS identifies this behavior with triangles of positive or negative correlations over time that are interrupted by a full column with weak correlation. Call G_t and G_{t+1} the correlation vectors in the timesteps t and $t+1$: their similarity can be assessed via the euclidean distance $d(G_t, G_{t+1}) = \sqrt{\sum_{i=1}^{n}(G_t^i - G_{t+1}^i)^2}$. We compute the euclidean distance d between each two subsequent correlation vectors in the test set and normalize it in the range $[0:1]$. When $d > 0.6$, in 65 % of the cases, we find a change of

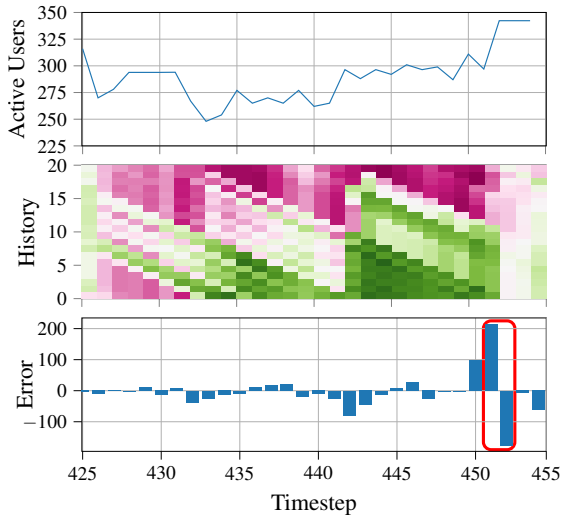
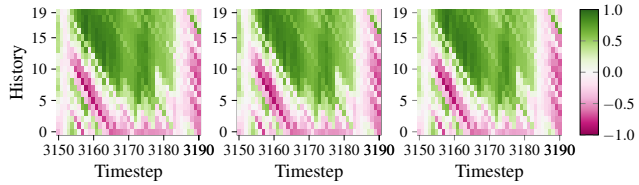


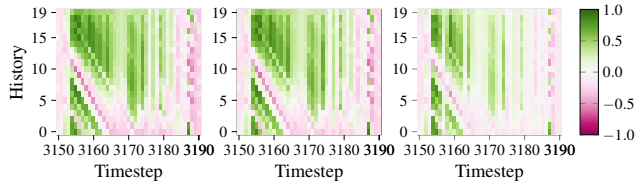
Fig. 8. Analysis consecutive errors with high magnitude that change of sign

error sign with a corresponding MAE $= 0.46$, much higher than the MAE computed over the entire dataset (*i.e.*, MAE $= 0.13$).

Finding R_3. To demonstrate qualitatively R_3, we portray in Fig. 9 examples of the correlation vectors in a window $W = 40$ timesteps in the test set for all the models in Table I. Here, the models with the lowest learning rate (on top) tend to exhibit a strong positive or negative correlation, with values approaching either 1 or -1. In contrast, the correlation scores tend to cluster around zero for models with higher learning rates, which indicates a weaker or negligible correlation. These behaviors are consistently observed across the test set and can be linked with the fact that a higher learning rate trades a higher training cost with a model that is more sensitive to changes and more rapid to adapt to new or unseen conditions. We thus conclude that AICHRONOLENS offers precise insights into



(a) From left to right: model A, B, C



(b) From left to right: model A_A, B_B, C_C

Fig. 9. Correlation vector for models with different learning rates: top 0.0001 and bottom 0.001

the heterogeneous accuracy of models trained with different learning rates. Orthogonal to the discussion on the learning rate, by analyzing Fig. 9 we also observe that the depth of the LSTM architecture, *i.e.*, the number of neurons, plays a marginal role for this specific dataset.

V. Related Work

Relevant to our work are studies on XAI techniques like visualization tools, and XAI, and LSTM-based forecasting applied to mobile networks. Despite promising, the application of imaging techniques to time series like GAF has found limited applicability in forecasting signal quality indicators [46].

XAI Visualization Tools. Visualization tools usually build on top of the legacy XAI techniques and make it possible to identify which part of the input was responsible for the output of the prediction and track the associated hidden state changes. TSViz [47] provides a 3D visualization tool for convolutional deep learning models. Long-Short Term Memories (LSTM)-Vis [48] and Sequence to Sequence (Seq2Seq)-Vis [49] are visualization tools that apply respectively to LSTM and Seq2Seq models. Both tools are tailored to NLP applications. Our work departs from the class of visualization tools because AIChronoLens provides a way to quantify the hidden relationships between explanations and the input.

XAI For Mobile Networks. Future 6G networks embrace the vision for native, explainable network intelligence. A seminal work [50] motivates the need for XAI and stress that the lack of explainability may lead to poor AI/ML model design. This has been proved detrimental in the presence of adversarial attacks [51]. All the areas where AI is applied to mobile networking tasks can benefit from explainability. These include the physical and MAC layer design, network security mobility management, and localization [52]. One of the shortcomings of the existing XAI tools is the lack of deep relation between input data and the explanations [53]. While the foundations of AIChronoLens lie in harnessing such relationship, our work goes beyond [53] as *(i)* we formally show that it exists an ambiguity as legacy XAI techniques may assign the same relevance scores to diverse input sequences, and *(ii)* we resolve such ambiguity and exploit the richer expressiveness of the outputs of AIChronoLens to better comprehend the LSTM operations and optimize models performance.

LSTM-based Forecasting Applications. 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 [54]. The prediction of future traffic volumes forms the cornerstone of several applications that include anomaly event detection [8], scheduling of pilot signals for channel estimation [9], user throughput [10], buffer status reports [12], and to infer PRB utilization [13]. While all the above works rely on simple LSTM models, the works [55], [56], [57] are more complex ML architectures proposed with the unifying theme of better exploiting temporal characteristics of the inputs.

VI. Conclusions

In this paper, we have investigated the timely and challenging problem of improving the understanding of AI models like LSTM for time series forecasting. We perform a quantitative and qualitative study that reveals the shortcoming of existing XAI techniques and propose AIChronoLens, a first-of-its-kind tool in the area of XAI. By linking the temporal characteristics of the input with relevance scores produced by existing XAI techniques, AIChronoLens can dive deep into the analysis of models' behavior. Via extensive evaluations with real-world mobile traffic traces, we show that AIChronoLens makes it possible to spot different categories of model errors, trace back the root causes, and possibly improve the poor model design. To this end, a combined targeted data augmentation, and minor changes to the hyperparameters can improve performance by 32 %.

The authors have provided public access to their code and/or data at: https://git2.networks.imdea.org/wng/aichronolens.

Acknowledgments

This work is partially supported by the Spanish Ministry of Science and Innovation through the Juan de la Cierva grant IJC2019-039885-I and PID2021-128250NB-I00 ("bRAIN"). P. Fernández received funding from the EU-NextGenerationEU and SEPE/PRTR called "Programa Investigo" (grant 2022-C23.I01.P03.S0020-0000038). The work of J. Widmer and M. Fiore are respectively supported by the European Union-NextGenerationEU through the UNICO 5G I+D projects TSI-063000-2021-63 ("MAP-6G"), TSI-063000-2021-59 ("RISC-6G"), and TSI-063000-2021-52 ("AEON-ZERO").

References

[1] Ericsson, "Mobility Report, June 2023. Technical Report." 2023, Accessed on 14/07/2023: https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/june-2023.

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

[3] C. Fiandrino, C. Zhang *et al.*, "A machine learning-based framework for optimizing the operation of future networks," *IEEE Communications Magazine*, vol. 58, no. 6, 2020.

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

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

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

[7] J. Lin, Y. Chen *et al.*, "A data-driven base station sleeping strategy based on traffic prediction," *IEEE Transactions on Network Science and Engineering*, pp. 1–1, 2021.

[8] H. D. Trinh, L. Giupponi *et al.*, "Urban anomaly detection by processing mobile traffic traces with LSTM neural networks," in *Proc. IEEE SECON*, 06 2019, pp. 1–8.

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

[11] D. Overbeck, N. A. Wagner *et al.*, "Proactive resource management for predictive 5G uplink slicing," in *Proc. of IEEE GLOBECOM*, 2022, pp. 1000–1005.

[12] Q. Zhang, A. Nikou *et al.*, "Predicting buffer status report (BSR) for 6G scheduling using machine learning models," in *Proc. of IEEE WCNC*, 2022, pp. 632–637.

[13] Y. Xu, F. Yin *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.

[14] S. Chinchali, P. Hu *et al.*, "Cellular network traffic scheduling with deep reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[15] H. D. Trinh, L. Giupponi *et al.*, "Mobile traffic prediction from raw data using LSTM networks," in *Proc. IEEE PIMRC*, Sep. 2018, pp. 1827–1832.

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

[17] D. Adesina, C.-C. Hsieh *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.

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

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

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

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

[22] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. of NIPS*, 2017, pp. 4768–4777.

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

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

[25] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Workshops at AAAI Conf. on Artificial Intelligence*, January 2015.

[26] D. Gunning and D. Aha, "DARPA's explainable artificial intelligence (XAI) program," *AI magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[27] S. E. Middleton, E. Letouzé *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] S. Wang, M. A. Qureshi *et al.*, "Explainable AI for B5G/6G: Technical aspects, use cases, and research challenges," 2021.

[29] C. Fiandrino, L. Bonati *et al.*, "EXPLORA: AI/ML explainability for the Open RAN," *Proc. ACM Netw.*, vol. 1, no. CoNEXT3, Nov 2023.

[30] D. A. Broniatowski *et al.*, "Psychological foundations of explainability and interpretability in artificial intelligence," *NIST, Tech. Rep*, 2021.

[31] 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.

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

[33] R. Mochaourab, A. Venkitaraman *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.

[34] W. Ge, J.-W. Huh *et al.*, "An interpretable icu mortality prediction model based on logistic regression and recurrent neural networks with lstm units." in *AMIA Annual Symposium Proceedings*, vol. 2018. American Medical Informatics Association, 2018, p. 460.

[35] M. Korobov and K. Lopuhin, "ELI5 is a python library - v. 0.11," 2021, available at (accessed 26/10/2021): https://eli5.readthedocs.io/en/latest/.

[36] U. Schlegel, H. Arnout *et al.*, "Towards a rigorous evaluation of XAI methods on time series," in *Proc. of IEEE/CVF ICCVW*, 2019, pp. 4197–4201.

[37] H. Turbé, M. Bjelogrlic *et al.*, "Evaluation of post-hoc interpretability methods in time-series classification," *Nature Machine Intelligence*, vol. 5, no. 3, pp. 250–260, 2023.

[38] F. Petitjean, A. Ketterlin *et al.*, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition*, vol. 44, no. 3, pp. 678–693, 2011.

[39] M. Cuturi and M. Blondel, "Soft-DTW: a differentiable loss function for time-series," in *Proc. of PMLR ICML*, D. Precup and Y. W. Teh, Eds., vol. 70, 08 2017, pp. 894–903.

[40] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[41] J. C. B. Gamboa, "Deep learning for time-series analysis," in *arXiv*, 2017.

[42] Y. Zhang, Y. Hou *et al.*, "Multi-scale signed recurrence plot based time series classification using inception architectural networks," *Pattern Recognition*, vol. 123, p. 108385, 2022.

[43] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1985.

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

[45] G. Attanasio, C. Fiandrino *et al.*, "In-depth study of RNTI management in mobile networks: Allocation strategies and implications on data trace analysis," *Computer Networks*, vol. 219, p. 109428, 2022.

[46] B. Y. L. Kimura, J. Almeida *et al.*, "Deep learning in beyond 5G networks with image-based time-series representation," *arXiv preprint arXiv:2104.08584*, 2021.

[47] S. A. Siddiqui, D. Mercier *et al.*, "TSViz: Demystification of deep learning models for time-series analysis," *IEEE Access*, vol. 7, 2019.

[48] H. Strobelt, S. Gehrmann *et al.*, "LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE TVCG*, vol. 24, no. 1, pp. 667–676, 2018.

[49] H. Strobelt, S. Gehrmann *et al.*, "Seq2seq-Vis: A visual debugging tool for sequence-to-sequence models," *IEEE TVCG*, vol. 25, no. 1, pp. 353–363, 2019.

[50] W. Guo, "Explainable artificial intelligence for 6G: Improving trust between human and machine," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, 2020.

[51] S. Moghadas Gholian, C. Fiandrino *et al.*, "Spotting deep neural network vulnerabilities in mobile traffic forecasting with an explainable AI lens," in *IEEE INFOCOM*, 2023.

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

[53] C. Fiandrino, G. Attanasio *et al.*, "Toward native explainable and robust AI in 6G networks: Current state, challenges and road ahead," *Computer Communications*, vol. 193, pp. 47–52, 2022.

[54] S. P. Sone, J. J. Lehtomäki *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.

[55] L. Mei, J. Gou *et al.*, "Realtime mobile bandwidth and handoff predictions in 4G/5G networks," *Computer Networks*, vol. 204, p. 108736, 02 2022.

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

[57] H. Nan, X. Zhu *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.