UNIVERSIDADE ESTADUAL DE CAMPINAS

Faculdade de Engenharia Elétrica e de Computação

Orlem Lima dos Santos

Application of Interpretability to Improve the Performance of an LSTM Classifier for Power System Event

Aplicação de Interpretabilidade para Melhorar o Desempenho de um Classificador LSTM para Eventos de Sistema de Potência

Campinas

2020

UNIVERSIDADE ESTADUAL DE CAMPINAS

Faculdade de Engenharia Elétrica e de Computação

Orlem Lima dos Santos

Application of Interpretability to Improve the Performance of an LSTM Classifier for Power System Event

Aplicação de Interpretabilidade para Melhorar o Desempenho de um Classificador LSTM para Eventos de Sistema de Potência

Dissertation presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Electrical Engineering, in the area of Electric Power.

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na área de Energia Elétrica

Supervisor: Prof. Dr. Daniel Dotta

Este trabalho corresponde à versão final da dissertação defendida pelo aluno Orlem Lima dos Santos, orientada pelo Prof. Dr. Daniel Dotta

Campinas

2020

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

Sa59a

Santos, Orlem Lima dos, 1991-
Application of interpretability to improve the performance of an LSTM classifier for power system events / Orlem Lima dos Santos. – Campinas, SP : [s.n.], 2020.

Orientador: Daniel Dotta.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Aprendizado de máquina. 2. Redes neurais profundas. 3. Teoria dos jogos. I. Dotta, Daniel, 1978-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Aplicação de interpretabilidade para melhorar o desempenho de um classificador LSTM para eventos de sistema de potência
Palavras-chave em inglês:
Machine learning
Deep neural networks
Game theory
Área de concentração: Energia Elétrica
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Daniel Dotta [Orientador]
Fernando José Von Zuben
Ildemar Cassana Decker
Data de defesa: 31-08-2020
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-3942-6418
- Currículo Lattes do autor: http://lattes.cnpq.br/8462551910761953

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato(a): Orlem Lima dos Santos RA: 211501
Data de defensa: 31 de agosto de 2020
Título da Dissertação: "Application of Interpretability to Improve the Performance of an LSTM Classifier for Power System Events"
Título em Português: "Aplicação de Interpretabilidade para Melhorar o Desempenho de um Classificador LSTM para Eventos de Sistema de Potência"

Prof. Dr. Daniel Dotta (Presidente)
Prof. Dr. Fernando José Von Zuben.
Prof. Dr. Ildemar Cassana Decker.

A Ata de Defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de Pós-Graduação da Faculdade de Engenharia Elétrica e de Computação.

To my family for their support and love.

Acknowledgements

First, I would like to thank God and Jesus Christ, for giving his life for us. I am very grateful to my mother Irenildes Lima dos Santos, my father Orlem Matias dos Santos, my wife Tais Barros, and my siblings Orlenildes, Ilem, and Iolanda for their care, love and support.

I would like to express my deep and sincere gratitude to my advisor, Dr. Daniel Dotta, for giving me the opportunity to be his master student, and patience in discussions and debates with me.

To the university of UNICAMP, for the great public infrastructure capable of providing and encouraging the science in Brazil.

To the university Federal University of Santa Catarina (UFSC), especially Prof. Dr. Ildemar Cassana Decker, Marco Antonio Delgado Zarzosa and MedFasee Project for the dataset of power system events.

To the university of Rensselaer Polytechnic Institute (RPI), especially Prof. Meng Wang and Prof. Joe H. Chow for the great learning obtained in research internship in the RPI.

This study was financed in part by the grants 2017/25425-5, 2019/08200-5, 2016/08645-9, 2018/20104-9, 2019/10033-0, São Paulo Research Foundation (FAPESP). Also, this study was financed in part by CAPES, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, and CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico - Brasil.

“Ever tried. Ever failed.
No matter.
Try Again. Fail again
Fail better.
(Samuel Beckett)

Abstract

Nowadays, vast amounts of data are collected by Wide Area Measurement Systems (WAMS). Therefore, there is an obvious necessity for Machine Learning (ML) methods, as useful knowledge to extract relevant and reliable information from this synchrophasor data. Among the ML approaches, the Deep Neural Network (DNN) models provide an important opportunity to advance direct learning from the data, making these approaches independent from feature extraction techniques. However, these deep models produce *black-box* classifiers that can be matter of concern when applying to high-risk environment (critical infrastructure) such as the EPS (Electric Power Systems). In this work, the application of an explainable data-driven method is carried out in order to inspect the performance of DNN classifier for event identification using synchrophasor measurements. The DNN classifier is a Long-Short Term Memory (LSTM) with positive performance in the extraction of dynamic features. The principal benefit of this approach is the use of an interpretability inspection named SHAP (SHapley Additive exPlanation) values, which are based on cooperative game theory (Shapley values). These SHAP values provide the means to evaluate the predictions of the LSTM, highlight the parts of the input time-series with the most contribution to the identification of the events, and detect possible bias. Moreover, by employing the SHAP inspection along with domain knowledge of the problem, the performance and coherence of the LSTM classifier will be improved by choosing the classifier that not only has highest Identification Accuracy Rate (IAR) but is also coherent with domain knowledge of the problem, minimizing detected bias. The application of this interpretable approach is desirable because: *i)* it explains how the LSTM classifier is making its decisions; *ii)* it helps the designer to improve the training of the classifier; *iii)* it certifies that the resulting classifier has a consistent and coherent performance according to domain knowledge of the problem; *iv)* it clearly reduces the concerns of the application of DNN methods in a critical infrastructure, in the cases that the user understands that the classifier is taking coherent decisions. The proposed method has been evaluated using real synchrophasor event records from the Brazilian Interconnected Power System (BIPS).

Keywords: Interpretability. SHAP. Shapley Values. PMU. WAMS. LSTM.

Resumo

Atualmente, uma grande quantidade de dados é coletada pelos WAMS (*Wide Area Measurement Systems*). Portanto, existe uma clara necessidade de métodos de aprendizagem de máquina (ML - *Machine Learning*), capazes de extrair informações relevantes e confiáveis dos dados de sincrofasores. Entre as abordagens de ML, os modelos de Rede Neural Profunda (DNN - *Deep Neural Network*) têm a vantagem de aprender diretamente com os dados, tornando essas abordagens não dependentes das técnicas de extração de atributos. No entanto, esses modelos profundos produzem classificadores caixa-preta (*black-box*) que podem suscitar preocupações quando aplicados a ambientes de alto risco (infraestrutura crítica), como o sistema elétrico de potência (EPS-*Electric Power Systems*). Neste trabalho, a aplicação de um método orientado a dados (*data-driven*) explicável é realizada a fim de inspecionar o desempenho do classificador DNN para identificação de eventos usando medições de sincrofasores. O classificador DNN é uma LSTM (*Long-Short Term Memory*) que tem demostrado bom desempenho na extração de características dinâmicas. A principal vantagem dessa abordagem é o uso de uma inspeção baseada em interpretabilidade denominada SHAP (*SHapley Additive exPlanation*), que é baseada na teoria dos jogos cooperativos (valores Shapley), que fornece os meios para avaliar as previsões da LSTM, destacando as partes das séries temporais de entrada que mais contribuíram para a identificação dos eventos e detecção de possíveis vieses. Além disso, usando a inspeção SHAP juntamente com o conhecimento de domínio (*domain knowledge*) sobre o problema, o desempenho e a coerência do classificador LSTM são aprimorados ao escolher o classificador que não apenas possui a maior acurácia de identificação (IAR - *Identification Accuracy Rate*), mas também é coerente com o conhecimento de domínio do problema, minimizando possíveis vieses detectados. O uso dessa abordagem interpretável é útil porque: *i)* explica como o classificador LSTM está tomando suas decisões; *ii)* ajuda o designer a melhorar o treinamento do classificador; *iii)* certifica que o classificador resultante tem um desempenho consistente e coerente de acordo com o conhecimento do domínio; *iv)* quando o usuário entende que o classificador está tomando decisões coerentes, reduz claramente as preocupações da aplicação dos métodos DNN em uma infraestrutura crítica. O método proposto é avaliado usando registros reais de eventos sincrofasores do Sistema Interligado Nacional (SIN).

Palavras-chave: Interpretabilidade. SHAP. Valores Shapley. PMU. WAMS. LSTM.

List of Figures

List of Tables

List of abbreviations and acronyms

LSTM	Long-Short Term Memory
IAR	Identification Accuracy Rate
BA	Balanced Accuracy
TP	True Positive number
FN	False Negative number
FP	False Positive number
TN	True Negative number
TPR	True Positive Rate
TNR	True Negative Rate
EPS	Electrical Power System
SVM	Support Vector Machine
MSVM	Multi-Class Support Vector Machine
RBF	Radial Basis Function (RBF)
MLP	Multi-Layer Perceptron
CNN	Convolutional Neural Network
SHAP	SHapley Additive exPlanation
DeepLIFT	Deep Learning Important FeaTures
NN	Neural Network
WAMS	Wide Area Measurement System
BIPS	Brazilian Interconnected Power System
SIN	Sistema Interligado Nacional
PMU	Phasor Measurement Unit
SCADA	Supervisory Control and Data Acquisition

PDC	Phasor Data Concentrator
ROCOF	Rate Of Change of Frequency
LSTM-SHAP	LSTM classifier combined with SHAP inspection
GT	Generation Trip event type
LS	Load Shedding event type
LT	Line Trip event type
OS	Oscillation event type

List of symbols

P(GT)	Output probability of LSTM for GT event
P(LS)	Output probability of LSTM for LS event
P(LT)	Output probability of LSTM for LT event
P(OS)	Output probability of LSTM for OS event
$\boldsymbol{x}^{(t)}$	Input of the LSTM in the time-step t
ϕ_j	SHAP values contribution of feature x_j
$\phi_j^{(t)}$	SHAP values contribution of sample j in the time-step t using LSTM
$\Phi^{(t)}$	SHAP values contribution of the time-step t of the LSTM
$E[f(x)]$	Base value of SHAP values
$E[f(x)]_{\mathrm{GT}}$	Base value of LSTM classifier for the GT event
$E[f(x)]_{\mathrm{LS}}$	Base value of LSTM classifier for the LS event
$E[f(x)]_{\mathrm{LT}}$	Base value of LSTM classifier for the LT event
$E[f(x)]_{\mathrm{OS}}$	Base value of LSTM classifier for the OS event
$E[f_\phi(x)]_{\mathrm{GT}}$	Base value of LSTM-SHAP classifier for the GT event
$E[f_\phi(x)]_{\mathrm{LS}}$	Base value of LSTM-SHAP classifier for the LS event
$E[f_\phi(x)]_{\mathrm{LT}}$	Base value of LSTM-SHAP classifier for the LT event
$E[f_\phi(x)]_{\mathrm{OS}}$	Base value of LSTM-SHAP classifier for the OS event
$E[x_j]$	Base value of feature x_j in SHAP values
$f_x(z') = f(h_x(z')) = E[f(z)\|z_S]$	Conditional expectations of SHAP values
h_x	Function that maps the coalition z' (binary) to a valid instance z
η	Learning rate of the LSTM
γ	decay parameter in learning rate η
α	Momentum parameter of the Batch Normalization

Contents

1 Introduction

1.1 Motivation and Literature Review

The Wide Area Measurement System (WAMS) is capable of providing simultaneous measurements of voltage and current phasors, known as synchrophasors. These WAMS networks capture and store a large amount of data that must be analyzed in order to extract relevant and reliable information about the power system performance. Therefore, there is a necessity to explore algorithms of data science, such as artificial Neural Network (NN), since they may allow fast and efficient extraction of significant information about the EPS (Electric Power Systems).

In the previous studies, the majority of the approaches have explored the application of machine learning techniques using feature extraction for event identification. Rafferty et al. (2016) proposed method for event detection and identification in real-time based on a moving window Principal Components Analysis (PCA). The authors were able to precisely detect and classify generation loss, load shedding, and islanding events using real synchrophasors data from the U.K. power system. In (JENA; PANIGRAHI; SAMANTARAY, 2018), the authors proposed an event identification method that performs a postmortem analysis on real synchrophasors data of the Indian power system. The method combines an Empirical Wavelet Transform (EWT) for feature extraction with a random forest classifier. In (YADAV; PRADHAN; KAMWA, 2018), a real-time event detection and classification were performed using a signal energy transformation. The detection was realized with the Teager-Kaiser Energy Operator (TKEO) and the identification was noticed using an Energy Similarity Measure (ESM), for feature extraction with a 1-nearest neighbor classifier. The authors were able to accurately detect and classify real synchrophasors events of the Indian grid.

Recently, Deep Neural Network (DNN) models can take advantage of representation learning, i.e., learning representations of the data that makes it easier to extract useful information when building classifiers or other predictors (BENGIO; COURVILLE; VINCENT, 2013). Therefore, by using this technique other representations can be learned directly from the data. As a result, the DNN models are less dependent on feature engineering (BENGIO; COURVILLE; VINCENT, 2013). In these DNN models, the representations are formed by the composition of a cascade of multiple non-linear transformations to yield more abstract and more useful representations (BENGIO; COURVILLE; VINCENT, 2013). In a recent paper (LI; DU, 2018), the authors emphasized the potential of DNN models to solve problems in different power system areas. One of these works is (MIRANDA et al., 2019), where the authors presented an interesting idea about application of Convolutional

Neural Network (CNN) for event identification using postmortem analysis without feature extraction methods by relaying on a DNN model, and taking advantage of the representation learning. Besides, the authors in Li e Wang (2019) proposed an identification of successive events using a CNN for classification. Also, the authors proposed a method to train on the extracted dominant eigenvalues of the dynamical system and the singular values of the data matrix instead of direct measurements of time-series.

One of the most relevant DNN models is the Long-Short Term Memory (LSTM), which is a special Recurrent Neural Network (RNN), and has been performing well in the extraction of features of time-series capable of solving long-term dependencies (HOCHREITER; SCHMIDHUBER, 1997). This recurrent model has shown superior performance in many research fields such as image captioning (MOU; GHAMISI; ZHU, 2017), NLP (Natural Language Processing) (YOUNG et al., 2018), involving a series of dynamic problems like text classification (ZUO et al., 2019), machine translation (SUTSKEVER; VINYALS; LE, 2014), and speech recognition (ZAZO et al., 2018). This superior performance in these dynamic problems is due to the recurrent nature of the LSTM and its great capability of learning time-series patterns (HOCHREITER; SCHMIDHUBER, 1997).

The LSTM has been applied to many power systems to solve problems such as:

- The real-time identification of power fluctuations (WEN et al., 2019), estimating the power fluctuations from real-time frequency signal using the LSTM;
- The short-term residential load forecasting (KONG et al., 2017), comparing the LSTM with some state-of-the-arts models in load forecasting;
- The wind power forecasting (YU et al., 2019), forecasting using the LSTM and extracting sequential correlation feature;
- The detection non-technical losses (CHATTERJEE et al., 2017), detecting the irregularities in power usages using the LSTM;
- The power disaggregation, extracting the target power signal of the appliance or sub-circuit based on a supervised trained LSTM.

Regarding the event identification, the authors in Zhang et al. (2017) proposed a method for line trip fault prediction in power systems using the LSTM and a Support Vector Machine (SVM). The authors proved that LSTM is suitable for extracting the features of numerous time-series in an EPS application. In (WEN et al., 2019), the LSTM is employed for real-time identification of power fluctuations, showing that these fluctuations could be accurately identified using the LSTM.

In all these studies, the performance evaluation is limited to the Identification Accuracy Rate (IAR), which is the ratio of the number of correctly classified examples to the total number of examples, of the resulting *black-box* classifier for a specified dataset.

These deep models lack the interpretability of their predictions in the sense that is not clear how and why they arrive at a particular prediction (MONTAVON et al., 2017). This is further aggravated by the cascade of non-linear transformations, which is becoming deeper and deeper in recent models. The lack of knowledge about the way the classifier performs the identification can raise concerns for high-risk environments (critical infrastructure) such as the EPS, and especially in event identification when controlling actions are taken after the event identification. In the cases with low reliability, the actions could face serious consequences in the EPS. In addition, when the classifier fails, the designer will not have a clear direction to improve the classifier performance. Even though the classifier is retrained and incorporated the fail events in the dataset, there is no indication that the problem was solved. This can be a clear barrier for the application of DNN methods in power systems.

To overcome this problem, we have proposed to use an interpretability technique called SHAP (SHapley Additive exPlanation) values (LUNDBERG; LEE, 2017), which is based on the game theory method named Shapley value (MOLNAR, 2019), to understand if the classifier takes its decisions in a reasonable and coherent way according to the domain knowledge of the power system events problem. The domain knowledge of power system events is defined as the main characteristic of the awareness of power system events. To reach this goal, we have explored a deep LSTM network (without the assistance of feature extraction techniques) for the identification of events in practical power systems. The performance evaluation is carried out by the IAR and the interpretability inspection using SHAP values. This technique allows us to identify what is really important in the input highlighting the parts of the time-series with the most contribution to the identification of each event type. Furthermore, this interpretability technique can also provide a better understanding of how a DNN can be trained. This helps the designer to improve the training process of the classifier.

1.2 Problem Statement and Our Approach

With the great amount of PMU (Phasor Measurement Unit) data in the control centers the main challenge is to explore the potential of data science techniques capable of extracting relevant information (correlations) that is not directly captured by the power system engineer. For instance, in the event identification problem, the power system operators need something more than a black-box classifier that just attribute a label to a specific event such as: Generation Tripping (GT), Loading Shedding (LS), Oscillation (OS) and Line Tripping (LT), and Islanding. It is also useful to know why and how the classifier is taking its decisions.

To achieve this goal, we have proposed an approach based on time-series by means of an LSTM network. For instance, we have 10 seconds frequency record time-series classified as OS event by the LSTM, and sub-divided into six time-steps $\boldsymbol{x}^{(t)}, t = 1, \ldots, 6$. The proposed approach is described in Figure 1. Therefore, using the SHAP inspection the magnitude of the contribution $\Phi^{(t)}$ for each time step $t = 1, \ldots, 6$ can be estimated separately. Finally, by evaluating the size of the magnitudes of the contributions, a ranking for contributions is built that helps to evaluate that which ones are more important (or not) to the identification of this event.

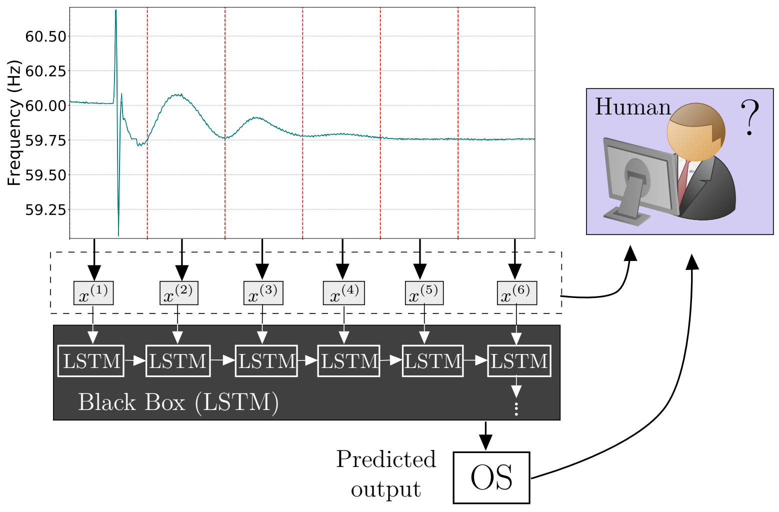

Frequency (Hz)
60.50
60.25
60.00
59.75
59.50
59.25
$x^{(1)}$ $x^{(2)}$ $x^{(3)}$ $x^{(4)}$ $x^{(5)}$ $x^{(6)}$
LSTM LSTM LSTM LSTM LSTM LSTM
Black Box (LSTM)
Human ?
Predicted output
OS

Figure 1 – Event Identification based on Usual LSTM Data-driven Method (Black-Box).

Realizing how the event was classified is mainly important to detect possible bias, patterns that are not according to domain knowledge of the events and inconsistencies. The application of SHAP inspection in EPS applications can bring both knowledge and understanding about EPS operation. Thus, the main focus of this work is to propose an explainable data-driven classifier, named LSTM-SHAP, presented in Figure 2b, that is re-trained after the SHAP inspection of the traditional LSTM classifier.

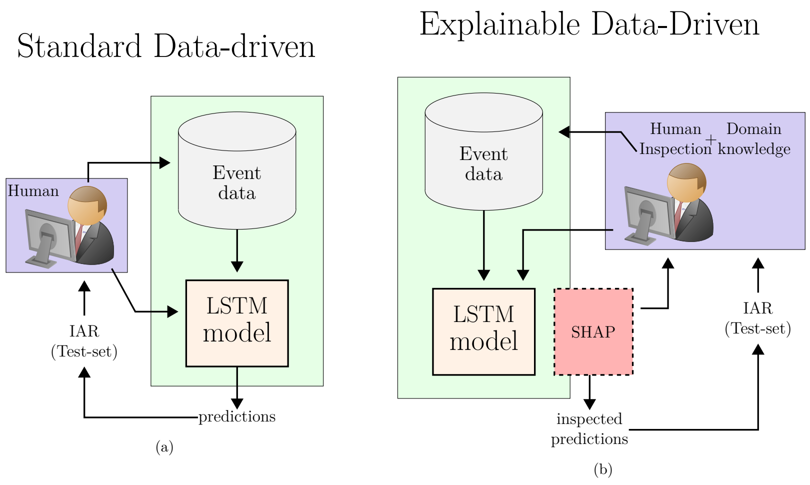


Figure 2 – Our approach using LSTM with SHAP Inspection compared with standard data-driven. (a). Standard Data-driven approach, trained focusing only to obtain the highest IAR of the Test-set. (b) Explainable Data-driven approach (LSTM-SHAP).

The LSTM is only trained focusing to obtain the highest IAR of the Test-set, following a standard data-driven approach (Figure 2a). The LSTM-SHAP will incorporate the obtained knowledge through the SHAP inspection, to correct bias and inconsistencies.

1.3 Objectives

The dissertation attempts to propose a methodology for the identification of events using an explainable LSTM classifier, based on the SHAP inspection, providing coherent decision-making processes. These decision-making processes must be in accordance with domain knowledge of the power system events. To achieve this goal, the following partial objectives are proposed:

- To explore the capabilities of the LSTM for classification of events;
- To compare the identification performance of the LSTM with some known methods in the literature, Multi-Class SVM (MSVM) and MLP (Multi-Layer Perceptron);
- To understand how the LSTM classifier is making its decisions in the identification of the events using the SHAP inspection;
- To locate the parts of the input time-series with the highest contributions $\Phi^{(t)}$ to the classification of the events;
- To identify relationships between the input $\boldsymbol{x}^{(t)}$ and the contributions $\Phi^{(t)}$;
- To detect possible bias and inconsistencies of the LSTM, based on the domain knowledge of events;
- To propose improvements of data and/or LSTM based on the knowledge extracted from the SHAP inspection, to train a new classifier LSTM-SHAP.

1.4 Contributions

In summary, the main contributions of the explainable data-driven approach are as follows:

- Identification and location of the parts of the input time-series with the most contribution to the classification of the events;
- Application of SHAP inspection for time-series analysis using the LSTM;
- Application of SHAP values in a high-risk environment (critical infrastructure) such as EPS in event identification;

- The performance of LSTM classifier is carried out using both the IAR and the interpretability inspection. Accordingly, confirming if the LSTM classifier makes reasonable and coherent decisions. This inspection may help us to reduce the concerns about the use of DNN as a tool for power system operation.

1.5 Thesis Outline

The overall structure of the thesis takes the form of seven chapters, including:

In Chapter 2: The dataset of Brazilian Interconnected Power System (BIPS) events records is described. Moreover, a description of an event detection, segmentation and identification applied to the BIPS is presented.

In Chapter 3: The design of the LSTM classifier is presented. The proposed LSTM classifier is formulated along with the definitions and theoretical background for the LSTM.

In Chapter 4: The formulation of SHAP values is presented, showing the fundamentals of game theory, based on Shapley values method. Also, the specific method used for NN, known as DeepSHAP, is presented along with mathematical formulations of the SHAP values computation.

In Chapter 5: The proposed methodology is detailed with putting all the concepts together to show how to apply the SHAP inspection on the LSTM for the event identification problem. Besides, the performance indices Balanced Accuracy (BA) and IAR for event identification, and the background dataset are defined.

In Chapter 6: The performance of classification is assessed using the real events records from BIPS. The identification performance of IAR of LSTM is compared with some known methods in the literature (MSVM and MLP). In addition, two classifiers LSTM and LSTM-SHAP are inspected. The LSTM is the trained classifier focusing only on the IAR alone, and LSTM-SHAP is the classifier obtaining with the improvements in data and/or LSTM model through the SHAP inspection.

Finally, conclusions and future works are presented in Chapter 7.

2 Identification of Power System Events Using Synchrophasors

2.1 Introduction

The objective of this chapter is to review the main concepts involved in the identification of power system events using synchrophasors measurements. Initially, the WAMS network and the dataset of events collected by the MedFasee Project are described. Only the frequency measurements are used due to the high coupling between these signals in high and low voltage level (DECKER et al., 2011).

The detection and segmentation steps used in processing of the events are described. The detection step is important to know the instant of the beginning of the event, and the segmentation step is used to cluster the PMUs signals when a single recording contains more than one event (BYKHOVSKY; CHOW, 2003). Also, the formal definitions of power system events are presented.

The chapter has been organized in the following way:

- Section 2.2 describes WAMS technology including the main parts such as PMUs, Phasor Data Concentrator (PDC), and communication channels;
- Section 2.3 describes the MedFasee Project;
- Section 2.4 describes the power system event definition and the applied methods for detection, and identification. Also, it describes the most relevant monitored events in the BIPS;
- Section 2.5 describes the dataset of events. Finally, Section 2.6 contains the conclusions of the chapter.

2.2 Wide Area Measurement Systems

The WAMS technology enables synchronized signals, measured at remote locations available in a control center (DECKER et al., 2006). The WAMS networks are composed of PMUs, synchronized in time by a signal of high precision, and connected to a Phasor Data Concentrator (PDC) through communication channels. Such systems can operate at 60 synchrophasor/second, well above the rates used by the SCADA (Supervisory Control and Data Acquisition) system. The main constituent elements of a WAMS network are:

- Phasor Measurement Unit (PMU) (Figure 3): The device that estimates synchronous phasor, frequency, the Rate Of Change of Frequency (ROCOF) based on voltages and/or currents and the temporal synchronization, obtained by the receiving equipment of GPS signal (MARTIN et al., 1998).

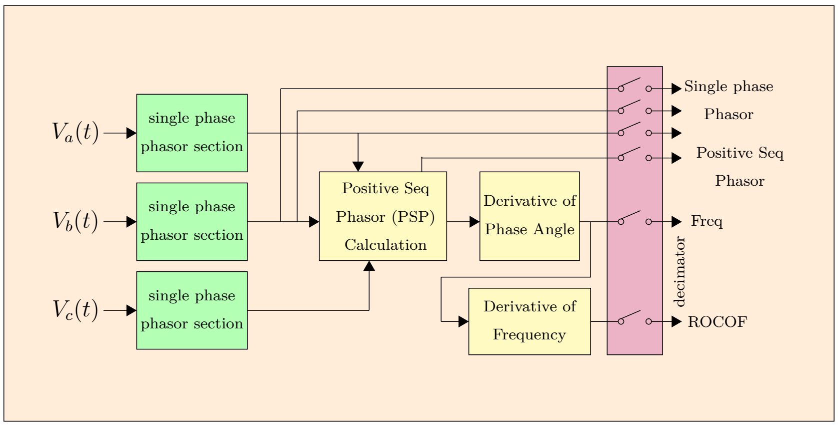

$V_a(t)$
$V_b(t)$
$V_c(t)$
single phase phasor section
single phase phasor section
single phase phasor section
Positive Seq Phasor (PSP) Calculation
Derivative of Phase Angle
Derivative of Frequency
decimator
Single phase Phasor
Positive Seq Phasor
Freq
ROCOF

Figure 3 – Phasor Measurement Unit (PMU).

- Phasor Data Concentrator (PDC): The data concentrator is intended to receive the synchrophasors sent by the PMUs, verify any transmission errors, and organize and make available data for other applications (Figure 4).

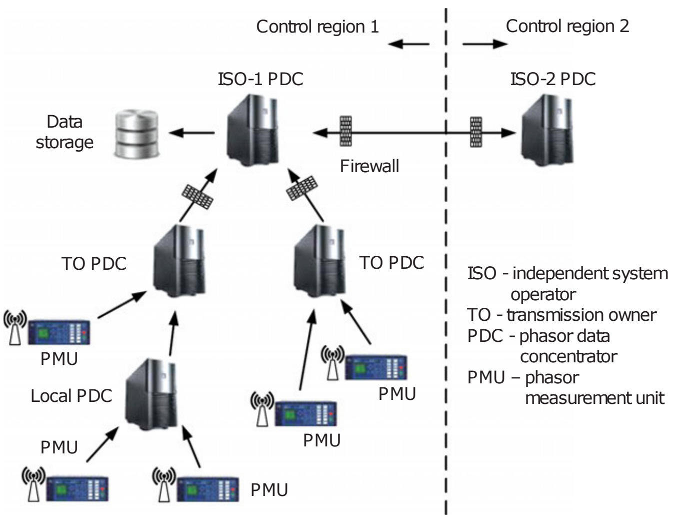


Figure 4 – Structure of a Synchrophasorial Network.

- Communication channels: The communication channels have the well-defined goal of making data transfer possible between the PMUs and the PDC, as well allowing exchange of information between PDCs of different areas. They can be used as connection systems between PMUs and PDCs, fiber optic links, microwave channels, Power Line Communication (PLC), modem system and even the internet itself with the Virtual Private Network (VPN) system. The most commonly used communication protocols for communication are TCP/IP and UDP/IP. The selection of the communication system to be used is directly linked to the application being considered. Furthermore, the monitoring applications do not require the same transfer rate and security of receipt required by a control application.

2.3 MedFasee Project

The MedFasee Project, led by UFSC (Federal University of Santa Catarina), illustrated in Figure 5, is composed of 26 PMUs installed in the low-voltage level at the universities. The PDCs of Medfasee are installed at the university campus of UFSC, and more recently at UNICAMP. The MedFasee network provides the synchrophasors (absolute voltage value, angle, and frequency) of a three-phase system from the universities monitored by the PMUs.

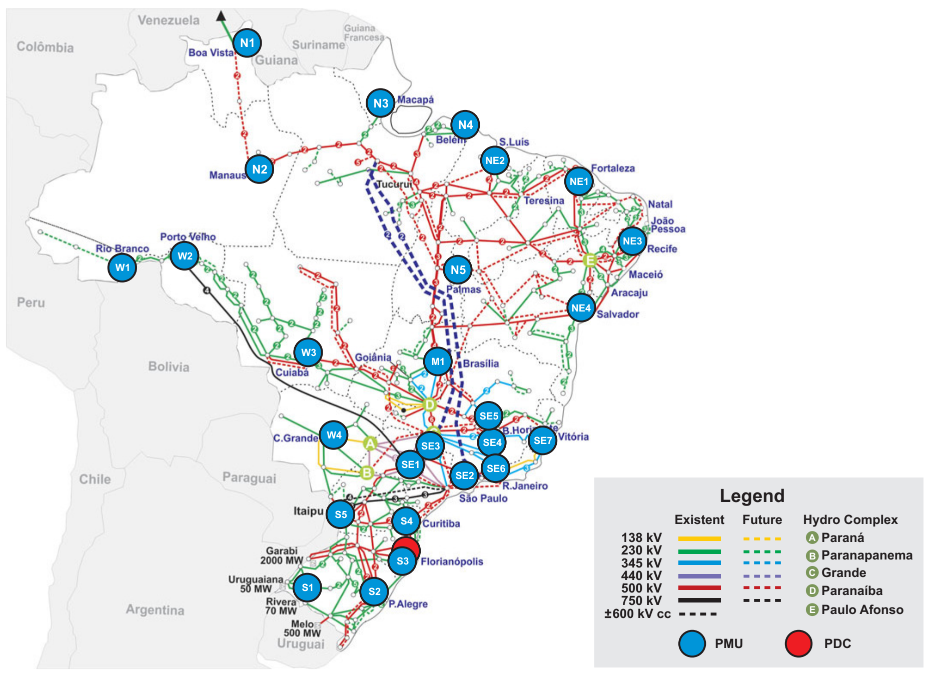


Figure 5 – Medfasee Project.

The MedFasee is a prototype used to collect the dataset of events using the history of events that have been stored since 2010. The project also provides real-time measurements that can be applied for the detection, and identification of the events.

2.4 Power System Events

As presented in Decker et al. (2011), the frequency of the EPS plays an important role in the detection, and identification of events. The frequency estimated by WAMS using synchrophasors is a voltage parameter, obtained using

$$f = \frac{1}{2\pi}\frac{d\theta}{dt} \tag{2.1}$$

where θ is the angle measured by the PMUs. This frequency is subjected to angle variations that happen during the events. These variations cause spikes on the frequency signal. These spikes are important to detect the events, and are a meaningful characteristic of Line Tripping (LT) events.

In this work, according to the terms described in Zimmer et al. (2013), Zarzosa et al. (2016), the term **event** refers to any disturbance that occurred in the EPS, electromagnetic or electromechanical, no matter what the dimension of the impact is. The term **perturbation** is considered as a set of events that characterize a large disturbance with a great impact on the EPS (ZIMMER et al., 2013). These terms were defined in grid codes developed by National Power System Operator (ONS - *Operador Nacional do Sistema Elétrico*) (ONS, 2020). However, in the international literature, there is a different formality. Usually, the term perturbation is defined as multiple-event (WANG et al., 2014; SONG et al., 2017; WEN et al., 2019). Also, the multiple events term can be sub-divided into simultaneous events when the events occurred within a very short time span (SONG et al., 2017), and cascading, sequential or successive events when the instants of beginning between the events are a bit longer (WANG et al., 2014; WEN et al., 2019). Therefore, the term used for a large disturbance with multiple-events in this work will be multiple events to be in accordance with the previous papers.

As described in Zimmer et al. (2013), the source of events are generally switching, short-circuit, and equipment defects. The small events usually have local impact triggering a few PMUs signals, referred to as local events. Large events, on the other hand alter the condition of operation causing imbalances between load and generation and affecting several areas of the EPS, referred to as systematic events. Generally, in a large disturbance the events happen simultaneously and a segmentation technique is required to separate them for proper analysis (BYKHOVSKY; CHOW, 2003).

Four types of events are considered: Loading Shedding (LS), Generation Tripping (GT), Line Tripping (LT), and Oscillation (OS) that will be presented as follows.

2.4.1 Monitored Events in the BIPS

According to Zarzosa, Zimmer e Decker (2016), the LS and GT events are defined as systematic events because in general they have a large systematic impact on the EPS triggering a large amount of PMUs. Furthermore, LT and OS are generally defined as local events since they trigger a small amount of PMUs.

2.4.1.1 Load Shedding (LS)

The Load Shedding event, also known as Load Trip, is the process of shutting down some loads to actuate protection of substation circuits that energize some cities and industries. This may be a systematic event if a large amount of load is suddenly disconnected. The main characteristic of this event is an increase in the system frequency, since the generators are producing more energy than the system can consume. One example of LS event occurred in BIPS is presented in Figure 6. In this case, the frequency reached 60.16Hz stabilizing in 60.03Hz.

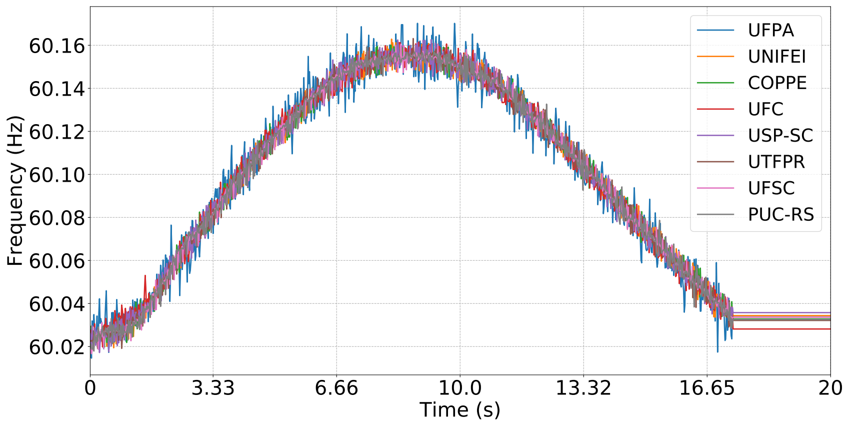


Figure 6 – Load Shedding Event.

2.4.1.2 Generation Tripping (GT)

The sign of Generation Tripping event is a frequency drop in the whole interconnection system. In this event the loads of energy exceed the generations, leading the generators to speed up. One example of GT event that happened in the BIPS is presented in Figure 7. This event is also a systematic event with a great impact in the system. In the Figure 7, the system frequency dropped to 59.66Hz stabilizing in 59.9Hz.

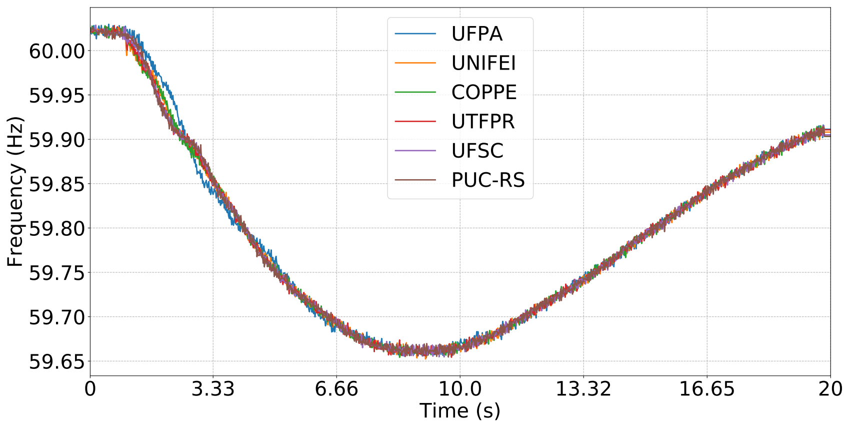


Figure 7 – Generation Tripping Event.

2.4.1.3 Line Tripping (LT)

The Line Tripping (LT) event is a kind of switching of network topology which refers to opening and closing of transmission line. The sign of LT local event is abrupt variations (spike) in the frequency followed by little oscillations between inter-areas. After the spike, the frequency can increase, decrease or stay in steady state, depending on the post-fault action. Figure 8 which refers to a transmission LT event, illustrates the characteristic of the frequency during this type of event. This kind of LT event the steady state frequency after the spike is the most usual type of LT, however, the other kinds of behavior (increase and decrease) can happen in simultaneous events.

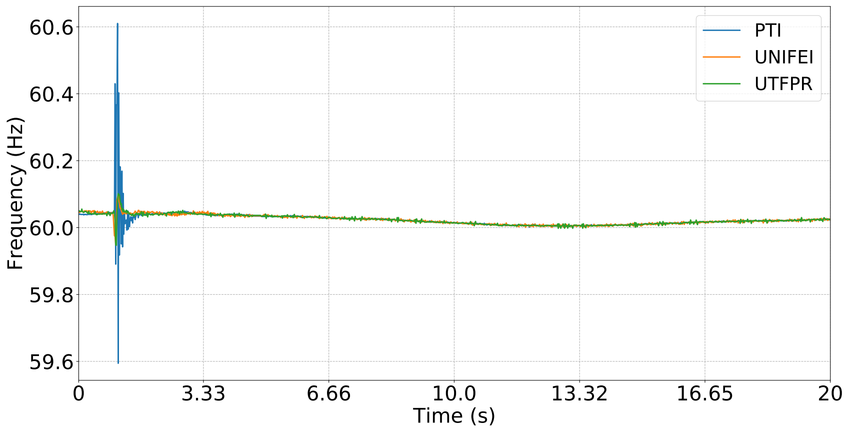


Figure 8 – Line Tripping Event.

2.4.1.4 Oscillation (OS)

The Oscillations events usually have their origin in the electromechanical oscillations of system machines. The main characteristic of these is the oscillation peaks (maximum and minimum) soon after the spike. These oscillations can be intra-plant (same power plant), same area (local) and inter-areas (different locations). The shutdown of a DC link can cause oscillations. As a matter of fact, most of the registered events of this kind in the BIPS happened on HVDC link (600kV) of Madeira, which had been detected by UFAC (Federal University of Acre) and UNIR (Federal University of Rondônia). One example of this event is presented in Figure 9. Thus, this event is also a local event that generally triggers these two PMUs (UFAC and UNIR).

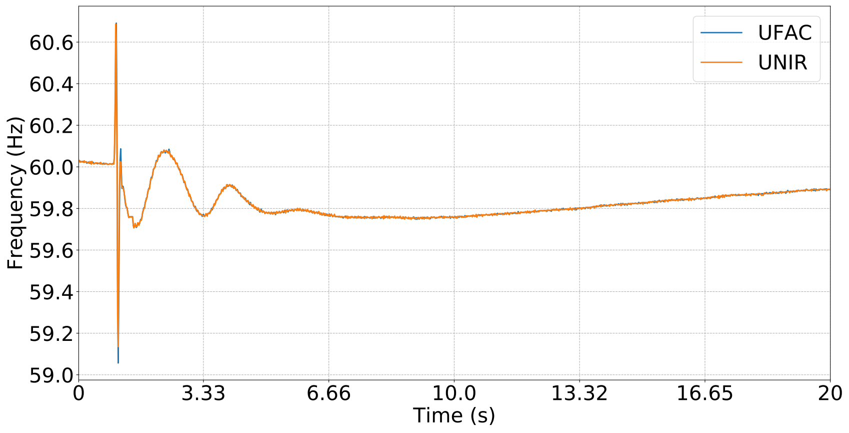


Figure 9 – Oscillation Event.

2.4.2 Steps in Event Identification

The usual steps involved in the identification of power system events using synchrophasors measurements are presented in Figure 10.

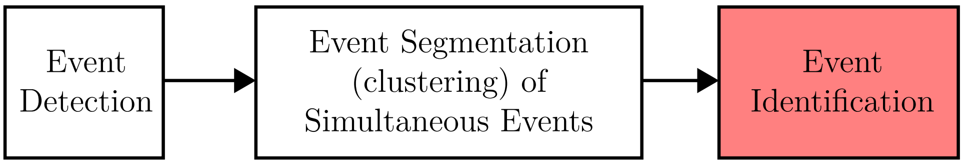


Figure 10 – Steps for Event Identification.

The steps applied in the BIPS are summarized below:

- Event Detection: The normalized wavelet method proposed in Kim et al. (2017) is used to detect the events of BIPS;
- Event Segmentation: Usually a single recording may contain more than one disturbance needs to be segmented for proper analysis (BYKHOVSKY; CHOW, 2003). Hence, an agglomerative hierarchical clustering (FOWLKES; MALLOWS, 1983) is used to categorize similar PMUs into groups (clusters). This algorithm is an unsupervised method that cluster the signals based on their similarities;
- Event Identification: The LSTM classifies each PMU signal of the cluster, thus a soft-majority vote (mean value of output probabilities) is taken to classify the whole event.

The event detection and segmentation are previous essential steps for the identification of the events. However, our focus of study here is only the event identification step, highlighted of red in Figure 10. Although, our focus is the event identification the detection and segmentation steps are necessary off-line, respectively, to:

- Locate the beginning instant of the event, establishing the time-window for identification with the T_{pre} and T_{pos};
- Segment the clusters of PMUs signals in multiple events, if that is the case. In this case has only one event, there is no need for segmentation.

After these steps the events are collected, and then labeled, i.e., attributed an event type LS, GT, LT, or OS. In Subsection 2.4.3 it is described that how the detection, and segmentation steps are applied to two multiple events.

2.4.3 Detection and Segmentation Steps: Case Studies

Multiple Events 1: Simultaneous LT and GT events

A multiple event with LT and GT events is presented in Figure 11. After the detection and segmentation steps (Figure 10) two clusters are obtained: an LT and a GT event (Figure 12). It can be observed that two events have occurred simultaneously in different areas of the system, without the segmentation step it would be very difficult to identify simultaneously both events. Probably, the classifier would identify only the large event (GT event), disregarding the LT event.

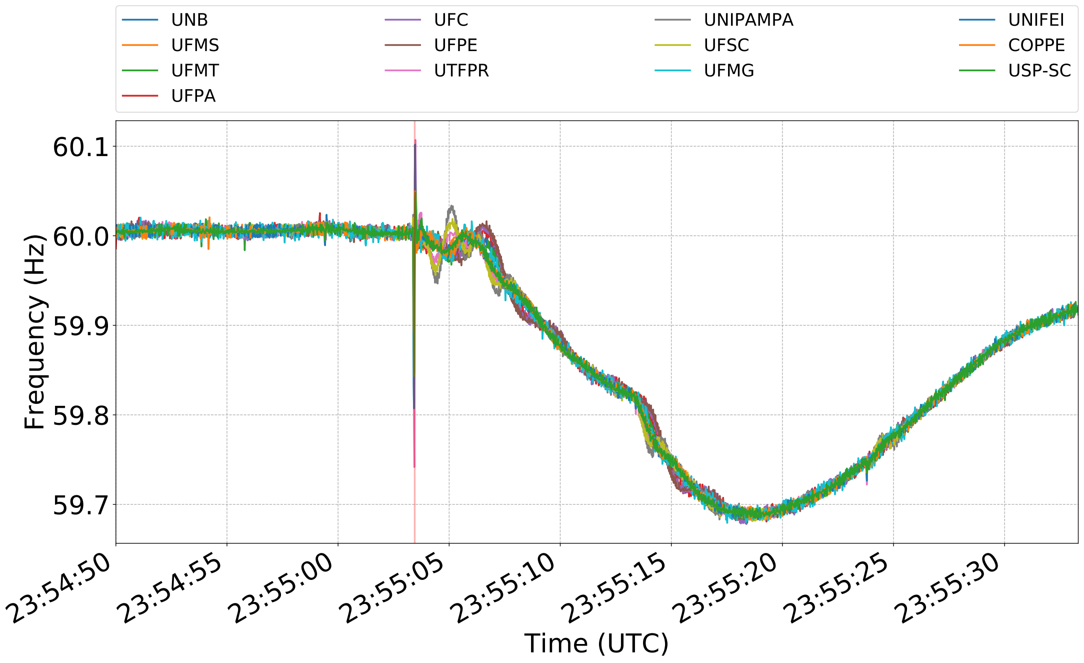


Figure 11 – Multiple Events 1 with LT and GT events. The red line indicates the beginning of the events.

The events (or clusters) are represented in a shorter windows of 20s, with $T_{pre} = 1$ second of pre-event and $T_{pos} = 19$ seconds of post-event. Figure 12a referring to the LT event, illustrates the cluster of PMUs signals with the spike in the beginning of the event. In this LT event the downfall in the frequency after the spike due to the practically simultaneous loss of generation (GT event) that happened in the other PMUs of the system (other area), as shown in Figure 12b, can be observed. As a result of the synchronized nature of synchrophasors and interconnection of the EPS, the LT event captured this redundancy (downfall in the frequency) indicating the PMUs where the anomaly (spike) happened.

Multiple Events 2: Simultaneous OS and GT events

A multiple event with OS and GT events is presented in Figure 13. After the detection and segmentation steps two clusters are obtained: an OS and a GT event

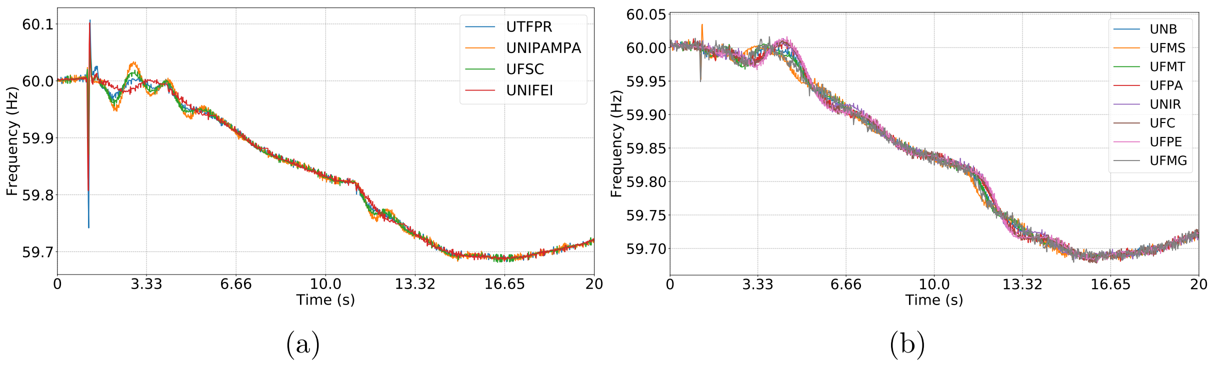


Figure 12 – Segmented Events from Multiple Events 1. (a) LT event. (b) GT event.

(Figure 14). Figure 14a illustrates the PMUs signals triggered by the OS event, and Figure 14b illustrates the PMUs signals of the GT event.

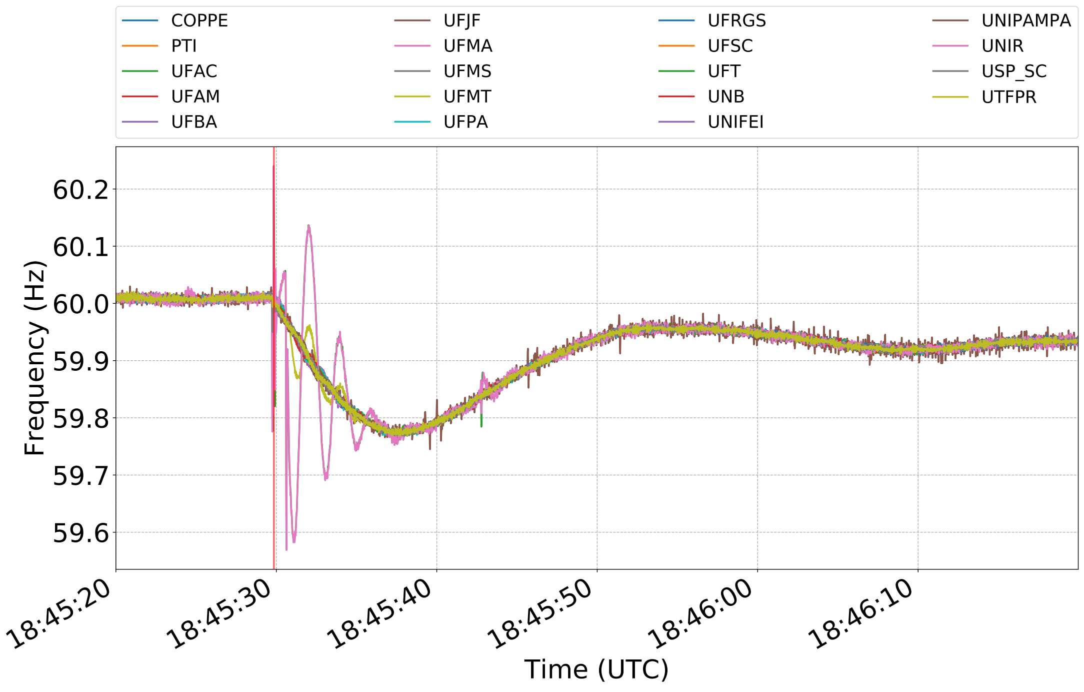


Figure 13 – Multiple Events 2 with OS and GT events. The red line indicates the beginning of the events.

The OS event again happened in UFAC, and UNIR related to HVDC link (600kV) of Madeira (Figure 14a). Also, it can be observed that the OS event has a downfall trend due to a simultaneous GT event that happened in the other PMUs of the system. The rest of EPS (Figure 14b) was triggered by the GT event that had not following the oscillation of UFAC and UNIR. It should be noted that in the segmentation of these events, it is not necessary to know the type of the event because the segmentation is unsupervised.

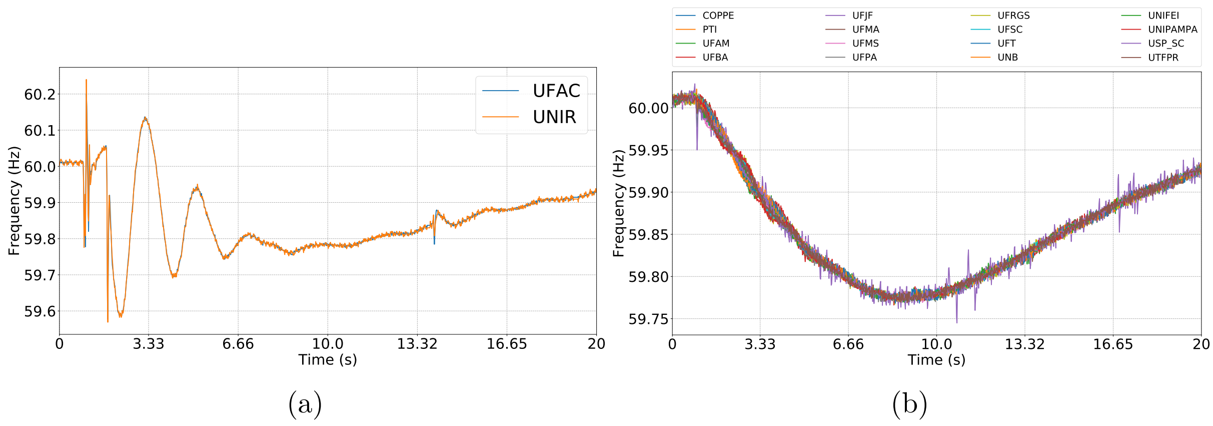


Figure 14 – Segmented Events from Multiple Events 2. (a) OS event. (b) GT event.

2.5 Events Data-set

The dataset is a collection of 168 real events (time-series) that occurred in the BIPS between June 2010 to July 2015 (ZARZOSA et al., 2016). All the events were acquired by the Medfasee Project Low Voltage synchrophasor system (LV-WAMS) which covers all the BIPS (DECKER et al., 2011). All the time-series used as input for the classifier are frequency records (acquired at 60 samples/second). A total time-window of 10 seconds (600 samples), consisting one second of pre-event ($T_{pre} = 1$s) (system frequency normal behavior) and 9 seconds of post-event ($T_{pos} = 9$s), is considered. In order to train the classifier the dataset was split into a Training-set (34.524%), and a Test-set (65.476%), represented in Table 1, by randomly selecting the events in the database.

Table 1 – Data-Set Splitting.

Set \ Event type	GT	LS	LT	OS	Total
Training	36	13	6	3	58
Test	49	34	18	9	110

It should be noted that by reflecting what happens in practical systems the majority of collected relevant events are GT and LS, resulting in an unbalanced dataset. However, one advantage of using WAMS is that in one event represented by a PMU data matrix (LI; WANG; CHOW, 2018) with PMUs signals (channels) as rows and the time-window as columns, multiple PMUs signals are triggered by the disturbance. Therefore, we can use this to increase the amount of essential data in training the classifiers, by taking each PMU signal as a labeled training sample. Therefore, in the events of the training-set all the PMUs signals triggered by the disturbance (cluster of PMUs signals) are taken as examples for training, increasing the amount of training samples, as presented in Table 2.

The resulting table with PMUs signals (Table 2) is still unbalanced that is very

Table 2 – PMUs signals per Event in the Training-Set.

Events type	GT	LS	LT	OS	Total
no of examples	523	176	16	6	721

problematic to most of ML techniques. So, to overcome the issue of the unbalanced dataset, the following data augmentation techniques (GOODFELLOW; BENGIO; COURVILLE, 2016) were applied for LS, OS and LT events: 1) *variable median filter*, which is used to change the size of the spike for LT events; 2) *time-shifting*, which is used to change the position in the time of the spike of the LT events, oscillations for OS events, and fall in the frequency LS events; 3) *injection of noise*, which is a recognized method in ML for data augmentation (GOODFELLOW; BENGIO; COURVILLE, 2016). The main idea is to inject noise to the new replicated training examples making the model more robust to noise examples. The objective of this augmentation is to obtain a balanced Training-set, which contains equal or almost equal number of examples for each class. The resulting training-set obtained by the application of the data augmentation techniques is presented in Table 3. It can be observed that the dataset has almost the same number of training examples per event.

Table 3 – Examples Generated in Training-Set

Events type	GT	LS	LT	OS	Total
no of examples	523	522	522	522	2089

Another important part of the ML process is the data normalization. The time-series of the events were normalized according to

$$\boldsymbol{x}_{scaled} = \left(\frac{\boldsymbol{x} - \boldsymbol{x}_{min}}{\boldsymbol{x}_{max} - \boldsymbol{x}_{min}}\right)(\max - \min) + \min \tag{2.2}$$

where $\min = -1$, $\max = 1$ are the feature ranges for scale. The limits are set to $\boldsymbol{x}_{min} = 59\text{Hz}$ and $\boldsymbol{x}_{max} = 61\text{Hz}$, which are the boundaries established for this normalization. This normalization rescales the input time-series into the fixed range (-1,1) making the training faster and improving the convergence of the training.

2.6 Summary

In this chapter, the main components of event identification using synchrophasors have been presented. The steps of detection, and segmentation necessary to obtain the dataset of events have been described. Moreover, the most relevant events that happen in BIPS, along with Training and Test sets which were split for identification have been shown. Also, the data augmentation which have been applied to make the Training-set a balanced one was described. Therefore, the Training and Test sets established in this chapter can be used in the training and evaluation of the proposed LSTM classifier for the event identification problems.

3 LSTM Classifier for Event Identification

3.1 Introduction

The main objective of an ML classification model is to predict the class label for unlabeled input instances. These predictions are realized based on background knowledge and knowledge extracted from an example of labeled instances (usually in the form of a Training-set) (KONONENKO et al., 2010).

In this chapter, the LSTM classifier that is used to identify the events is presented. The main advantage of using the LSTM in compare with other methods from literature, such as MSVM and MLP, is the capability of the LSTM to efficiently extract dynamic patterns of a time-series. In the LSTM model the time-series needs to be subdivided into τ time-steps which is then optimized to automatically learn dynamic patterns between these subdivisions or time-steps ($t = 1, \ldots, \tau$), using a set of gates that select and forget the information, depending on the importance of the learned information. The main disadvantage of the LSTM is that the model is not explainable, thus creating a *black-box*.

The chapter has been organized in the following way: Section 3.2 describes the main components of the LSTM network, detailing the inside gates that improve the performance over the classic RNN, and other known models from literature. Section 3.3 describes the proposed LSTM classifier used for the classification of the events, detailing the input of the LSTM and the output of the classifier. Finally, Section 3.4 contains the conclusions of the chapter.

3.2 LSTM Network

The LSTM is based on a gradient-based method proposed by (HOCHREITER; SCHMIDHUBER, 1997). The LSTM is an upgraded RNN model with a special memory cell. The architecture of the LSTM memory cell is presented in Figure 15.

The recurrent transition of the LSTM model is given by

$$\begin{pmatrix} \tilde{\boldsymbol{f}}^{(t)} \\ \tilde{\boldsymbol{i}}^{(t)} \\ \tilde{\boldsymbol{o}}^{(t)} \\ \tilde{\boldsymbol{g}}^{(t)} \end{pmatrix} = \boldsymbol{W}_h \boldsymbol{h}^{(t-1)} + \boldsymbol{W}_x \boldsymbol{x}^{(t)} + \boldsymbol{b} \tag{3.1}$$

$$\boldsymbol{c}^{(t)} = \sigma(\tilde{\boldsymbol{f}}^{(t)}) * \boldsymbol{c}^{(t-1)} + \sigma(\tilde{\boldsymbol{i}}^{(t)}) * \tanh(\tilde{\boldsymbol{g}}^{(t)}) \tag{3.2}$$

$$\boldsymbol{h}^{(t)} = \sigma(\tilde{\boldsymbol{o}}^{(t)}) * \tanh(\boldsymbol{c}^{(t)}) \tag{3.3}$$

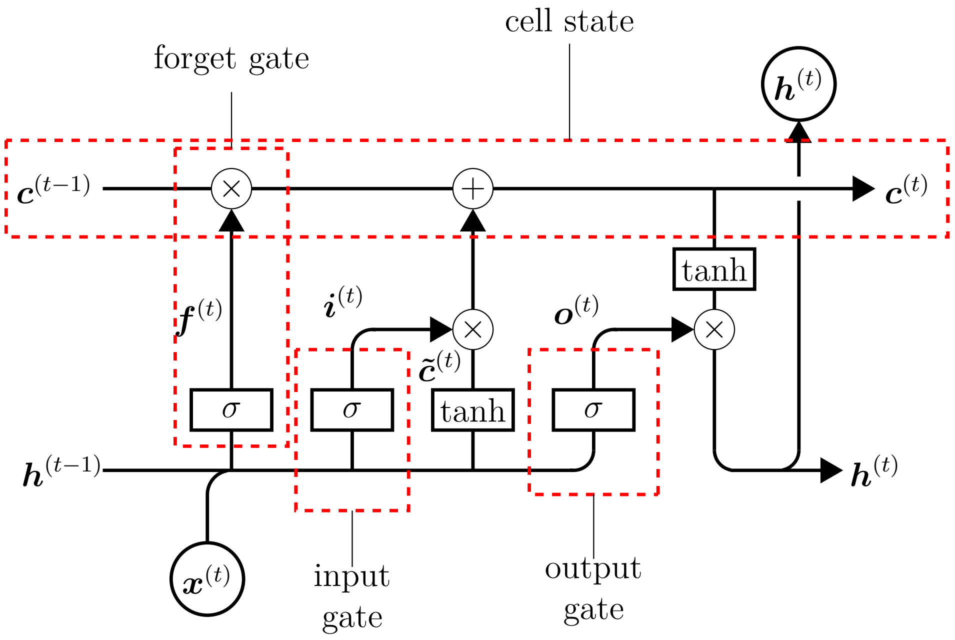


Figure 15 – LSTM Memory Cell.

where $\boldsymbol{W}_h \in \mathbb{R}^{d_h \times 4d_h}$ denotes the input weight matrix, $\boldsymbol{W}_x \in \mathbb{R}^{d_h \times 4d_h}$ denotes the recurrent weight matrix, $\boldsymbol{b} \in \mathbb{R}^{4d_h}$ denotes the bias matrix and the initial states $\boldsymbol{h}^{(0)} \in \mathbb{R}^{d_h}$; $\boldsymbol{c}^{(0)} \in \mathbb{R}^{d_h}$ are model parameters. The $*$ operator denotes the Hadamard product and d_h is the number of the units.

Each variable t presented in Figure 15 and in the set of equations 3.1, 3.2 and 3.3 is know as time-step. Following the definition of the term time-step used in (ABADI et al., 2015) we refer to the term time-step in this work as set of samples subdivided from the time-window defined that establish the order of the time-series. Usually in electrical engineering especially for transient studies the term *time-step* refers to the integration step used in the solution of the differential equations. However, in Deep Learning the term *time-step* is related to the order of a sequence, for example in a NLP problem such as text classification each word of a sentence will be the time-step t used in the LSTM for classify the text.

One of the main differences of this model in compared with RNNs is that the LSTM has an additional memory cell $\boldsymbol{c}^{(t)}$ with nearly linear update which allows the gradient to flow back through time more easily (CUBUK et al., 2018). The LSTM memory cell is regulated by the set of gates $\tilde{\boldsymbol{i}}^{(t)}$, $\tilde{\boldsymbol{f}}^{(t)}$ and $\tilde{\boldsymbol{o}}^{(t)}$.

- The **forget gate** $\tilde{\boldsymbol{f}}^{(t)}$ determines the extent to which information is carried over from the previous time-step $t-1$;
- The **input gate** $\tilde{\boldsymbol{i}}^{(t)}$ controls the flow of information from the current input $\boldsymbol{x}^{(t)}$;
- The **output gate** $\tilde{\boldsymbol{o}}^{(t)}$ allows the model to read from the cell.

The LSTM model can be represented by

$$\boldsymbol{h}^{(t)} = LSTM([\boldsymbol{c}^{(t-1)}, \boldsymbol{h}^{(t-1)}], \boldsymbol{x}^{(t)}, \boldsymbol{\theta}) \tag{3.4}$$

where the $LSTM(\cdot)$ computes Equation 3.1, Equation 3.2 and Equation 3.3. At the time-step t, the $LSTM(\cdot)$ uses the previous state of the network $[\boldsymbol{c}^{(t-1)}, \boldsymbol{h}^{(t-1)}]$ and the current time-step t of the sequence $\boldsymbol{x}^{(t)}$ to compute the output $\tilde{\boldsymbol{o}}^{(t)}$ and the updated cell state $\boldsymbol{c}^{(t)}$. The LSTM layer, represented in Figure 16, illustrates the flow of the information for a time-series $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(\tau)}\}$ with m features (channels) of length τ, where each point $\boldsymbol{x}^{(t)} \in \mathbb{R}^m$. The parameters of the LSTM layer are represented by $\boldsymbol{\theta} = \{\boldsymbol{W}_h, \boldsymbol{W}_x, \boldsymbol{b}\}$.

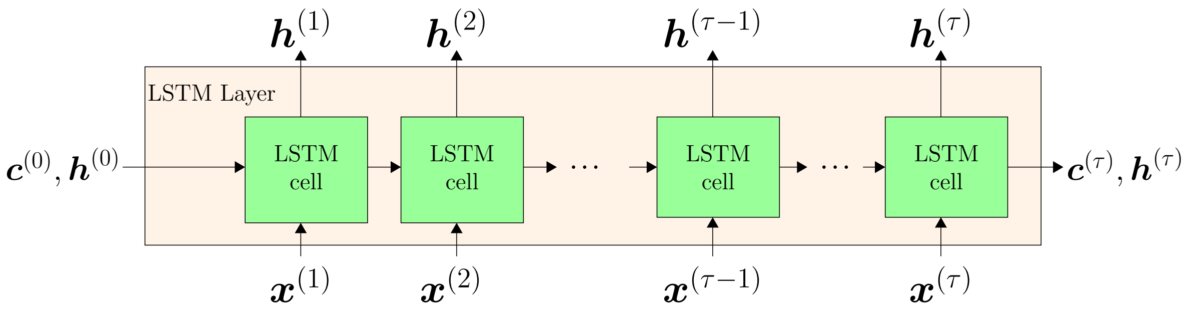

$\boldsymbol{h}^{(1)}$
$\boldsymbol{h}^{(2)}$
$\boldsymbol{h}^{(\tau-1)}$
$\boldsymbol{h}^{(\tau)}$
LSTM Layer
$\boldsymbol{c}^{(0)}, \boldsymbol{h}^{(0)}$
LSTM cell
LSTM cell
LSTM cell
LSTM cell
$\boldsymbol{c}^{(\tau)}, \boldsymbol{h}^{(\tau)}$
$\boldsymbol{x}^{(1)}$
$\boldsymbol{x}^{(2)}$
$\boldsymbol{x}^{(\tau-1)}$
$\boldsymbol{x}^{(\tau)}$

Figure 16 – LSTM Layer. $\boldsymbol{c}^{(\tau)}$ and $\boldsymbol{h}^{(\tau)}$ are the final states.

3.3 Proposed LSTM Classifier

In this section, we present an overview of the LSTM classifier used in this work. The input of the LSTM is 1D frequency record time-series of one PMU signal ($\boldsymbol{x} \in \mathbb{R}^{600}$), which represents the time-window of 10s with 600 samples ($60\frac{\text{samples}}{s} \times 10\text{s}$). Therefore, in a set of PMUs signals that represent the event each PMU signal is classified by the LSTM, then the mean value of output probabilities of the LSTM is taken to classify the event.

For the LSTM the input vector is $\boldsymbol{x} \in \mathbb{R}^{600}$ and the output vector is $\hat{\boldsymbol{y}} \in \mathbb{R}^4$, representing the specified classes (GT, LS, OS and LT). For the given Training-set, described in Section 2.5, $\mathbb{D} = \{\boldsymbol{x}_i \in \mathbb{R}^{600}, \boldsymbol{y}_i \in \mathbb{R}^4, i = 1, \cdots, N\}$ contains $N = 2089$ pairs of training data and the corresponding labels. $\boldsymbol{y}_i^k$ is a binary vector where the only kth entry is 1, if $\boldsymbol{x}_i$ belongs to the class k. This is known as one-hot encoding. When the input to the LSTM classifier is $\boldsymbol{x}_i$, the output class score for $\boldsymbol{x}_i$ is $\hat{\boldsymbol{y}}_i$.

In this work, better generalization results have been obtained with the function ReLU (see Equation 3.5)

$$\text{ReLU}(z) = \max(0, z), \tag{3.5}$$

where z is the current input of this layer, in place of $\mathbf{tanh}(\cdot)$ as described in Equation 3.2 and Equation 3.3. Also, the output layer activation function is $\mathbf{softmax}(\cdot)$. The LSTM classifier structure is represented in Figure 17. The penultimate layer is a Fully Connected

(FC) layer with the main purpose of reducing the over-fitting of the previous LSTM layers with the use of dropout technique in this layer.

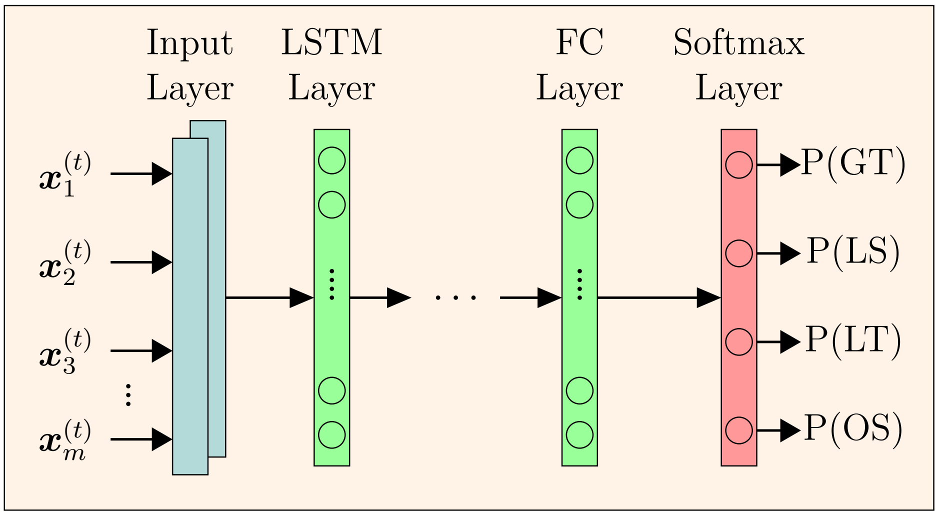


Figure 17 – LSTM Classifier.

The input vector $\boldsymbol{x} \in \mathbb{R}^{600}$ is translated (or subdivided) into τ time-steps, as described in Figure 16. Then, the LSTM receives a time-series in the form $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(\tau)}\} \in \mathbb{R}^{\tau \times m}$, where each point $\boldsymbol{x}^{(t)} \in \mathbb{R}^m$ in the time series is a m-dimensional vector $\boldsymbol{x}^{(t)} = [x_1^{(t)}, x_2^{(t)}, \ldots, x_m^{(t)}]^T$. m is a dependent of τ computed as $m = \frac{600}{\tau}$.

In this structure, the LSTM hidden layers, for the given time-step t, are given by

$$\begin{aligned} \boldsymbol{h}^{(s,t)} = LSTM_{\text{ReLU}}([\boldsymbol{c}^{(s,t-1)}, \boldsymbol{h}^{(s,t-1)}], \boldsymbol{x}^{(t)}, \boldsymbol{\theta}^{(s)}), \\ s = 1 \cdots n_h - 1 \end{aligned} \tag{3.6}$$

where n_h denotes the last hidden layer (number of hidden layers). The information flows to all LSTM layers along all the times-steps $t = 1, \cdots, \tau$, as described in Figure 16. Then, the output of the last LSTM layer in the last time-step $\boldsymbol{h}^{(n_h-1,\tau)}$ is fed to the FC layer with the ReLU to increase the sparsity. Therefore, the FC output is computed as

$$\boldsymbol{h}_{FC} = \text{ReLU}(\boldsymbol{W}^T \boldsymbol{h}^{(n_h-1,\tau)} + \boldsymbol{B}) \tag{3.7}$$

where $\boldsymbol{W} \in \mathbb{R}^{d_h^{(n_h-1)} \times d_h^{(n_h)}}$ denotes the weight matrix, $\boldsymbol{B} \in \mathbb{R}^{d_h}$ denotes the bias matrix of the FC layer, $d_h^{(s')}, s' = 0, \cdots, n_h$ is the dimension of the layer s', $d_h^{(0)} = 600$ is the dimension of the input layer and $d_h^{(n_h)}$ is the dimension of the FC layer. The output class scores $\hat{\boldsymbol{y}} \in \mathbb{R}^4$ are computed from

$$\hat{\boldsymbol{y}} = \text{softmax}((\boldsymbol{W}^o)^T \boldsymbol{h}_{FC} + \boldsymbol{B}^o) \tag{3.8}$$

where $\boldsymbol{W}^o \in \mathbb{R}^{d_h^{(n_h)} \times 4}$, denotes the output weight matrix, $\boldsymbol{B}^o \in \mathbb{R}^4$ denotes an output bias matrix.

It should be noted that the LSTM classifies each PMU signal in the cluster of PMUs signals, then final output probabilities (P(GT), P(LS), P(LT), and P(OS)) are obtained with the mean value of output probabilities of the LSTM over the PMUs signals of the cluster.

To train a suitable parameter set $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(s)}, s = 1, \cdots, n_h - 1, \boldsymbol{W}, \boldsymbol{B}, \boldsymbol{W}^o, \boldsymbol{B}^o\}$, we minimize the cross-entropy loss function, and the optimal parameter set $\boldsymbol{\Theta}$ can be computed as

$$\boldsymbol{\Theta} = \arg\min_{\boldsymbol{\Theta}} \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{4} \boldsymbol{y}_i^k \log \hat{\boldsymbol{y}}_i^k \tag{3.9}$$

where $N = 2089$ is the number of training examples, $\boldsymbol{y}_i^k$ corresponds to the kth element of one-hot encoded label of the example $\boldsymbol{x}_i$, $\hat{\boldsymbol{y}}_i^k$ corresponds to the kth element of $\hat{\boldsymbol{y}}_i$. In practice, these process is executed in mini-batch training, in this work a batch size of 32 was used. This means that the parameters of the LSTM are updated every 32 training examples.

The stochastic gradient descent method RMSprop (TIELEMAN; HINTON, 2012) is employed with a decay $\gamma = 1 \times 10^{-6}$ and learning rate $\eta = 0.001$. Also, the Xavier initialization (GLOROT; BENGIO, 2010) and batch normalization with momentum α are used. In order to improve the generalization capacity the dropout technique is used. The dropout is applied to LSTM layers and FC. Due to a acknowledged noise problem of applying dropout to the standard LSTM (ZAREMBA; SUTSKEVER; VINYALS, 2014), the dropout applied to the LSTM layers is known as variational dropout that uses the same dropout mask at each time-step, the standard dropout uses different masks at different time-steps (GAL; GHAHRAMANI, 2016). Furthermore, the dropout technique is used along with the constraint $\|\boldsymbol{w}\| \leq c$, where $\boldsymbol{w}$ represents the vector of weights incident on any hidden unit and c is a fixed constant. This constraint is imposed during training by projecting $\boldsymbol{w}$ onto the surface of a ball of radius c, whenever $\boldsymbol{w}$ goes out of it (SRIVASTAVA et al., 2014). The combination between dropout and constraint of the weights has been proven to be one most efficient ways of avoiding over-fitting (SRIVASTAVA et al., 2014).

3.4 Summary

In this chapter, the LSTM model was presented, showing the LSTM cell and gates. Also, the LSTM classifier was presented, showing the structure of the LSTM used in the classification of the events. The regularization techniques (dropout and constraint) applied to the LSTM was also presented.

The LSTM classifier established in this chapter will be used in the event identification problem. Also, this classifier will be inspected later using the SHAP values to provide explanations about the event identification.

4 SHAP Inspection - Interpreting LSTM Predictions

4.1 Introduction

In this chapter, the interpretability method known as SHAP values used to inspect the LSTM classifier is presented. This method is presented in detail, starts with the classic cooperative game theory method known as Shapley values method and ends with the sampling procedure known as SHAP values that are used to approximate the Shapley values.

Understanding how complex models take decisions is a relevant problem for data science applications (MOLNAR, 2019). Several methods were proposed in the literature to cope with this issue, such as LIME (RIBEIRO; SINGH; GUESTRIN, 2016), DeepLIFT (SHRIKUMAR; GREENSIDE; KUNDAJE, 2017), Layer-wise Relevance Propagation (LRP) (BACH et al., 2015). However, these methods still lack a theoretical background in order to be properly applied in real-world applications. Recently, the authors in (LUNDBERG; LEE, 2017) proposed the SHAP values method that is embedded with some formal definitions, axioms, and proprieties based on cooperative game theory and helps to fill this gap. The authors stated that any explanation of a prediction model must be a model itself. Therefore, they introduced the term *explanation model* g which is the best interpretable approximation of the original prediction model f (LUNDBERG; LEE, 2017). We decide to adapt this method as a way of evaluate the coherence of the LSTM classifier to synchrophasors events data. However, another similar methods could be also adopted.

Using the SHAP values method we can extract which parts of the input (which in the case of time-series are the time-steps) were more relevant in identifying the events. As well as, which parts of the input most harm the identification in the case of misclassifications. As a result, based on game theory concepts of Shapley values, they showed theoretical procedure that guarantee a measure of the importance features that can just be approximated in other methods.

The remaining part of the chapter has been organized in the following way: Section 4.3 describes the classic Shapley values method along with its properties. Section 2.3 describes in detail the definition of SHAP values, computation of SHAP values for NN, method known as DeepSHAP, and application of SHAP for multi-class classification. Section 4.5 describes the visualizations tools used by SHAP values for interpretation of the results. Finally, Section 4.6 contains the conclusions of the chapter.

4.2 Game Theory

The area of game theory is generally divided into two branches called non-cooperative game theory and cooperative game theory. In the non-cooperative game, the actors in the game are individuals players, i.e., they do not cooperate with each other in any way (SERRANO, 2007). These players are said to be independent which means that they play individually. One example of non-cooperative is two rivals companies that take into account each other's likely behaviors and independently determine a pricing or advertising strategy, aiming to increase its market share (AMBONI et al., 2001). One advantage of this approach is that it is possible to observe how specific details of interaction among individual players may impact the final payout (SERRANO, 2007). However, one observed limitation is that the contributions may be very sensible to those details that require fine-tuning parameterization. In the cooperative game, the players in the game are coalitions (group of players) where they cooperate with other forming sets with no order or hierarchy. This is why given M players we have 2^M possible coalitions of players. Thus, given the coalitions and their sets of feasible payoffs as primitives, the question tackled is the identification of final payoffs awarded to each player (SERRANO, 2007).

The cooperative game theory has proven to be more appropriate for credit allocation problems. Therefore, in this work we are going to give focus on cooperative game theory.

4.2.1 Cooperative Game Theory: Credit Allocation

The cooperative game theory studies the interactions between the coalitions of players. The coalitions of players, which represent a group of players, is defined as a set of players and the output of the coalition is the value of total payout. So, there are a set Z (of M players), usually called *grand coalition*, and a function $f_x(S) = E[f(x)|S]$ that maps subsets S of players to the real numbers: $f_x(S) : 2^S \rightarrow \mathbb{R}$, with $f_x(S)(\emptyset) = 0$, where $\emptyset$ denotes the empty set. The function $f_x(S)$ is called a characteristic function or conditional expectation function. Each coalition can be assigned a single value of its payout.

The problem of credit allocation is direct related to cooperative game theory working with coalitions of players instead of individuals players (AMBONI et al., 2001). The process of credit allocation is based on subdivide the credit of an activity between the players according to some responsibilities and benefits (AMBONI et al., 2001). This credit allocation must encourage the cooperation between the players to induce the efficient use of the resources. In general, the methods of credit allocation must guarantee the total recovery of the credits proving insights that induce the efficient in the application of the resources. Thus, a method of credit allocation is a function φ defined for all Z and for all

$f_x(S)$ such that

$$\varphi(f_x(S)) = (\phi_1, \ldots, \phi_Z) \in \mathbb{R}^M \quad \text{and} \quad \sum_M \phi_j = f_x(Z) \tag{4.1}$$

where ϕ_j is the credit allocated to each player j.

The solutions that allocate credit to coalitions must guarantee that all the players of the coalition feel itself better or equal than if they were alone. This is related to the concepts of *stand-alone* and *incremental* test (AMBONI et al., 2001).

The *stand-alone* test is defined as

$$\sum_S \phi_j \leq f_x(S) \tag{4.2}$$

ϕ_j is the credit of the player j that belongs to the coalition S. $f_x(S)$ is the credit function for the coalition S. $f_x(S)$ represents the minimum credit allocated to the players of the coalition S as efficiently as possible. The *incremental* (or marginal) credit of any coalition S is defined as $f_x(Z) - f_x(Z - S)$, Thus, the incremental test require that $\phi \in M$ be

$$\sum_S \phi_j \geq f_x(Z) - f_x(Z - S) \tag{4.3}$$

for all $S \subseteq Z$. Thus, in a fair credit allocation any player j of the coalition must have a credit that is less than the stand-alone credit and greater or equal than the incremental credit. The first condition offer incentive to the collaboration of the players, and the second ensure that no group subsidize the other (AMBONI et al., 2001). The Equation 4.2 encourages the voluntary cooperation and Equation 4.3 guarantee the fair allocation, preventing the coalition $Z - S$ from subsidizing S.

In the context of interpretability, the credit is optimally allocated to the features of the classifier in a way that the sum of the contributions of the features is the payout. The contribution is the credit allocated to each feature j of the classifier, i.e., it is defined here as the value of relevant of the feature to the classifier.

4.3 Shapley values

The Shapley values $\phi_j(f, x)$ provide the fundamentals to allocate the contributions for each member of a specific coalition. Lundberg et al. (2018) states that Shapley values $\phi_j(f, x)$ are the only method of allocation that obeys a set of desirable properties for explanation methods known as Shapley properties (Subsection 4.3.1).

4.3.1 Shapley values properties

Local Accuracy

The local accuracy property is given by the following equation

$$f(x) = \phi_0(f,x) + \sum_{j=1}^{Z} \phi_j(f,x) \tag{4.4}$$

where $\phi_0(f,x) = E[f(x)]$. The local accuracy property forces the attribution values to correctly capture the difference between the expected model output $E[f(x)]$ and the output of the current prediction $f(x)$ (LUNDBERG et al., 2018).

Consistency

For any two models f and f', if

$$f'_x(S \cup \{j\}) - f'_x(S) \geq f_x(S \cup \{j\}) - f_x(S) \tag{4.5}$$

For all $S \in Z \setminus \{j\}$, then $\phi_j(f',x) \geq \phi_j(f,x)$. This property guarantee that if a feature j is more important (greater) in one model f' than another f, then the importance attributed ϕ_j to that feature j should also be higher.

These properties are important to guarantee that the resulting explanation model g, when using the SHAP values method, is able to properly interpret the original one f.

4.3.2 Exact Computation of Shapley Values

The Shapley values $\phi_j(f,x)$ are used to explain a prediction $f(x)$ by a set of single numerical values representing the impact of each feature j on the prediction model $f(x)$ with the single input x (LUNDBERG et al., 2018). The feature value contribution $\phi_j(f,x)$ represents the contribution to the payout, weighted and summed over all possible feature value combinations.

According to game theory considerations (LUNDBERG; LEE, 2017), it can be proven that only one solution of credit allocation satisfies these properties and that solution is the Shapley values $\phi_j(f,x)$ given by

$$\phi_j(f,x) = \sum_{S \subseteq Z \setminus \{j\}} \frac{|S|!(M-|S|-1)!}{M!} [\underbrace{f_x(S \cup \{j\})}_{\substack{\text{player } j \\ \text{presented}}} - \underbrace{f_x(S)}_{\substack{\text{player } j \\ \text{not presented}}}] \tag{4.6}$$

where S is a subset of the features used in the model, $f_x(S) = E[f(x)|S]$ is the expected value of the model over the training subset S, M is the number of input features, Z is the set of all M input features and $\phi_j \in \mathbb{R}$.

The Equation 4.6 can be interpreted as

$$\phi_j(f,x) = \frac{1}{\text{number of features}} \sum_{\substack{\text{coalitions}\\ \text{excluding } j}} \frac{\substack{\text{marginal contribution}\\ \text{of } j \text{ to coalition}}}{\substack{\text{number of coalitions excluding}\\ j \text{ of this size}}} \tag{4.7}$$

The term $f_x(S \cup \{j\}) - f_x(S)$ is the marginal contribution that player j generates after his adhesion to a coalition S. Practically, to compute the Shapley values of each prediction, it is necessary to estimate the predictions of the model f when some specific input features are missing (those not in the subset S). Also, the sum in Equation 4.6 has 2^M coalitions, i.e., it is necessary to estimate (retrain) 2^M models to compute the Shapley values. As might be expected, depending on the type of the model these 2^M models would be too many to be completely evaluated. In practice, a sampling process approximates the terms of Equation 4.6 such as Monte-Carlo (ŠTRUMBELJ; KONONENKO, 2014), and SHAP (LUNDBERG; LEE, 2017).

4.4 SHapley Additive exPlanation (SHAP) framework

The major drawback of computing the traditional Shapley values using Equation 4.6 is the high computational burden because of the 2^M possible coalitions of the feature values that need to re-trained. For the case of $M = 600$ features it would be necessary to re-train

$$2^M = 4.149516 \times 10^{180} \tag{4.8}$$

models. The SHAP framework (LUNDBERG; LEE, 2017) is part of a class of additive feature attribution methods and introduces the perspective of viewing any explanation of a model's prediction as a model itself called an *explanation model* (LUNDBERG; LEE, 2017). This framework turns the Shapley values method into an optimization problem, enabling both fast and accurate results and taking insights from others additive feature attribution methods (defined in Subsection 4.4.1) such as LIME, DeepLIFT, LRP, and Classic Shapley value estimation (LIPOVETSKY; CONKLIN, 2001; DATTA; SEN; ZICK, 2016; ŠTRUMBELJ; KONONENKO, 2014). For DNN models, SHAP combines some intuitions from DeepLIFT (Deep Learning Important FeaTures) method and Shapley values.

4.4.1 Additive feature attribution methods

The additive feature attribution methods have an explanation model that is a linear function of binary variables, known as simplified inputs z'. Let f be the original prediction model to be explained and g the explanation model. Lundberg e Lee (2017) defined additive feature attribution methods as

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{4.9}$$

where $z' \in \{0,1\}^M$ is the simplified inputs, M is the number of simplified inputs features, and $\phi_j \in \mathbb{R}$. Explanation models often use simplified inputs z', also known as the coalition vector, that map to the original inputs through a mapping function $z = h_x(z')$. In the coalition vector, an entry of 1 means that the corresponding feature value is present and 0 represents absent feature value. This is very similar to Shapley values, where we need to simulate that only some features values are playing ("present") and some are not ("absent") (MOLNAR, 2019).

The additive feature attribution methods are the basis on that the SHAP framework compute its values. The main innovation in SHAP is the use of the linear model (Equation 4.9) to represent the classic Shapley values. So, SHAP specifies the explanation for an instance x as described in Equation 4.9 (MOLNAR, 2019).

4.4.2 SHAP values

The SHAP values provide a unique important solution as an additive feature for Equation 4.6 that comply with Shapley properties and uses conditional expectations $f_x(z')$, defined as

$$f_x(z') = f(h_x(z')) = E[f(z)|z_S], \tag{4.10}$$

where S is the set of non-zero indexes in z', and used to define simplified inputs (LUNDBERG; LEE, 2017). Therefore, SHAP values are the Shapley values of the conditional expectation function $f_x(z')$ of the original function f. To compute the SHAP values we must combine the conditional expectations $f_x(z')$ with the classic coalition game theory from Shapley value (Equation 4.6). The Shapley values properties are important to guarantee that the resulting explanation model g is able to properly interpret the original one f. SHAP values attribute to each feature the change in the expected model prediction when conditioning on that feature x_j.

For example, consider a simple binary classification problem, represented by the logistic regression model f, with output $y = f(x) \in \{0,1\}$. This model, used for probability estimation, is composed of a linear regression z and a sigmoid function:

$$\Pr(x) = \frac{1}{1 + \exp\left(-g(x)\right)} \tag{4.11}$$

where $\Pr(x) = \Pr(y = 1|x)$, which ranges from 0 to 1, is the output probability of class 1, $g(x) = w_0 + \sum_{j=1}^{4} w_j x_j$ is a linear regression with bias w_0, weights $w_j, j = 1, \ldots, 4$ and input features $\boldsymbol{x} = [x_1, x_2, x_3, x_4]$.

The main purpose of SHAP is to provide a methodology to compute the individual contribution ϕ_j of a particular input feature x_j to the overall model estimative. The procedure is illustrated in Figure 18 for the case with $n = 4$, representing one single ordering. This is realized that computing the contribution ϕ_j using the conditional expectations

$E[f(z)|z_S]$ (Equation 4.10) starts with a null entry (no feature) that results in base value ($\phi_0 = E[f(x)]$). Afterward, the other features are added one by one and their respective contributions are estimated by $E[f(z)|z_S]$ using the base value as a reference. It should be noted that the output of the contribution $\phi_j(f, x)$ is a real number that can be positive $\phi_j(f, x) > 0$ (in red) or negative $\phi_j(f, x) < 0$ (in blue). In this particular case, like the classification problem, the positive and negative values increase or decrease the value of the estimated probability, respectively. There are $n = 4!$ possible orderings (or permutations) that can have different contributions. However, only the ones that follow the Shapley properties are evaluated and their average is used as the final result. Finally, the sum of contributions ($\sum_{j=0}^{4} \phi_j = \Pr(x)$) must satisfy the local accuracy property (Equation 4.4).

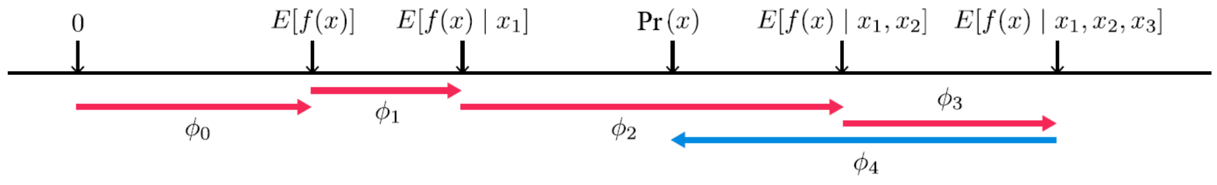
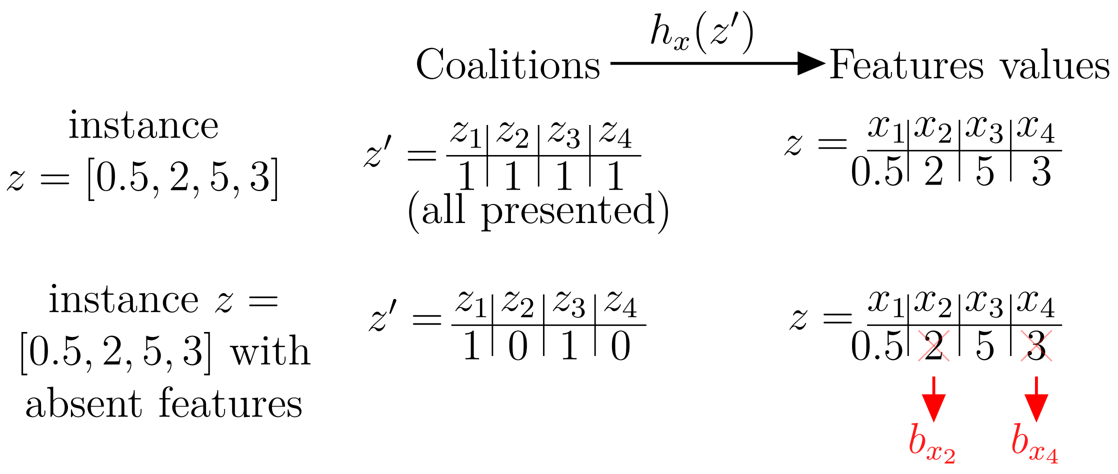


Figure 18 – Adapted from (LUNDBERG; LEE, 2017). SHAP values explain how to get from the base value $E[f(x)]$ that would be predicted if we did not know any features to the current probability output $\Pr(x)$. This diagram shows a single ordering $[x_1, x_2, x_3, x_4]$ of the 4!. For non-linear functions the order in which features are introduced is important. SHAP values $\phi_j(f, x)$ arise from averaging the ϕ_j values across all possible orderings that follow the Shapley properties.

In Figure 18, the features $\phi_j, j = 1, \ldots, 3$ are positive because they push the $E[f(z)|z_S]$ higher (increasing the probability output $\Pr(x)$) and ϕ_4 is negative because it pushes the $E[f(z)|z_S]$ lower. These values explain the output probability $\Pr(x)$ as sum of the effects ϕ_j of each feature being introduced into the conditional expectation (LUNDBERG; LEE, 2017). In practice, the base value $E[f(x)]$ is computed by calculating the average of $\Pr(x)$ over a background dataset, a sub-set of the training set. Also, for NN it is very difficult to emulate missing features and compute the $E[f(z)|z_S]$. So, the mapping function h_x must emulate the missing features from features samples of the background data b_{x_j}, as represented in Figure 19. Furthermore, the background data is used by the SHAP values to estimate the base values.

The exact computations of SHAP values are still complicated, but one advantage is that it is not necessary to retrain 2^M models as in the classic Shapley values estimation. Therefore, in order to accelerate the computation of SHAP values, these values are approximated using the insights from other additive feature attribution methods (LIME, DeepLIFT). The SHAP framework provides methods for both model agnostic approximations (KernelSHAP, and TreeSHAP), which represents learning the model on predicting of the black-box, perturbing inputs and seeing how the black-box reacts without knowing



$h_x(z')$
Coalitions
Features values
instance
$z = [0.5, 2, 5, 3]$
$z' = \frac{z_1 | z_2 | z_3 | z_4}{1 | 1 | 1 | 1}$
(all presented)
$z = \frac{x_1 | x_2 | x_3 | x_4}{0.5 | 2 | 5 | 3}$
instance $z = [0.5, 2, 5, 3]$ with absent features
$z' = \frac{z_1 | z_2 | z_3 | z_4}{1 | 0 | 1 | 0}$
$z = \frac{x_1 | x_2 | x_3 | x_4}{0.5 | \cancel{2} | 5 | \cancel{3}}$
b_{x_2}
b_{x_4}

Figure 19 – Adapted from (MOLNAR, 2019). Function h_x maps the coalition z' to a valid instance z. For present features 1, maps to the feature values of z. For absent features 0, maps to the values of a randomly sampled background data instance b_{x_j}. So, the $E[f(z)|z_S] = f(h_x(z')) = f(z)$.

what is inside (STRUMBELJ; KONONENKO, 2010; RIBEIRO; SINGH; GUESTRIN, 2016), and model specific approximations (Linear SHAP, and Deep SHAP). Since we are working on a LSTM neural network, we will give focus on the Deep SHAP method, which takes insights from the DeepLIFT method. The Linear SHAP and DeepSHAP are presented as follows. The linear SHAP is important to understand the computation process of SHAP vaues in a simple model, especially the process of computing the base values in practice.

4.4.3 Linear SHAP

For linear models, if we assume independence input feature, SHAP values can be directly approximated from the model's weight coefficients. Given a linear model $f(x) = \sum_{j=1}^{M} w_j x_j + b$, and $\phi_0(f, x) = b$. The SHAP values $\phi_i(f, x)$ are

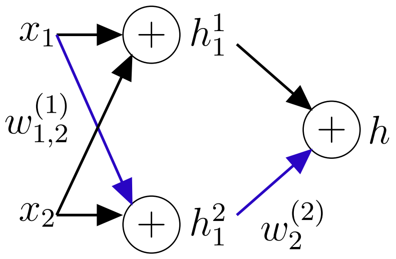
$$\phi_j(f, x) = w_j(x_j - E[x_j]) \tag{4.12}$$

where $E[x_j]$ is mean value of feature x_j over a background data. The same method of computing the base values $E[x_j]$ and $E[f(x)]$ using this background dataset is used in DeepSHAP.

4.4.4 Deep SHAP

The Deep SHAP method takes advantage of compositional nature of NN. As presented in (LUNDBERG; LEE, 2017), the Deep SHAP is a combination between DeepLIFT and Shapley values. The DeepLIFT is a recursive prediction explanation method for deep networks that attributes a value $C_{\Delta x_j \Delta y}$ to each input x_j and represents the effect of set the input value to a reference value as opposed to its original value (SHRIKUMAR; GREENSIDE; KUNDAJE, 2017).

The basic idea for computing SHAP values using DeepSHAP can be understood considering a fully linear model, as presented in Figure 20. In this simple model the input feature x_j can follow two possible paths. The exact SHAP value for this input is obtained by summing the attributions along all possible paths between that input x_j and the model's output y (CHEN; LUNDBERG; LEE, 2019).



$$h_1^i = w_{1,i}^{(1)} x_1 + w_{2,i}^{(1)} x_2$$
$$y = w_1^{(2)} h_1^1 + w_2^{(2)} h_1^2$$

Figure 20 – Adapted from (CHEN; LUNDBERG; LEE, 2019). Fully linear model.

Focusing on the blue path highlighted, the path's contribution $\phi(x_1)^{blue}$ is the product of the weights along the path with the difference among x_1 and its base value ($E[x_1]$) given by:

$$\phi(x_1)^{blue} = w_2^{(2)} w_{1,2}^{(1)} (x_1 - E[x_1]). \tag{4.13}$$

The contribution of the blue path to h_1^2 is

$$\phi(h_1^2)^{blue} = w_2^2 (h_1^2 - E[h_1^2]) \quad \Rightarrow \quad w_2^2 = \frac{\phi(h_1^2)^{blue}}{h_1^2 - E[h_1^2]} \tag{4.14}$$

Substituting Equation 4.14 in Equation 4.13 results in the contribution to $\phi(x_1)^{blue}$ in terms of $\phi(h_1^2)^{blue}$

$$\phi(x_1)^{blue} = \frac{\phi(h_1^2)^{blue}}{h_1^2 - E[h_1^2]} w_{1,2}^{(1)} (x_1 - E[x_1]) \tag{4.15}$$

For Deep models, with hidden activation's functions (non-linearities) such as **ReLU**, **sigmoid**, and **tanh** It is not simply possible to backward the products weights along the paths. Therefore, DeepSHAP uses some of the DeepLIFT properties known as rules that simplify the NN, and linearize the non-linear components of the NN. DeepLIFT established the following properties (SHRIKUMAR; GREENSIDE; KUNDAJE, 2017): Chain Rule, Rescale rule, and RevealCancel rule. The chain rule is an important rule to backpropagating the multipliers dy/dx across all neurons. Both Rescale and RevealCancel rules can be used to simplify the non-linearities and propagate the attributions to get ϕ_j.

In Figure 21a, a non-linear neuron model g is presented as an example to show how SHAP values are approximated using the Rescale rule. Under this rule SHAP values are computed for h as $\phi(h) = g(h) - g(E[h])$ given the local accuracy property and the g node has only one input, so

$$\frac{dy}{dh} = \frac{\phi(h)}{h - E[h]} \tag{4.16}$$

Then, dy/dh is propagated back linearly using the chain rule, obtaining

$$\phi(x_j) = \frac{\phi(h)}{h - E[h]} w_j(x_j - E[x_j]),$$

approximating the non-linear attributions (CHEN; LUNDBERG; LEE, 2019).

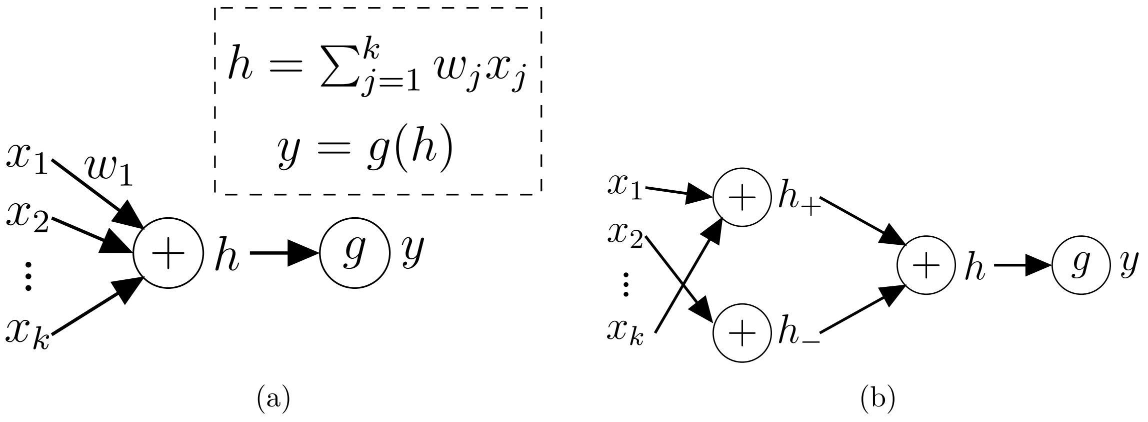


Figure 21 – Adapted from (CHEN; LUNDBERG; LEE, 2019). Neuron model where g is a non-linear function and $h = \sum_j^k w_j x_j$. (a) Rescale rule (b) RevealCancel rule.

The RevealCancel rule, presented in Figure 21b, partitions x_j into positive and negative components (intermediate nodes h_+ and h_-) based on the condition $w_j(x_j - E[x_j]) < t$ (where t = mean value of $w_j(x_j - E[x_j])$ across j). These components will be processed by the virtual intermediate nodes (h_+ and h_-)) that can be described as:

$$h_+ = \sum_j 1\left\{w_j(x_j - E[x_j]) > t\right\} w_j x_j, \text{ and} \tag{4.17}$$

$$h_- = \sum_j 1\{w_j(x_j - E[x_j]) < t\} w_j x_j. \tag{4.18}$$

This rule computes the exact SHAP attributions for h_+ and h_-

$$\begin{aligned}\phi_{h_+} = \frac{1}{2}[&g(h_+ + h_-) - g(E[h_+] + h_-) + \\ &g(h_+ + E[h_-]) - g(E[h_+] + E[h_-])]\end{aligned} \tag{4.19}$$

$$\begin{aligned}\phi_{h_-} = \frac{1}{2}[&g(h_+ + h_-) - g(E[h_+] + h_-) + \\ &g(h_+ + E[h_-]) - g(E[h_-] + E[h_-])]\end{aligned} \tag{4.20}$$

then propagates the resultant SHAP values back linearly.

$$\phi_j = \phi(x_j) = \begin{cases} \frac{\phi_{h_+}}{h_+ - E[h_+]} w_j(x_j - E[x_j]), & \text{if } w_j(x_j - E[x_j]) > t. \\ \frac{\phi_{h_-}}{h_- - E[h_-]} w_j(x_j - E[x_j]), & \text{otherwise.} \end{cases} \tag{4.21}$$

Note that the contribution ϕ_j is derived in term of the intermediary nodes using the chain rule (CHEN; LUNDBERG; LEE, 2019). The RevealCancel rule exactly explains the non-linearity and a partition of the inputs to the linearity as a single function prior to backpropagating, thus improving the rescale rule as demonstrated in (CHEN; LUNDBERG; LEE, 2019). This is why the RevealCancel rule is recent DeepSHAP method for estimating the SHAP values.

4.4.5 Application of SHAP Values for Multi-class Classification

In a multi-class classification problem (for example MNIST digit classification using a CNN classifier) for each class type $(0, 1, \ldots, 9)$ there is a base value $E[f(x)]_i, i = 0, \ldots, 9$. These base values are computed by taking the mean of the model output probabilities of the CNN over the background dataset. 100 training samples are taken, the output probabilities are computed, and a matrix (100×9) is obtained, so the mean of outputs for the 100 samples are the base values $E[f(x)]_i, i = 0, \ldots, 9$, presented in Table 4.

Table 4 – Base values $E[f(x)]_i, i = 0, \ldots, 9$.

$E[f(x)]_0$	$E[f(x)]_1$	$E[f(x)]_2$	$E[f(x)]_3$	$E[f(x)]_4$	$E[f(x)]_5$	$E[f(x)]_6$	$E[f(x)]_7$	$E[f(x)]_8$	$E[f(x)]_9$
0.06	0.1503	0.11	0.1	0.05	0.05	0.09	0.1597	0.13	0.091

The computed SHAP values for the MNIST digit classification are presented in Figure 22. The Figure explains ten outputs (digits 0-9) for four images. Red pixels increase the model's output while blue pixels decrease the output. The input images are shown on the left, and as nearly transparent gray-scale background behind each of the explanations. For each column the SHAP values are computed by taking the base value $E[f(x)]_i, i = 0, \ldots, 9$ of the corresponding digit.

In the correct images for the 'zero' image, the blank middle is important, while for the 'four' image, lack of connection on top makes it a four instead of a nine. In the last image a 'four' image was classified as 'one'. It is interesting to inspect the SHAP values of the predicted class 'one', using the base value $E[f(x)]_1$, and the correct class label using the base value $E[f(x)]_4$. The misclassified 'four' digit is different from the correct one in having a connection on top. This connection is the main reason for the missclassification, as highlighted in blue in the SHAP values computed using $E[f(x)]_4$.

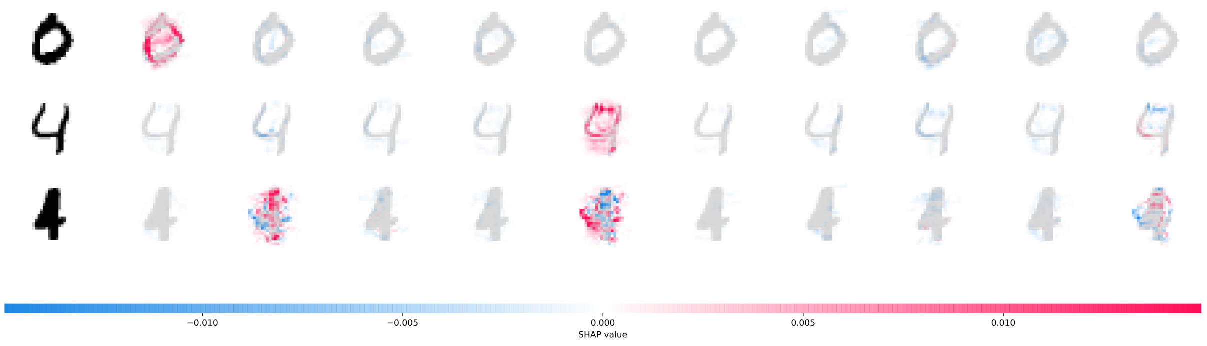


Figure 22 – SHAP values for MNIST multi-class

4.5 Visualizations tools for SHAP Inspection

Two important and useful visualizations tools in the explanations of SHAP values are: force plot, and summary plot (bar plot and beewarm plot). The force plot is used for local explanations, and the summary plot for global explanation. The local explanation stands for explaining single predictions, and the global explanation deals with understanding the predictions globally, in general by combining multiples local explanations and providing summaries of the classifier and features. As presented in (LUNDBERG et al., 2020), combining many local explanations is an effective way to obtain global explanations about the classifier.

As an example, we present the adult income dataset which predicts whether income exceeds $\$50K$ per year based on census data. The applied classifier is a simple logistic regression. Two computed base values for P(>50$ per year) and P(>50$ per year) are 0.2411 and 0.7589, respectively.

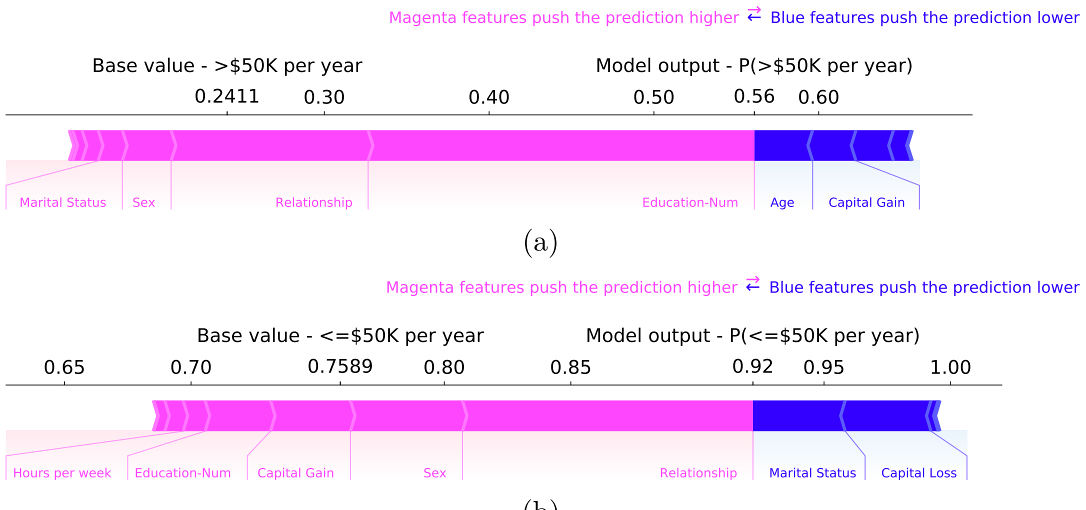


Figure 23 – (a) Force Plot of P(>$50K per year). (b) Force Plot of P(≤$50K per year).

The local explanations (Figure 23) show features contributing to pushing the model output from the base value to the model output. Features pushing the prediction higher are shown in magenta, those pushing the prediction lower are shown in blue. The force plot of P(>\$50K per year) (Figure 23a) shows that the Education-Num (number of years of education), Relationship features (Not-in-Family, Unmarried, Other-relative, Own-child, Husband, and Wife), Sex (Female, Male), and Marital Status (Divorced, Never-married, Separated, Widowed, etc) have the greatest contribution in increasing the prediction of exceeding income \$50K per year P(>\$50K per year). Figure 23b presents the force plot of P(≤\$50K per year) shows that the Relationship, and Sex (Female, Male) have the greatest contribution in increasing the prediction of whether income is less than \$50K per year P(≤\$50K per year).

Figure 24 represents the summary plot, combining multiples local explanations. Figure 24a presents the mean absolute value of the SHAP values for each feature to get a standard bar chart of the average magnitude. This plot is useful to get an absolute overview of the most important features of a model. Figure 24b is a beeswarm plot, and each dot corresponds to an individual prediction. The dot's position on the x-axis shows the impact of that feature on the model's prediction for that person. When multiple dots land at the same x position, they pile up to show density. The beeswarm plot sorts the features by summing SHAP value magnitudes over all samples. This plot uses SHAP values to show the distribution of the impacts of each feature on the model output. The colors represent the feature value (yellow high, purple low). This plot is useful to visualize the relationship between the input features and the contributions ϕ_j. Thus, it can be observed from Figure 24b that if you are married (husband or wife) the probability of the individual makes an increase over \$50k per year. The same relationship can be observed from Education-Num, Capital Gain, hours per week, Age, and Sex.

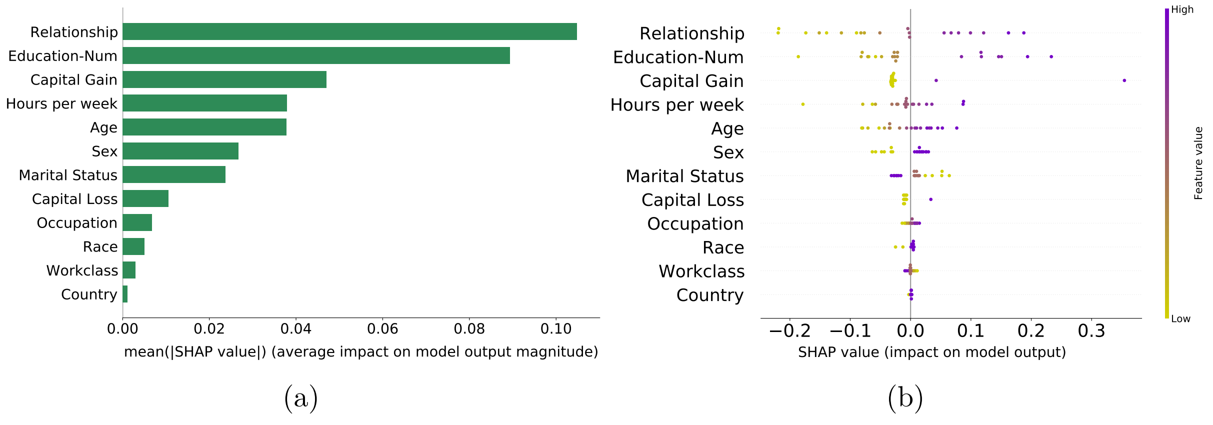


Figure 24 – (a) Bar chart of average SHAP value magnitude. (b) Beeswarm plot of Adult Income Data-set.

4.6 Summary

In this chapter, the main fundamentals of SHAP values were described. First, the game theory method of Shapley values, which is the theoretical background of SHAP values, was presented and detailed, focusing on the Shapley values properties. Then, the definition of SHAP values was presented, thus the specific method applied for NN models known as DeepSHAP was described. Finally, the essential visualizations tools for understanding the SHAP values were described.

The DeepSHAP established in this chapter will be used to compute the SHAP values. This method can be applied to DNN models, enabling its explanation. The described DeepSHAP method will be applied to the LSTM classifier to provide local and global explanations about the classifier.

5 Methodology

5.1 Introduction

In this chapter, the proposed methodology is presented in detail to put all the concepts together and to show the procedure for applying the SHAP inspection on the LSTM for the event identification problem. Also, the performance indices of Balanced Accuracy (BA), and IAR for event identification, and the background dataset are defined.

The remaining part of the chapter has been organized in the following way: Section 5.2 describes in detail the proposed methodology applied, explaining steps for establishing the work. Finally, Section 5.3 contains the conclusions of the chapter.

5.2 Framework

The proposed approach is illustrated in Figure 25 presenting the procedure used to apply the SHAP inspection on the LSTM for the event identification problem.

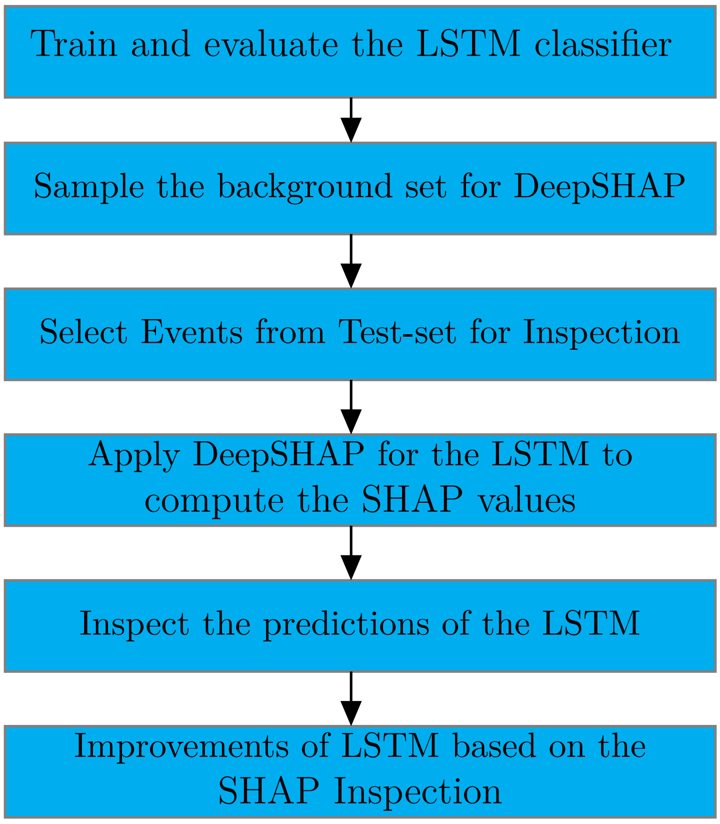

Train and evaluate the LSTM classifier
Sample the background set for DeepSHAP
Select Events from Test-set for Inspection
Apply DeepSHAP for the LSTM to compute the SHAP values
Inspect the predictions of the LSTM
Improvements of LSTM based on the SHAP Inspection

Figure 25 – Framework of the LSTM-based Event identification using SHAP Inspection.

Step 1: Train and evaluate the LSTM classifier

The LSTM classifier is trained on the Training-set, minimizing the cross-entropy loss according to Equation 3.9 with batch size of 32. To evaluate the identification accuracy, the BA and IAR are introduced as performance indices. These two criteria are defined below.

$$\mathrm{IAR} = \frac{\mathrm{N_{correct}}}{\mathrm{N_{total}}} \tag{5.1}$$

where $\mathrm{N_{correct}}$ is the number of correctly classified events, and $\mathrm{N_{total}}$ is the total number of the events.

The BA is calculated as the average of the proportion of each class individually. TP, FN, FP, TN denote the True Positive number, False Negative number, False Positive number, and True Negative number, respectively. $K = 4$ is the number of events, and TP_k denotes the TP of the kth event type. BA is defined as

$$\mathrm{BA} = \frac{\mathrm{TPR}_{avg} + \mathrm{TNR}_{avg}}{2} \tag{5.2}$$

where the average True Positive Rate (TPR) is

$$\mathrm{TPR}_{avg} = \frac{\sum_{k=1}^{K} \mathrm{TP}_k}{\sum_{k=1}^{K} (\mathrm{TP}_k + \mathrm{FN}_k)},$$

and average True Negative Rate (TNR) is

$$\mathrm{TNR}_{avg} = \frac{\sum_{k=1}^{K} \mathrm{TN}_k}{\sum_{k=1}^{K} (\mathrm{TN}_k + \mathrm{FP}_k)}$$

.

The BA criteria are essential due to the imbalance number of classes in the Test-set.

Step 2: Sample the background set for DeepSHAP

The background dataset are sampled from the Training-set to compute the base values $E[f(x)]_{\mathrm{GT}}, E[f(x)]_{\mathrm{LS}}, E[f(x)]_{\mathrm{LT}}$ and $E[f(x)]_{\mathrm{OS}}$ (one base value for each event type GT, LS, LT and OS), $E[x_j], j = 1, \ldots, 600$ and conditional expectations $f_x(z') = E[f(z)|z_S]$. The background dataset has 721 training examples (time-series), representing the non-augmented training-set of PMUs signals (Table 3). As described in Table 3, we have $N_{\mathrm{GT}} = 523$, $N_{\mathrm{LS}} = 176$, $N_{\mathrm{LT}} = 16$, and $N_{\mathrm{OS}} = 6$. Thus, the base values are computed as follows:

$$\begin{aligned}
E[f(x)]_{\mathrm{GT}} &= \frac{\sum_{i=1}^{N_{\mathrm{GT}}} \mathrm{P(GT)}_i}{N_{\mathrm{GT}}} \\
E[f(x)]_{\mathrm{LS}} &= \frac{\sum_{i=1}^{N_{\mathrm{LS}}} \mathrm{P(LS)}_i}{N_{\mathrm{LS}}} \\
E[f(x)]_{\mathrm{LT}} &= \frac{\sum_{i=1}^{N_{\mathrm{LT}}} \mathrm{P(LT)}_i}{N_{\mathrm{LT}}} \\
E[f(x)]_{\mathrm{OS}} &= \frac{\sum_{i=1}^{N_{\mathrm{OS}}} \mathrm{P(OS)}_i}{N_{\mathrm{OS}}}
\end{aligned} \tag{5.3}$$

$\mathrm{P(GT)}_i, \mathrm{P(LS)}_i, \mathrm{P(LT)}_i, \mathrm{P(OS)}_i$ denote, receptively, $\mathrm{P(GT)}, \mathrm{P(LS)}, \mathrm{P(LT)}, \mathrm{P(OS)}$, which are predicted by the LSTM of the ith example.

The base value $E[x_j], j = 1, \ldots, 600$ is computed for every sample $j = 1, \ldots, 600$ of the time-series as

$$E[x_j] = \frac{\sum_{i=1}^{N_T} x_j^i}{N_T} \tag{5.4}$$

where $N_T = 721$, and x_j^i denotes the sample x_j of the ith example.

Step 3: Select Events from Test-set for Inspection

Some events of the Test-set are selected for inspection. We select one correctly classified event of each type, to give insights about what is being learned to correctly classify the events, and also we select the misclassified events, to give insights about why the classifier made the mistake.

Step 4: Apply DeepSHAP for the LSTM to compute the SHAP values

The DeepSHAP method is applied to the LSTM classifier for computing the SHAP values in order to provide explanations about the selected events. The most important parts are highlighted in the time-series input. The direct results of application of the Deep SHAP method are the sample value contributions $\bar{\phi}_j^{(t)}$ (contribution of the sample j in the time-step t), representing the average contribution over the PMUs signals of the event. These results allow the user to see the samples with most significant role in all the time series to identify a specific event. Another way to extract information from SHAP values is to verify which time-steps t (set of samples) has the contribution to event identification. This is realized by calculating the summation of the sample contribution $\bar{\phi}_j^{(t)}$ values for each time-step t, combining set of samples of the same time-step t, using

$$\Phi^{(t)} = \sum_{j=1}^{m} \bar{\phi}_j^{(t)} \tag{5.5}$$

where m is the dimension of $\boldsymbol{x}^{(t)}$, and $t = 1, \cdots, \tau$. An interesting point of using the time-steps contributions $\Phi^{(t)}$ instead of using the sample contribution $\bar{\phi}_j^{(t)}$ is that the local accuracy propriety is still maintained (Equation 4.4), so

$$\sum_{t=1}^{\tau} \Phi^{(t)} = \sum_{t=1}^{\tau} \sum_{j=1}^{m} \bar{\phi}_j^{(t)} = f(x) - E[f(x)] \tag{5.6}$$

Step 5: Inspect the predictions of the LSTM

For each event type (GT, LS, LT and OS) the SHAP inspection is applied to identify the main contributions (SHAP values) involved in classification of these events, both local and global explanations. The local explanation is used for explaining single event, presenting with force plot. The global explanation is presented using the visualizations tools of SHAP values: bar chart of the SHAP average magnitude, and beeswarm plot. These contributions are computed using the DeepSHAP with the RevealCancel rule. The predictions of the LSTM are inspected to observe the coherence of the classifier and to detect possible bias in the LSTM predictions based on the domain knowledge of the events.

Step 6: Improvements of LSTM based on the SHAP Inspection

The obtained knowledge through the SHAP inspection is used to improve the LSTM by creating a new classifier LSTM-SHAP, correcting the possible bias and inconsistencies, and improving the coherence of the classifier predictions. Therefore, after applying modifications to data and/or model, for every trained LSTM model with different initials conditions and hyper-parameters, we observe not only the IAR and BA of the Test-set, but also observe the SHAP values of the selected events, both locally and globally. This process is represented in Figure 26.

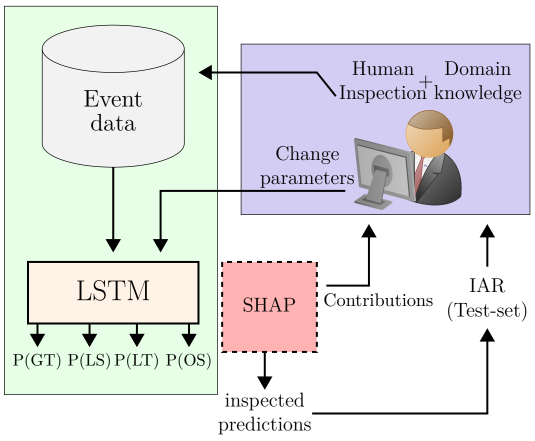


Figure 26 – LSTM-SHAP classifier training process.

The main purpose of SHAP inspection is to obtain an LSTM that have the main contributions $\Phi^{(t)}$ according to domain knowledge of the events. This process is described in Subsection 2.4.1. For example, the LSTM should be able:

1. To identify the downfall in frequency for GT events;
2. To identify the rise in frequency for LS events;
3. To identify the spikes in the beginning of the time-series for LT events;
4. To identify the oscillation peaks (maximum and minimum) soon after the spike for OS events.

Thus, implicitly what we are doing is introducing this domain knowledge into the LSTM. One problem that can be observed is the difficulty to obtain the LSTM that follows these patterns by only changing the hyper-parameters and initial conditions. So, a mechanism of introducing this knowledge into the loss function (Equation 3.9) must be studied in the future.

5.3 Final Comments

In this chapter, the proposed methodology employed in this study was presented, showing the steps associated with the procedure. It can be observed, that the proposed methodology is useful not only to understand the decision-makings of the LSTM, but also to incorporate the explanations into the LSTM. These explanations represent knowledge of data and model.

One difficulty that still can be observed from LSTM-SHAP was the time and energy consuming process to obtain the classifier by only hyper-parameters changes and initial conditions. However, the same hard-work was observed using traditional ML techniques, and there is still need to try parameters for optimizing the IAR of the Test-set.

The proposed methodology described in this chapter is executed, and the results are presented in the next chapter.

6 Performance Evaluation in BIPS events

6.1 Introduction

In this chapter, the results obtained with the methodology described in Chapter 5 are presented. The evaluations of the LSTM classifier are performed using both the identification rate (IAR and BA) and the interpretability inspection. The performance of identification of the LSTM is compared with the MSVM, and the MLP methods. The LSTM is optimized using the regularization techniques, and the best number of the time-steps τ of the LSTM is obtained (Subsection 6.2.1). The MLP and SVM are also optimized using regularization techniques to improve the generalization capacity. The interpretability inspection of the LSTM is performed using the DeepSHAP method, described in Subsection 4.4.4, to compute the SHAP values.

The remaining part of this chapter has been organized in following way: Section 6.2 presents the classification performance of LSTM compared to the MSVM, and MLP. Then, Section 6.3 presents the interpretability inspection of the LSTM, inspecting each event type (GT, LS, LT, and OS), both locally and globally, in order to understand the decisions of the LSTM. Also, the misclassified events of the LSTM are inspected to understand the limitations of the classifier. Finally, Section 6.4 presents the results of the LSTM-SHAP classifier obtaining by the knowledge of the SHAP inspection, and incorporating into the data to improve the coherence of the classifier. This new classifier LSTM-SHAP is then trained with the improvements, and is inspected using SHAP values.

6.2 Performance of Classifying the Events (IAR and BA)

The LSTM classifier is trained offline using the Training-set. We chose different initialization values of LSTM parameters and obtain multiple sets of parameters by trial and error. The resulting LSTMs are evaluated on the Test-set, and the best one evaluated in the Test-set is selected. According to Table 5, the LSTM classifier has three hidden layers with 60, 40 and 30 neurons, respectively. The FC layer has 30 neurons. The dropout rate applied to each layer is 0.3, 0.4, and 0.5, respectively.

Table 5 – Best parameters of LSTM classifier.

n_h	neurons	dropout rate	α
3	60|40|30	[0.3, 0.4, 0.5]	0.55

The results presented in Table 6 displays that the LSTM classifier shows a high IAR for GT, LS, LT and OS events. The classifier can identify the events with an overall IAR equal to 98.182%. Also, the BA% achieved for the LSTM classifier is presented.

Table 6 – Performance of LSTM classifier for identifying the events.

GT%	LS%	LT%	OS%	Overall %	BA%
97.959	100.0	94.444	100.0	98.182	98.101

6.2.1 Performance relation to time-steps τ

The number of time-steps τ, described in Figure 16, is important in the performance of LSTM. The results in Figure 27 indicate the IAR and BA in the test-set of the LSTM classifier as a function of τ. Usually, the LSTM has an optimal τ, in most case the LSTM has worse performance when τ is higher than this optimal value. The LSTM classifier has the best performance with $\tau = 8$. An interesting point is the terrible performance of LSTM with higher time-steps values $\tau > 40$. This is due to the saturation in the extraction of LSTM in learning the dynamic features.

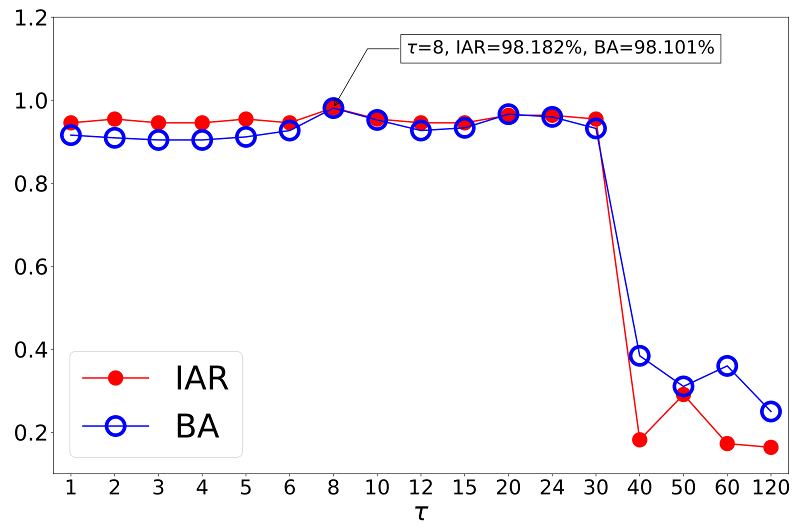


Figure 27 – Performances of LSTM classifier in relation to the number of times-steps τ. Best performance at $\tau = 8$.

6.2.2 Comparison of Different Classifiers

In order to show the advantage of our classifier in compare with other known methods from literature, we compared the LSTM with MLP and Multi-Class SVM (MSVM). The IARs of the MSVM, MLP and LSTM are compared in Table 7.

The misclassifications presented in Table 8 show that the LSTM classifier made only one error for GT and OS events. Therefore, the LSTM incorrectly classifies only 2

Table 7 – Performances of MSVM, MLP, and LSTM.

Classifier	GT%	LS%	LT%	OS%	Overall %	BA%
SVM	85.714	94.118	66.667	88.889	87.273	83.847
MLP	100.0	91.176	66.667	88.889	92.727	86.683
LSTM	97.959	100.0	94.444	100.0	98.182	98.101

Table 8 – Misclassifications of MSVM, MLP, and LSTM.

Classifier	GT	LS	LT	OS	Total
MSVM	7	2	2	3	14
MLP	0	3	2	3	8
LSTM	1	0	0	1	2

events in the total of the Test-set. The misclassifications are labeled as Event 9 and Event 98, representing the position in the Test-set. Event 9 is a GT event that was classified as LT, and Event 98 is a LT event that was classified as GT.

The MSVM is the extension of binary-class SVM with one-vs-rest scheme using the Radial Basis Function (RBF) as kernel and regularization parameter $C = 6.722$. The parameter C was obtained using 10-fold Cross-Validation (CV) and BA% as score function, this score as a function of C is represented in Figure 28. Also, the MLP classifier has three layers, similar to LSTM, and the main difference is replacement of the LSTM layers with the FC layers.

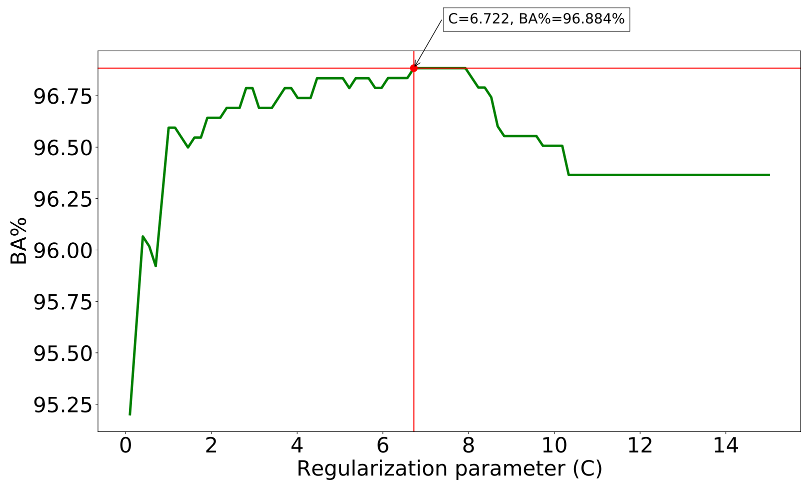


Figure 28 – CV score of MSVM as function C.

The parameters of the MLP classifier are represented in Table 9. The same RMSprop optimizer is employed to train the MLP. The other hyper-parameters include $\eta = 0.001$, $\gamma = 1 \times 10^{-6}$, which are similar to LSTM training parameters. The results in Table 7 and

Table 8 indicate that the LSTM has achieved the highest overall IARs among these three classifiers.

Table 9 – Best parameters of MLP classifier.

n_h	neurons	dropout rate	α
3	100|60|50	[0.3, 0.5, 0.5]	0.55

6.3 Interpretability Inspection of LSTM Classifier

In this work, we applied the interpretability inspection on the LSTM classifier using the DeepSHAP method. In our inspections, we analyzed the correctly classified events, to understand how the classifier takes its decisions, and incorrectly classifies events, and also we tried to understand the limitations of the classifier. The base values (one from each event type) $E[f(x)]$ displayed in Table 10 are computed from averaging the predictions of the LSTM classifier over the non-augmented training-set of PMUs signals, known as background dataset.

Table 10 – Base values $E[f(x)]$.

$E[f(x)]_{\text{GT}}$	$E[f(x)]_{\text{LS}}$	$E[f(x)]_{\text{LT}}$	$E[f(x)]_{\text{OS}}$
0.7725	0.1958	0.0247	0.0071

6.3.1 Inspection of the Events

Some of the identified events are discussed as follows. The main purpose of this inspection is to show the way the LSTM classifier takes its decisions and inspects the coherence of the classifier. For each event type (GT, LS, LT and OS) the inspection is applied to identify the most relevant SHAP values involved in the identification of these events, both global and local explanations.

6.3.1.1 **GT Events**

The main characteristic of the GT events are the downfall in the frequency due to the loss of generation in the system. As suggested in (LUNDBERG et al., 2020) the inspection is carried out by using local and global explanations.

1) Local Explanation: The focus of the local explanation is to interpret each event separately. Looking at a single event we can numerically estimate the positive and negative contributions given the base value $E[f(x)]_{\text{GT}}$. In this case, the Event 3, correctly classified as GT with a probability of P(GT) = 100%, was selected and it's force plot is presented in Figure 29. The magenta time-steps are the ones that contribute to push the probability P(GT) up and the blue time-steps are the ones that contribute to push the P(GT) down. These $\Phi^{(t)}$ values were computed using Equation 5.5 with $\tau = 8$ and $m = \frac{600}{\tau} = \frac{600}{8} = 75$. The temporal evolution of this event, highlighting the positive/negative sample contributions, is presented in Figure 30. According to this graph it is clear to identify parts of the time-series which are more relevant (positive/negative) to the classifier decision making. As presented, the time-steps $t = 4, 5$ have the greater contributions. These time-steps represent the downfall in the middle of the time-series. The negative time-steps $t = 7, 8$ did not have a relevant impact on P(GT), according its higher value.

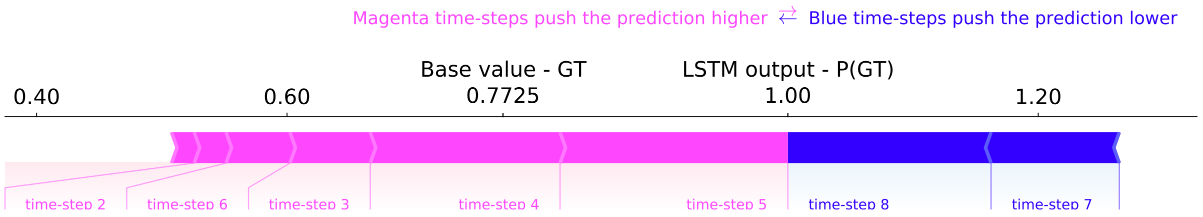


Figure 29 – Force plot of LSTM for Event 3.

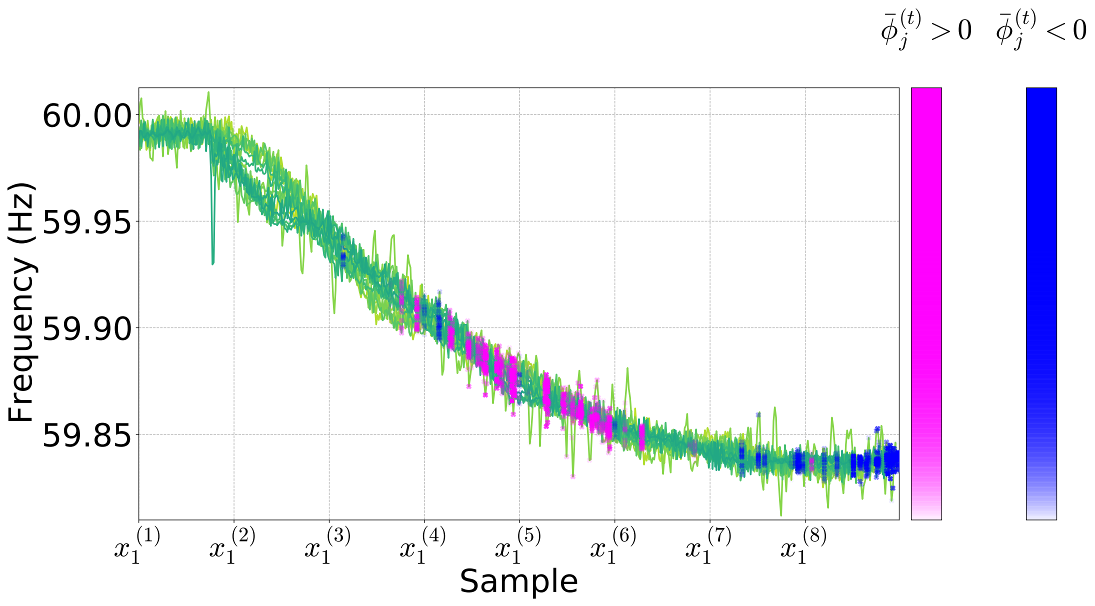
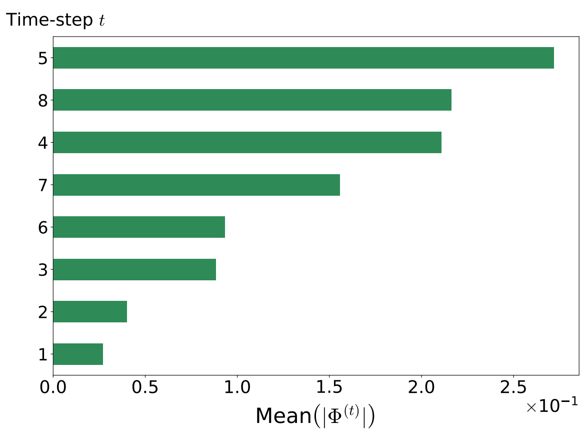


Figure 30 – Event 3. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f(x)]_{\text{GT}} = 0.7725$.

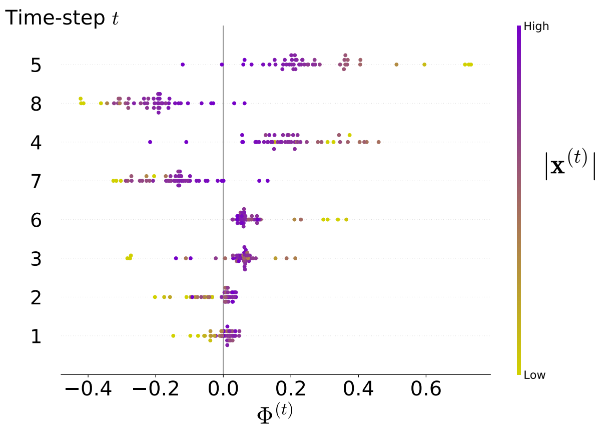
It can be observed from Figure 30 that the stream of PMUs signals representing the GT event. The LSTM classifies each PMU signal, and a soft-majority vote is taken to classify the event. Also, the sample contributions $\bar{\phi}_j^{(t)}$ are computed by taking the average of contributions over the PMUs signals of this event.

2) Global Explanation: The main goal of the global explanation is to verify which time-steps t have more influence in the classifier decision for a set of events (in this case GT events). This is obtained by combining multiples local explanations, expressing by the average absolute magnitude of the contributions ($\left|\Phi^{(t)}\right|$) for each time-step t. The beeswarm plot is useful to evaluate the relation between the local contributions ($\Phi^{(t)}$) for each time-step t and the input values $\boldsymbol{x}^{(t)}$.

The global explanation for GT events is presented in Figure 31. Figure 31a exhibits the mean absolute value of the SHAP values for each time-step t as a standard bar chart, and Figure 31b presents the beeswarm plot. In this plot, each dot corresponds to an individual GT event. The dot's position on the x-axis shows the impact of that time-step t has on the LSTM prediction P(GT), i.e., the $\Phi^{(t)}$ value. The colors represent the value of the input $\left|\boldsymbol{x}^{(t)}\right|$ from low to high (purple to yellow). This plot is applicable for visualization the relationship between the input ($\left|\boldsymbol{x}^{(t)}\right|$) with the contributions $\Phi^{(t)}$.



(a)



(b)

Figure 31 – Global Explanation of GT events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

As presented, the time-steps $t = 1, 2$ (beginning of the time-series) have a small contribution to the prediction P(GT), and these contributions get higher negative values with lower values of $\left|\boldsymbol{x}^{(t)}\right|, t = 1, 2$. Moreover, the time-steps $t = 5, 8$, and $t = 4$ have the greater contributions, with positive values for $t = 5, 4$ and negative values for $t = 8$ (Figure 31b). As presented in Figure 31b, focusing on the main time-steps $t = 5, 4$, we can observe that the events with lower values of $\left|\boldsymbol{x}^{(5)}\right|$ and $\left|\boldsymbol{x}^{(4)}\right|$ (in blue) have greater values of $\Phi^{(5)}$ and $\Phi^{(4)}$ (x-axis), respectively. This relationship is in agreement with the domain knowledge of GT events that greater deviations of Δf indicate greater loss of generation. However, GT events with no great deviation could have problems in being classified as GT based on this relation, as will be observed for the misclassification (Event 9).

The time-step $t = 8$ also presents an interesting relation with $\boldsymbol{x}^{(8)}$. The $\Phi^{(8)}$ gets higher values in magnitude when the input $\left|\boldsymbol{x}^{(8)}\right|$ is lower (see Figure 31b). This represents that P(GT) is pushed down by time-step $t = 8$ with higher deviations Δf. As observed, these inconsistencies will be common in our trained LSTM models, representing deceptive temporal patterns learned by the LSTM. These inconsistencies are important to understand the reasons for the misclassifications that are usually related to these deceptive patterns.

6.3.1.2 **LS Events**

The main characteristic of LS events are rise up in the frequency due to the gain of energy in the system, by disconnecting of a significant amount of loads.

1) Local Explanation: The LS event (Event 33) was selected, and correctly classified with a probability of P(LS) $= 100\%$. The force plot of the Event 33 is presented in Figure 32, given base value of $E[f(x)]_{\text{LS}}$. The time-series of the event are displayed in Figure 33, highlighting the main contributions. The time-steps $t = 5, 4$ have shown significant contributions, representing the rise up in the middle of the time-series.

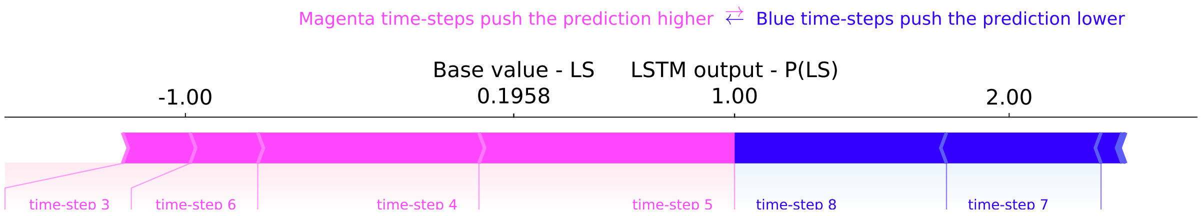


Figure 32 – Force plot of LSTM for Event 33.

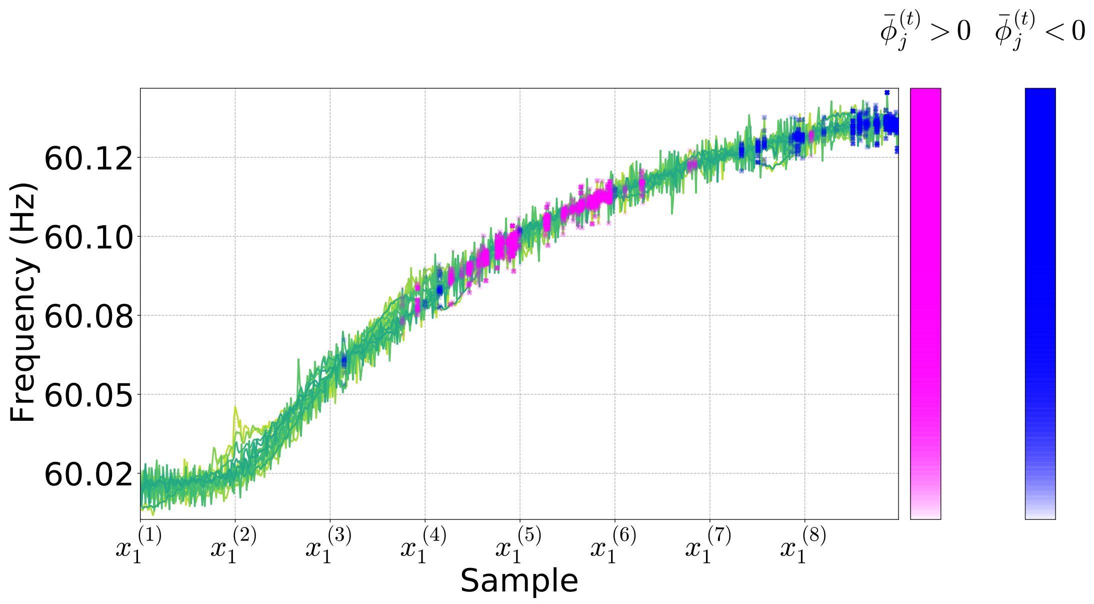
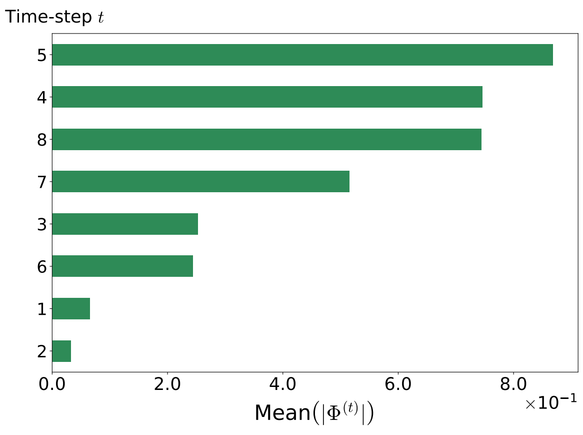
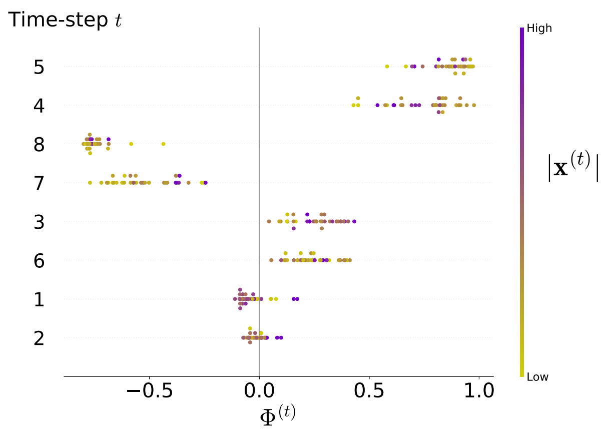


Figure 33 – Event 33. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f(x)]_{\text{LS}} = 0.1958$.

2) Global Explanation: The global explanation of LSTM for the LS events is presented in Figure 34. The time-steps $t = 5, 4, 8$, and $t = 7$ have the greater contributions, in which $t = 5, 4$ are mostly positive and $t = 8, 7$ are mostly negative (see Figure 34b). In these events, the contributions from $t = 8, 7$ are always negative, independently of the $\left|\boldsymbol{x}^{(t)}\right|, t = 8, 7$ values. The contributions $\Phi^{(t)}, t = 4, 5$ have always a positive contribution, independently of the $\left|\boldsymbol{x}^{(t)}\right|$ values, thus, there is not a clear correlation with the $\left|\boldsymbol{x}^{(t)}\right|$ values.



(a)



(b)

Figure 34 – Global Explanation of LS events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

6.3.1.3 LT Events

The LT events have the main characteristic of the spike frequency in the beginning of the disturbance. As described in Subsection 2.4.1, after the spike the frequency can rise up, go down or stay in steady state.

1) Local Explanation: The LT event selected (Event 39) is shown as follows. The force plot of this event is presented in Figure 35, given the base value of $E[f(x)]_{\text{LT}}$. The event was classified with a probability of P(LT) = 100%. The time-series of the event are presented in Figure 36, highlighting the main contributions. It can be notice from Figure 36 that the spikes and the nominal frequency samples having the most contribution to the identification. However, the higher contributions are from the steady state part (time-steps $t = 5, 4$) after the disturbance. This is a problematic issue because according to the domain knowledge of this event the most important contributions should be from the spike's frequency.

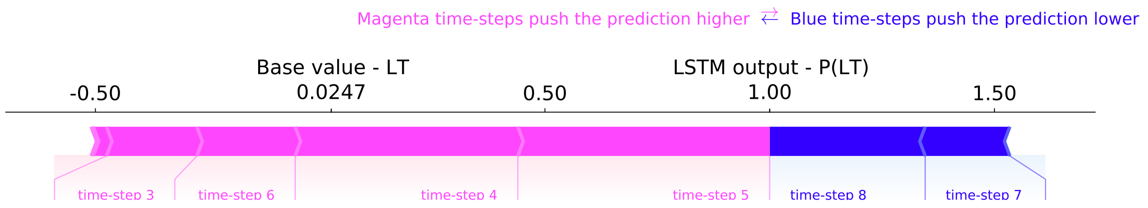


Figure 35 – Force plot of LSTM for Event 39.

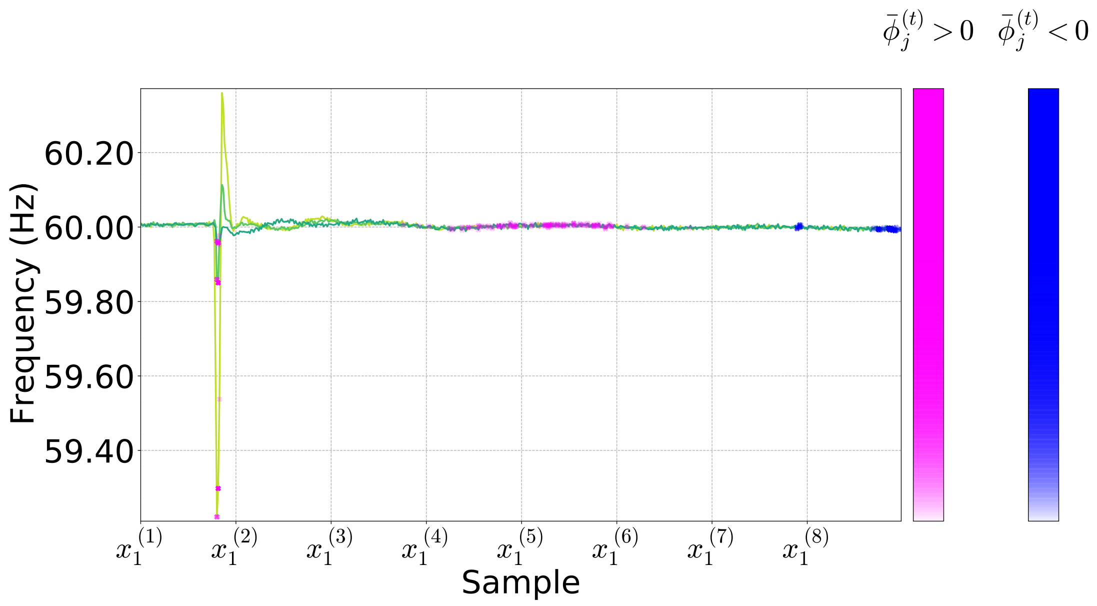


Figure 36 – Event 39. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f(x)]_{\text{LT}} = 0.0247$.

2) Global Explanation: The global explanation of LSTM for the LT events is presented in the Figure 37. According to Figure 37b the time-steps $t = 5$, and 4 are the ones with greater contributions, and both of these time-steps are positive. The spike, represented by the time-step $t = 1$, has the lowest contribution and does not have any impact on identification of this type of event. Also, for the inspected LT events, the contributions $\Phi^{(t)}, t = 5, 4$ are higher with greater values of $\left|\boldsymbol{x}^{(t)}\right|, t = 5, 4$, respectively, in this case representing values close to the nominal frequency.

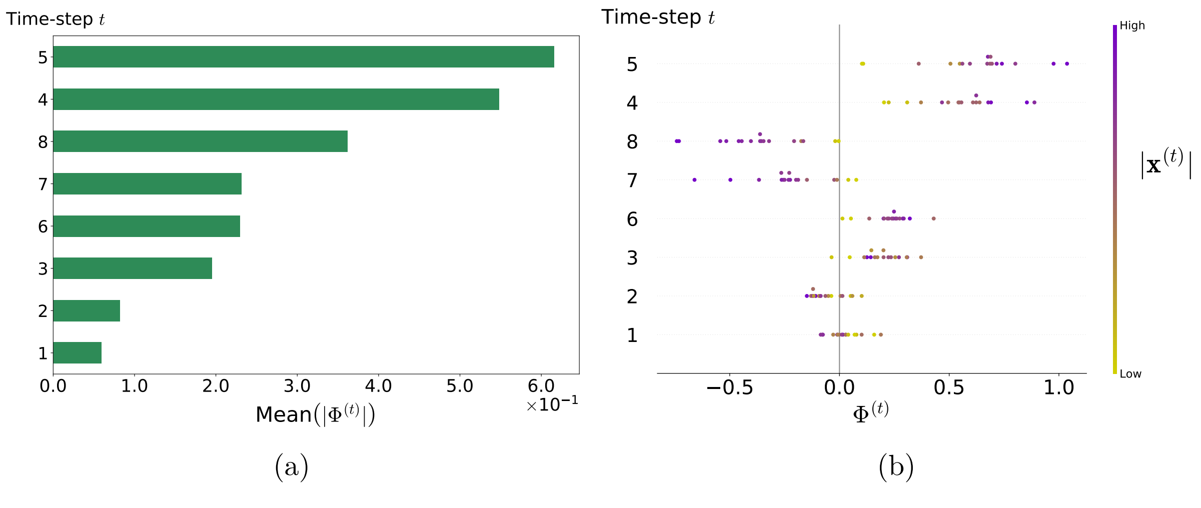


Figure 37 – Global Explanation of LT events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

This means that the classifier is learning to identify LT events by examining the steady state part after the disturbance. Therefore, this biased behavior could cause problems in other type of LT events that do not reach the steady state frequency operation after the disturbance. So, even though the LSTM classifier presents a high IAR and BA their performance behavior is biased and this can only be identified using the interpretability inspection.

6.3.1.4 OS Events

The OS events have the main characteristic of oscillation peaks (maximum and minimum) soon after the disturbance.

1) Local Explanation: The selected OS event is Event 48 that was correctly classified with a probability of P(OS) = 100%. The force plot of this Event 48 is presented in Figure 38, given base value of $E[f(x)]_{\mathrm{OS}}$. Also, the time-series of the event are presented in Figure 39, highlighting the main contributions $\bar{\phi}_j^{(t)}$. As presented, the time-steps $t = 2, 3$ are the ones with greater contributions. The samples with the most contribution to event identification are the oscillation peaks (time-steps $t = 2, 3$) after the spike disturbance.

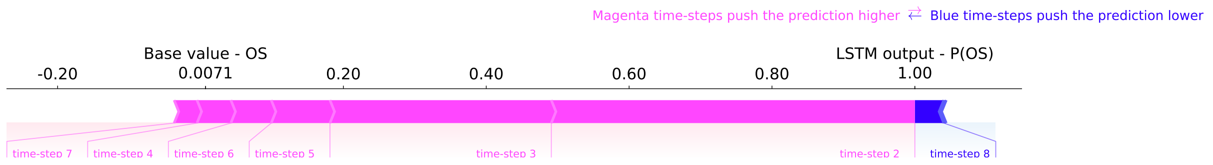


Figure 38 – Force plot of LSTM for Event 48.

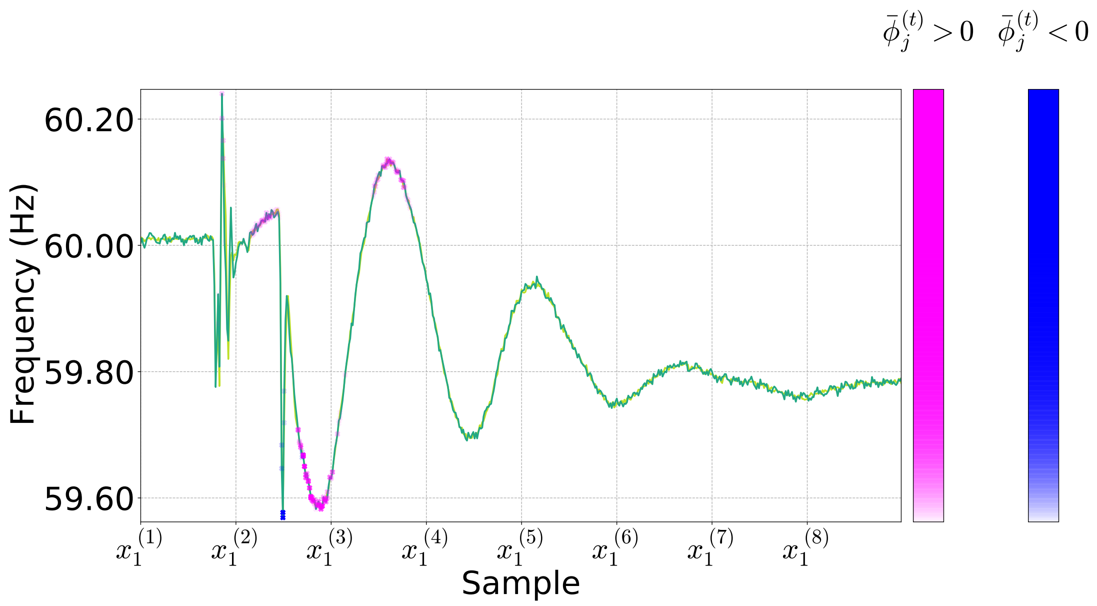
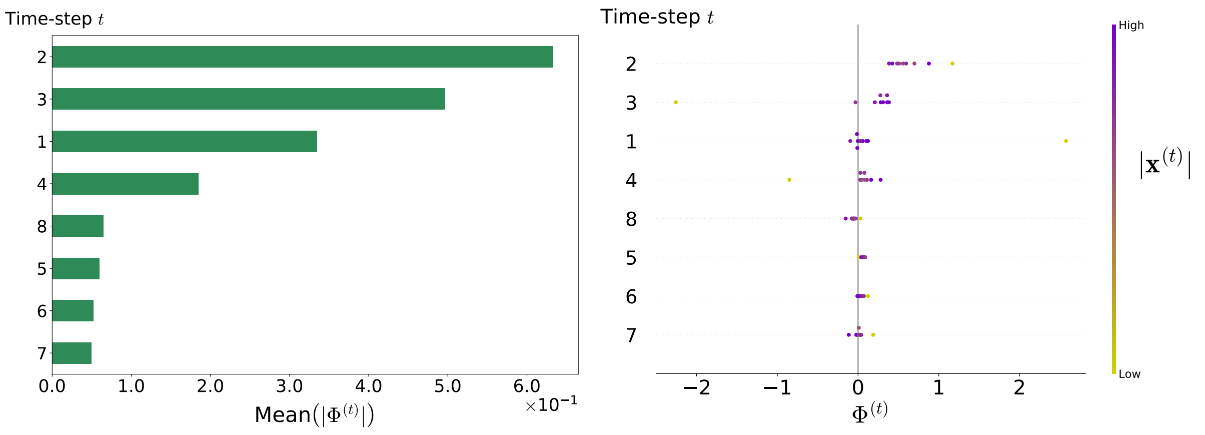


Figure 39 – Event 48. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f(x)]_{\mathrm{OS}} = 0.0071$.

2) Global Explanation: The global explanations of LSTM for OS events are presented in the Figure 40. Figure 40a displays the bar plot of average SHAP magnitude and Figure 40b exhibits beeswarm plot. According to the patterns of the Event 48, globally the most important time-steps are $t = 2, 3$ (oscillation peaks). Also, the contribution $\Phi^{(2)}$

increases with lower values of $\left|\boldsymbol{x}^{(2)}\right|$, representing that the contributions are higher for greater amplitudes in the oscillations.



(a) (b)

Figure 40 – Global Explanation of OS events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

6.3.2 Inspection of Misclassified Events

In this part, two misclassified events (Events 9 and 98) by the LSTM, presented in Table 8, will be discussed.

6.3.2.1 **Event 9**

The Event 9 was classified as LT with a probability of P(LT) = 100%, however this is an GT event. The SHAP values are computed using Equation 4.6 with the base value of $E[f(x)]_{\text{GT}}$. These contributions are related to the correct GT event affecting P(GT). The output probability is P(GT) = 0%. The force plot of this GT event is presented in Figure 41. Also, the contributions are displayed in Figure 42.

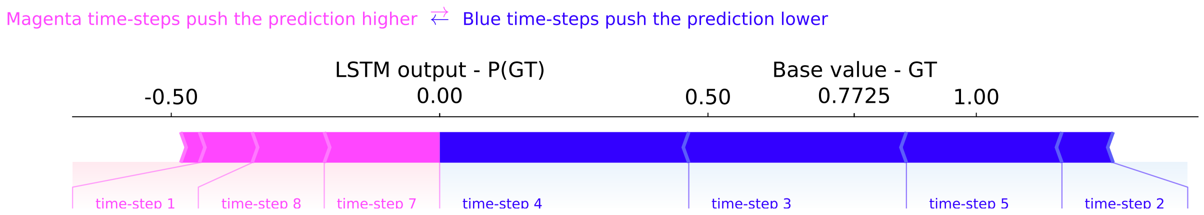


Figure 41 – Event 9. Force plot of the misclassified GT event.

It should be noted that the most relevant contributions $\bar{\phi}_j^{(t)}$ are negative. The negative contributions are the ones decreasing the probability P(GT). We also identified that these negative contributions are the positive contributions of the predicted class LT (Figure 43). Thus, we attribute this misclassification due to small deviation $\Delta f = 0.05\text{Hz}$.

It occurs in the time-steps $t = 4, 3$ and 5 close to a steady state behavior leading to incorrectly classify this event as LT.

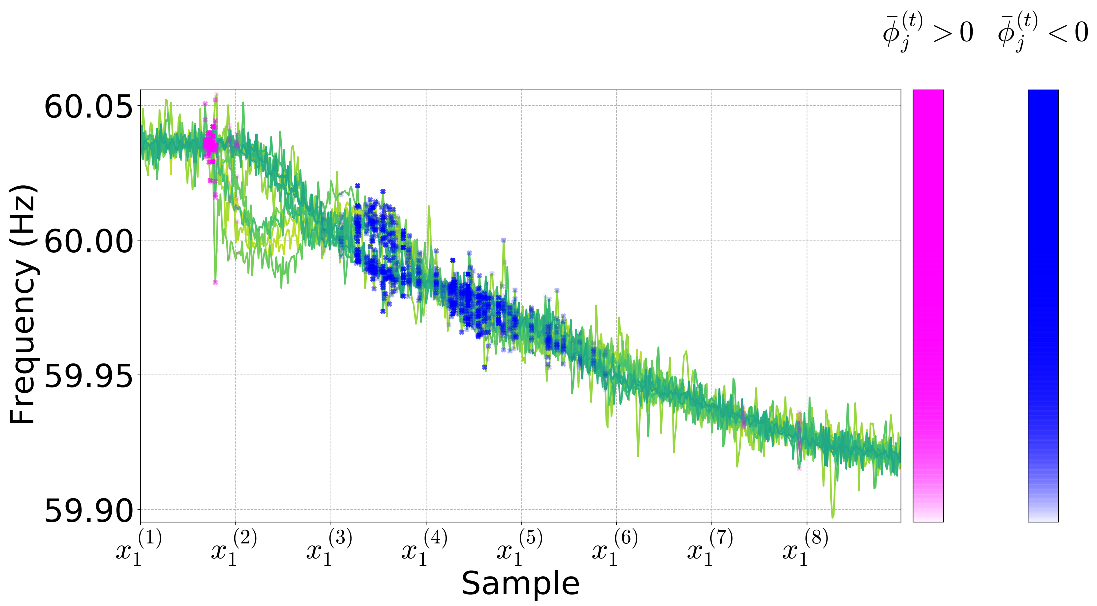


Figure 42 – Event 9. Misclassified GT event with the most relevant contributions highlighted. Base value of GT event $E[f(x)]_{\text{GT}} = 0.7725$.

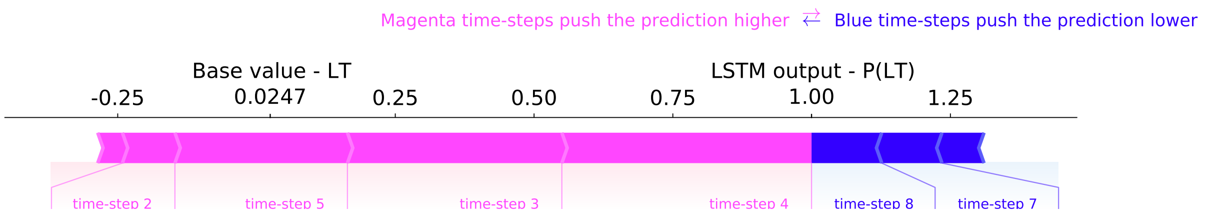


Figure 43 – Event 9. Force plot of the misclassified GT event.

6.3.2.2 **Event 98**

The Event 98 was classified as GT with a probability of P(GT) = 100%, however the correct class is LT. The SHAP values are computed with Equation 4.6 with the base value of $E[f(x)]_{\text{LT}}$. These contributions are related to identification of the correct event of LT, affecting P(LT). The output probability of LT is P(LT) = 0%. The force plot of this LT event is presented in Figure 44.

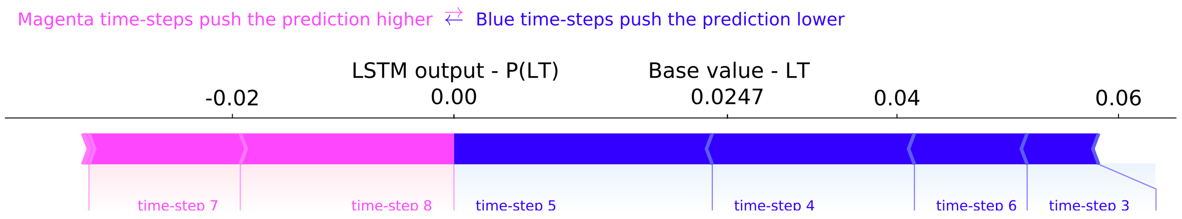


Figure 44 – Event 98. Force plot of the misclassified LT event.

Figure 45 displays the LT event, highlighting the main contributions. Also, Figure 46 exhibits the force plot of the predicted class of LT, presenting P(GT). It can be observed that fall in the frequency after the spike has a negative contribution in identification of LT, making the LSTM to classify the event as GT due to the deviation.

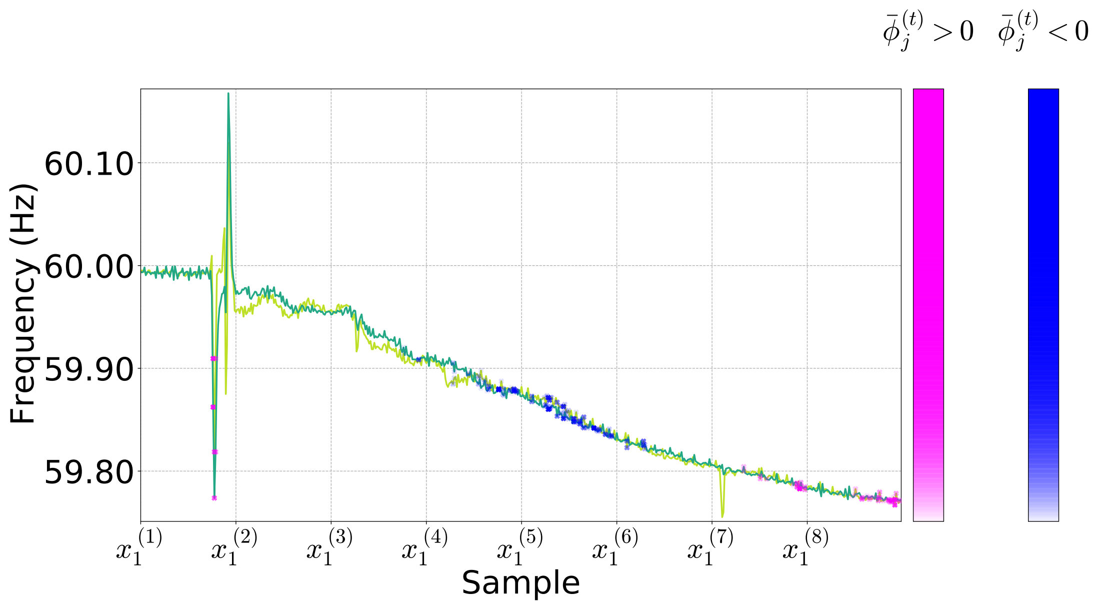


Figure 45 – Event 98. Misclassified LT event with the most relevant contributions highlighted. Base value of LT event $E[f(x)]_{\text{LT}} = 0.0247$.

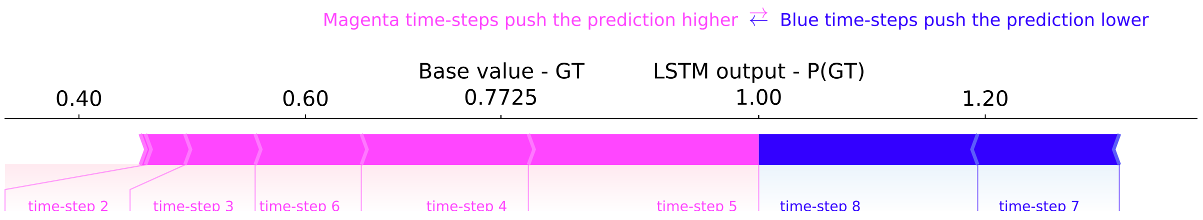


Figure 46 – Force plot of the Event 98. Presenting P(GT) and its base value $E[f(x)]_{\text{GT}} = 0.7725$.

One interesting point is that contribution of time-step t ($\Phi^{(t)}$), is the combination of samples contributions $\bar{\phi}_j^{(t)}$ in the same time-step t, and is more important in compared with the samples contributions $\bar{\phi}_j^{(t)}$.

6.4 Improvements of LSTM based on the Interpretability Inspection

Based on the inspection of the correct and incorrect classified events, some improvements can be applied in order to improve the performance of the classifier. First, the main identified problem is the bias for LT events, since based on EPS domain knowledge the spikes are supposed to be the most relevant parts in the identification of this type of event in compared with steady state part of time window. Second, the initial and final time-steps of the time-series are not useful for the classification of any event.

One clear strategy is reduction of the time-window. The following improvements are proposed to re-train the LSTM classifier: 1) the pre-time (T_{pre}) be reduce to 0.25s (15 samples). 2) the pos-event time (T_{pos}) be reduce from 9s to 1.75s (105 samples), as shown in Figure 47.

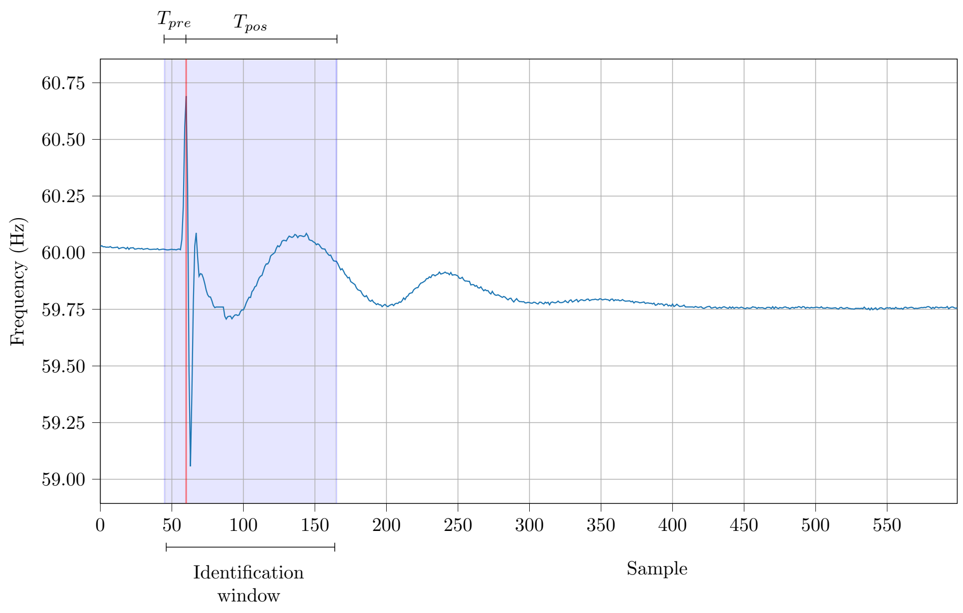


Figure 47 – New Identification Window after SHAP Inspection.

This new LSTM classifier named LSTM-SHAP now is trained by not only looking at IAR of Test-set, but also inspecting the SHAP values and analyzing the obtained contributions. Therefore, we try different hyper-parameters in multiples initial conditions. The IAR of Test-set for the obtained models are compared along with SHAP values, selecting the model with the higher IAR and contributions and mainly considering the spike to identify LT events. The hyper-parameters of the LSTM-SHAP classifier are presented at Table 11. The IARs of the LSTM and LSTM-SHAP are compared in Table 12, and misclassifications in Table 13. The results in Figure 48 indicate the IAR in the test-set of the LSTM-SHAP classifier as a function of τ. As presented, the LSTM-SHAP classifier has the best performance with $\tau = 10$. The misclassification labeled as event 7 will be inspected later.

Table 11 – Best Hyper-parameters of LSTM-SHAP classifier.

n_h	neurons	dropout rate	α
3	60|50|80	[0.1, 0.5, 0.5]	0.55

Table 12 – Performances of LSTM, LSTM-SHAP.

Classifier	GT%	LS%	LT%	OS%	Overall %	BA%
LSTM	97.959	100.0	94.444	100.0	98.182	98.101
LSTM-SHAP	97.959	100.0	100.0	100.0	99.490	99.091

Table 13 – Misclassifications of LSTM, and LSTM-SHAP.

Classifier	GT	LS	LT	OS	Total
LSTM	1	0	0	1	2
LSTM-SHAP	1	0	0	0	1

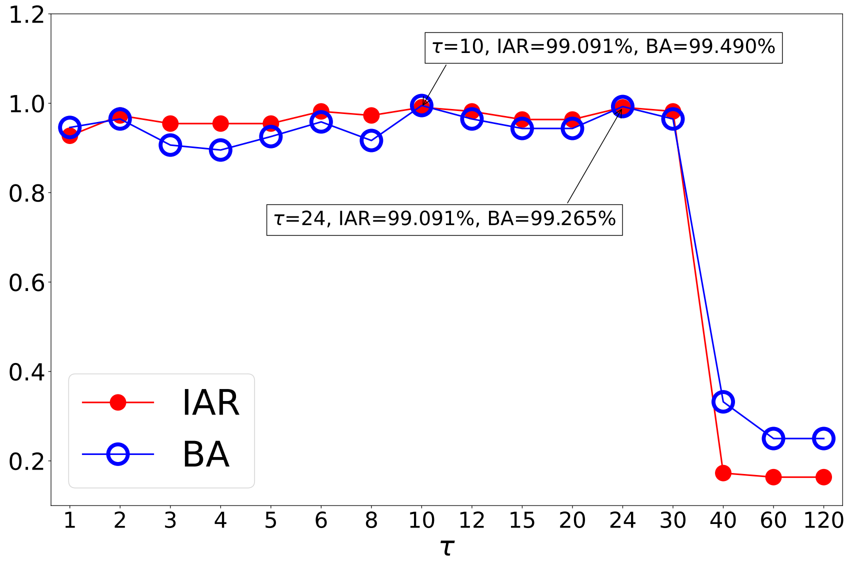


Figure 48 – Performances of LSTM-SHAP classifier in relation to the number of times-steps τ. Best performance at $\tau = 10$.

In the same way as to the inspection executed for the LSTM classifier we are going to inspect the predictions of the LSTM-SHAP classifier. Again, the base values of LSTM-SHAP $E[f_\phi(x)]$ (Table 14) are computed over the background dataset.

Table 14 – Base values $E[f_\phi(x)]$ of LSTM-SHAP.

$E[f_\phi(x)]_{\mathrm{GT}}$	$E[f_\phi(x)]_{\mathrm{LS}}$	$E[f_\phi(x)]_{\mathrm{LT}}$	$E[f_\phi(x)]_{\mathrm{OS}}$
0.7724	0.1956	0.0249	0.0071

6.4.1 Inspection of the Events using LSTM-SHAP

In this section, the same events discussed for the LSTM classifier are now presented using the LSTM-SHAP classifier.

6.4.1.1 GT Events

1) Local Explanation: The force plot of the Event 3 is presented in Figure 49, given base value of $E[f_\phi(x)]_{\mathrm{GT}}$. The event was correctly classified with a probability of P(GT) = 100%. Also, Figure 50 displays the Event 3, highlighting the main sample contributions. The $\Phi^{(t)}$ values were calculated using Equation 5.5 with $\tau = 10$ and $m = \frac{120}{10} = 12$. The parts of the time-series with most contribution to classification of this event still face the downfall in the frequency, identifying the deviation or drop similar to LSTM classifier. Although, the time-step $t = 9$ had the negative values for this case and did not affect the prediction P(GT).

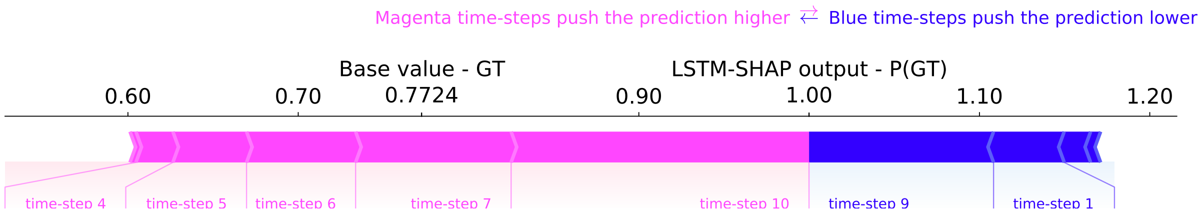


Figure 49 – Force plot of LSTM-SHAP for Event 3.

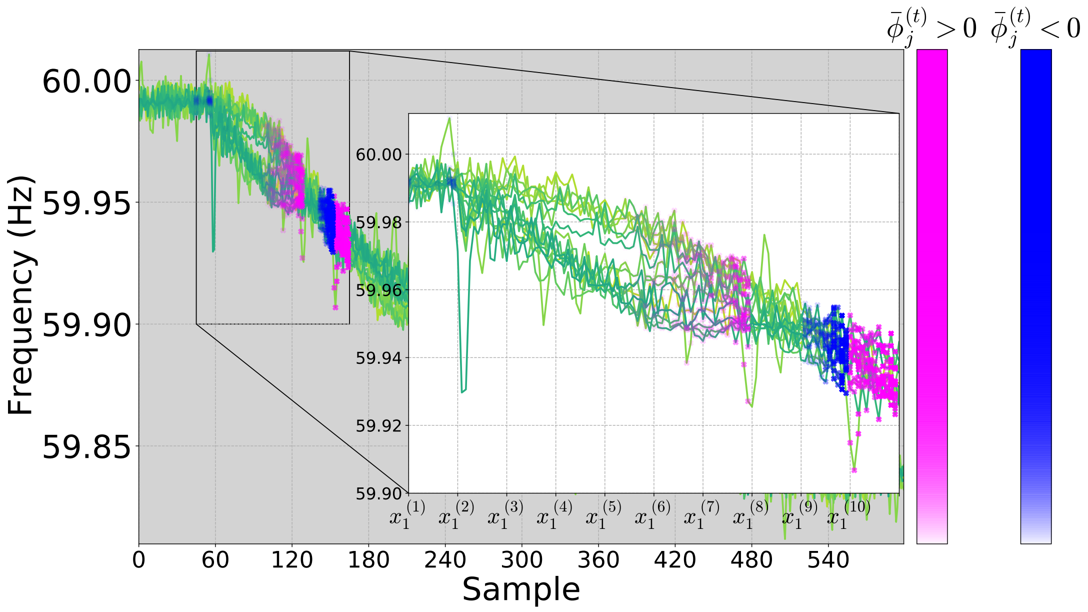
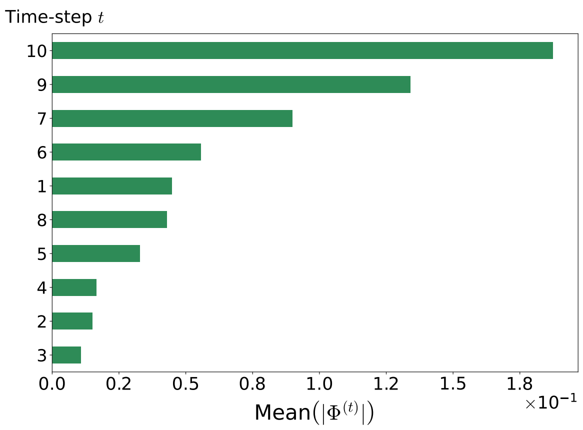


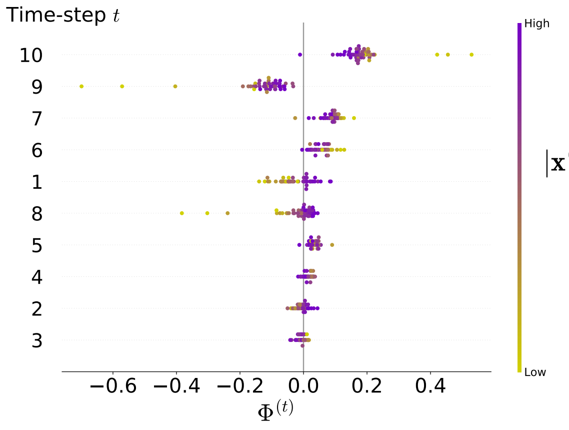
Figure 50 – Event 3. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f_\phi(x)]_{\mathrm{GT}} = 0.7724$.

2) Global Explanation: The global explanation of LSTM-SHAP for GT events are presented in Figure 51. Figure 51a exhibits the bar plot of average SHAP magnitude and Figure 51b presents beeswarm plot. The time-steps $t = 10, 9, 7$ have the most important contributions, where time-steps $t = 10, 7$ are positively captured the downfall in the frequency. Also, the events with lower values of $\left|\boldsymbol{x}^{(10)}\right|$ (in blue) have greater values of $\Phi^{(10)}$, which is coherent with domain knowledge of GT event.

According to the beeswarm plot, it can be observed that the time-step $t = 9$ hs a negative contribution, and that the contributions $\Phi^{(9)}$ increase with higher values of $\left|\boldsymbol{x}^{(9)}\right|$. This inconsistency could cause misclassifications for events with higher Δf, depending on the values of other time-steps $t = 7, 6$. As will be discussed later, this is the reason for the misclassification of the Event 7.



(a)



(b)

Figure 51 – Global explanation of LSTM-SHAP for GT events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

6.4.1.2 LS Events

1) Local Explanation: The force plot of the Event 33 is presented in Figure 52, given base value of $E[f_\phi(x)]_{\mathrm{LS}}$. This event was correctly classified with a probability of $\mathrm{P(LS)} = 100\%$ and Figure 53 displays the time evolution, highlighting the main sample contributions. It is clear that the main contributions to the identification of this event are again the rise in the frequency, represented by time-steps $t = 7, 10$, has the time-step $t = 9$ are the negative contributions, not affecting the prediction P(GT).

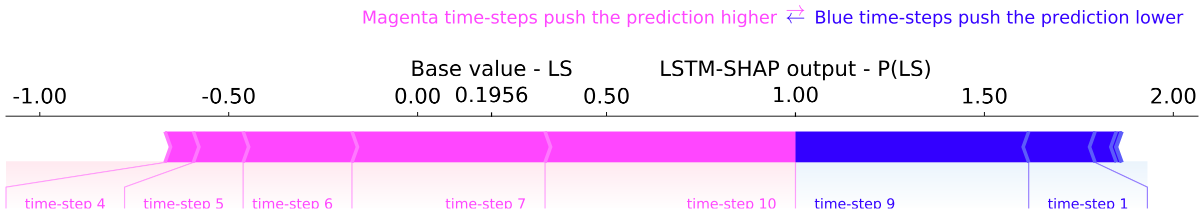


Figure 52 – Force plot of LSTM-SHAP for Event 33.

2) Global Explanation: The global explanation of LSTM-SHAP for LS events is presented in Figure 54. Figure 54a exhibits the bar plot of average SHAP magnitude, and Figure 54b presents beeswarm plot using the LSTM-SHAP. The time-steps $t = 10, 9, 7$ have the most important contributions, where time-steps $t = 10, 7$ are positively captured the rise in the frequency. Furthermore, the SHAP values of $\Phi^{(10)}$ and $\Phi^{(7)}$ increase with the rise in $\left|\boldsymbol{x}^{(10)}\right|$ and $\left|\boldsymbol{x}^{(7)}\right|$, respectively.

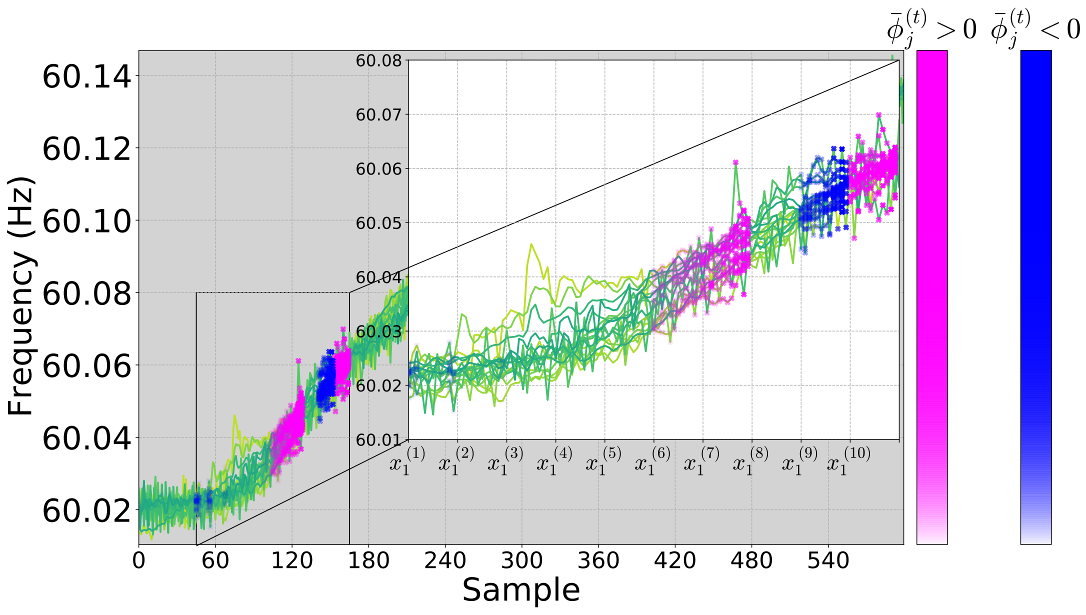


Figure 53 – Event 33. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f_\phi(x)]_{\text{LS}} = 0.1956$.

The time-step $t = 9$ presents in general negative contributions, and the magnitude of $\Phi^{(9)}$ decreases with increase in values of $\left|\boldsymbol{x}^{(9)}\right|$. Therefore, in the same way as to GT events (see Figure 51), the time-step $t = 9$ also has an inconsistency, because it does not follow the similar patterns of the time-step $t = 10$, which was expected since they are close.

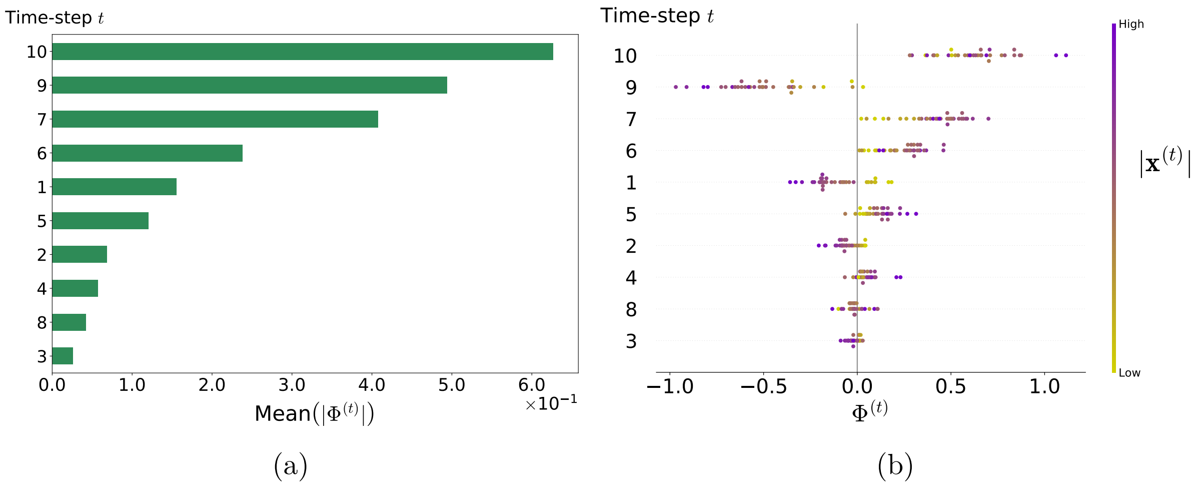


Figure 54 – Global explanation of LSTM-SHAP for LS events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

6.4.1.3 LT Events

1) Local Explanation: The force plot for Events 39 is presented in Figure 55, given the base value of $E[f_\phi(x)]_{\text{LT}}$. This event was correctly classified with a probability P(LT) of 78%. Also, the time-series of this event is presented in Figure 56 showing that the samples with most contribution to the correct identification are now the spikes (time-step $t = 2$ and $t = 3$). However, there are still contributions from the time-steps $t = 10, 8$. This inspection shows that the predictions of LSTM-SHAP are different from LSTM, which identified the LT from the ambient data part. Now, the spikes play a significant role in the identification of LT.

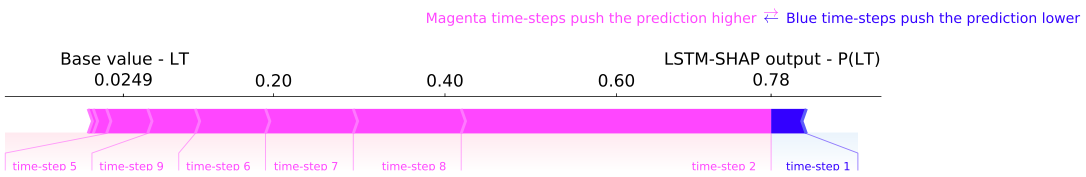


Figure 55 – Force plot of LSTM-SHAP for Event 39.

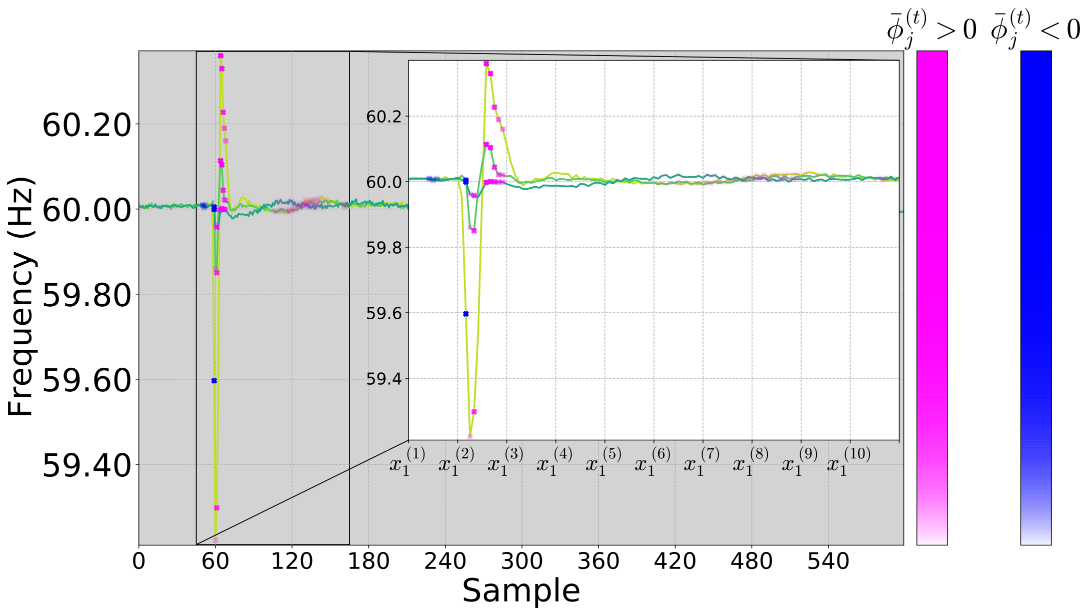


Figure 56 – Event 39. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f_\phi(x)]_{\text{LT}} = 0.0249$.

2) Global Explanation: Figure 57a displays the bar plot of average SHAP magnitude, and Figure 57b presents beeswarm plot using the LSTM-SHAP. The most important time-steps contributions are $t = 10, 2, 3$, mostly with positive contributions. The time-step $t = 10$ represents the transitory after spike with the greatest contribution. Also, the spikes (time-step $t = 2$ and $t = 3$) are now having a relevant contribution to the identification minimizing the bias presented to LSTM for the LT event, because as presented in Figure 37 the spikes had the lowest contribution.

6.4.1.4 OS Events

1) Local Explanation: The Event 48 is inspected using following procedure. The event was correctly classified with a probability of P(OS) = 100%, and the force plot of this event is presented in Figure 58, given the base value of $E[f_\phi(x)]_{\text{OS}}$. Also, Figure 59

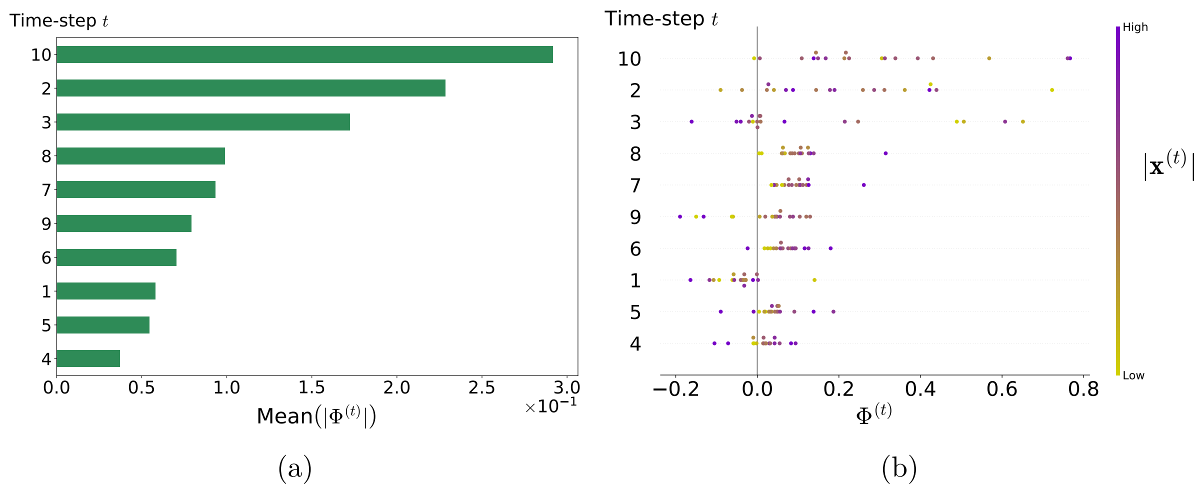


Figure 57 – Global explanation of LSTM-SHAP for LT events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

displays the time-series of Event 48, highlighting the main sample contributions $\bar{\phi}_j^{(t)}$. The samples with the most contribution to correct identification face the minimum oscillation peak soon after the disturbance, representing by time-steps $t = 8, 9$ (Figure 58).

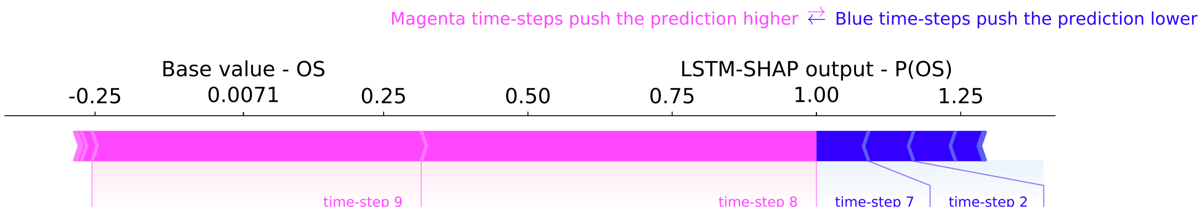


Figure 58 – Force plot of LSTM-SHAP for Event 48. Base value $E[f_\phi(x)]_{\mathrm{OS}} = 0.0071$

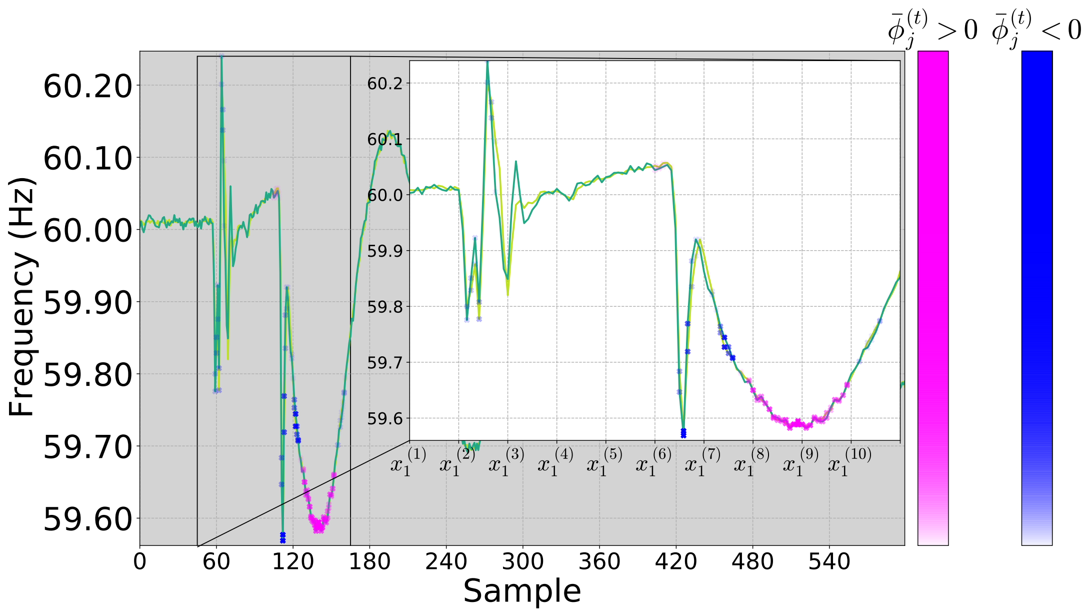


Figure 59 – Event 48. Frequency PMUs signals with the most relevant contributions highlighted. Base value $E[f_\phi(x)]_{\mathrm{OS}} = 0.0071$.

2) Global Explanation: The global explanation of LSTM-SHAP for OS events are presented in the Figure 60. It is clear that, the most important time-steps are $t = 3, 5$ and 8, where the time-steps $5, 8$ have positive contributions related to the peaks. In most cases $t = 5$ and $t = 8$ are related to first peak (zenith) to final peak of the window (nadir), respectively. The time-step $t = 3$ presents some very high negative SHAP values (Figure 60b) that push the absolute of global mean upward, this shows the reason that time-step $t = 3$ presents the highest mean contribution. As observed from other OS events, this is related to the spikes that usually presenting in the oscillations events, since the spikes are important to identify LT and they contribute to push P(OS) down. Also, the negative contributions of $\Phi^{(3)}$ increase in magnitude with higher values of $\left|\boldsymbol{x}^{(3)}\right|$ (higher spikes in frequency). An example of correctly classified as OS is Event 58, presented in Figure 61, with the force plot displayed in Figure 61b. This time-step $t = 3$ represents the spike with a negative contribution. Also, the time-steps $t = 5, 6$ represent the zenith which are relevant part for the correct classification.

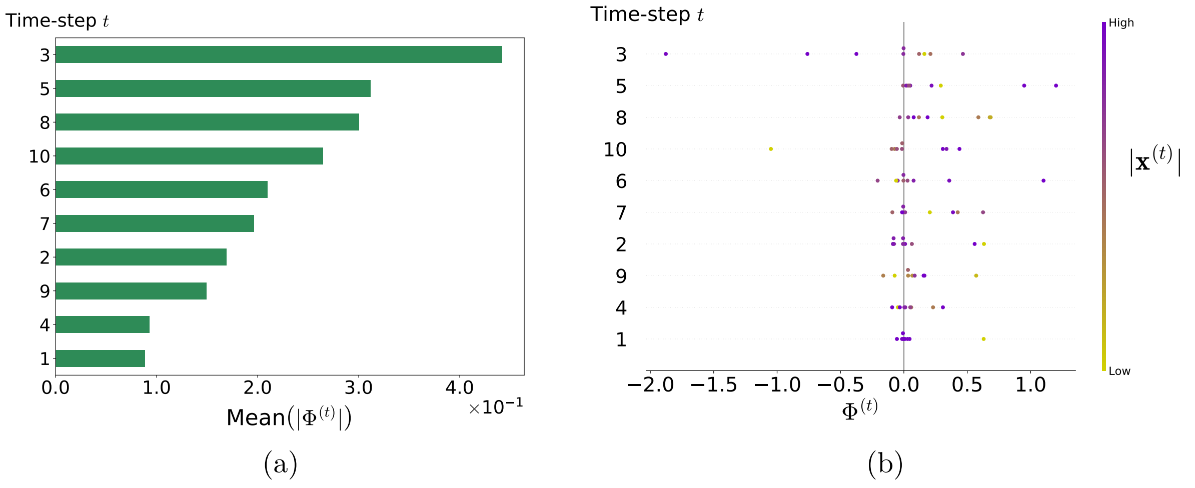
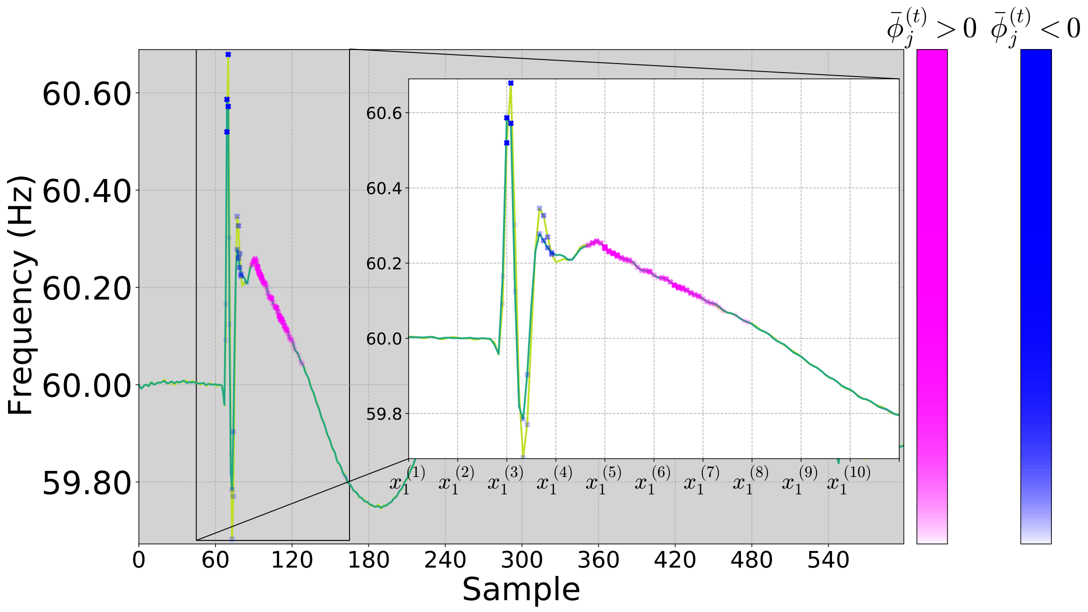


Figure 60 – Global explanation of LSTM-SHAP for OS events. (a) Bar chart of average $\Phi^{(t)}$ magnitude. (b) Beeswarm plot of $\Phi^{(t)}$.

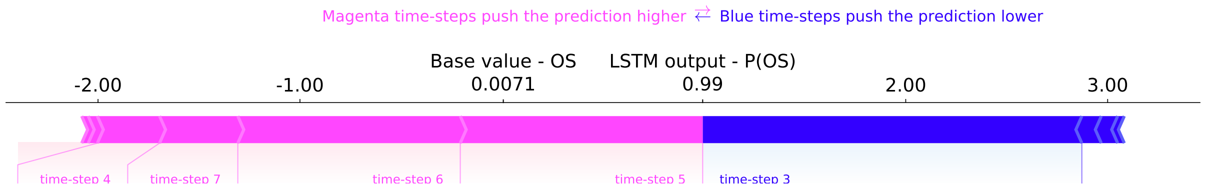
In general, the LSTM-SHAP classifier is learning to identify the OS only by the lowest peak at the final step of the identification window. Even though the time-step $t = 3$ has a great negative contribution, this will not affect the P(OS) in most OS events. Briefly, the summation of other positive contributions overcome this negative contribution.

6.4.2 Inspection of the Misclassified Event (Event 7) of LSTM-SHAP

The Event 7 was classified as OS with a probability of P(OS) = 54%, however the correct class is GT. The $\Phi^{(t)}$ are computed using Equation 4.6 using the base value $E[f_\phi(x)]_{\text{GT}} = 0.7724$. These contributions are related to the identification of the correct class GT affecting P(GT). The output probability for PG is P(GT) = 46%. The force plot



(a)



(b)

Figure 61 – Event 58. (a) Frequency PMUs signals with the most relevant contributions highlighted. (b) Force plot of LSTM-SHAP for this event.

of this GT event is presented in Figure 62. The Event 7 is presented in the Figure 63, highlighting the main sample contributions.

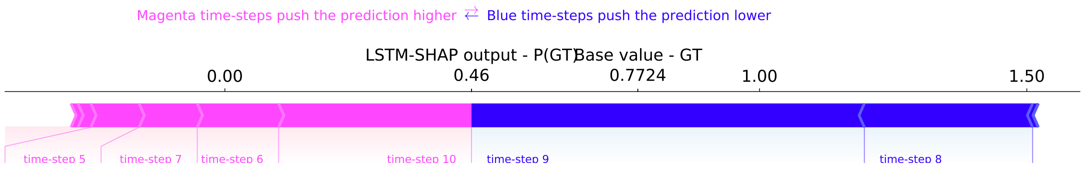


Figure 62 – Event 7. Force plot of LSTM-SHAP for the misclassified GT event.

The time-steps $t = 10, 6, 7$ have positive contribution for classification of the event as GT, but the combined contribution of these time-steps had limitations to overcome the negative contributions (Figure 62). The time-steps $t = 8, 9$, and especially the $t = 9$, present a high negative contribution that push P(GT) downward this is due to high deviation of Δf that presents lower values of $\left|\boldsymbol{x}^{(9)}\right|$ (Figure 51).

In a practical power system operation, this type of GT with a deviation of $\Delta f =$ 0.8Hz is extremely rare and unusual. So, this inconsistency is not very problematic for most GT events, making the LSTM-SHAP classifier an appropriate candidate for most

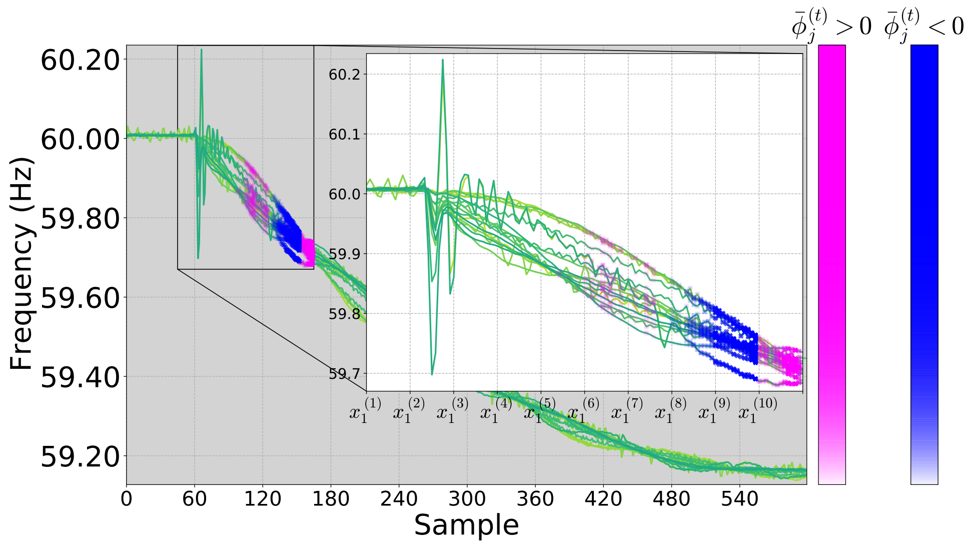


Figure 63 – Event 7. Misclassified GT event with the most relevant contributions highlighted. Base value of GT event $E[f_\phi(x)]_{\text{GT}} = 0.7724$.

real cases. Also, if others GT events have a deviation greater or equal to 0.8Hz we have a good indication in not trusting the prediction.

6.5 Discussion

In this section, the main outcomes of this study are discussed:

Even though, the LSTM classifier presented a high IAR of 98.182%, the decision-makings for LT events were biased. Because based on the domain knowledge of power system the spikes were supposed to be the most relevant parts in the identification of this type of event in compare with the steady state part of time window.

The LSTM-SHAP presented a higher BA than LSTM, but above all else the predictions are more coherent with the knowledge of power system events. Furthermore, we were able to minimize the bias for LT in the LSTM classifier. However, there is still an inconsistency with GT events presented in time-step $t = 9$. According to our observations, these types of inconsistencies were normal in our LSTM models and generally related to the misclassifications. But the positive side is that SHAP inspection allows us to discover these inconsistencies and biases, and as a result be aware about the limitations of these models. This inconsistency can also be solved (or minimize) using the same procedure described in Section 6.4 now focusing on this inconsistency.

The main difficult that was observed from LSTM-SHAP was the hard-work in obtaining the classifier by only changing hyper-parameters and initialization values. Thus, a mechanism of introducing the domain knowledge into the loss function (Equation 3.9) as constraints using the SHAP values should be a good enhancement to be studied in future applications.

7 Conclusion and Future Works

7.1 Conclusions

In this dissertation a methodology for identification of events using an explainable LSTM classifier was presented. The performance of this method was evaluated using real events acquired in the BIPS by the low-voltage WAMS prototype of the MedFasee Project. This methodology is based on the SHAP values method, more specifically with the DeepSHAP. In summary, the following conclusions can be drawn based on the findings of this research:

- The identification of the LSTM, evaluated using the IAR and BA, had superior performance compared to MSVM, and MLP;
- The interpretability inspection was very useful for understanding and certification of LSTM coherent performance. Moreover, by applying the SHAP values identification of the most relevant parts of the time-series were achieved by ranking the contributions $\Phi^{(t)}$;
- The SHAP values method had the ability to provide consistencies explanations about the LSTM. Therefore, the explanations were in accordance with the domain knowledge about the event identification problem;
- For each event type (GT, LS, LT, and OS) the SHAP inspection was applied providing local and global explanations. The local explanations were capable of explaining individuals events, yielding the most important contributions $\Phi^{(t)}$ for that specific event. Nonetheless, using the global explanations we were able to generally identify in general the most relevant time-steps t for each event type.
- Also using the global explanations we identified interesting correlation and interrelationship between the inputs of the LSTM ($\boldsymbol{x}^{(t)}$) and the contributions $\Phi^{(t)}$. These correlations were essential for identification of predictions that were in accordance with the domain knowledge of the problem, and biased predictions;
- The decision-makings of the LSTM for LT events, based on the domain knowledge of power system events, were biased. These biased decisions happened even though the LSTM classifier presented high identification indices (IAR of 98.182% and BA% of 98.101%). This exhibits that identification performance indices alone are not enough for power system events;

- By knowledge extracted from the inspection, we were able to detect the bias of the LSTM classifier for LT events. Moreover, the classifier was re-trained with improvements in the input data making a new model LSTM-SHAP. This new classifier not only had a greater IAR and BA, but also had a more consistent and coherent performance, correcting the detected bias;
- The new classifier LSTM-SHAP, obtained after the SHAP inspection, had identification performance that overcame the performance of the LSTM classifier, and consequently MSVM and MLP.

7.2 Future Work

It is recommended to:

1. Develop a methodology in multiple event for detection, segmentation and identification of several multiple successive events in real-time;
2. Establish deeper studies about inconsistencies of the LSTM, and identifying the main causes and solutions;
3. Incorporate the SHAP values into the loss function of the LSTM, introduce the domain knowledge as constraints, automatize the training process described in Figure 26, and improve the coherence of predictions;

8 References

ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>. Cited in page 39.

AMBONI, M. K. et al. Alocação do sobrecusto operativo de sistemas de energia elétrica via teoria dos jogos. Florianópolis, SC, 2001. Cited 2 times in pages 44 and 45.

BACH, S. et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, Public Library of Science, v. 10, n. 7, p. e0130140, 2015. Cited in page 43.

BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 8, p. 1798–1828, 2013. Cited in page 20.

BYKHOVSKY, A.; CHOW, J. H. Power system disturbance identification from recorded dynamic data at the northfield substation. *International journal of electrical power & energy systems*, Elsevier, v. 25, n. 10, p. 787–795, 2003. Cited 3 times in pages 26, 29, and 32.

CHATTERJEE, S. et al. Detection of non-technical losses using advanced metering infrastructure and deep recurrent neural networks. In: IEEE. *2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. [S.l.], 2017. p. 1–6. Cited in page 21.

CHEN, H.; LUNDBERG, S.; LEE, S.-I. Explaining models by propagating shapley values of local components. *arXiv preprint arXiv:1911.11888*, 2019. Cited 5 times in pages 9, 10, 51, 52, and 53.

CUBUK, E. D. et al. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. Cited in page 39.

DATTA, A.; SEN, S.; ZICK, Y. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In: IEEE. *2016 IEEE symposium on security and privacy (SP)*. [S.l.], 2016. p. 598–617. Cited in page 47.

DECKER, I. et al. Performance of a synchronized phasor measurements system in the brazilian power system. In: IEEE. *Power Engineering Society General Meeting, 2006. IEEE*. [S.l.], 2006. p. 8–pp. Cited in page 26.

DECKER, I. C. et al. Experience and applications of phasor measurements to the brazilian interconnected power system. *European Transactions on Electrical Power*, Wiley Online Library, v. 21, n. 4, p. 1557–1573, 2011. Cited 3 times in pages 26, 28, and 35.

FOWLKES, E. B.; MALLOWS, C. L. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, Taylor & Francis Group, v. 78, n. 383, p. 553–569, 1983. Cited in page 32.

GAL, Y.; GHAHRAMANI, Z. A theoretically grounded application of dropout in recurrent neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2016. p. 1019–1027. Cited in page 42.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. [S.l.: s.n.], 2010. p. 249–256. Cited in page 42.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Cited in page 36.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Cited 2 times in pages 21 and 38.

JENA, M. K.; PANIGRAHI, B. K.; SAMANTARAY, S. R. A new approach to power system disturbance assessment using wide-area postdisturbance records. *IEEE Transactions on Industrial Informatics*, IEEE, v. 14, n. 3, p. 1253–1261, 2018. Cited in page 20.

KIM, D.-I. et al. Wavelet-based event detection method using pmu data. *IEEE Transactions on Smart grid*, IEEE, v. 8, n. 3, p. 1154–1162, 2017. Cited in page 32.

KONG, W. et al. Short-term residential load forecasting based on lstm recurrent neural network. *IEEE Transactions on Smart Grid*, IEEE, v. 10, n. 1, p. 841–851, 2017. Cited in page 21.

KONONENKO, I. et al. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, v. 11, n. Jan, p. 1–18, 2010. Cited in page 38.

LI, F.; DU, Y. From alphago to power system ai: What engineers can learn from solving the most complex board game. *IEEE Power and Energy Magazine*, IEEE, v. 16, n. 2, p. 76–84, 2018. Cited in page 20.

LI, W.; WANG, M. Identifying overlapping successive events using a shallow convolutional neural network. *IEEE Transactions on Power Systems*, IEEE, 2019. Cited in page 21.

LI, W.; WANG, M.; CHOW, J. H. Real-time event identification through low-dimensional subspace characterization of high-dimensional synchrophasor data. *IEEE Transactions on Power Systems*, IEEE, v. 33, n. 5, p. 4937–4947, 2018. Cited in page 35.

LIPOVETSKY, S.; CONKLIN, M. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, Wiley Online Library, v. 17, n. 4, p. 319–330, 2001. Cited in page 47.

LUNDBERG, S. M. et al. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, v. 2, n. 1, p. 2522–5839, 2020. Cited 2 times in pages 54 and 65.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2017. p. 4765–4774. Cited 8 times in pages 9, 22, 43, 46, 47, 48, 49, and 50.

LUNDBERG, S. M. et al. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature biomedical engineering*, Nature Publishing Group, v. 2, n. 10, p. 749, 2018. Cited 2 times in pages 45 and 46.

MARTIN, K. E. et al. IEEE standard for synchrophasors for power systems. *IEEE Transactions on Power Delivery*, IEEE, v. 13, n. 1, p. 73–77, 1998. Cited in page 27.

MIRANDA, V. et al. Through the looking glass: Seeing events in power systems dynamics. *International Journal of Electrical Power & Energy Systems*, Elsevier, v. 106, p. 411–419, 2019. Cited in page 20.

MOLNAR, C. *Interpretable Machine Learning*: A guide for making black box models explainable. [S.l.: s.n.], 2019. <https://christophm.github.io/interpretable-ml-book/>. Cited 5 times in pages 9, 22, 43, 48, and 50.

MONTAVON, G. et al. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, Elsevier, v. 65, p. 211–222, 2017. Cited in page 22.

MOU, L.; GHAMISI, P.; ZHU, X. X. Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, IEEE, v. 55, n. 7, p. 3639–3655, 2017. Cited in page 21.

ONS, P. d. R. *PROCEDIMENTOS DE REDE*. 2020. Disponível em: <http://www.ons.org.br/paginas/sobre-o-ons/procedimentos-de-rede/vigentes>. Cited in page 29.

RAFFERTY, M. et al. Real-time multiple event detection and classification using moving window pca. *IEEE Transactions on Smart Grid*, IEEE, v. 7, n. 5, p. 2537–2548, 2016. Cited in page 20.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. Why should i trust you?: Explaining the predictions of any classifier. In: ACM. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. [S.l.], 2016. p. 1135–1144. Cited 2 times in pages 43 and 50.

SERRANO, R. *Cooperative games: Core and Shapley value*. [S.l.], 2007. Cited in page 44.

SHRIKUMAR, A.; GREENSIDE, P.; KUNDAJE, A. Learning important features through propagating activation differences. In: JMLR. ORG. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. [S.l.], 2017. p. 3145–3153. Cited 2 times in pages 43 and 51.

SONG, Y. et al. Multiple event detection and recognition for large-scale power systems through cluster-based sparse coding. *IEEE Transactions on Power Systems*, IEEE, v. 32, n. 6, p. 4199–4210, 2017. Cited in page 29.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Cited in page 42.

STRUMBELJ, E.; KONONENKO, I. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, JMLR. org, v. 11, p. 1–18, 2010. Cited in page 50.

ŠTRUMBELJ, E.; KONONENKO, I. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, Springer, v. 41, n. 3, p. 647–665, 2014. Cited in page 47.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2014. p. 3104–3112. Cited in page 21.

TIELEMAN, T.; HINTON, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, v. 4, n. 2, p. 26–31, 2012. Cited in page 42.

WANG, W. et al. Multiple event detection and recognition through sparse unmixing for high-resolution situational awareness in power grid. *IEEE Transactions on Smart Grid*, IEEE, v. 5, n. 4, p. 1654–1664, 2014. Cited in page 29.

WEN, S. et al. Real-time identification of power fluctuations based on lstm recurrent neural network: A case study on singapore power system. *IEEE Transactions on Industrial Informatics*, IEEE, v. 15, n. 9, p. 5266–5275, 2019. Cited 2 times in pages 21 and 29.

YADAV, R.; PRADHAN, A. K.; KAMWA, I. Real-time multiple event detection and classification in power system using signal energy transformations. *IEEE Transactions on Industrial Informatics*, IEEE, 2018. Cited in page 20.

YOUNG, T. et al. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, IEEE, v. 13, n. 3, p. 55–75, 2018. Cited in page 21.

YU, R. et al. Lstm-efg for wind power forecasting based on sequential correlation features. *Future Generation Computer Systems*, Elsevier, v. 93, p. 33–42, 2019. Cited in page 21.

ZAREMBA, W.; SUTSKEVER, I.; VINYALS, O. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014. Cited in page 42.

ZARZOSA, M.; ZIMMER, V.; DECKER, I. C. Classificador de eventos no sin baseado em redes neurais artificiais e sincrofasores. In: *VI Simpósio Brasileiro de Sistemas Elétricos*. [S.l.]: ResearchGate, 2016. Cited in page 29.

ZARZOSA, M. A. D. et al. *Análise de eventos em sistemas de energia elétrica usando sincrofasores*. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2016. Cited 2 times in pages 29 and 35.

ZAZO, R. et al. Age estimation in short speech utterances based on lstm recurrent neural networks. *IEEE Access*, IEEE, v. 6, p. 22524–22530, 2018. Cited in page 21.

ZHANG, S. et al. Data-based line trip fault prediction in power systems using lstm networks and svm. *IEEE Access*, IEEE, v. 6, p. 7675–7686, 2017. Cited in page 21.

ZIMMER, V. et al. Detecção, identificação e localização de eventos usando sincrofasores. 2013. Cited in page 29.

ZUO, L.-Q. et al. Natural scene text recognition based on encoder-decoder framework. *IEEE Access*, IEEE, v. 7, p. 62616–62623, 2019. Cited in page 21.