ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

FINAL DEGREE THESIS

ANALYTICAL CALCULATION OF PARTIAL DERIVATIVES IN RNN

Autor: Alonso Ortiz González

Director: Jaime Pizarroso Gonzalo

Contents

List of Figures

Abstract

The complex architecture of artificial intelligence models generates very accurate outcomes, however these are then very difficult to interpret. This "black box" problem poses a significant challenge to the acceptance and trust of AI systems. Explainable Artificial Intelligence (XAI) techniques are essential for making AI's decision-making processes transparent and understandable, ensuring AI can be applied in sensitive decision-making areas. The importance of XAI lies in its ability to unveil the complexity of AI models, allowing users to gain insights into how and why decisions are made, generating trust in AI while also complying with recent policy.

To address this issue, this final degree thesis focuses on the development of a sensitivity analysis method for Recurrent Neural Networks (RNNs). This method has already been successfully applied to MLPs and focuses on the analytical calculation of partial derivatives of output variables with respect to their inputs, providing interpretable insights into the decision-making processes of RNNs.

This study also reviews existing XAI techniques applied to RNNs. These techniques are evaluated both theoretically and through practical implementation on RNN models. Each method's strengths and limitations are discussed, highlighting the need for more effective and efficient interpretability solutions.

The sensitivity analysis method developed in this thesis offers several advantages over traditional XAI techniques. It provides more comprehensive and easily interpretable explanations. This makes the method more robust and reliable, especially when dealing with complex, non-linear relationships in data. Additionally, the sensitivity analysis method is more computationally efficient, making it suitable for large datasets with numerous samples and variables. This efficiency ensures quicker and more accessible interpretability, which is crucial for practical applications.

Validation of the sensitivity analysis method was conducted through three different cases. The first case used a synthetic simple regression model with known derivatives to confirm the correctness of the calculated partial derivatives. This initial validation ensured the method's accuracy and reliability. The method was then applied to two additional use cases: predicting the sine function and forecasting electric demand. In the sine function case, the method demonstrated its ability to capture and explain the periodic nature of the data, providing clear insights into the model's decision-making process. For the electric demand forecasting case, the method highlighted the significance of various time lags and their non-linear effects on the output, offering a detailed understanding of the model's behaviour.

Resumen

La compleja arquitectura de los modelos de inteligencia artificial contribuye a la generación cada vez más precisa de predicciones, sin embargo, también dificulta la comprensión de cómo el modelo ha llegado a esas predicciones. Este problema, conocido como “black box” o “caja negra” plantea un serio reto para la aceptación y la confianza en los sistemas de IA. Las técnicas de Inteligencia Artificial Explicable (XAI) son esenciales para hacer más transparentes y explicables los procesos de toma de decisión de la IA. La importancia de XAI reside en su capacidad para descubrir la complejidad de los modelos de IA, permitiendo a los usuarios obtener insights de cómo se toman las decisiones, fomentando la confianza en la IA, a la vez que cumpliendo con la regulación existente.

Para abordar este problema, este trabajo de fin de grado tiene como objetivo el desarrollo de un método de análisis de sensibilidad para las redes neuronales recurrentes (RNN). Este método ya se ha aplicado con éxito a los MLP y se centra en el cálculo analítico de derivadas parciales de las variables de salida con respecto a sus variables de entrada, proporcionando información fácilmente interpretable sobre los procesos de toma de decisiones de los RNN.

Este estudio también hace una revisión de las técnicas de XAI ya existentes aplicadas a los RNN. Estas técnicas se evalúan tanto teóricamente como a través de la implementación práctica en varios casos de aplicación. Se discuten las ventajas y desventajas de cada método, destacando la necesidad de soluciones de explicabilidad más efectivas y eficientes.

El método de análisis de sensibilidad desarrollado en este trabajo ofrece varias ventajas sobre las técnicas tradicionales de XAI. Proporciona explicaciones más completas y fáciles de interpretar, esto hace que el método sea más robusto y fiable, especialmente cuando se trata de relaciones complejas y no lineales en los datos. Además, el método de análisis de sensibilidad es más eficiente desde el punto de vista computacional, lo que lo hace adecuado para grandes conjuntos de datos con numerosas muestras y variables. Esta eficiencia garantiza una interpretabilidad más rápida y accesible, lo que es crucial para las aplicaciones prácticas.

La validación del método de análisis de sensibilidad se llevó a cabo a través de tres casos diferentes. El primer caso utilizó un modelo de regresión simple sintético con derivadas parciales conocidas para validar los cálculos realizados. A continuación, el método se aplicó a dos casos de uso adicionales: predecir la función seno y predecir la demanda eléctrica. En el caso de la función seno, el método demostró su capacidad para capturar y explicar la naturaleza periódica de los datos, proporcionando información clara sobre el proceso de toma de decisiones del modelo. Para el caso del pronóstico de la demanda eléctrica, el método destacó la importancia de varios lags temporales y sus efectos no lineales en la salida, ofreciendo una comprensión detallada del comportamiento del modelo.

1 Introduction

Artificial intelligence (AI) is a technical and scientific field devoted to the engineered system that generates outputs such as content, forecasts, recommendations, or decisions for a given set of human-defined objectives. It is a field of research in computer science that develops and studies methods and software that enable machines to perceive their environment and use learning and intelligence to take actions that maximize their chances of achieving defined goals.

Machine learning (ML), a subset of artificial intelligence, combines the computational discipline dedicated to the analysis and comprehension of patterns and structures inherent in data (Mahesh, 2020). Its aim is to facilitate autonomous learning, reasoning, and decision-making processes, mirroring the complexity of intelligent human behaviour.

Machine learning can be broadly classified into three types:

1. Supervised Machine Learning: This type involves supervision, where machines are trained on labelled datasets and enabled to make predictions based on the patterns learned.
2. Unsupervised Machine Learning: This type involves no supervision, where machines are trained on unlabelled datasets and enabled to identify patterns and group data points based on similarities.
3. Semi-supervised Machine Learning: This type involves a combination of supervised and unsupervised learning, where machines are trained on both labelled and unlabelled datasets.

In today's world, machine learning holds immense importance, playing a crucial role across various aspects of our daily routines and industries. As part of artificial intelligence, machine learning empowers systems to learn and adapt based on data, allowing them to decipher intricate patterns and make informed decisions. Whether it's delivering personalized recommendations on online platforms (Sarma, 2020), optimizing manufacturing processes through predictive maintenance, or revolutionizing medical diagnostics (Deo, 2015) and autonomous vehicles (Min, 2019), machine learning has proven to be an excellent tool to model reality from data. Its ability to analyse extensive datasets and derive meaningful insights has not only boosted efficiency and accuracy but has also paved the way for innovation in fields spanning healthcare (Esteva et al., 2019), finance (Ngai et al., 2009), and beyond. The ongoing integration of machine learning into diverse sectors highlights its essential role in shaping the present and future landscape of technological advancements and societal progress.

1.1 Artificial Neural Networks

Artificial Neural Networks (ANN) are one of the most prominent models among Machine Learning. ANNs mimic the behaviour of biological nervous systems through mathematical modelling (Abraham, 2005). ANNs models consist of interconnected neurons, also called nodes, which are organized in layers. Each neuron receives input data which is processed using a mathematical function, called activation function. The neuron then passes the result onto the next layer. Each connection between neurons in different layers is weighted to optimize the performance of the model. The superiority of Artificial Neural Networks lies in their ability to capture complex, non-linear patterns inherent in the data, presenting a more adaptive and nuanced approach to modelling intricate relationships.

There are many types of Artificial Neural Networks, each one varying in the complexity of its architecture and its applicability. The simplest models are the feedforward neural networks (FNN) (Rosenblatt, 1958), which consists of layers of nodes through which information is passed forward. Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986) are another class of artificial neural networks specifically designed to process sequential data, i.e., time series data; by introducing the concept of memory. Unlike traditional feedforward neural networks, RNNs possess internal memory that enables them to retain information about previous inputs in the sequence. This memory mechanism makes RNNs particularly well-suited for tasks involving sequential or time-

dependent data, such as natural language processing, speech recognition, and time series forecasting. More complex forms of ANNs, such as Convolutional Neural Networks (CNNs) (LeCun et al., 1998), Long Short-Term Memory (LSTM) Networks (Hochreiter & Schmidhuber, 1997), Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) or Transformer Networks (Vaswani et al., 2017) serve very different purposes but will not be discussed in this project.

1.1.1 Time series Forecasting

One of the most important fields where machine learning is playing a key role is forecasting new values of a time series based on past values (Menon, 2022). Forecasting time series is indispensable for several reasons. Primarily, it facilitates the extraction of meaningful insights from historical data, enabling the identification of underlying patterns and correlations. These insights empower decision-makers to anticipate future changes and make well-informed choices. Furthermore, forecasting time series is vital in mitigating risk and optimizing resource optimization (Box, Jenkins, & Reinsel, 2015), allowing to make decisions with enhanced information and preventing potentially unwanted situations.

Time series analysis encompasses several steps:

- Describing the data through statistical measures or graphical representations.
- Constructing a model that explains historical and current variations in the variables.
- Utilizing the model to forecast future values and creating diverse scenarios.

Before Machine Learning, traditional models such as AutoRegressive Integrated Moving Average (ARIMA) (Contreras, 2003) were commonly used to forecast time series in scenarios where historical values show autocorrelation. An evolution of this model, Seasonal ARIMA (SARIMA) (Vagropoulos, 2016) was specially designed for time series data exhibiting seasonality, where the autocorrelation is shown between values between fixed periods of time. These models are effective when there are repeating patterns at regular intervals of time and have been proven effective when forecasting time-series data with linear relationship, but they fail at handling non-linear dynamic time series structures.

Nowadays, with the evolution of ML, Neural Networks have surpassed the traditional methods previously described. The superiority of neural networks, specially RNN, lies in their ability to capture complex, non-linear patterns inherent in the data, presenting a more adaptive and nuanced approach to modelling intricate relationships and therefore, have been proven very effective to model complex scenarios that involve time-series forecasting (Tokgöz, 2018). The architecture of Recurrent Neural Networks is a crucial feature when forecasting involves understanding patterns that span extended periods, providing a distinct advantage over methods like SARIMA that may struggle to capture such temporal intricacies. Furthermore, neural networks do not require feature engineering nor a prior functional specification, allowing to capture intricate relationship directly from the data without the need of prior knowledge about the relationship between inputs and outputs.

1.1.2 The Black Box Problem: Interpretability vs Performance

The rapid developing of Neural Network models has facilitated the creation of extremely accurate models. However, these models, due to their high accuracy, often possess complex architectures that are difficult to understand. The “Black Box” problem refers to this lack of transparency or interpretability of the neural networks, where the model is a black box that is fed with data and returns predictions without giving information on how it has calculated these predictions. The Black box problem is even more important when the neural networks are used for decision making in sensitive matters. Without a clear understanding of its inner workings, neural networks cannot be fully trusted by people, even if they excel at predicting an outcome (Doshi-Velez & Kim, 2017).

To address this problem, different government organizations have passed laws to regulate AI. The European Union proposed in April 2021 its AI act (Veale, 2021) The proposed legal framework focuses on the specific utilization of AI systems, as well as including provisions for transparency and explainability of AI systems. Moreover, the General Data Protection Regulation (GDPR) (Regulation, 2018) recognizes the right to explanation of individuals about decisions made by automated systems. Therefore, it is imperative to develop methods that allow us to obtain information about the inner workings of Neural Networks in order to use them in decision making processes.

1.2 Explainable Artificial Intelligence (XAI)

Before getting into the objectives of this project, it must be understood how a model can be classified as understandable. Three terms are prevalent in academic literature: interpretability, explainability and transparency. There is no consensus on the definition of the first two terms, and they are often treated as synonyms in academic literature. However, this project will use the following definitions for these terms:

- Interpretability is seen as a passive characteristic of a model that refers to the level at which a given system makes sense to a human observer. A model can be classified as interpretable if a cause-and-effect relationship between the inputs and the output can be stablished. Interpretability has to do with how the model make their decisions (Ali et al., 2023).
- Explainability on the other hand is an active characteristic of the model. It's about how well a model can convey to a human how it arrived at a particular decision. It refers to the understanding of what each node of the model represents and its importance on the overall model's performance. According to Ali et al. (2023), explainability has to do with why the model make their decisions.
- Transparency in AI is a broader concept that encompasses both interpretability and explainability. It refers to the extent to which all stakeholders can clearly understand the workings of an AI system, including how it makes decisions and processes data. Transparency also involves being open about data handling, model training, and evaluation processes. It is crucial for building trust in AI systems, particularly in high-risk applications. Without transparency, there is a risk of creating AI systems that could inadvertently perpetuate harmful biases, make inscrutable decisions, or even lead to undesirable outcomes.

In this project we will focus on making recurrent neural networks more explainable via developing new techniques based on the analysis of partial derivatives, which has been successfully stated as a powerful neural network explainability technique in recent literature (Pizarroso et al. (2022), Pizarroso et al. (2023a), Pizarroso et al. (2023b)). These techniques, as well as all the different techniques and methods used to explain the decision-making process and the output of a model (Schwalbe, Finzel, 2023), are part of what is known as Explainable Artificial Intelligence or XAI.

Explainable Artificial Intelligence has four main goals according to Meske (2022):

1. Build a sufficient understanding about the system's behaviour to detect unknown vulnerabilities and flaws of the model.
2. Provide a better understanding of the inner workings of the model, allowing the developer to improve the model's accuracy and value.
3. Discover unknown correlations with causal relationships in data.
4. Provide explanations about how the model works, accomplishing the requirements from existing regulations.

Various criteria can be employed to categorize the diverse methods and techniques constituting XAI.

Figure 1: XAI method's classification.

1.2.1 Model inherent interpretability

Models can be broadly categorized into interpretable and non-interpretable models. Interpretable models are those where the internal workings are understandable by design. These models, such as linear regression or decision trees, are inherently interpretable, and do not require additional methods to explain them unless the structure is too complex to be humanly processed (Schwalbe, Finzel, 2023).

Interpretable models offer several advantages in various domains. Firstly, they foster trust and transparency by allowing users to comprehend the decision-making process, enhancing trust and facilitating regulatory compliance while also leading to increased acceptance and adoption of these models. Moreover, the architecture of inherent interpretable models aids in model debugging, enabling the identification and correction of errors, biases and overfitting. However, interpretable models also come with certain disadvantages. One notable drawback, as described in Figure 2, is the trade-off between interpretability and accuracy, with simpler models often sacrificing predictive performance by not capturing complex patterns as effectively as the non-interpretable counterpart.

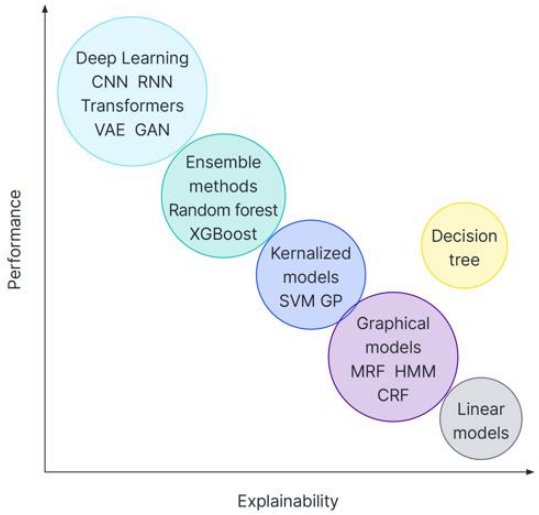

Figure 2: The relation between performance and explainability across different types of models.
Source: https://www.borealisai.com/research-blogs/explainability-i-local-post-hoc-explanations/

Non-interpretable Models, also referred to as black box models, are those where the internal workings are not directly understandable. These models often deliver higher accuracy due to the capacity to capture intricate patterns. Nevertheless, the main drawback of non-interpretable models is their inherent lack of transparency. This opacity challenges the understanding of the decision-making process, undermining trust and potentially impeding adoption of these models, especially in sensitive matters. To unveil the complexity of these models, post-hoc explainability methods are often used. Post-hoc explainability refers to the application of interpretation methods after a model has been trained (Madsen, Reddy, Chandar, 2022).

1.2.2 Specific vs Agnostic methods

This categorization refers to whether the method can explain just a particular type of model or many types of models.

Model-agnostic methods possess a versatility that extends beyond the specific characteristics of any given AI algorithm (Ribeiro et al., 2016). The main advantage is that they serve as a generic tool, enabling developers to unveil the decision-making processes of any black-box model without being constrained by the intricacies of its underlying algorithm, typically by analysing changes on the outputs based on changes on the inputs. However, model-agnostic methods can be less insightful and accurate since they do not take advantage of the specific workings of the black-box model.

On the other hand, model-specific methods are tailor-made to scrutinize the distinctive characteristics of a particular algorithm, offering a deeper understanding and more accurate insights (Lundberg & Lee, 2017). While offering a more in-depth comprehension of how a specific algorithm enhances decision-making within a model, these methods demand a higher level of proficiency from developers during the analysis. Moreover, it is more challenging to compare the interpretability of different models if each requires a unique model-specific method.

1.2.3 Local vs Global explanations.

The subsequent categorization in XAI's taxonomy addresses whether the method explains a specific sample (or region of the input space) or gives information of the model as a whole. Local explainers focus on detailing how a model arrives at a specific decision for an individual case, while global explainers examine the broader patterns in the model's behaviour.

Local explanations offer one main advantage: they provide specificity by offering detailed insights into individual decisions made by the model (Du, 2019). However, local explanations have their limitations. They offer only a limited scope, providing insights into individual predictions but not necessarily offering a comprehensive understanding of the model's overall behaviour. Furthermore, local interpretations may be misinterpreted without a global context, leading to incorrect conclusions (Du, 2019).

On the other hand, global explanations aim to provide a comprehensive understanding of how the model operates across all inputs, contributing to a greater understanding of the model internal mechanism. However, there's a risk that global explanations might oversimplify the model's operations, leading to a loss of critical details about how specific decisions are made (Du, 2019).

1.2.4 Contrastive vs non-contrastive methods.

Following up with XAI's taxonomy, methods can be classified as contrastive or non-contrastive. Contrastive methods clarify why an output was obtained in contrast to another (Jacovi et al., 2021). These methods usually generate an explanation comparing the actual output with a reference value, which leads to a more robust explanation against noise and variations in the data (Balestriero, 2022; Yang, 2023). Nevertheless, contrastive methods are often computationally expensive and memory-intensive due to the large memory banks needed to store negative pairs. Furthermore, contrastive explanations vary if the reference value varies, hence making more difficult the comparison between different explanations.

On the contrary, non-contrastive methods focus on explaining the actual outcome without contrasting it with other potential outcomes. By eliminating the need for a reference sample, non-contrastive methods simplify the explanation process and can reduce the computational burden (Garrido, 2022). Nevertheless, non-contrastive methods might result in less discriminative explanations.

1.3 Motivation

Current XAI methods, although effective, have limitations in terms of computational efficiency and specificity when applied to complex models like RNNs. The proposed research aims to explore and develop an interpretability method based on sensitivity analysis. This method will focus on analysing the changes in the output of an RNN with respect to small variations in its inputs. By adapting sensitivity analysis to RNNs, we aim to provide a more intuitive and direct way to interpret the decision-making process of these networks.

This approach is expected to offer several advantages:

- Direct Interpretability: By examining how changes in input affect output, it provides a straightforward way to understand the model's behaviour.
- Model Specificity: Tailored specifically for RNNs, this method promises deeper insights into their unique architecture and functioning.
- Efficiency: It aims to be less computationally intensive than existing methods, facilitating quicker and more accessible interpretability.

The ultimate goal of this research is to enhance the trust and reliability of RNNs in practical applications. By making RNNs more transparent and understandable, we can ensure their responsible and ethical use, particularly in critical areas where the stakes of decisions are high.

This research will not only contribute to the field of machine learning but also pave the way for more accountable and trustworthy AI systems in the future.

1.4 Project's Objectives

- Review the state of the art of explainability methods applied to simple RNNs:

In the first phase of this research, a comprehensive review of the current state of explainability methods applied to Recurrent Neural Networks (RNNs) will be conducted. This involves examining various techniques and approaches used to interpret and understand the decision-making processes within RNN models and the importance of inputs in the output. This review will synthesize existing academic literature and will then take a more practical approach by evaluating each method.

- Develop a new method to conduct sensitivity analysis based on the calculation of partial derivatives.

Building upon the insights gained from the literature review, the second objective focuses on the development of a new method for conducting sensitivity analysis on RNNs. The proposed method involves the calculation of partial derivatives, providing a mathematical framework to quantify the impact of input features on the model's output. The goal is to enhance interpretability by offering a more granular understanding of how individual features contribute to the overall decision-making process.

- Demonstrate why the proposed method is more efficient and overall superior to the existing methods.

The final objective seeks to demonstrate the superiority and efficiency of the newly developed sensitivity analysis method over existing techniques. Through empirical evaluations and comparative studies, the research aims to showcase the advantages of the proposed method in terms of accuracy, computational efficiency, and ease of interpretation. This empirical evidence will validate the practical significance of the new approach, establishing it as a valuable contribution to the field of explainability in RNNs.

1.5 Sustainable Development Goals Alignment

The Sustainable Development Goals serve as a global blueprint for addressing pressing challenges and achieving a sustainable future. The research contributes significantly to these goals by advancing transparency and interpretability in artificial intelligence, particularly within Recurrent Neural Networks (RNNs). As we delve into the specific SDGs, it becomes evident that the research not only fosters technological innovation and education but also resonates with broader societal aspirations such as economic growth, justice, healthcare improvement, and the pursuit of gender equality. This alignment underscores the multifaceted impact of the research on global sustainability, reinforcing its relevance within a broader framework of social, economic, and environmental objectives.

1.5.1 SDG 4 - Quality Education

By reviewing and synthesizing the state of the art in explainability methods, the research directly aligns with SDG 4's aim to ensure inclusive and quality education for all. The state-of-the-art review hopes to contribute to disseminating valuable insights within the academic and research community, supporting the global pursuit of quality education.

1.5.2 SDG 8 - Decent Work and Economic Growth

By contributing to the technological landscape through the advancements in RNN interpretability, this study hopes to foster economic growth, given the wide variety of RNN applications.

1.5.3 SDG 9 - Industry, Innovation, and Infrastructure

The research significantly contributes to SDG 9 by advancing technology and innovation within the realm of artificial intelligence, particularly RNNs. Through the development of a new sensitivity analysis method, the study fosters advancements in industry, innovation, and infrastructure.

1.5.4 SDG 16 - Peace, Justice, and Strong Institutions

Transparent and interpretable AI models are essential for ensuring justice and fairness in decision-making processes. By providing a method for sensitivity analysis, this research contributes to the establishment of strong and accountable institutions, aligning with SDG 16.

1.5.5 SDG 17 - Partnerships for the Goals

This research promotes collaboration and partnerships within the scientific community through the empirical demonstration of the efficiency and superiority of the proposed method. This aligns with SDG 17, emphasizing the importance of global partnerships to achieve sustainable development goals.

2 State of the art: Explainability methods applied to RNNs

This section will explore some common XAI techniques that have already been developed and that can be applied to the specific architecture of Recurrent Neural Networks.

2.1 LIME

Local Interpretable Model-Agnostic Explanations, or LIME, explain machine learning model predictions, including Recurrent Neural Networks. Introduced by Ribeiro in 2016, this model-agnostic technique involves approximating the model with a local, interpretable model focused on the prediction of interest.

This method generates interpretations for individual predictions by generating simulated data proximal to the input of interest. These simulated points are then employed to train a simpler interpretable model, such as a linear regression, thereby facilitating the explanation of the original black-box model's predictions (Linardatos, 2020).

The optimization problem behind LIME is described in the following equation proposed by Ribeiro in 2016.

$$\xi(x^{(j)}) = argmin_{g \in G}(L(f, g, \pi_{x^{(j)}}) + \Omega(g))$$

The notation $g \in G$ defines an explanation as a model, with G representing a class of potentially interpretable models. $\Omega(g)$ serves as a metric for the complexity of the explanation, as not every $g \in G$ is inherently interpretable. The proximity measure, $\pi_{x^{(z)}}$, quantifies the closeness of an instance z to a given input x. Further, $L(f, g, \pi_{x^{(j)}})$ is a metric indicating how faithful the explanation g is in approximating the model f within the locality defined by π_x. The objective is to minimize $L(f, g, \pi_{x^{(j)}})$ while maintaining $\Omega(g)$ at a sufficiently low level to ensure interpretability for human comprehension.

The utilization of LIME for machine learning model explanation presents several advantages:

- Its model-agnostic nature allows LIME to provide explanations across a wide array of machine learning models, including RNNs. Additionally, LIME supports explanations for different types of data, including tabular data, text and images. In terms of usability, the implantation of LIME in popular programming languages ensures accessibility and ease of use.
- LIME supports explanations for different types of data, including tabular data, text and images.

However, the LIME method also presents some disadvantages:

- The method is susceptible to suboptimal parameter choices, potentially leading to the oversight of silent features (Garreau, 2020).
- Zafar and Khan (2019) also noted the instability of LIME-generated interpretations, as it can provide different interpretations for the same prediction due to the utilization of random perturbation and feature selection methodologies.
- Small changes in the input can lead to significant variations in the explanations.

2.2 SHAP

Shapley Additive explanations (SHAP) is a model-agnostic method based on cooperative game theory based on Shapley values. Introduced by Lloyd Shapley in 1951, Shapley values determine the average marginal contribution of each feature to the model's output, therefore explaining the prediction's distribution among the individual inputs. To calculate the marginal contribution, the SHAP algorithm creates different subsets with every possible combination of features and then average the contribution of a feature to each subset. The

contribution of each feature can be calculated as the difference between the input when that feature is included and the input when that feature is withheld.

The following equation describes the mathematical calculation of the Shapley value, ϕ_i , for a feature i.

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!\,(|N| - |z'| - 1)!}{|N|!} [f_x(z') - f_x(z' \backslash i)]$$

In this equation, f denotes the Blackbox model, x is the input datapoint, and z' represents every combination of subsets which include feature i.

The equation represents a weighted average of the difference of the model's output with and without feature i.

The term $|z'|!\,(|N| - |z'| - 1)!$ Represent the number of orderings that place feature i in a particular position, out of all the possible orderings. The term $[f_x(z') - f_x(z' \backslash i)]$ represents the marginal contribution when the feature i is included in the subset z'.

The weighting is based on how many features, N, there are in each subset, giving more weight to subsets with both a high number of features in the subset (as that would mean the contribution of feature *i* is very important) and a very small number of features (as the features are isolated).

SHAP has some advantages over other methods:

- SHAP satisfies the three desirable properties that every feature attribution method should meet (Lundberg, 2017). The first property is local accuracy, which means that the explanation model should match the output produced by the original model for the simplified input. The second property is missingness, which requires that, given than the simplified outputs represent feature presence, features that are missing in the original input should have no impact. The last property is consistency, which requires that the contribution of an input should not vary if some simplified input contributions' do vary when the model changes.
- SHAP offers both local and global explanations (Lundberg, 2017)
- SHAP considers feature dependencies, making the method more accurate when compared to other methods that assume feature independence (Lundberg, 2017)

However, SHAP also has some drawbacks:

- SHAP is computationally expensive. This is due to the extensive number of subsets that need to be calculated. For a model with n features, there are 2^n possible subsets.
  To address this problem, a variation of the SHAP method called KernelShap has been developed.
  KernelShap is a combination of LIME and SHAP (Lundberg, 2017) and the general idea behind it is to approximate the Shapley values instead of calculating each combination. KernelShap samples features subsets and fits a linear regression model based on the samples. The variables on this linear regression represent the presence or absence of a feature. After training, the coefficients of the linear regression are approximations of the Shapley values.
- SHAP values are difficult to interpret when dealing with a large number of features.
- May not be accessible to every user.

2.3 Partial Dependence Plots (PDPs)

Partial Dependence Plots (PDPs) are a model-agnostic, post hoc, explainability method that helps to visualize the relationship between a subset of input features and the predicted outcome of a machine learning model. Partial Dependence plots show the marginal effect of input variables on the predicted outcome of the model (Friedman, 2001).

Mathematically, for a feature x_i in a model f, the partial dependence function is given by:

$$PDP(f, x_i) = \frac{1}{n}\left[\sum_{j=1}^{n} f\left(x_i^{(i)}, x_{\setminus i}^{(j)}\right), \ldots, f\left(x_i^{(n)}, x_{\setminus i}^{(j)}\right)\right]$$

Where $x_{\setminus i}$ represents every feature excluding x_i. This equation represents a curve that shows the relationship between feature x_i and the predicted outcome. By marginalizing over the other features, the function depends only on x_i, including the interaction between x_i and the other features. PDPs help visualize how the model's prediction change as the feature of interest varies.

Some advantages of PDPs are:

- PDPs offer a global perspective on the model's behaviour, enabling users to comprehend overall trends and patters in the decision-making process.
- The simplicity and ease of interpretation of PDPs make them accessible to a wide audience. They can effectively illustrate both linear and non-linear relationship between features and predictions.

However, PDPs also present some disadvantages:

- As described before, the function includes interactions between variables that could make the predictions difficult to explain when the model does not possess feature independence.
- This method can become computationally expensive for modes with a large number of feature or instances (Goldstein et al., 2015).
- The information provided by PDPs is merely qualitative, which makes the comparison for several features complicated.

2.4 Individual Conditional Explanation (ICE)

Individual Conditional Explanations (ICE) is another model-agnostic, post hoc, and local explanation technique that extends the concept of partial dependence plots (PDPs) by displaying the relationship between input features and predictions at an individual level (Goldstein et al,. 2015). Unlike PDP, which averages the effect of a feature across all instances, ICE plots describe the variation of the prediction with a single feature is varied.

Given a dataset with *n* instances, a predictive model *f,* and a feature x_i, the equation that describe the ice plot is given by:

$$ICE\left(\mathrm{f}, x^{(j)}, x_i\right) = \left[f\left(x_i^{(1)}, x_{\setminus i}^{(j)}\right), \ldots, f\left(x_i^{(n)}, x_{\setminus i}^{(j)}\right)\right]$$

A curve is created for every instance *j,* showing how the model's prediction changes when a feature x_i is varied.

Advantages of ICE:

- ICE allows user to interpret individual predictions, providing deeper insights into the model's behaviour.

Disadvantages:

- Like PDPs, ICE plots assume feature independence, which can lead to inaccurate interpretations if the features are correlated.
- ICE plots can be challenging to interpret when dealing with a large number of instances, as the plots can become cluttered and difficult to read.

2.5 Permutation Importance

Permutation importance is a model-agnostic, post-hoc explainability method used to assess the importance of individual features by measuring the change in model performance when the values of a feature are randomly shuffled (Altmann et al, 2010). This technique evaluates the decrease in model accuracy or increase in error when the association between the feature and the target is broken (Breiman, 2001) providing insights into the feature's impact on the model's prediction Mathematically, the permutation importance of a individual feature x_i is calculated by first measuring the baseline performance of the model using an evaluation metric. The feature values are then permuted, and the performance is measured again. Permutation importance of feature x_i is quantified as:

$$Permutation\ importance\ (x_i) = Baseline\ Performance - Performace\ after\ permutation$$

Advantages of permutation importance:

- Its implementation is very straightforward and intuitive to interpret, as it directly quantifies the effect of each feature on the model's performance.
- Permutation importance is applicable to any type of machine learning model.
- Permutation importance considers feature interactions, as it directly measures the impact of permuting a feature on the final prediction.

Disadvantages:

- Permutation importance can be computationally expensive when working with large datasets, as it involves multiple performance evaluations.
- Permutation importance provides global explanations, so interpreting local prediction is not possible with this technique.

3 Partial Derivatives of Recurrent Neural Network Models

3.1 Introduction

The utilization of partial derivatives to analyse ANNs dates to early studies focused on backpropagation, where the derivatives of the loss function with respect to the weights were computed to optimize network training (Rumelhart et al., 1986). This principle has been extended to sensitivity analysis, as partial derivatives serve to quantify the impact of individual input variables on the network's output (Saltelli et al., 2008). Additionally, sensitivity analysis aids in model validation and robustness assessment by revealing the stability of the network's performance against variations in input data (Gevrey et al., 2003).

Sensitivity analysis based on partial derivatives have already been developed successfully in recent years (Pizarroso et al. (2022), Pizarroso et al. (2023a), Pizarroso et al. (2023b)) for MLP models. This thesis aims to adapt this method to RNNs, as these partial derivatives shall be able to analyse the relationship between inputs and outputs through time (Pascanu et al.,2013).

3.2 Analytical Calculation of the Partial Derivatives of Recurrent Neural Networks

Recurrent Neural Network (RNN) are a type of artificial neural network that incorporates the concept of memory through one or more loops. Unlike traditional feedforward neural networks like MLP where the information flows only in one direction, the information flow in a RNN is bi-directional, as the output of a node can also affect subsequential inputs of the same node. This model is well suited for tasks involving sequential data such as Natural Language processing, speech recognition, time series prediction, or image captioning

A RNN model can have a different number of layers, which can be recurrent or not. In a not recurrent layer, the information is processed in the same way as in a traditional feedforward model. The outputs of the previous layer are multiplied by the weights and then summed together with a constant before being applied to an activation function that produces the output of the neuron.

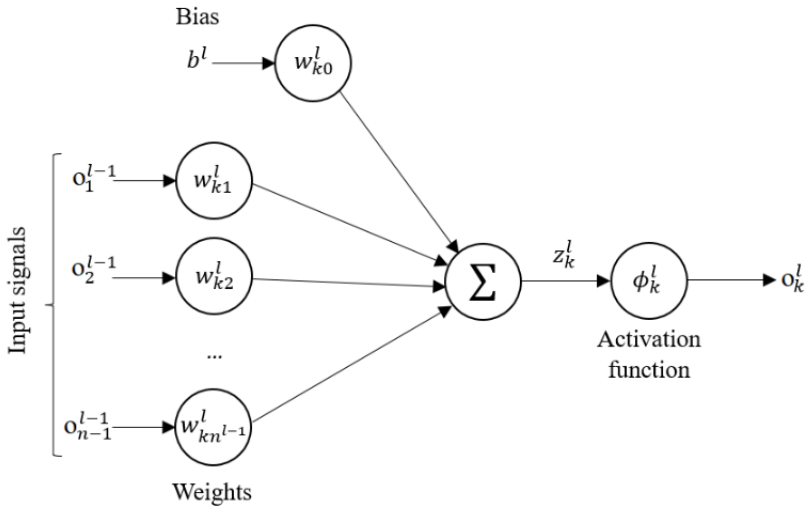

Figure 3: Structure of a feedforward neuron.

In a recurrent layer, the information comes from two directions, the previous layer and from the same layer but at the previous timestep. The information is processed as follows: At each neuron, there is a sum of two types of terms and the bias or constant. The first type are the outputs of the neurons of the previous layer, multiplied

by the feedforward weights. The second type are the previous outputs of the neurons of that layer, which are multiplied by the recurrent weights.

Regardless of the layer type, after all inputs of each unit are summed, an activation function is applied to the result. This result serves as both an input for the next layer and as an input in that layer in the following timestep.

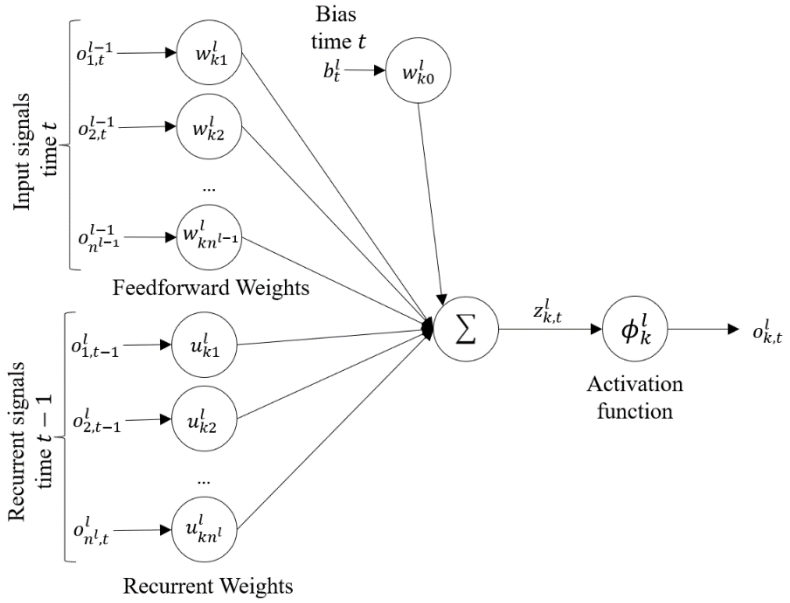

Figure 4: Structure of a recurrent neuron

Figure 4 shows the scheme of a neuron in a RNN model and represent graphically the operations described in the following equation.

For each neuron in a recurrent layer, the output $o_{k,t}^l$ of the k^{th} neuron in the l^{th} layer can be calculated by:

$$o_{k,t}^l = \phi_k^l\left(z_{k,t}^l\right) = \phi_k^l\left(\sum_{i=1}^{n^{l-1}}\left(w_{ki}^l \cdot o_{i,t}^{l-1}\right) + \sum_{j=1}^{n^l}\left(u_{kj}^l \cdot o_{j,t-1}^l\right) + w_{k0}^l \cdot b^l\right)$$

where $z_{k,t}^l$refers to the weighted sum of the neuron inputs, n^{l-1} refers to the number of neurons in the $(l-1)^{th}$ layer, w_{ki}^l refers to the weight of the connection between the i^{th} neuron in the $(l-1)^{th}$ layer and the $k$neuron in the l^{th} layer, u_{kj}^l refers to the weight of the connection between the j^{th} neuron in the l^{th} layer in the $(t-1)^{th}$ timestep and the k^{th} neuron in the l^{th} layer in the t^{th} timestep, ϕ_k^l refers to the activation function of the k^{th} neuron in l^{th} layer, b^l refers to the bias in the l^{th} layer and $\cdot$ refers to the scalar product operation. For the input layer thus holds $l = 1$, $o_{k,t}^{l=1} = x_k$, $w_{ki}^l = 1$, u_{kj}^l and $b^l = 0$

Figure 5 shows a general RNN model. A RNN model can have L layers and each layer l $(1 \leq l \leq L)$ has n^l neurons. Moreover, each layer can have either have a recurrent connection or not.

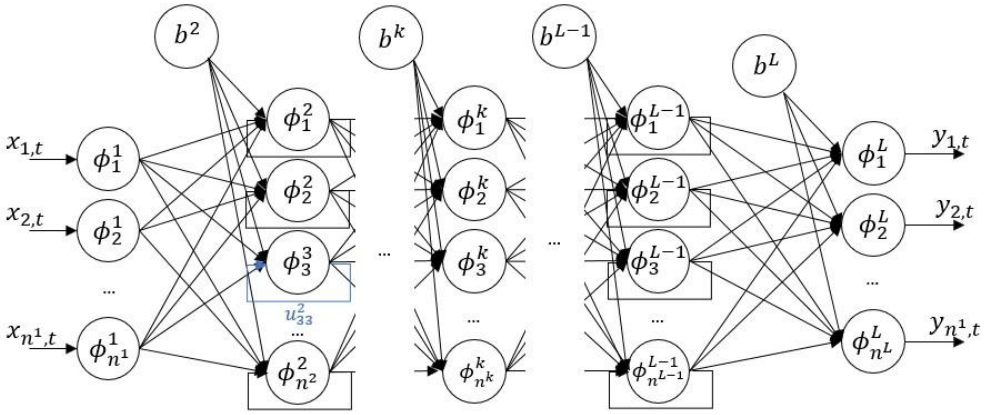

Figure 5: Structure of a recurrent neural network with an arbitrary number of feedforward and recurrent layers.

3.2.1 Partial Derivatives

The method of Sensitivity analysis based on partial derivatives, requires calculating the derivatives of the output with regards to the inputs of the neural network.

These partial derivatives are called sensitivities, and are defined as:

$$PD_{s,i,t} = \frac{\partial y_{s,t}}{\partial x_{s,i,t-j}}$$

In order to calculate them, the chain rule needs to be applied to the partial derivatives of the inner layers.

- Derivative of $z_{k,t}^l$ with respect to $o_{i,t}^{l-1}$ .This derivative corresponds to the weight of the connection

$$\frac{\partial z_{k,t}^l}{\partial o_{i,t}^{l-1}} = w_{ki}^l$$

between the k^{th}neuron in the l^{th} layer and the i^{th} neuron in the $(l-1)^{th}$ layer:

- Derivative of $z_{k,t}^l$ with respect to $o_{i,t-1}^l$ This derivative corresponds to the recurrent weight of the connection between the k^{th} neuron in the l^{th} layer and the i^{th} neuron in the l^{th} layer in the $(t-1)^{th}$ timestep.

$$\frac{\partial z_{k,t}^l}{\partial o_{i,t-1}^l} = u_{ki}^l$$

- Derivative of $o_{k,t}^l$ with respect to $z_{k,t}^l$. This derivative corresponds to the partial derivative of the activation function with respect to the input of the k^{th} neuron in the l^{th} layer evaluated for the input $z_{k,t}^l$ of the k^{th} neuron in the l^{th} layer.

$$\frac{\partial o_{k,t}^l}{\partial z_{k,t}^l} = \frac{\partial \phi_k^l}{\partial z_{k,t}^l}(z_{k,t}^l)$$

With the partial derivatives of the inner layers calculated, the chain rule can now be applied to obtain the sensitivities.

To illustrate the application of chain rule, we will use the following RNN model:

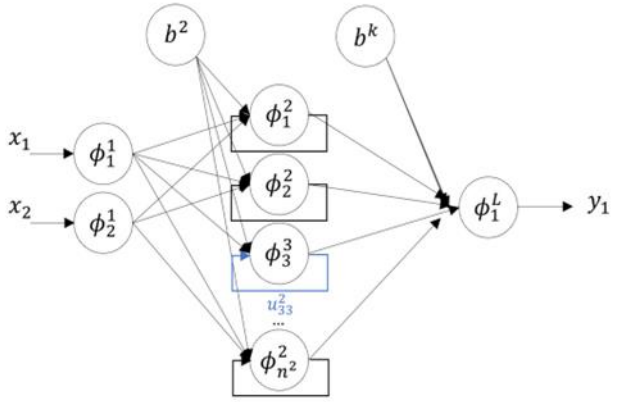

Figure 6: Recurrent Neural Network (RNN) model to be analysed in this project.

The model shown in Figure 6 consists of an input layer, a hidden layer with recurrent neurons, and an output layer.

The sensitivities needed correspond to the partial derivatives of the final output of the model $(y_{k,t})$ with respect to the initial inputs of the model $y_{i,t-j}$ in the $(t-j)^{th}$timestep.

Applying the equations described above we can obtain:

First, the derivative between the output of the model with regards to the input of the hidden layer is calculated.

$$\frac{\partial y_t}{\partial z_t^2} = \frac{\partial y_t}{\partial z_t^3} * \frac{\partial z_t^3}{\partial o_t^2} * \frac{\partial o_t^2}{\partial z_t^2} = \frac{\partial \phi^3}{\partial z_t^3}(z_t^3) * W^2 * \frac{\partial \phi^2}{\partial z_t^2}(z_t^2)$$

This expression can be simplified as in most cases the activation function of the output layer corresponds to the identity and the weights of said layer are all equal to 1

$$\frac{\partial y_t}{\partial z_t^2} = \frac{\partial \phi^2}{\partial z_t^2}(z_t^2)$$

Then, we can continue applying the chain rule until we get to input of the model in the desired timestep:

$$\frac{\partial z_t^2}{\partial x_t} = \frac{\partial z_t^2}{\partial o_t^1} \cdot \frac{\partial o_t^1}{\partial z_t^1} \cdot \frac{\partial z_t^1}{\partial x_t} = \mathrm{I} \cdot \mathrm{I} \cdot W^1 = W^1$$

$$\frac{\partial z_t^2}{\partial x_{t-1}} = \frac{\partial z_t^2}{\partial o_{t-1}^2} * \frac{\partial o_{t-1}^2}{\partial z_{t-1}^2} * \frac{\partial z_{t-1}^2}{\partial o_{t-1}^1} \cdot \frac{\partial o_{t-1}^1}{\partial z_{t-1}^1} * \frac{\partial z_{t-1}^1}{\partial x_{t-1}} = U * \frac{\partial \phi^2}{\partial z_{t-1}^2}(z_{t-1}^2) * W^1$$

And as a generalization:

$$\frac{\partial z_t^2}{\partial x_{t-j}} = \frac{\partial \phi^1}{\partial z_{t-j}^2}\left(z_{t-j}^2\right) * W^0 * W^1 * \prod_0^{i=j}\left(\frac{\partial \phi^2}{\partial z_{t-i}^2}\left(z_{t-i}^2\right) * U\right)$$

Rationale: derivatives shall be calculated first backwards in time then backwards in layers

Combining the two parts of the derivative:

$$\frac{\partial y_t}{\partial x_{t-j}} = \frac{\partial y_t}{\partial z_t^2} * \frac{\partial z_t^2}{\partial x_{t-j}} = \frac{\partial \phi^3}{\partial z_t^3}(z_t^3) * W^2 * \frac{\partial \phi^2}{\partial z_t^2}(z_t^2) * \frac{\partial \phi^1}{\partial z_{t-j}^2}\left(z_{t-j}^2\right) * W^0 * W^1 * \prod_0^{i=j}\left(\frac{\partial \phi^2}{\partial z_{t-i}^2}\left(z_{t-i}^2\right) * U\right)$$

3.2.2 Sensitivity analysis

Once all the sensitivities have been obtained, the results can be analysed through the following measures. Most of them have been used in recent literature to analyse MLP models (Pizarroso et.al, 2022). The other measures are variations of the previous ones to measure the variability across time of the sensitivities in a RNN model.

- Mean sensitivity of the final output with respect to the input i^{th} in timestep t^{th}:

$$\overline{PD}_{i,t} = \frac{\sum_{s=0}^{n} PD_{s,i,t}}{n}$$

  Where *n* is the number of samples in the dataset

- Standard deviation of the final output with respect to the input i^{th} in timestep t^{th}:

$$\sigma\left(PD_{i,t}\right) = \sqrt{\frac{\sum_{s=0}^{n}\left(PD_{s,i,t} - \overline{PD}_{i,t}\right)^2}{n}}$$

- Mean squared sensitivity of the final output with respect to the input i^{th} in timestep t^{th}:

$$\overline{PD_{i,t}^2} = \sqrt{\frac{\sum_{s=0}^{n} PD_{s,i,t}^2}{n}}$$

- Mean sensitivity of the output with respect to the input i^{th} :

$$\overline{PD}_i = \frac{\sum_{t=0}^{T} PD_{s,i,t}}{T}$$

- Standard Deviation of the output with respect to the input i^{th} :

$$\sigma(PD_i) = \sqrt{\frac{\sum_{t=0}^{T}(\overline{PD_{i,t}^2} - \left(\overline{PD}_{i,t}\right)^2)}{T}}$$

- Mean squared sensitivity of the output with respect to the input i^{th} :

$$\overline{PD_i^2} = \sqrt{\frac{\sum_{t=0}^{T}\left(\overline{PD_{i,t}^2}\right)^2}{T}}$$

3.3 Implementation of the partial derivatives method

After explaining the theorical foundations of the proposed method, this section will describe its implementation.

The RNN_XAI package contains three functions that serve different purposes:

- The `jacobian_rnn` function first calculates all the intermediate values of the RNN model, this are all the inputs and outputs of every neuron in the recurrent network. Then, based on the formulas of section 3.2.1, calculates all the partial derivatives of every output variable with respect to every input variable in every timestep.
- The `compute_sensitivities` function calculates all the sensitivity measures described in section 3.2.2
- Lastly, the `plot_partial_derivative_analysis` generates different plots to visualize the previously mentioned sensitivity measures. The explanation of each graph will be discussed in the next section with a practical example.

3.4 Synthetic Example: Predicting a simple regression model

This section shows the application of the RNN_XAI package to a synthetic example to perform sensitivity analysis. The dataset used in this example will be constructed using a simple regression model with known derivatives. By doing this, we aim to validate the calculation of the partial derivatives used in our method.

The are three randomly generated input variables using a normal distribution with zero mean and standard deviation equal to 1. The input variables have then been reshaped into the appropriate form to train the RNN model. After reshaping, the dataset consists of three variables, each with 9996 rows of observations and three columns for three timesteps. An observation of one of the three input variables will be the following:

$$X_i = [x_{i,t}, x_{i,t-1}, x_{i,t-2}]$$

The output Y is created using the following equation:

$$y_{[t]} = (x_{1,t})^2 + x_{2,t-1} - x_{3,t-2}$$

In this simple regression model the partial derivatives can be easily calculated,

1. Partial derivative of output y_t with respect to $x_{1,t}$:

$$\frac{\partial y_t}{\partial x_{1,t}} = 2 \cdot x_{1,t}$$

2. Partial derivative of output y_t with respect to $x_{2,t-1}$:

$$\frac{\partial y_t}{\partial x_{2,t-1}} = 1$$

3. Partial derivative of output y_t with respect to $x_{3,t-2}$:

$$\frac{\partial y_t}{\partial x_{3,t-2}} = -1$$

In this example, what we expect is that the derivatives with respect to $x_{2,t-1}$ and $x_{3,t-2}$ being constant (equal to 1 and -1 respectively), and the derivatives with respect to $x_{1,t}$ depending on the value of $x_{1,t}$. Therefore, what we expect is that: $\sigma(PD_{2,t-1}) = \sigma(PD_{3,t-2}) \approx 0$, $\sigma(PD_{1,t}) \gg 0$, $\overline{PD}_{2,t-1} = 1$ and $\overline{PD}_{3,t-3} = -1$. The derivative with respect to the rest of timesteps are zero, as they do not have relationships with the output.

To test the functionality of the sensitivity analysis designed, we trained a RNN model with 20 neurons in its only recurrent hidden layer using the previously described dataset. Figure 7 shows the plot for both the true values of the output and the predicted values obtained with the model.

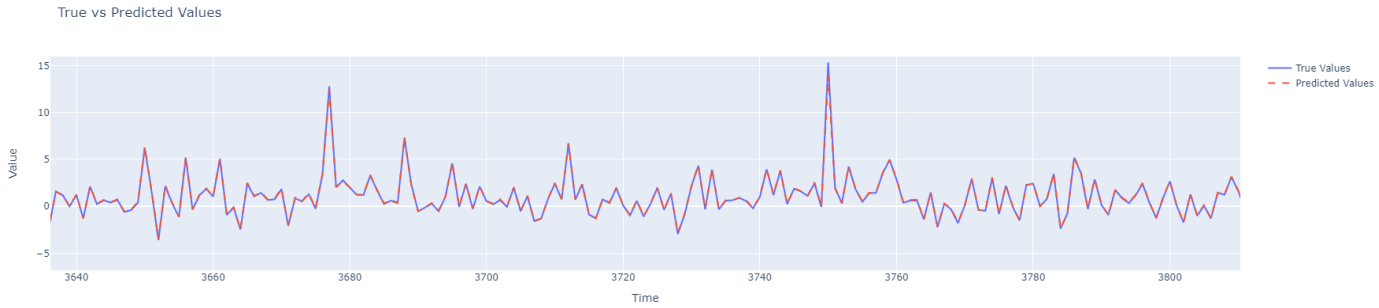

Figure 7: Comparison between true and predicted values of the simple regression model

After training the model, the partial derivatives method was applied to the model and the sensitivities were computed, obtaining the following sensitivity metrics:

Mean_PD

	t	t-1	t-2
X1	1.546889	-0.116307	0.188530
X2	0.193668	0.943034	-0.192784
X2	0.002252	-0.003083	-1.001983

Std_Dev

	t	t-1	t-2
X1	1.133926	0.121913	0.062677
X2	0.448836	0.068737	0.057184
X3	0.019635	0.010386	0.018273

Mean_sqrt_PD

	t	t-1	t-2
X1	3.678655	0.028390	0.039472
X2	0.238962	0.894038	0.040436
X3	0.000391	0.000117	1.004304

Figure 8: Numerical results of the main sensitivity measures obtained in the simple regression model

The mean sensitivities obtained are similar to what we have expected from deriving the equation of the model with respect to each variable, proving that the partial derivatives are well calculated. The difference between the expected sensitivity metrics and the obtained sensitivity metrics is caused by the inherent modelling error, as the RNN do not make a perfect approximation to the real function. In this case, after the training of the model was complete, the final value of the loss function (prediction error) was 0.0023. As presented in Figure 7, the predictions seem highly accurate.

The values of the sensitivity measures presented in Figure 8 indicate the following information about the variables:

- $X_{1,t}$: its mean sensitivity $(\overline{PD}_{i,t})$ of approximately 2 and standard deviation $(\sigma(PD_{i,t}))$ close to 1, suggest that this variable in this timestep has a non-linear effect on the output variable. Moreover, the high value of the mean squared sensitivity $(\overline{PD^2_{i,t}})$ indicates the importance of the variable.

- The rest of the timesteps of the first input variable ($X_{1,t-1}, X_{1,t-2}$) all present $\overline{PD}_{i,t}$ close to zero, as well as $\sigma(PD_{i,t})$ close to zero. These suggest the variables have almost no effect, with little influence on the output variable.
- $X_{2,t-1}$: with $\overline{PD}_{2,t-1} \approx 1$ and $\sigma(PD_{2,t-1}) \approx 0$, it suggests that this variable possess a notable linear effect on the output variable.
- $X_{2,t}$ and $X_{2,t-2}$ both present a mean sensitivity close to zero and low standard deviation, which indicates almost no effect of these timesteps on the output.
- $X_{3,t-2}$: with $\overline{PD}_{3,t-2} \approx -1$ and $\sigma(PD_{3,t-2}) \approx 0$, it suggests that this variable possess a significant negative linear effect on the output variable.
- $X_{3,t}$ and $X_{3,t-1}$ both present a mean sensitivity close to zero and low standard deviation, which indicates almost no effect of these timesteps on the output.

For models with more variables and for a deeper analysis on the model's predictions, we can also make use of the graphs produced by the `plot_partial_derivative_analysis` function.

1. The scatter plot presented in Figure 9 shows the relationship between $\overline{PD}_{i,t}$ and $\sigma(PD_{i,t})$. This graph serves to rapidly analyse the linearity of the effect of every variable on the output. The extent to which a variable separates from the horizontal axis, representing $\sigma(PD_{i,t})$=0, corresponds to the magnitude of its non-linear influence on the output. Similarly, the extent to which a variable separates from the vertical axis, representing $\overline{PD}_{i,t} = 0$, measures the significance of its effect on the output variable. As represented in this graph, $X_{1,t}$ presents a non-linear effect, which is the most significant effect to the output. $X_{2,t-1}$ and $X_{3,t-2}$ show a linear effect, the former one being a positive effect and the latter one being a negative effect. The rest of the variables show little effect on the output variables, but their non-zero distances with the axis can be explained by spurious relationships found by the model due to the correlation between the variables and their lags. This effect would be further discussed in section 4.4

2. The bar plot expands the information of the previous plot, as it shows the $\overline{PD^2_{i,t}}$ for each input variable. Moreover, darker greens represent larger positive effects of the input variable on the outputs, while darker reds represent larger, negative effects. The information of this graph is aligned with the numerical analyses conducted before, as it shows that $X_{1,t}$, $X_{2,t-1}$ and $X_{3,t-2}$ are the ones with the more impact in the model, the former one being the most important with a positive effect.

3. The density plot of Figure 9 shows the distribution of output sensitivities for each variable at each timestep. A narrow distribution, such as the ones for $X_{2,t-1}$ and $X_{3,t-2}$, indicate a linear relationship with the output variable, while a wider distribution, like the one for $X_{1,t}$, suggest a non-linear effect.

4. The fourth graph of Figure 9**¡Error! No se encuentra el origen de la referencia.** displays the evolution of the partial derivatives, or sensitivities, for each input and timestep across all samples. This serves as a visual representation of the variability of the effect of each variable on the output feature. When analysing the graph, we arrive to the same results as with the other plots.

5. The last graph in Figure 9 shows the evolution of the mean sensitivity, standard deviation and the mean of squared partial derivatives for each variable across the different timesteps.

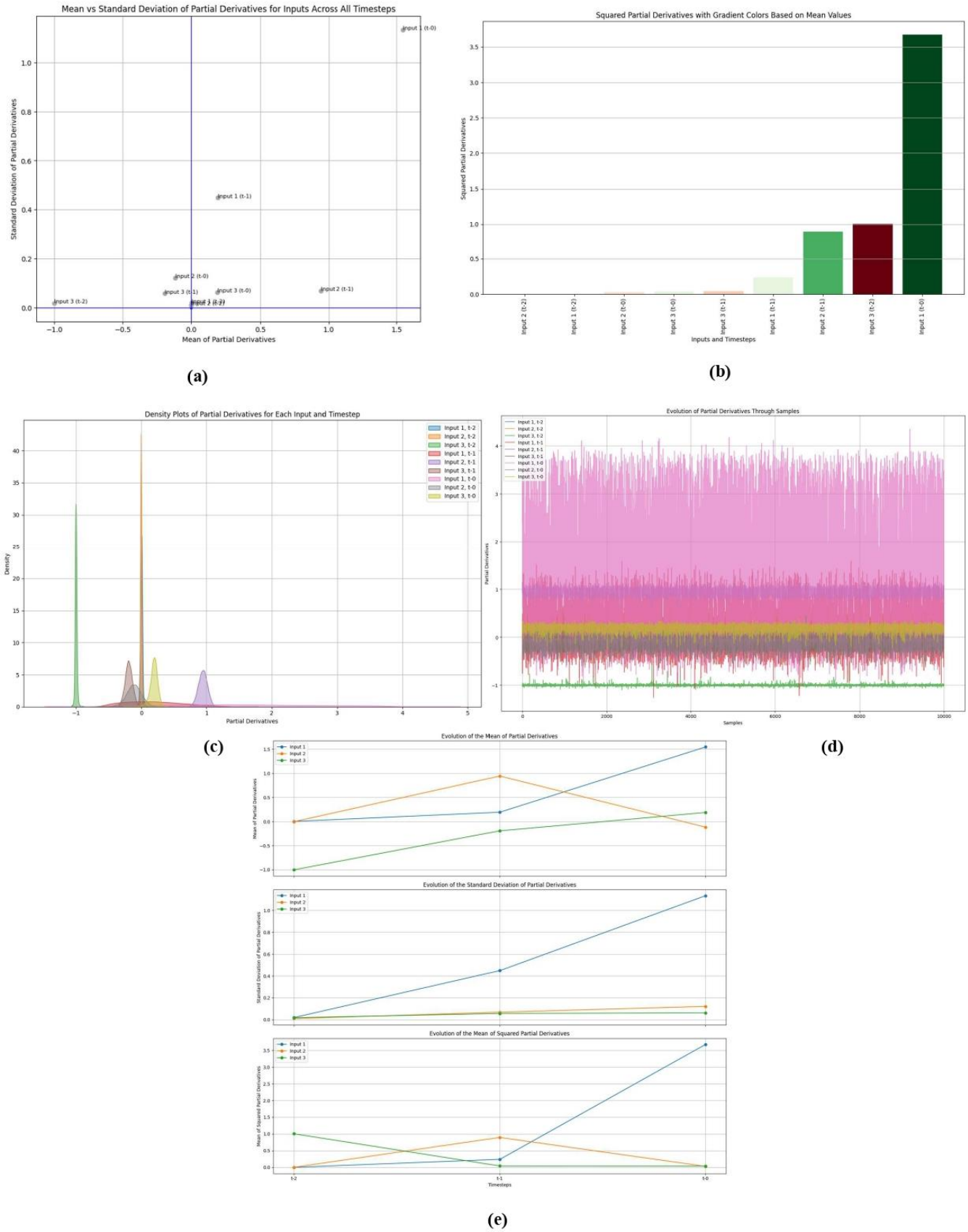

Figure 9: Sensitivity plots for the simple regression case. (a) Scatter plot. (b) Bar blot. (c) Density plot. (d) Partial derivatives through samples. (e)Sensitivity measures through timesteps

4 Use cases

4.1 Introduction

In this chapter, the previously described method of sensitivity analysis based on partial derivatives will be practically examined via two different uses cases. One of the cases being a synthetic dataset, which serve as further validation of the model and of the analytical calculation of the sensitivities and partial derivatives. The second case is a real-life dataset that serve to further analyse the performance of the partial derivative method as a new XAI technique.

Moreover, the partial derivative method will be compared with the application of the existing XAI methods described in State of the art: Explainability methods applied to RNNs section of this project.

The two application cases are as follows:

1. **Case 1: Predicting the sin function:** In this case, a RNN model will be trained to predict the sin function and the different methods described previously will be applied to it.
2. **Case 2: Predicting electric demand:** In this case, the methods are applied to a RNN model that predicts the electric demand based on two input variables

4.2 Case 2: Predicting the sine function

As previously discussed in this project, Recurrent Neural Networks excel at handling sequential data, making them particularly suitable for tasks involving time series and temporal patterns. One of the standard applications to test the advantages of RNNs is the sine function. This application is often utilized as a standard benchmark, because the sine function is inherently periodic and continuous, possessing a clear and predictable pattern over time. These characteristics make the sine function a simple, yet effective test case for demonstrating the capability of RNNs to capture temporal dependencies and learn from sequential data (Sherstinsky, 2020).

Moreover, simpler models such as MLP are not well equipped to handle sequential dependencies. MLPs treat each input independently, lacking the internal mechanism that enables RNNs to decipher temporal dependencies. The inability to capture these temporal dependencies result in suboptimal performance for MLPs on tasks like predicting the sine function outside the training data range (Hochreiter et al., 1997).

To construct the dataset, we generated 600 samples of equally spaced points between 0 and 2π to be taken as inputs and then calculated the output of the sine function. After that, the dataset was reshaped to the appropriate form to train a RNN model. We chose 8 timesteps for each instance ending obtaining a dataset with an input variable with 591 rows of observations and 8 columns (one for each timestep) and an output variable with 591 rows of observations.

After constructing the dataset, a simple RNN model was trained to predict the sine function. The model consists of a single recurrent hidden layer with 12 neurons. Figure 10 plots the generated sine wave with the predictions obtained from the RNN model.

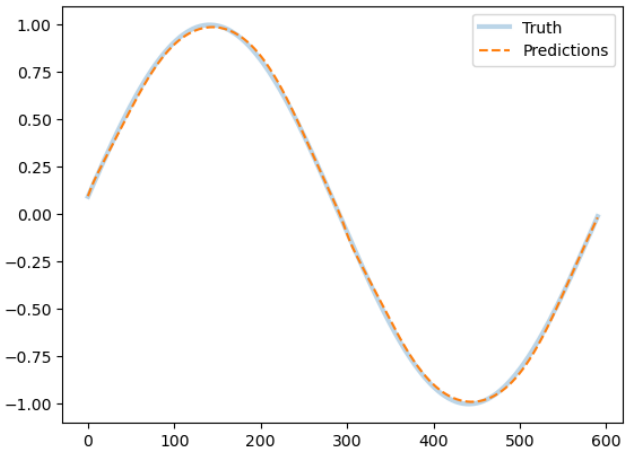

Figure 10: Comparison between true and predicted values for the sine function

After training the model and predicting the output values for the sine function, the following explainability methods were applied to the model's predictions:

- First, the sensitivity method based on partial derivatives developed in this thesis.
- Second, the LIME method was applied to provide instance-specific explanations and identifying key features for specific predictions.
- Third, the SHAP method offers a global understanding of feature importance and consistent individual predictions.
- PDP and ICE curves help understand the effect of each variable to the model's output and detect non-linear interactions.
- Lastly, Permutation importance offers a simple understanding of the importance of each variable on the model's predictions.

From the partial derivative method, as shown in the scatter and bar plot of Figure 11, we can see that the most influential variables are the timesteps $X_{t-3}, X_{t-4}, X_{t-2}$ as they present the higher absolute mean values. Moreover, these three variables all have a positive effect on the model's output, as seen by the colour of the bar plot. By looking at the density plots, we can see somewhat wide distributions for these variables, which suggest the non-linearity of its effects. Other variables such as X_t, X_{t-1} have a smaller contribution on the model's output, but such contributions are negative. These variables also present a non-linear effect on the model's output, as presented in the density plot. On the other hand, the latter timesteps ($X_{t-5}, X_{t-6}, X_{t-7}$) are the least important of all the variables, as shown by their position in the scatter plot. The relatively stable and lower values of their partial derivatives indicate that contribution of these variables is smaller and somewhat linear.

Overall, the sensitivity analysis performed suggests the following:

- The model constructs the sine function based primarily on the positive, non-linear contributions of intermediate timesteps ($X_{t-3}, X_{t-4}, X_{t-2}$) and then uses the most recent timesteps (X_t, X_{t-1}) to correct the predictions.
- The most distant timesteps ($X_{t-5}, X_{t-6}, X_{t-7}$) are of little significance to the model's prediction and show a more linear effect on the output variable.

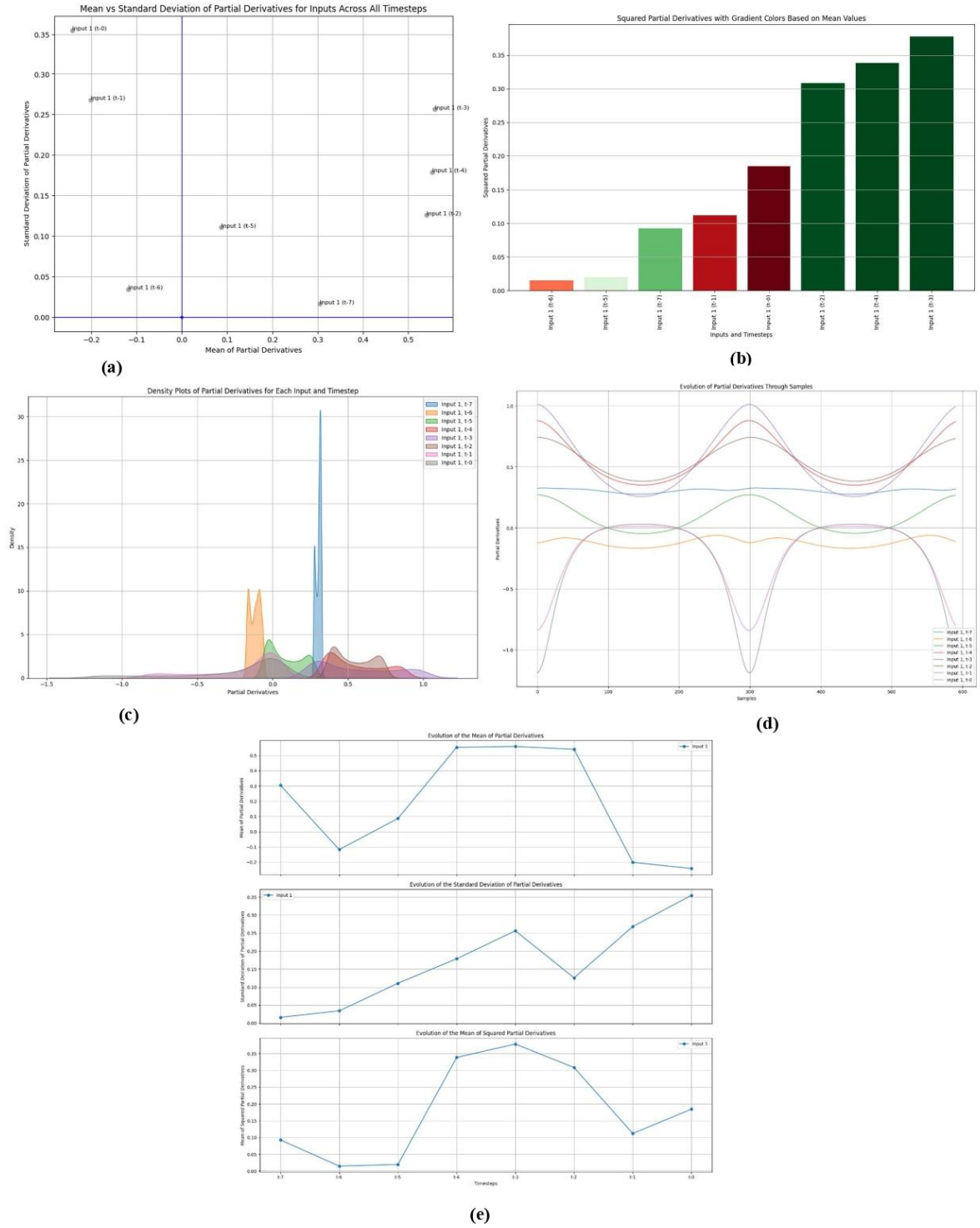

Figure 11. Sensitivity plots for the sine function. (a) Scatter plot. (b) Bar blot. (c) Density plot. (d) Partial derivatives through samples. (e)Sensitivity measures through timesteps

The explanations offered by LIME take a different approach, as the LIME method consists of creating a local, linear approximation of the model's behaviour around specific instances. Therefore, the effects of each variable, as represented in the first graph of Figure 12 (a), are all constant across instances. In the bar plot of Figure 12(a), the lime coefficients of every feature are shown. LIME attributes the most significance to X_{t-2}, X_t and less

significance to X_{t-3}, X_{t-4} in contrast with the results obtain from the partial derivative method. However, LIME also considers that the most distant timesteps $(X_{t-5}, X_{t-6}, X_{t-7})$ are the least significant to the model.

In the last plot of Figure 12(a), we plotted the true sine values, against the RNN model predictions, and the lime predictions based on the lime coefficients obtained. The high inaccuracy of the lime prediction is due to the nature of lime explanations. As explained before, LIME tries to fit a linear model to explain individual instances, and is failing at capturing the smooth, periodic nature of the sine function accurately.

The explanations offered by SHAP are similar regarding the importance of the variable. As shown in the bar plot of Figure 12(b), the most important variables for SHAP are X_t, X_{t-3}, X_{t-4} and the least important are again $(X_{t-5}, X_{t-6}, X_{t-7})$. The beeswarm plot of Figure 12(b) indicate a linear effect for intermediate feature values. This linear effect is only distorted by the extreme feature values, which corresponds to the peaks of the sine function.

Moreover, we can make a similar analysis to the one performed with the partial derivative methods regarding the direction of the contribution of each variable. $(X_{t-3}, X_{t-4}, X_{t-2})$ seem to contribute to the creation of the sine function, as the SHAP values are negative for negative feature values and positive for positive feature values. X_t, X_{t-1} on the other hand, seem to go in the other direction, as they present positive SHAP values for negative feature values and vice versa.

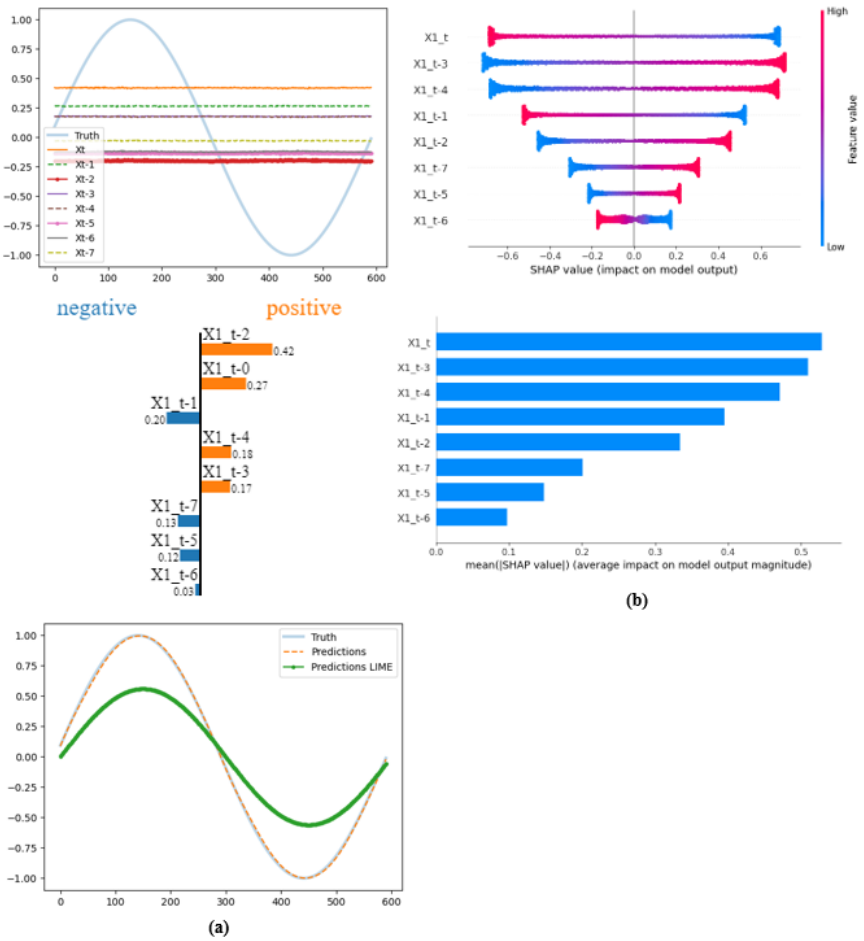

Figure 12: (a) LIME graphs for the sine case. (b) SHAP graphs for the sine case

Continuing with the application of XAI methods to the sine case, we plotted the PDP and ICE curves for the input variable at every time step and are presented in Figure 13. The PDP lines, shown in red, describe the average effect of each feature on the model's output, while the ICE curves, in grey, show the impact on individual

instances. The PDP for the current timestep, X_t, reveals a negative slope, indicating that higher values of this feature generally lead to lower predictions. However, the spread in the ICE curves suggests some variability among individual instances. X_{t-1} also shows a less pronounced negative slope, with less spread, indicating a consistent but minor negative influence. In contrast, X_{t-2} presents a slight positive slope, and X_{t-3}, X_{t-4} have the most pronounced positive slopes, indicating these features significantly increase the model output. On the other hand, X_{t-5}, X_{t-7} also show positive slopes but with considerable ICE curve variability, suggesting moderate and less consistent influences. The least influential feature, X_{t-6}, has a flat PDP and minimal ICE curve variation, indicating that changes in this feature have little effect on the model's output.

The results of PDP and ICE curves agree with the partial derivatives and the SHAP method on the importance of X_{t-3}, X_{t-4} and with those said methods and LIME in considering the most distant timesteps as the least significant to the model's output.

Lastly, the Permutation Importance method offers a simple overview of feature relevance. In this method, X_{t-3}, X_{t-4} are still the most relevant, but the importance of the rest variables differs from the other methods described. However, Permutation Importance does not give any information regarding the nature of the effect of the feature on model outputs, making this method just a complement for other more complete methods.

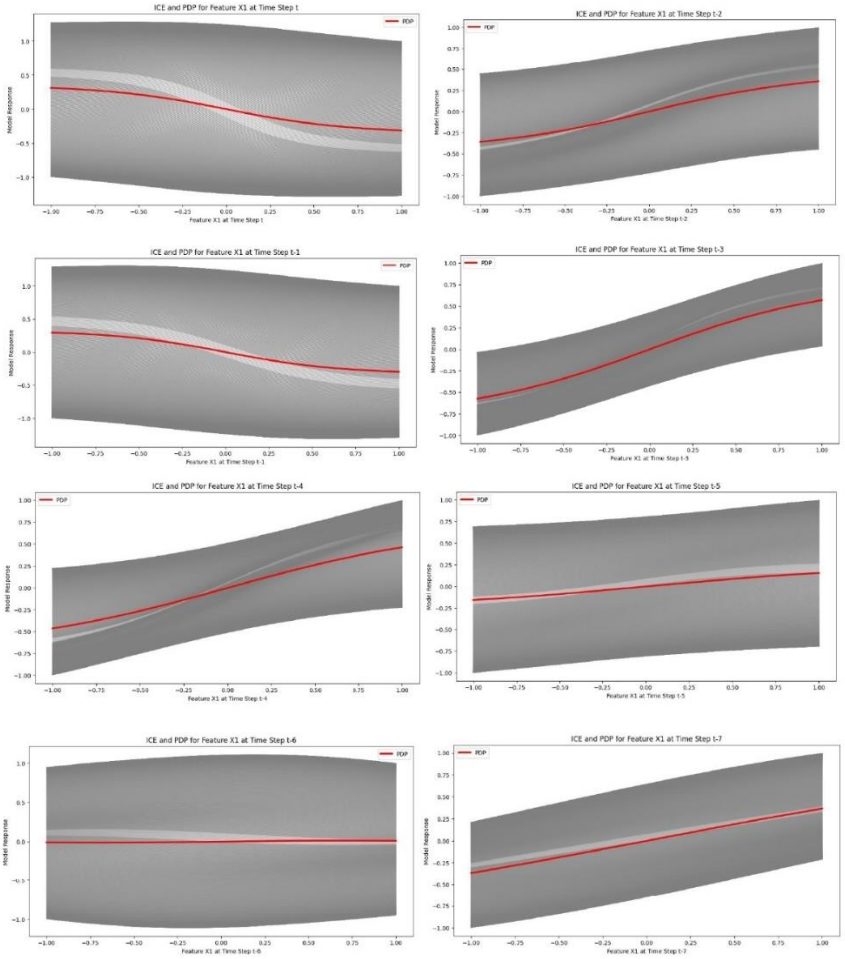

Figure 13: PDPs and ICE curves for the sine case

```
{(0, 0): 0.1204011395907097,
 (1, 0): 0.10551480620308265,
 (2, 0): 0.1343106820587593,
 (3, 0): 0.3737239766493224,
 (4, 0): 0.2203026310759624,
 (5, 0): 0.02601401156073681,
 (6, 0): 0.0027450158600067416,
 (7, 0): 0.14184071324869846}
```

Figure 14: Permutation Importance results for the sine case

4.3 Case 2: Predicting electric demand

Predicting electric demand is a crucial aspect of energy management and planning, providing benefits ranging from economic efficiency to enhanced grid reliability. The dataset used in this analysis contains daily records of electric demand, working conditions, and temperatures over a period. Understanding and predicting electric demand can significantly aid utility companies and grid operators in making informed decisions about energy production and distribution.

The dataset contains 1980 instances, comprising records from July 1, 2007, onward and includes four main attributes:

1. DATE: The date of the recorded observation.
2. DEM: The electric demand measured in consistent units.
3. WD: A measure of how much work is done on that day. For instance, Sundays typically have lower WD values compared to weekdays, indicating a less labour-intensive day.
4. TEMP: The mean temperature recorded on that particular day.

Therefore, this case contains two input variables (WD and TEMP), and the target variable is the electric demand (DEM).

Before training the model, it is crucial to reshape the data in the appropriate form to train the RNN. We chose to work with three timesteps for every instance. The timesteps selected correspond to the previous day $(t-1)$, the day before $(t-2)$, and the same day of previous week $(t-7)$. The reason behind the selection of these timesteps is to obtain a simpler dataset, as it is a simplification of including all the timesteps that correspond to the whole previous week. No timesteps further than a week have been considered, because as plotted in Figure 15, the electric demand presents 7-day-cycles.

The model we used to predict the electric demand is a simple RNN with 32 neuron in its only hidden recurrent layer. Figure 15 shows the plot for the true electric demand and the predicted by the model. The model seems to have captured correctly the periodic nature of the electric demand and predicts quite accurately intermediate and values and low peaks of demand. On the other hand, prediction for high peaks show some error in the model.

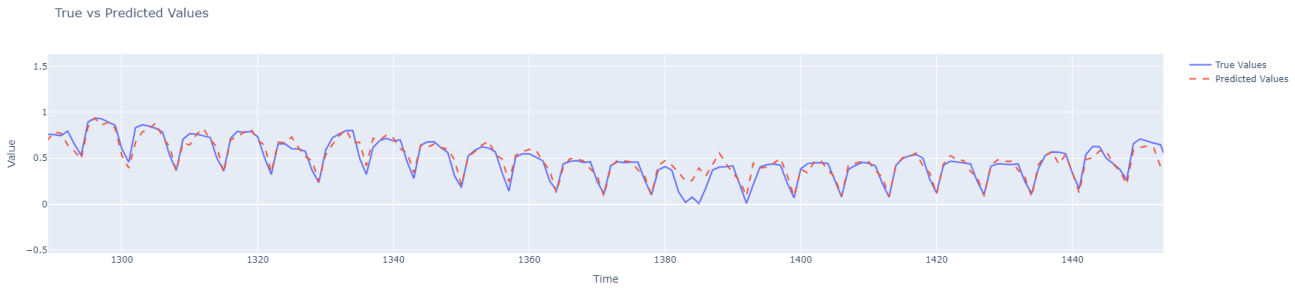

Figure 15: Comparison between true and predicted values for the electric demand case

After training the model, the same methods as the previous use case were applied, with the exception of LIME, which failed to produce reasonable explanations of the model.

The Partial Derivatives Method offers a complete analysis regarding feature importance, the effects of every variable and the evolution across time and samples. As shown in the scatter plot in Figure 16, WD_{t-1} exhibits high variability and a positive mean, indicating a significant and consistent influence on the output. WD_{t-2} and WD_{t-7} also shows notable variability but with a lower mean compared to WD_{t-1}. Temperature variables, particularly $TEMP_{t-7}$ and $TEMP_{t-2}$ have a lower variability and near-zero mean, suggesting a weaker impact on the model. The squared partial derivatives graph reassures the information obtain from the previous graph, regarding feature importance. WD_{t-1} is the most significant, creating a positive impact on the model, followed by WD_{t-2} and $TEMP_{t-1}$, both with negative impacts on the model. Regarding the linearity of the effects, we can look at the density plot in Figure 16 to see that WD_{t-1} and WD_{t-2} present very wide distributions, suggesting a non-linear effect. $TEMP_{t-1}$ and $TEMP_{t-2}$ also seems to be non-linear, while $WD_{t-2}$and WD_{t-7} present a narrow distribution, suggesting a more linear, though less significant effect.

Overall, we can extract the following conclusions.

- WD_{t-1} and WD_{t-2} are the most relevant features, with opposite non-linear effects. At first hand, one could expect WD_{t-7} to be the most significant, as it corresponds to the working day of the same day on previous week, instead it is one of the less relevant features of the model. However, the feature importance found by the partial derivative method makes sense. By taking another look at Figure 15, we can see that there are very strong low peaks, conformed by single points, attributable to Sundays, while high peaks are more extended, with a couple points with similar values, corresponding to central weekdays. This explains why the model might be focusing on the last two values of WD and compensating the effects of one another instead of just focusing directly in the WD of the past week.
- The other relevant feature, $TEMP_{t-1}$, presents a non-linear effect. This seems logical, as higher values of temperature probably increase demand, but lower values also increase demand.

The SHAP method suggests a very similar feature importance to the sensitivity method. Looking at the bar plot in Figure 17, WD_{t-2} ,$TEMP_{t-1}$ and WD_{t-1} are the most significant variables, in that order. On the other hand, $TEMP_{t-7}$ and WD_{t-7} are considered the least important, same as the sensitivity method. Regarding the nature of the effects, the results are similar to the sensitivity method. The pair WD_{t-1} , WD_{t-2} presents opposite non-linear effects. The $TEMP_{t-1}$ also shows a non-linear effect, agreeing with the results of the partial derivative method.

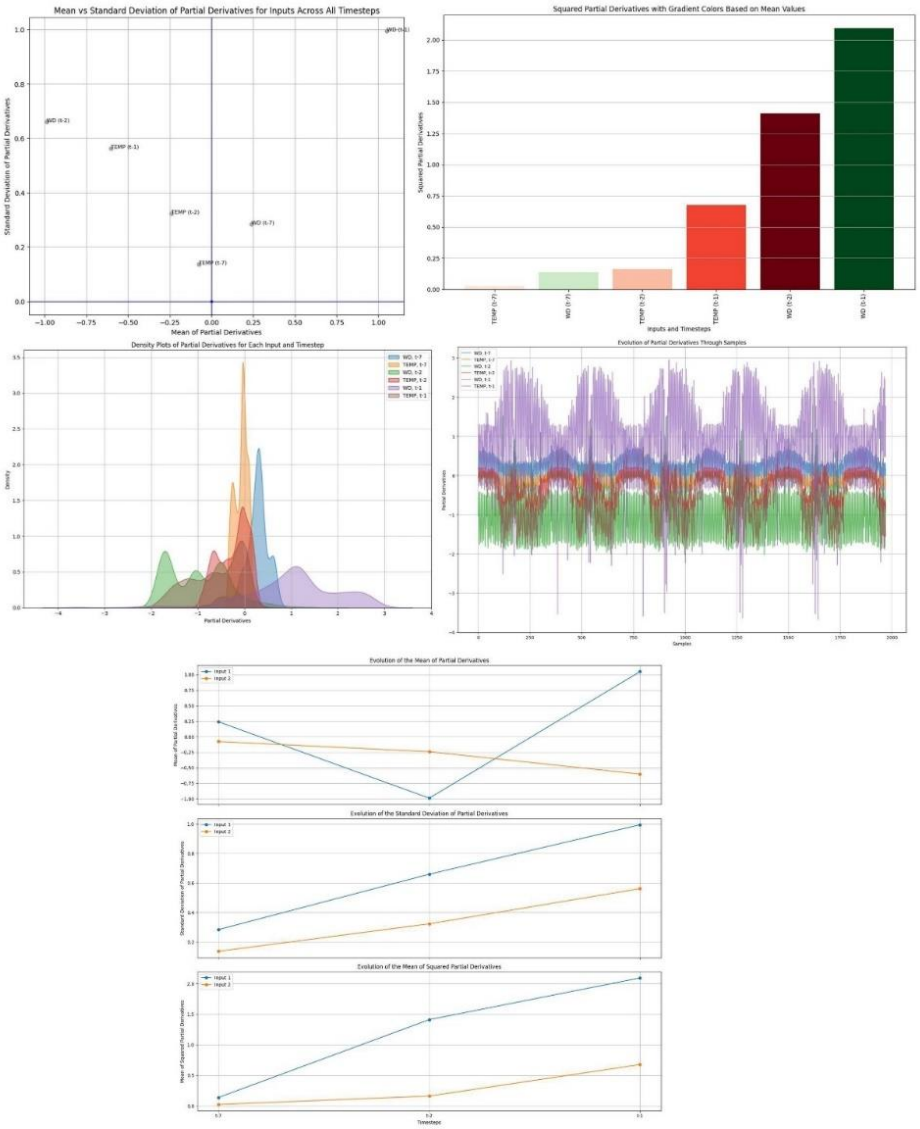

Figure 16:Sensitivity plots for the electric demand case

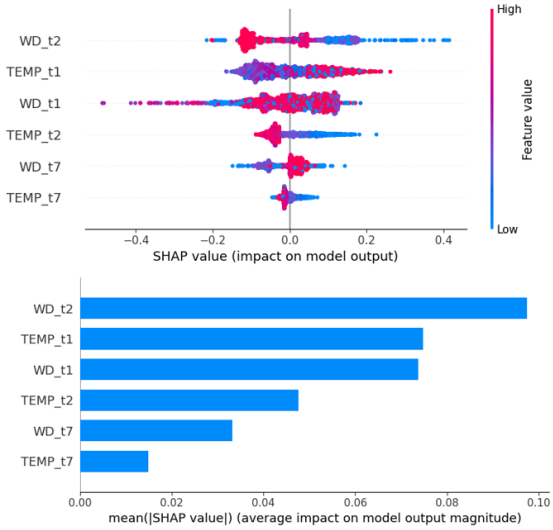

Figure 17: SHAP plots for the electric demand case

Continuing with the application of the XAI methods, Figure 18 show the PDP and ICE curves for the predictions of the electric demand. Regarding the WD metrics, WD_{t-1} shows a very small non-linear effects that could even be considered linear with a mild negative effect for some samples. WD_{t-2} on the other hand, shows a non-linear effect. Last WD_{t-7} shows a linear effect with very small slope, which means that the variation on the features' contribution is minimal. For the temperature metrics, $TEMP_{t-1}$ exhibits a negative non-linear effect, while $TEMP_{t-2}$ and $TEMP_{t-7}$ show a very mild negative linear effect.

However, it is important to mention that the feature importance attributed by the PDP and ICE curves differs considerably from the other methods describes. As shown in Figure 18, all the features exhibit similar importance, as all the curves range around the same value of 0.4.

Lastly, Permutation Importance presents a quick overview of feature significance. As shown in Figure 19, WD_{t-1} and $TEMP_{t-1}$ are the most relevant features and they exhibit the same importance. WD_{t-2} is the next most important feature, and the rest of the variables show very little significance in comparation with the other three variables. This feature importance analysis is similar to the sensitivity method and SHAP method but differs from the PDP and ICE curves.

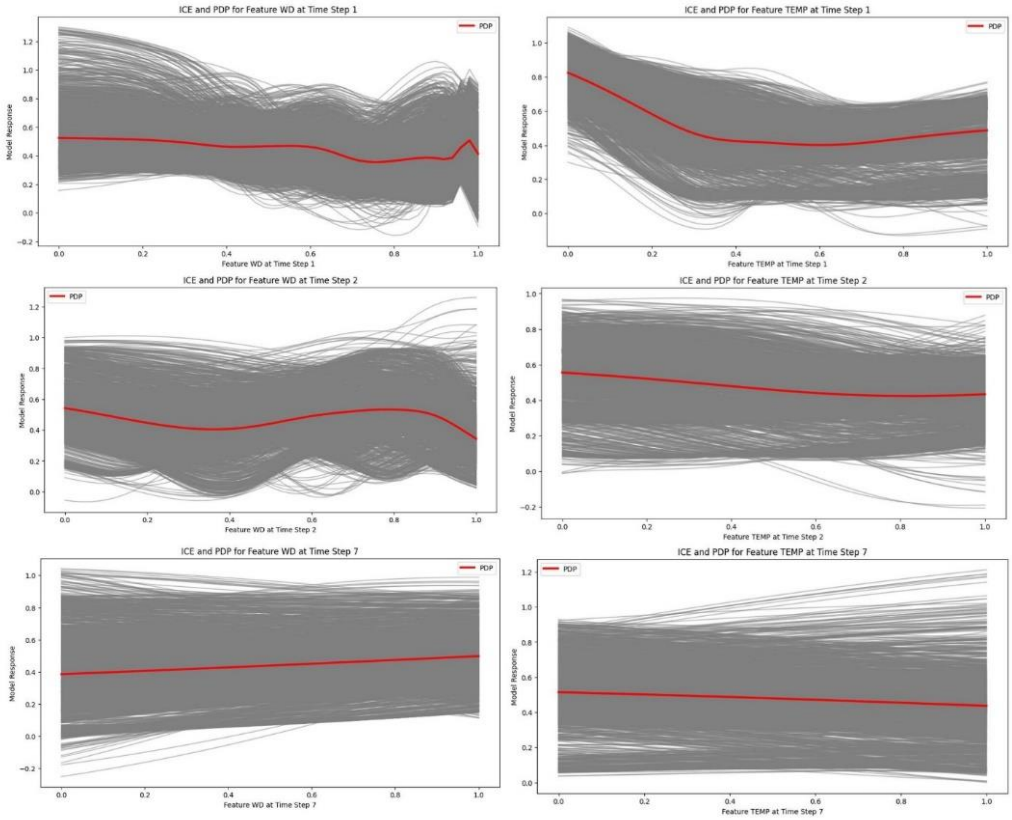

Figure 18: PDP and ICE curves for the electric demand case

```
{(0, 0): 0.027212113745875288,
 (0, 1): 0.01706275281698539,
 (1, 0): 0.027867198666592895,
 (1, 1): 0.007489294858045068,
 (2, 0): 0.007575904486082136,
 (2, 1): 0.002435004628550192}
```

Figure 19: Permutation Importance results for the electric demand case

4.4 Discussion of Results

In this chapter the sensitivity analysis method based on partial derivatives was applied to two different use cases, predicting the sine function and forecasting demand. In each case, besides applying the partial derivatives method, we also studied the explanations obtained by existing XAI methods.

In the first use case, the sensitivity analysis offered key insights into the importance of different time lags to predict the sine function and the effect of each lag in the model's output. While some of the other XAI methods show similar explanations regarding the importance of the time lags, sensitivity analysis provides the most complete and easy to interpret explanations. Moreover, the explanations provided are much more accurate than others such as LIME explanations, which offered a very poor prediction.

In the second use case, the partial derivative method unveiled the relationship between inputs and outputs found by the model. This relationship attributed more importance to certain time lags that are not the ones that are supposed to be in relation with the output. In this use case, the model attributed more importance to WD_{t-1} and WD_{t-2} instead of WD_{t-7} which should be the one bearing the most importance of all the time lags of that variable as it corresponds to the same day of the week before. This has to do with the correlation between the time lags and the original input variable, which induces the RNN to search for a relationship that predicts the data accurately but without being the true relationship.

Overall, the explanations offered by the sensitivity analysis method have the following advantages against the other XAI methods:

- The information offered is more complete than any other XAI method. Sensitivity analysis does not assume feature independence, providing more trustable explanations than the rest of the XAI methods.
- The graphs offered by our method ensure ease of interpretation and makes the sensitivity analysis understandable for every user.
- Sensitivity analysis is a more computationally efficient method, as it is much less memory intensive. This is crucial for datasets with higher number of samples and variables, as other methods such as LIME or SHAP, which require much more execution time than the sensitivity analysis method.
- The sensitivity analysis method is easy to implement in any workflow, as it does not require any particular version of python. SHAP on the other hand, required a very old and outdated version of python to function, which makes it hard to implement in state-of-the-art architectures.

With all these advantages, the sensitivity analysis method makes a significant contribution to the field of XAI, as it is a more accurate, complete, and easy to use and implement than any other existing method.

5 Conclusions

5.1 Summary and Conclusions

The aim of this project was the development and validation of sensitivity analysis on Recurrent Neural Networks using the partial derivatives of the output variables with respect to the input. This motivation comes from the pressing need of addressing the black box problem given the rapid development of artificial intelligence models. The opacity of artificial intelligence models often hinders their acceptance and trust, particularly in sensitive decision -making areas like finance or healthcare. This project therefore seeks to contribute to the development of Explainable Artificial Intelligence.

Besides the development of the sensitivity analysis, this project has also reviewed the most common XAI techniques that are being applied to RNNs. Methods such as LIME, SHAP, PDP curves, ICE curves and Permutation Importance have not only been revised from a theoretical point of view but has also been practically implemented to RNN models.

The development of the sensitivity analysis method consisted of two main steps. The first one being the analytical calculation of the partial derivatives in a Recurrent Neural Network and the selection of statistical measures. The theoretical framework was then implemented in the RNN_XAI package, which includes functions for the calculation of the Jacobian matrix of the RNN, computing the sensitivity measures, and providing several user-friendly plots to visualize the results. This implementation makes our method accessible and usable for practical applications, ensuring that it can be easily implemented into existing workflows.

The sensitivity analysis method was then validated and put into comparison with the other XAI methods via three different cases. A synthetic simple regression model with known derivatives was used to validate the partial derivatives calculated by our method, proving its correctness. Following this, the method was applied to two different use cases: predicting the sine function and forecasting electric demand. These two cases highlighted the ease of interpretation of our partial derivatives method and the more complete information provided. Moreover, our method was demonstrated to be much more computationally efficient, as well as being compatible with more recent versions of python.

Another key finding of this project was the tendency of RNNs to attribute importance to variables that are not related with the output of the dataset. This has to do with the correlation between the time lags used in RNNs and the original variables. However, it is important to make clear that XAI techniques do not try to validate nor correct the models, but instead, provide explanations for the predictions of a given model which is assumed to be correct.

Overall, this project has proven the superiority of the sensitivity analysis method on both quality of the explanations and ease of use over more traditional methods and hope to have achieved the goals of a more human-friendly, understandable and regulation compliant Artificial Intelligence.

5.2 Future developments

This Project has taken the developments in XAI for MLP and applied them successfully into the architecture of RNN. However, there are still some lines of research to be explored:

- Generalise the partial derivatives calculation to larger RNNs: This project has only focused on RNNs with a single recurrent hidden layer, but now that sensitivity analysis has been proven successful for this type of neural networks, the calculations could be extended to RNNs with more than recurrent layer. However, the number of terms present in the partial derivatives grow exponentially the more recurrent layer are included in the model, which could difficult the notation of a general formula that can be used for any number an arbitrary number of recurrent layers.
- Extend the sensitivity analysis method based on partial derivatives to other neural networks architecture: the partial derivatives method has already been proven successful in MLPs and RNNs but could still be applied to more complex networks such as LSTM or CNN.

Annex A: Calculation of Partial Derivatives with more than one recurrent layer

One of the future lines of research is extending the calculation of partial derivatives of RNNs with a single recurrent hidden layer to RNNs with an arbitrary number of recurrent hidden layers.

This project only focuses on RNNs with one single recurrent layer, as the aim of the project was to test the functionality of sensitivity analysis for these architectures. However, as a pure academic exercise, this section contains the calculation of the partial derivative of the output of one neuron in one layer, in one timestep, with respect to the output of the previous layer in the previous timestep, considering more than one recurrent layer.

$$\frac{\partial o_{k,t}^{l}}{\partial o_{k,t-1}^{l-1}} = \frac{\partial o_{k,t}^{l}}{\partial z_{k,t}^{l}} \cdot \frac{\partial z_{k,t}^{l}}{\partial o_{k,t-1}^{l-1}} = \frac{\partial \phi_{k}^{l}}{\partial z_{k,t}^{l}}\left(z_{k,t}^{l}\right) \cdot \left(\frac{\partial z_{k,t}^{l}}{\partial o_{k,t}^{l-1}} \cdot \frac{\partial o_{k,t}^{l-1}}{\partial o_{k,t-1}^{l-1}} + \frac{\partial z_{k,t}^{l}}{\partial o_{k,t-1}^{l}} \cdot \frac{\partial o_{k,t-1}^{l-1}}{\partial o_{k,t-1}^{l-1}}\right) =$$

$$= \frac{\partial \phi_{k}^{l}}{\partial z_{k,t}^{l}}\left(z_{k,t}^{l}\right) \cdot \left(w_{k}^{l} \cdot \frac{\partial o_{k,t}^{l-1}}{\partial z_{k,t}^{l-1}} \cdot \frac{\partial z_{k,t}^{l-1}}{\partial o_{k,t-1}^{l-1}} + u_{k}^{l} \cdot \frac{\partial o_{k,t-1}^{l}}{\partial z_{k,t-1}^{l}} \cdot \frac{\partial z_{k,t-1}^{l}}{\partial o_{k,t-1}^{l-1}}\right) =$$

$$= \frac{\partial \phi_{k}^{l}}{\partial z_{k,t}^{l}}\left(z_{k,t}^{l}\right) \cdot \left(w_{k}^{l} \cdot \frac{\partial \phi_{k}^{l-1}}{\partial z_{k,t}^{l-1}}\left(z_{k,t}^{l-1}\right) \cdot u_{k}^{l-1} + u_{k}^{l} \cdot \frac{\partial \phi_{k}^{l}}{\partial z_{k,t-1}^{l}}\left(z_{k,t-1}^{l-1}\right) \cdot w_{k}^{l}\right)$$

This calculation still needs to be extended to calculate the final output of the model with respect to the initial input, considering the desired number of recurrent layers.

6 Bibliography

[1] Abraham, A. (2005). Artificial neural networks. Handbook of measuring system design.

[2] Ali, S., Abuhmed, T., El-Sappagh, S., Muhammad, K., Alonso-Moral, J. M., Confalonieri, R., ... & Herrera, F. (2023). Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. Information Fusion, 99, 101805.

[3] Balestriero, R., & LeCun, Y. (2022). Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. Advances in Neural Information Processing Systems, 35, 26671-26685.

[4] Box, G. E. P., Jenkins, G. M., & Reinsel, G. C. (2015). Time Series Analysis: Forecasting and Control. John Wiley & Sons.

[5] Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

[6] Contreras, J., Espinola, R., Nogales, F. J., & Conejo, A. J. (2003). ARIMA models to predict next-day electricity prices. IEEE transactions on power systems, 18(3), 1014-1020.

[7] Deo, R. C. (2015). Machine learning in medicine. Circulation, 132(20), 1920-1930.

[8] Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.

[9] Du, M., Liu, N., & Hu, X. (2019). Techniques for interpretable machine learning. Communications of the ACM, 63(1), 68-77.

[10] Esteva, A., Chou, K., Yeung, S., Naik, N., Madani, A., Mottaghi, A., ... & Dean, J. (2019). Deep learning-enabled medical computer vision. NPJ Digital Medicine, 2, 38.

[11] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

[12] Garreau, D., & Luxburg, U. (2020, June). Explaining the explainer: A first theoretical analysis of LIME. In International conference on artificial intelligence and statistics (pp. 1287-1296). PMLR.

[13] Garrido, Q., Chen, Y., Bardes, A., Najman, L., & Lecun, Y. (2022). On the duality between contrastive and non-contrastive self-supervised learning. arXiv preprint arXiv:2206.02574.

[14] Goldstein, A., Kapelner, A., Bleich, J., & Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. journal of Computational and Graphical Statistics, 24(1), 44-65.

[15] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.

[16] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[17] Jacovi, Alon, Swabha Swayamdipta, Shauli Ravfogel, Yanai Elazar, Yejin Choi, and Yoav Goldberg. "Contrastive explanations for model interpretability." arXiv preprint arXiv:2103.01378 (2021).

[18] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[19] Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2020). Explainable ai: A review of machine learning interpretability methods. Entropy, 23(1), 18.

[20] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. Advances in neural information processing systems, 30.

[21] Madsen, A., Reddy, S., & Chandar, S. (2022). Post-hoc interpretability for neural nlp: A survey. ACM Computing Surveys, 55(8), 1-42.

[22] Mahesh, B. (2020). Machine learning algorithms-a review. International Journal of Science and Research (IJSR).[Internet], 9(1), 381-386.

[23] Menon, R. S., & Ranjan, R. (2022). The evolution of forecasting techniques: Traditional versus machine learning methods. genpact.
[24] Meske, C., Bunde, E., Schneider, J., & Gersch, M. (2022). Explainable artificial intelligence: objectives, stakeholders, and future research opportunities. Information Systems Management, 39(1), 53-63.
[25] Min, K., Kim, D., Park, J., & Huh, K. (2019). RNN-based path prediction of obstacle vehicles with deep ensemble. IEEE Transactions on Vehicular Technology, 68(10), 10252-10256.
[26] Molnar, C. (2020). Interpretable machine learning. Lulu. com.
[27] Ngai, E. W., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2009). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision Support Systems, 50(3), 559-569.
[28] Regulation, P. (2018). General data protection regulation. Intouch, 25, 1-5.
[29] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144).
[30] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological review, 65(6), 386.
[31] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533-536.
[32] Sarma, D., Mittra, T., & Hossain, M. S. (2021). Personalized book recommendation system using machine learning algorithm. International Journal of Advanced Computer Science and Applications, 12(1).
[33] Schwalbe, G., & Finzel, B. (2023). A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts. Data Mining and Knowledge Discovery, 1-59.
[34] Sherstinsky, A. (2020). Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena, 404, 132306.
[35] Tokgöz, A., & Ünal, G. (2018, May). A RNN based time series approach for forecasting turkish electricity load. In 2018 26th Signal processing and communications applications conference (SIU) (pp. 1-4). IEEE.
[36] Vagropoulos, S. I., Chouliaras, G. I., Kardakos, E. G., Simoglou, C. K., & Bakirtzis, A. G. (2016, April). Comparison of SARIMAX, SARIMA, modified SARIMA and ANN-based models for short-term PV generation forecasting. In 2016 IEEE international energy conference (ENERGYCON) (pp. 1-6). IEEE.
[37] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
[38] Veale, M., & Zuiderveen Borgesius, F. (2021). Demystifying the Draft EU Artificial Intelligence Act—Analysing the good, the bad, and the unclear elements of the proposed approach. Computer Law Review International, 22(4), 97-112.
[39] Yang, H., Ren, Z., Yuan, H., Xu, Z., & Zhou, J. (2023). Contrastive self-supervised representation learning without negative samples for multimodal human action recognition. Frontiers in Neuroscience, 17.
[40] Zafar, M. R., & Khan, N. (2021). Deterministic local interpretable model-agnostic explanations for stable explainability. Machine Learning and Knowledge Extraction, 3(3), 525-541.
[41] Zafar, M. R., & Khan, N. M. (2019). DLIME: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. arXiv preprint arXiv:1906.10263.