UNIVERSITÉ DE RENNES 1

Inria

THESIS PRESENTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN COMPUTER SCIENCE

University of Rennes

DOCTORAL SCHOOL N° 601
*Mathematics, Information and Communication Sciences*

by

**Kevin FAUVEL**

# Enhancing Performance and Explainability of Multivariate Time Series Machine Learning Methods: Applications for Social Impact in Dairy Resource Monitoring and Earthquake Early Warning

**Thesis Defended in Rennes, France, on October 13, 2020**
**Research Unit: LACODAM, Inria**

## Rapporteurs

| | |
|---|---|
| Fosca GIANNOTTI | Director of Research - CNR, Italy |
| Germain FORESTIER | Full Professor - UHA, France/Monash University, Australia |

## Jury Members

| | | |
|---|---|---|
| Chair | Sébastien LEFÈVRE | Full Professor - UBS/IRISA, France |
| Examiners | Fosca GIANNOTTI | Director of Research - CNR, Italy |
| | Véronique MASSON | Associate Professor - University of Rennes, France |
| | Germain FORESTIER | Full Professor - UHA, France/MU, Australia |
| | Aurélien MADOUASSE | Associate Professor - ONIRIS/INRAE, France |
| Thesis Co-Dir. | Philippe FAVERDIN | Director of Research - INRAE, France |
| Thesis Dir. | Alexandre TERMIER | Full Professor - Univ. Rennes/Inria, France |

# Enhancing Performance and Explainability of Multivariate Time Series Machine Learning Methods

*Applications for Social Impact in Dairy Resource Monitoring and Earthquake Early Warning*

By

Kevin Fauvel



Thesis Presented for the Degree of
Doctor of Philosophy in Computer Science

University of Rennes, France - October 2020

Rapporteurs

| | |
|---|---|
| Fosca GIANNOTTI | Director of Research - CNR, Italy |
| Germain FORESTIER | Full Professor - UHA, France/Monash University, Australia |

Jury Members

| | | |
|---|---|---|
| Chair | Sébastien LEFÈVRE | Full Professor - UBS/IRISA, France |
| Examiners | Fosca GIANNOTTI | Director of Research - CNR, Italy |
| | Véronique MASSON | Associate Professor - Univ. Rennes/IRISA, France |
| | Germain FORESTIER | Full Professor - UHA, France/MU, Australia |
| | Aurélien MADOUASSE | Associate Professor - ONIRIS/INRAE, France |
| Thesis Co-Dir. | Philippe FAVERDIN | Director of Research - INRAE, France |
| Thesis Dir. | Alexandre TERMIER | Full Professor - Univ. Rennes/Inria, France |

# Foreword

The research work presented in this thesis is the result of three collaborations. The first one concerns the dairy resource monitoring application and is a collaboration between the French National Institute for Research in Computer Science and Automation (Inria) and the French National Institute for Agriculture, Food and Environment (INRAE), under the French Digital Agriculture Convergence Lab (#DigitAg). On the same application, the second one is a collaboration that I have built between #DigitAg and Zhejiang University, China. The last one concerns the earthquake early warning application and it is a collaboration that I have jointly built between Inria and the National Science Foundation (NSF), United States. Information about the collaborations will be given in introduction of each research work.

This thesis was supported by the French National Research Agency under the Investments for the Future Program (ANR-16-CONV-0004).




Figure 1 – Logos of the collaborating institutions on the research work of this thesis

# Acknowledgements

I am deeply grateful to my advisors Philippe Faverdin, Véronique Masson and Alexandre Termier for guiding me through my PhD, and for a lifetime worth of sharp insights and supportive advices. Among other things, I would like to thank them for their constant sharing of experiences and the inspiring discussions, also for allowing me to explore different fields and trusting me with great responsibilities. I express my sincere gratitude to Élisa Fromont, Luis Galárraga and Thomas Guyet who always made time from their busy schedules to exchange with me and provided me with instrumental inputs on my research work.

And, a particular thank you to Germain Forestier and Fosca Giannotti for having accepted to evaluate my thesis and to all the members of my PhD defense jury for their invaluable feedback. I would also like to thank Marie-Pierre Etienne and Pascal Poncelet who have served on my thesis committee and provided precious advices and encouragements during my time at Inria.

I have had the privilege to work with numerous great researchers from different institutions during these last three years. I would like to thank the researchers from the PEGASE team at INRAE, France, and in particular Nicolas Bedere and Amélie Fischer who generously shared with me their domain knowledge. I would also like to thank Manish Parashar and his team who have welcomed me at the Rutgers Discovery Informatics Institute, United States, and especially Daniel Balouek-Thomert and Anthony Simonet who gave me generous support for and beyond my research. My work on earthquake early warning would not have been possible without Diego Melgar. I am grateful to Diego for his enthusiastic lessons about earthquake detection. It has been enthralling to work with Gabriel Antoniu, Alexandru Costan and Pedro Silva from the Inria KerData team on an exciting project that bridges our research fields. I would like to particularly thank Pedro for the captivating discussions about distributed computing and the opportunity to work on this fascinating project. I had the chance to work at Zhejiang University in China thanks to Tao Lin and his team, and among others Henrique Vinicius de Holanda, Guoqiang Ren, Jie Yang and Renhai Zhong, to whom I would like to express my gratitude for their warm welcome and the sharing of their exciting experiences. And, a big thank you to Issei Harada whom I had the privilege to work with; his commitment, proactivity and

enthusiasm helped us to move forward on the pattern-based approach.

My appreciation also goes to Véronique Bellon-Maurel, Frédérick Garcia, Carole Giansily, Gabrielle Lartia and Pierre Péré for the organization and animation of the different engaging events of the #DigitAg institute. I would also like to thank François Brun, Florent Duyme, Mohammed El Jabri, David Makowski, Aurore Philibert and Alexandre Termier for the nice and enriching moments shared during the teaching of our data science training program across France and the writing of the book associated with it.

It has been a pleasure to work every day in a convivial environment, I thank all the Inria LACODAM team for this, and especially my colleagues Colin Leverger, Raphaël Gauthier and Yichang Wang. And, everything would have been harder without the flawless administrative support from Nadia Derouault, Marie-Noëlle Georgeault, Marie Le Roïc, Laurence Thébault and Gaëlle Tworkowski. I am particularly grateful to Gaëlle for handling all my arrangements.

More than anyone else, I want to thank my family for their unfailing love and support. More than words could ever express, I am grateful for my amazing partner Mallory, whose constant and loving support is precious. Finally, a thousand thanks go to my sisters Chloé and Marion, and to my mother Laurence and my late father Patrick to whom I dedicate this thesis.

# Résumé en Français

Le déploiement massif de capteurs couplé à leur exploitation dans de multiples secteurs génère une masse considérable de données multivariées qui se sont révélées clés pour la recherche scientifique, les activités des entreprises et la définition de politiques publiques [Cussins Newman, 2019; Esteva et al., 2019; Ransbotham et al., 2019]. L'exploitation de ces données, de nature diverse et en grande quantité, est rendue possible grâce à l'intelligence artificielle. Prof. Wolfram Burgard propose une définition de l'intelligence artificielle : "systems that can interpret sensory data, create internal models, and then develop activities out of this, reason about this and create the next best action to take" [1]. En particulier, nous pouvons observer une tendance cette dernière décennie à utiliser l'apprentissage automatique pour automatiser la prise de décision. L'apprentissage automatique est un sous-domaine de l'intelligence artificielle qui se réfère aux algorithmes capables d'apprendre à partir des données [Goodfellow et al., 2016]. Comme établi dans [WIPO, 2019], l'apprentissage automatique est la technique la plus représentée dans les brevets (89% des brevets relatifs à l'intelligence artificielle) et est incluse dans plus d'un tiers de toutes les inventions identifiées (Mars 2018). Cependant, pour de nombreuses applications, l'adoption d'algorithmes d'apprentissage automatique ne peut se reposer uniquement sur la performance. Par exemple, le règlement général sur la protection des données de l'Union européenne, entré en application le 25 Mai 2018, introduit un droit à l'explication pour tous les individus afin qu'ils obtiennent des "meaningful explanations of the logic involved" lorsque la prise de décision automatisée a des "legal effects" sur les individus ou les affecte significativement [2].

Plus spécifiquement, les données multivariées qui intègrent une évolution temporelle, c'est-à-dire des séries temporelles, ont reçu une attention toute particulière ces dernières années, notamment à travers des applications critiques de monitoring (e.g. santé [Li et al., 2018a], mobilité [Jiang et al., 2019]). Une série temporelle est une séquence de valeurs réelles ordonnées en fonction du temps; et lorsque que plusieurs séries temporelles sont enregistrées simultanément par un jeu de capteurs, elles forment une série temporelle multivariée. Les séries temporelles multivariées associées avec les évènements correspondant

1. https://www.itu.int/en/ITU-T/AI/2018
2. https://ec.europa.eu/info/law/law-topic/data-protection_en

peuvent être exploitées pour des applications de classification en apprentissage automatique. L'objectif de la classification de séries temporelles multivariées est d'apprendre la relation entre une série temporelle multivariée et son annotation. Un exemple de série temporelle multivariée annotée est donné en Figure 2. Cette Figure représente la première série temporelle multivariée du jeu de données test de l'UEA Atrial Fibrilation [Bagnall et al., 2018] appartenant à la classe *Non-Terminating Atrial Fibrilation*. Cette série temporelle multivariée est composée de deux dimensions (deux signaux ECG) avec une longueur égale à 640 (une période de 5 secondes avec 128 échantillons par seconde). L'objectif d'un classifieur de séries temporelles multivariées est d'être capable de prédire que cette série appartient à la classe *Non-Terminating Atrial Fibrilation*.

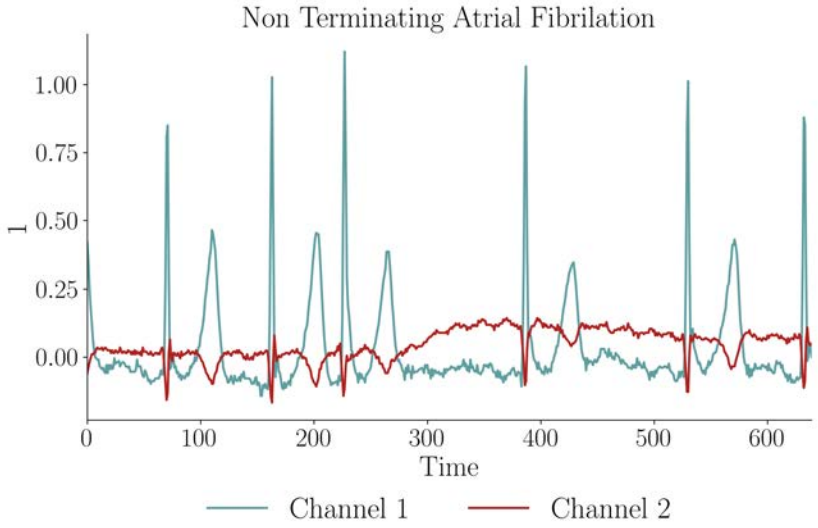



Figure 2 – La première série temporelle multivariée du jeu de données test de l'UEA Atrial Fibrilation. Elle appartient à la classe *Non-Terminating Atrial Fibrilation* et est composée de deux signaux ECG sur une période de 5 secondes (128 échantillons par seconde).

Cette thèse étudie la classification de séries temporelles multivariées. Les classifieurs de séries temporelles multivariées de l'état de l'art [Karim et al., 2019; Schäfer et al., 2017] sont des modèles difficiles à comprendre ("black-box" [Lipton, 2016]), qui se reposent sur des méthodes d'explicabilité applicables à n'importe quel modèle d'apprentissage automatique (post-hoc modèle-agnostique). L'axe de travail principal au sein des méthodes d'explicabilité post-hoc modèle-agnostique consiste à approximer la surface de décision d'un modèle en utilisant un modèle de remplacement explicable [Guidotti et al., 2019; Lakkaraju et al., 2017; Lundberg et al., 2017; Ribeiro et al., 2016, 2018]. Cependant, les explications du modèle de remplacement ne peuvent pas être parfaitement exactes au re-

gard du modèle original [Rudin, 2019], ce qui constitue un prérequis pour de nombreuses applications. Par conséquent, cette thèse a pour objectif d'améliorer la performance et l'explicabilité des méthodes d'apprentissage automatique de séries temporelles multivariées, et d'établir à partir des méthodes développées de nouvelles connaissances concernant deux applications réelles. Cette thèse analyse comment la performance et l'explicabilité peuvent être combinées ensemble, alors que celles-ci sont souvent opposées.

Deux applications réelles sont étudiées dans cette thèse. La première application étudiée concerne la détection et la caractérisation des séismes au moment où ceux-ci se déclarent, afin de pouvoir alerter avant que les secousses atteignent des zones sensibles et de pouvoir prendre des mesures de protection. Le problème réside dans le fait que les solutions existantes de détection avancée de séismes ne sont pas capables de détecter tout le spectre de séismes pouvant causer des dommages (séismes moyens et forts) [Allen et al., 2019]. La seconde application vise à améliorer la détection d'évènements déterminants pour la production de lait dans les exploitations laitières, ce qui est l'une des étapes les plus importantes pour atteindre nos objectifs de prodution alimentaire et environnementaux. Comme publié dans le rapport *Creating A Sustainable Food Future* [Searchinger et al., 2018], les ruminants (bovins, ovins, caprins) destinés à produire du lait et de la viande occupent les deux tiers des terres agricoles mondiales et sont responsables d'environ la moitié des émissions liées à la production agricole. La détection des évènements déterminants pour la production de lait (p. ex. estrus) est cruciale pour une utilisation optimale des ressources et représente une part importante du travail des éleveurs. Le problème réside dans le fait que les solutions existantes de détection, basées sur des données issues de capteurs abordables, font face à une adoption limitée [Steeneveld et al., 2015]. En effet, ces solutions ont des performances insuffisantes (fausses alertes, couverture incomplète). De plus, l'absence d'explications pour appuyer les alertes de détection génère de la confusion auprès des éleveurs et une défiance vis-à-vis de ces solutions.

# Contributions

## Une Grille d'Analyse Performance-Explicabilité

Tout d'abord, une nouvelle grille d'analyse pour évaluer et comparer les méthodes d'apprentissage automatique est proposée. Compte tenu des différentes catégories de classifieurs de la thèse et leurs formes d'explications respectives, une grille d'analyse a été

développée pour comparer et discuter les avantages/limites de ces approches d'apprentissage automatique au regard de leur performance et explicabilité. Cette grille d'analyse introduit un jeu de caractéristiques qui systématise l'évaluation des méthodes d'apprentissage automatique. Ces travaux ont été publiés au *IJCAI-PRICAI 2020 Workshop on Explainable Artificial Intelligence* [Fauvel et al., 2020d].

## Vers des explications exactes, informatives et accessibles pour des méthodes ensemblistes performantes

Puis, cette thèse présente trois nouvelles méthodes ensemblistes. Les méthodes ensemblistes sont les méthodes de référence de l'état de l'art concernant la classification de données multivariées (Random Forest [Breiman, 2001], Extreme Gradient Boosting [Chen et al., 2016]) et de séries temporelles univariées (HIVE-COTE [Lines et al., 2016]). Cependant, elles ne sont pas présentes dans les méthodes de l'état de l'art des classifieurs de séries temporelles multivariées. Les travaux de cette thèse montrent que les méthodes ensemblistes peuvent également être performantes sur les séries temporelles multivariées, tout en offrant une certaine explicabilité.

La première, DMSEEW, est une nouvelle méthode ensembliste, basée sur le stacking, pour la détection avancée de séismes et qui fournit des explications exactes au niveau local. DMSEEW améliore la détection des séismes pouvant causer des dommages. Tout particulièrement, DMSEEW détecte tous les séismes forts avec une précision de 100% sur un jeu de données réel, ce qui est primordial pour une détection avancée des séismes. Ces travaux ont été publiés à *AAAI 2020* [Fauvel et al., 2020a] et ont reçu le prix "AAAI 2020 Outstanding Paper Award" dans la catégorie "AI for Social Impact".

La seconde, LCE, est une nouvelle méthode ensembliste hybride pour la gestion des ressources dans les exploitations laitières. Elle se repose sur une méthode d'explicabilité applicable à n'importe quelle méthode d'apprentissage automatique (post-hoc modèle-agnostique), SHAP, qui offre des explications plus informatives et accessibles à une plus grande audience que celles de la première méthode. Ces travaux ont été publiés à *KDD 2019* [Fauvel et al., 2019b].

Cependant, cette amélioration des explications pour LCE s'effectue au détriment de l'exactitude, ce qui est un prérequis pour de nombreuses applications. Aussi, la troisième méthode, XEM, est une extension de la méthode ensembliste hybride qui intègre une explicabilité intrinsèque afin d'assurer l'exactitude des explications. XEM rivalise avec le

niveau d'information et l'accessibilité de la méthode post-hoc modèle-agnostique tout en maintenant la performance. XEM est plus performante que les méthodes de l'état de l'art en classification de séries temporelles multivariées sur les jeux de données publiques UEA. Ces travaux sont disponibles sur ArXiv [Fauvel et al., 2020b].

### Une approche basée sur des motifs fréquents avec une explicabilité intrinsèque exacte, informative et accessible

Ensuite, une nouvelle approche basée sur des motifs fréquents est présentée. Les motifs sont de petites conjonctions de symboles avec une sémantique prédéfinie. L'utilisation de ces motifs avec un classifieur facile à comprendre offre un fort potentiel d'explicabilité. Or, l'état de l'art des classifieurs de séries temporelles multivariées ne comprend pas de classifieur basé sur des motifs. Aussi, un nouveau classifieur se reposant sur des motifs fréquents (XPM) est proposé pour la gestion des ressources dans les exploitations laitières, un classifieur facile à comprendre avec des explications accessibles à une audience plus large que celles des méthodes ensemblistes.

### Un réseau de neurones à convolution combinant la performance avec une explicabilité post-hoc modèle-spécifique exacte et informative

Enfin, cette thèse présente un nouveau réseau de neurones à convolution. Suite au succès des méthodes d'apprentissage profond dans la reconnaissance d'images [Huang et al., 2017] et dans le traitement automatique du langage naturel [Devlin et al., 2019], celles-ci ont commencé à être adoptées pour l'analyse de séries temporelles. Le classifieur de séries temporelles multivariées de l'état de l'art obtenant la meilleure performance sur les jeux de données publiques de l'UEA est une méthode d'apprentissage profond (MLSTM-FCN [Karim et al., 2019]). Cependant, les predictions de MLSTM-FCN ne peuvent être appuyées par des explications parfaitement exactes car MLSTM-FCN se repose sur des méthodes post-hoc modèle-agnostique [Rudin, 2019], ce qui peut empêcher son utilisation sur de nombreuses applications. Ainsi, cette thèse présente un nouveau réseau de neurones à convolution, XCM, se révelant plus performant que XEM sur les jeux de données publiques UEA. De plus, l'architecture de XCM permet, grâce à l'utilisation de la méthode Grad-CAM (post-hoc modèle-spécifique), une identification exacte et pré-

cise des variables observées et des timestamps des données d'entrée importants pour les prédictions. XCM avec Grad-CAM présente de meilleures performances que la méthode ensembliste LCE avec SHAP sur l'application relative à la gestion des ressources dans les exploitations laitières, tout en améliorant l'explicabilité avec des explications exactes et plus informatives. En outre, XCM détecte environ 20% d'évènements clés en plus qu'une solution commerciale de référence sur un jeu de données réel en exploitation laitière, tout en préservant la même précision. Ces travaux sont disponibles sur ArXiv [Fauvel et al., 2020c].

# Table of Contents


**1 Introduction** **19**

1.1 Motivation . . . 19

1.2 Overview of the Contributions . . . 21

1.2.1 A Performance-Explainability Analytical Framework . . . 21

1.2.2 Towards Faithful, Informative and Human-Friendly Explainability in Performant Ensemble Methods . . . 22

1.2.3 A Pattern-Based Approach with Human-Friendly Explainability by Design . . . 22

1.2.4 A Convolutional Neural Network Combining Performance with Faithful and Informative Post-Hoc Model-Specific Explainability . . . 23

**I State-of-the-Art** **24**

**2 Methods** **25**

2.1 Classification . . . 25

2.1.1 Ensemble Methods . . . 26

2.2 Multivariate Time Series Classification . . . 28

2.2.1 Similarity-Based Classifiers . . . 29

2.2.2 Feature-Based Classifiers . . . 30

2.2.3 Deep Learning Classifiers . . . 35

2.3 Explainability . . . 37

2.3.1 Explainability by Design . . . 37

2.3.2 Post-Hoc Model-Specific Explainability . . . 39

2.3.3 Post-Hoc Model-Agnostic Explainability . . . 40

**3 Applications** **42**

3.1 Dairy Resource Monitoring . . . 42

3.1.1 Background . . . 42

3.1.2 Machine Learning Solutions . . . 44

3.2 Earthquake Early Warning . . . 45
3.2.1 Background . . . 45
3.2.2 Machine Learning Solutions . . . 47

**II Performance-Explainability Analytical Framework 48**

**4 Analytical Framework 49**
4.1 Introduction . . . 49
4.2 Framework . . . 50
4.3 Example . . . 58
4.4 In The Next Parts... . . . 60

**III Towards Faithful, Informative and Human-Friendly Explainability in Performant Ensemble Methods 61**

**5 A Distributed Multivariate Time Series Ensemble Method to Earthquake Early Warning 63**
5.1 Introduction . . . 64
5.1.1 Contributions . . . 65
5.2 DMSEEW . . . 66
5.3 Evaluation . . . 67
5.3.1 Real-World Dataset . . . 68
5.3.2 Experimental Setting . . . 70
5.4 Results . . . 71
5.4.1 Sensor-Level Predictions . . . 72
5.4.2 Combined Predictions at Central Level . . . 72
5.5 Performance-Explainability Analysis . . . 75

**6 A Hybrid Ensemble Method for Dairy Resource Monitoring 77**
6.1 Introduction . . . 78
6.1.1 Contributions . . . 79
6.2 Local Cascade Ensemble . . . 80
6.2.1 Local Cascade - LC . . . 80
6.2.2 Local Cascade Ensemble - LCE . . . 81

6.3 Evaluation . . . 83
6.3.1 Datasets . . . 83
6.3.2 Experimental Setting . . . 86
6.4 Results . . . 88
6.4.1 Public Datasets . . . 88
6.4.2 Real-World Application . . . 90
6.5 Peformance-Explainability Analysis . . . 96

**7 An Explainable by Design Ensemble Method for Multivariate Time Series Classification 100**

7.1 Introduction . . . 101
7.1.1 Contributions . . . 102
7.2 XEM . . . 103
7.2.1 Dataset Transformation . . . 103
7.2.2 Classification . . . 104
7.2.3 Explainability by Design . . . 105
7.2.4 Properties . . . 106
7.2.5 Time Complexity . . . 107
7.2.6 Implementation . . . 108
7.3 Evaluation . . . 108
7.3.1 Public Datasets . . . 108
7.3.2 Algorithms . . . 111
7.3.3 Hyperparameters . . . 112
7.3.4 Metrics . . . 112
7.4 Results . . . 112
7.4.1 Performance . . . 112
7.4.2 Explainability . . . 118
7.4.3 Discussion . . . 123
7.5 Performance-Explainability Analysis . . . 124

**IV Towards Individual Methods Combining Performance with Faithful, Informative and Human-Friendly Explainability 127**

**8 A Pattern-Based Approach with Human-Friendly Explainability by Design 128**

8.1 Introduction . . . 129
8.1.1 Contributions . . . 130
8.2 XPM . . . 131
8.2.1 Detection Algorithm . . . 131
8.2.2 Explainability by Design . . . 133
8.3 Evaluation . . . 134
8.3.1 Real-World Dataset . . . 134
8.3.2 Algorithms . . . 136
8.3.3 Hyperparameters . . . 137
8.3.4 Metrics . . . 138
8.4 Results . . . 138
8.4.1 Performance . . . 138
8.4.2 Explainability . . . 143
8.5 Performance-Explainability Analysis . . . 146

**9 A Convolutional Neural Network Combining Performance with Faithful and Informative Post-Hoc Model-Specific Explainability 150**

9.1 Introduction . . . 151
9.1.1 Contributions . . . 153
9.2 XCM . . . 153
9.2.1 Architecture . . . 153
9.2.2 Explainability . . . 157
9.3 Evaluation . . . 158
9.3.1 Datasets . . . 158
9.3.2 Algorithms . . . 159
9.3.3 Hyperparameters . . . 160
9.3.4 Metrics . . . 160
9.4 Results . . . 161
9.4.1 Performance . . . 161
9.4.2 Explainability . . . 164

9.4.3 Real-World Application . . . . . 169
9.5 Performance-Explainability Analysis . . . . . 173

**10 Conclusion** **177**
10.1 Summary of the Contributions . . . . . 177
10.2 Perspectives . . . . . 178
10.2.1 A Consensus About a Performance-Explainability Framework . . . . . 178
10.2.2 Pattern-Based Post-Hoc Model-Specific Explainability Methods for Black-Box Machine Learning Models . . . . . 180

**Bibliography** **182**

**Appendix** **196**

**A Publications** **197**

Chapter 1

# Introduction

## 1.1 Motivation

The prevalent deployment and usage of sensors in a wide range of sectors generate an abundance of multivariate data which has proven to be instrumental for researches, businesses and policies [Cussins Newman, 2019; Esteva et al., 2019; Ransbotham et al., 2019]. The exploitation of the diverse and large amount of data is rendered possible through Artificial Intelligence (AI). Prof. Wolfram Burgard offered a definition of AI as "systems that can interpret sensory data, create internal models, and then develop activities out of this, reason about this and create the next best action to take" [1]. In particular, there has been a trend over the past decade to leverage machine learning to automate decision-making processes. Machine learning is a subfield of AI which refers to algorithms that are able to learn from data through experience [Goodfellow et al., 2016]. As stated in [WIPO, 2019], machine learning is the dominant AI technique disclosed in patents (89% of patent families related to an AI technique) and is included in more than one-third of all identified inventions (as of March 2018). However, for many applications, the adoption of machine learning algorithms cannot rely solely on their prediction performance. For example, the European Union's General Data Protection Regulation, which became enforceable on 25 May 2018, introduces a right to explanation for all individuals so that they can obtain "meaningful explanations of the logic involved" when automated decision-making has "legal effects" on individuals or similarly "significantly affecting" them [2].

More specifically, multivariate data which integrates temporal evolution, i.e. time series, has received significant interests in recent years, driven by automatic and high resolution monitoring applications (e.g. healthcare [Li et al., 2018a], mobility [Jiang et al., 2019]). A time series is a sequence of real values ordered according to time; and when a set of co-evolving time series are recorded simultaneously by a set of sensors, it is called a Multivariate Time Series (MTS). MTS associated with their corresponding events can

1. https://www.itu.int/en/ITU-T/AI/2018
2. https://ec.europa.eu/info/law/law-topic/data-protection_en

be leveraged in machine learning classification applications. The objective of MTS classification is to learn the relationship between a MTS sample and its label. An example of a labeled MTS is given in Figure 1.1. It is the first MTS of the UEA Atrial Fibrilation [Bagnall et al., 2018] test set that belongs to the class *Non-Terminating Atrial Fibrilation*. This MTS is composed of two dimensions (two channels ECG) with a length of 640 (5 second period with 128 samples per second). The objective of a MTS classifier is to be able to predict that this MTS belongs to the class *Non-Terminating Atrial Fibrilation*.

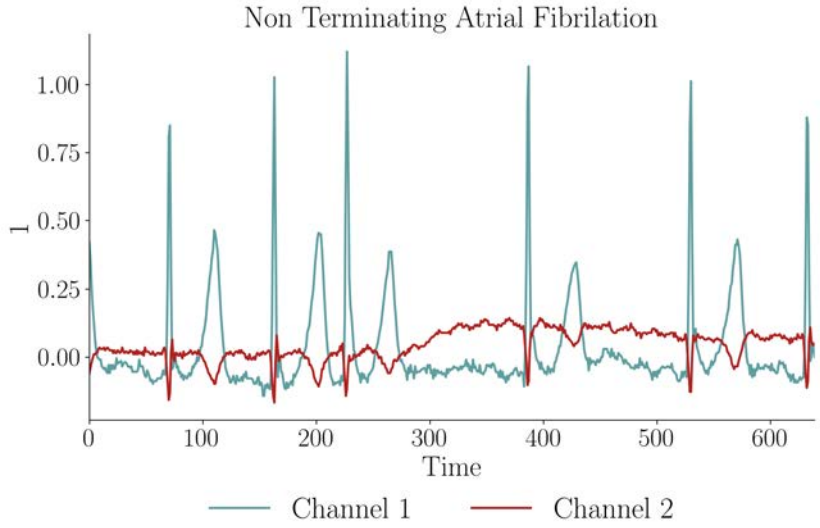



Figure 1.1 – The first MTS sample of the UEA Atrial Fibrilation test set. It belongs to the class *Non-Terminating Atrial Fibrilation* and is composed of two channels ECG on a 5 second period (128 samples per second).

This thesis addresses the issue of MTS classification. The best performing state-of-the-art MTS classifiers are "black-box" classifiers [Karim et al., 2019; Schäfer et al., 2017], i.e. complicated-to-understand models [Lipton, 2016], which rely on explainability methods providing explanations from any machine learning model to support their predictions (post-hoc model-agnostic). The main line of work in post-hoc model-agnostic explainability methods approximates the decision surface of a model using an explainable surrogate model [Guidotti et al., 2019; Lakkaraju et al., 2017; Lundberg et al., 2017; Ribeiro et al., 2016, 2018]. However, the explanations from the surrogate models cannot be perfectly faithful with respect to the original model [Rudin, 2019], which is a prerequisite for numerous applications. Therefore, this thesis aims to enhance the performance and explainability of MTS classifiers, and derive new insights from the new methods developed about two real-world applications. Performance and explainability are often opposed

whereas this thesis analyzes how they can be best combined together along their different aspects.

Two real-world applications will be studied in this thesis. The first application studied concerns the automatic detection and characterization of earthquakes as they happen, in order to deliver alerts before the ground motion actually reaches sensitive areas so that protective measures could be taken. The challenges lies in the fact that current earthquake early warning solutions are unable to cover the whole spectrum of potentially damaging earthquakes (medium and large) [Allen et al., 2019]. The second application aims to improve the detection of determining events for milk production in dairy farms, which is one of the most important steps toward meeting both food production and environmental goals. As underlined in the report *Creating A Sustainable Food Future* [Searchinger et al., 2018], ruminant livestock (cattle, sheep, and goats), used for dairy and meat production, occupy two-thirds of global agricultural land and contribute roughly half of agriculture's production-related emissions. The detection of determining events for milk production (e.g. estrus) in dairy farms is crucial for an optimal resource use and represents an important part of dairy farmers' work. The challenge lies in the fact that current detection solutions based on affordable sensor data face a moderate adoption rate [Steeneveld et al., 2015]. First, these solutions lack of performance (false alerts, incomplete coverage). Second, the absence of explanations behind detection alerts generate confusion and mistrust from farmers.

## 1.2 Overview of the Contributions

### 1.2.1 A Performance-Explainability Analytical Framework

Firstly, this thesis introduces a new performance-explainability framework in chapter 4 to assess and compare the proposed machine learning methods. The different classifier categories of this thesis along with their own explainability approach suggested the need for an analytical framework to compare and discuss the strengths/limitations of these machine learning approaches with regard to their performance and explainability. The framework details a set of characteristics that systematize the performance-explainability assessment of the machine learning methods. It has been published in *IJCAI-PRICAI 2020 Workshop on Explainable Artificial Intelligence* [Fauvel et al., 2020d].

### 1.2.2 Towards Faithful, Informative and Human-Friendly Explainability in Performant Ensemble Methods

Then, three new ensemble methods are presented in part III. Ensemble methods are the current state-of-the-art classifiers for traditional multivariate data classification (Random Forest [Breiman, 2001], Extreme Gradient Boosting [Chen et al., 2016]) as well as for univariate time series classification (HIVE-COTE [Lines et al., 2016]). However, there is no ensemble method among the state-of-the-art MTS classifiers. Hence, this part shows that ensemble methods can also be performant in the MTS classification setting, while providing some explainability.

This thesis presents:

- DMSEEW in chapter 5, a new stacking ensemble method for earthquake early warning combining performance with faithful and local explainability by design. It has been published in *AAAI 2020* [Fauvel et al., 2020a] and received the "AAAI 2020 Outstanding Paper Award" in the category "AI for Social Impact";
- LCE in chapter 6, a new hybrid ensemble method for dairy resource monitoring combining performance with informative post-hoc model-agnostic explainability. It has been published in *KDD 2019* [Fauvel et al., 2019b];
- XEM in chapter 7, a new ensemble method for MTS classification combining performance with faithful, informative and local explainability by design. It is available on ArXiv [Fauvel et al., 2020b].

### 1.2.3 A Pattern-Based Approach with Human-Friendly Explainability by Design

Next, a new pattern-based approach is presented in chapter 8. Patterns are small conjunctions of symbols with a predefined semantic. The use of patterns with an easy-to-understand classifier has great potential for explainability. Nonetheless, there is no pattern-based classifier among the state-of-the-art MTS classifiers. Therefore, this thesis presents XPM, a new pattern-based approach for dairy resource monitoring providing faithful, informative and human-friendly explainability.

### 1.2.4 A Convolutional Neural Network Combining Performance with Faithful and Informative Post-Hoc Model-Specific Explainability

Finally, this thesis presents a new deep learning approach in chapter 9. Following the good performance of deep learning architectures in image recognition [Huang et al., 2017] and natural language processing [Devlin et al., 2019], they have started to be adopted for time series analysis. The current best performing state-of-the-art MTS classifiers on the public UEA datasets [Bagnall et al., 2018] is a deep learning approach (MLSTM-FCN [Karim et al., 2019]). However, MLSTM-FCN cannot provide perfectly faithful explanations as it can only rely on post-hoc model-agnostic explainability methods [Rudin, 2019], which could prevent its use on numerous applications. Hence, this thesis presents XCM, a new convolutional neural network for MTS classification combining performance with faithful and informative post-hoc model-specific explainability. It is available on ArXiv [Fauvel et al., 2020c].

Before detailing the different contributions, I present in the next part the state-of-the-art for the methods and applications of this thesis.

Part I

# State-of-the-Art

Chapter 2

# Methods

## Contents


**2.1 Classification . . . . . 25**
2.1.1 Ensemble Methods . . . . . 26
**2.2 Multivariate Time Series Classification . . . . . 28**
2.2.1 Similarity-Based Classifiers . . . . . 29
2.2.2 Feature-Based Classifiers . . . . . 30
2.2.3 Deep Learning Classifiers . . . . . 35
**2.3 Explainability . . . . . 37**
2.3.1 Explainability by Design . . . . . 37
2.3.2 Post-Hoc Model-Specific Explainability . . . . . 39
2.3.3 Post-Hoc Model-Agnostic Explainability . . . . . 40


In this chapter, I first define the learning task of this thesis, i.e. supervised learning for classification, and introduce the current state-of-the-art classifiers on which the new hybrid ensemble method LCE is positioned. Then, I present the state-of-the-art MTS classifiers; the machine learning algorithms designed to handle the data type encountered in the applications of this thesis. Finally, I present the categories of explainability methods supporting machine learning models predictions.

## 2.1 Classification

This thesis addresses the issue of supervised learning for classification. Classification consists in learning a function that maps an input data to its label.

**Definition 1** (**Classification**). *Given an input space $X$, an output space $Y$, an unknown distribution $P$ over $X \times Y$, a training set sampled from $P$, the mathematical expectation $\mathbb{E}$, compute a function $h^*$ such as*

$$h^* = \arg\min_h \quad \mathbb{E}_{(x,y)\sim P}\left[h(x) \neq y\right]$$

In the traditional multivariate data setting, there is no explicit relationship among samples or variables and every sample has the same set of variables (also called attributes or dimensions). The most popular (and often best performing) machine learning classifiers on the traditional multivariate data belong to the following classes: k-nearest neighbors, regularized logistic regressions, support vector machines, neural networks and ensemble methods. Nevertheless, to undertake the task of the general multivariate classification, no single classifier can claim to be superior to any of the others [Wolpert, 1996] (known as the "No Free Lunch theorem"). Thus, the combination of different classifiers - an ensemble method - is often considered a good method to obtain a better generalizing classifier. There are three main reasons that justify the use of ensembles over single classifiers [Dietterich, 2000]: statistical (reduce the risk of choosing the wrong classifier by averaging when the amount of training data available is too small compared to the size of the hypothesis space), computational (local search from many different starting points may provide a better approximation to the true unknown function than any of the individual classifier), and representational (expansion of the space of representable functions).

Based on these reasons, I have worked on developing a new ensemble method, called Local Cascade Ensemble (LCE), which will be presented in the chapter 6. Thus, I detail in the next section the related work on ensemble methods on which I position the new ensemble method LCE.

### 2.1.1 Ensemble Methods

Ensemble methods are structured around two approaches (explicit, implicit) which have their own strengths and limitations. Therefore a hybrid ensemble method is encouraged [Masoudnia et al., 2014]. The *implicit approach* involves creating diverse classifiers on the original training data, whereas the *explicit approach* emphasizes classifiers diversity through the creation of different training sets by probabilistically changing the distribution of the original training data.

There are two methods adopting an implicit approach: Mixture of Experts (ME) [Jacobs et al., 1991] and Negative Correlation Learning (NCL) [Liu et al., 1999].
*ME* uses a divide-and-conquer algorithm to split the problem space, and each individual classifier learns a part of the training data. The advantage of this method is that each individual classifier is concerned with its own individual error. However, individual classifiers are trained independently so there is no control over the bias-variance trade-off. The bias-variance trade-off defines the capacity of the learning algorithm to generalize

beyond the training set. The *bias* is the component of the classification error that results from systematic errors of the learning algorithm. A high bias means that the learning algorithm is not able to capture the underlying structure of the training set (underfitting). The *variance* measures the sensitivity of the learning algorithm to changes in the training set. A high variance means that the algorithm is learning too closely the training set (overfitting). The objective is to minimize both the bias and variance.
Next, *NCL* is an ensemble method which is trained on the entire training data simultaneously and interactively to adjust the bias-variance trade-off. Individual classifiers interact through the correlation penalty terms of their error functions. The correlation penalty term is a regularization term that is integrated into the error function of each individual classifier. This term quantifies the amount of error correlation and is minimized during the training, which leads to negatively correlated individual classifiers and balances the bias-variance trade-off. The disadvantage of this method is that each classifier is concerned with the whole ensemble error due to the training of each classifier on the same data. Some studies combine NCL and ME features to address their limitations (Local Cascade [Gama et al., 2000], Mixture of Negatively Correlated Experts - MNCE [Masoudnia et al., 2012]). For instance, Masoudnia et al. [2012] proposed MNCE, an augmented version of ME by integrating a regularization term in the error function (NCL) to better balance the bias-variance trade-off.

However, a combination of implicit approaches does not benefit from the diversification of generating classifiers by perturbing the distribution of the original training data (explicit approach). There are two methods adopting an explicit approach with complementary effects on the bias-variance trade-off (bagging [Breiman, 1996] - variance reduction, boosting [Schapire, 1990] - bias reduction). *Bagging* is a method for generating multiple versions of a predictor (bootstrap replicates) and using these to get an aggregated predictor. *Boosting* is a method for iteratively learning weak classifiers and adding them to create a final strong classifier. After a weak learner is added, the data weights are readjusted, allowing future weak learners to focus more on the examples that previous weak learners misclassified. Bagging and boosting methods have been combined [Kotsiantis et al., 2005] but without integrating the diversification benefit of an implicit approach.

There is a study which combines the explicit boosting method with the implicit ME divide-and-conquer principle [Ebrahimpour et al., 2012]. Nonetheless, the only bias reduction distribution change of boosting does not ensure a bias-variance trade-off. Therefore, a hybrid ensemble method called Local Cascade Ensemble (LCE) is proposed in this thesis.

LCE combines an explicit boosting-bagging approach to handle the bias-variance trade-off and an implicit divide-and-conquer approach (decision tree) to learn different parts of the training data. LCE algorithm is detailed in the chapter 6 and how it is leveraged for building a MTS classifier (XEM) in the chapter 7. In the next section, I present the state-of-the-art MTS classifiers, i.e. the machine learning algorithms designed to handle the data type and the learning task encountered in the applications of this thesis.

## 2.2 Multivariate Time Series Classification

The state-of-the-art MTS classifiers can be categorized into three families: similarity-based, feature-based and deep learning methods.

Before presenting in the following sections the MTS classifiers of each category, I define the MTS data type which is used in the different applications of this thesis.

**Definition 2** (**Multivariate Time Series**). *A multivariate time series (MTS)* $M = \{x_1, ..., x_d\} \in \mathcal{R}^{d*l}$ *is an ordered sequence of* $d \in \mathcal{N}$ *streams with* $x_i = (x_{i,1}, ..., x_{i,l})$*, where* $l$ *is the length of the time series and* $d$ *is the number of multivariate dimensions.*

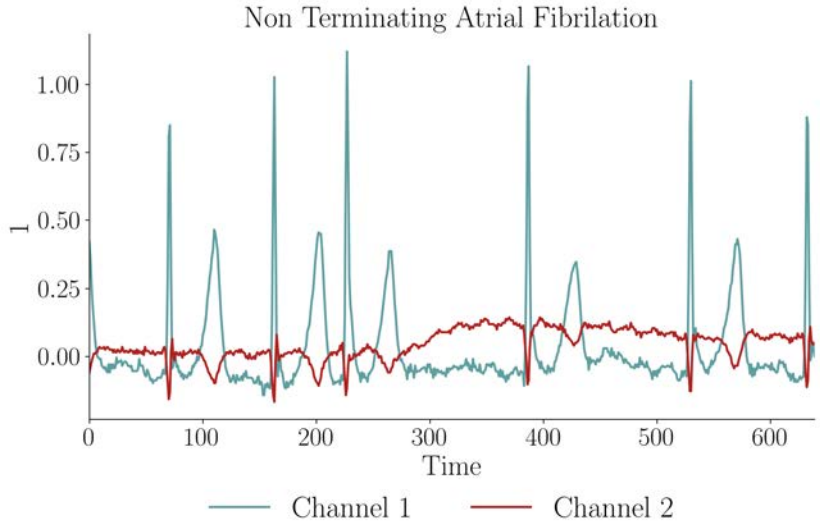



Figure 2.1 – The first MTS sample of the UEA Atrial Fibrilation test set. It belongs to the class *Non-Terminating Atrial Fibrilation* and is composed of two channels ECG on a 5 second period (128 samples per second).

In the different datasets of this thesis, MTS are generated from automatic sensors with

a fixed and synchronized sampling along all dimensions. An example of a MTS is given in Figure 2.1, it is the same example as in the Introduction of this thesis.

### 2.2.1 Similarity-Based Classifiers

Similarity-based methods make use of similarity measures (e.g., Euclidean distance) to compare two MTS. Dynamic Time Warping (DTW) has been shown to be the best similarity measure to use along the k-Nearest Neighbors (k-NN) [Seto et al., 2015], this approach is called kNN-DTW. DTW is not a distance metric as it does not fully satisfy the required properties (the triangle inequality in particular), but its use as similarity measure along with the NN-rule is valid [Vidal et al., 1985]. Unlike the Euclidean distance's strict one-to-one alignment (see alignment in the top-left corner image of Figure 2.2), DTW allows a one-to-many alignment (see alignment in the bottom-left corner image of Figure 2.2). To align $Q$ and $C$ sequences using DTW, an n-by-n matrix is constructed with the $(i, j)$ element being the squared Euclidean distance $(q_i, c_j)$ between the points $q_i$ and $c_j$. A warping path $P$ is a contiguous set of matrix elements defining a mapping between $Q$ and $C$. The path minimizing the warping cost is selected. Figure 2.2, sourced in [Seto et al., 2015], illustrates a DTW alignment between two time series.

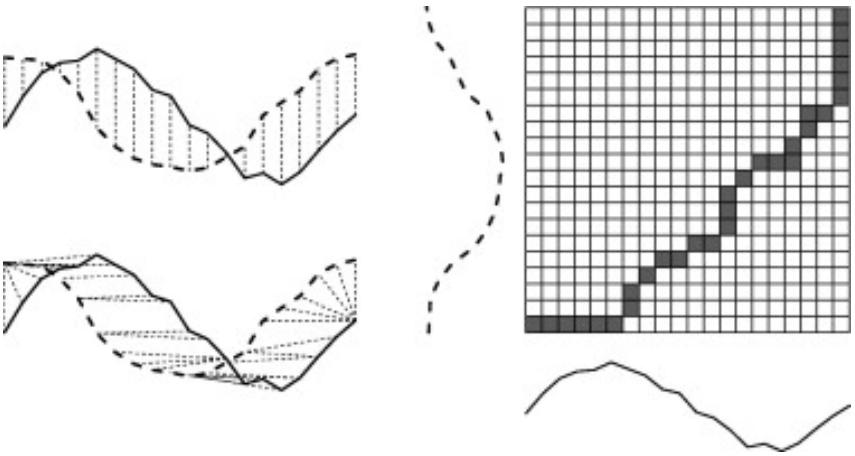


Figure 2.2 – DTW image showing the alignment procedure. The two original time series shown on the left (dotted) and bottom (solid) of the image on the right are shown aligned in the bottom-left image according to the optimal path shown as the dark black line on the right. Top-left image shows Euclidean distance's strict one-to-one alignment.

There are two versions of kNN-DTW for MTS: dependent ($\text{DTW}_D$) and independent ($\text{DTW}_I$). Neither dominates over the other [Shokoohi-Yekta et al., 2017]. $\text{DTW}_I$ measures

the cumulative distances of all dimensions independently measured under DTW. $\text{DTW}_D$ uses a similar calculation with single-dimensional time series; it considers the squared Euclidean cumulated distance over the multiple dimensions.

### 2.2.2 Feature-Based Classifiers

The feature-based MTS classifiers can be categorized into two families: shapelet/pattern-based and bag-of-words classifiers. The first family uses subseries (shapelets) or small conjunctions of symbols with a predefined semantic (patterns) to transform the original time series into a lower-dimensional space that is easier to classify. The second family breaks up MTS into windows, converts the windows into a bag of discrete words, and builds a histogram of feature counts as basis for classification. The state-of-the-art classifiers belonging to these two families are presented in the next two sections.

#### Shapelet-Based and Pattern-Based Classifiers

Both shapelet-based and pattern-based classifiers transform the original time series into a lower-dimensional space that is easier to classify. Nevertheless, *shapelet-based* classifiers use subseries to perform classification whereas *pattern-based* classifiers can exploit different types of patterns (e.g. itemsets, sequences, chronicles) [Han et al., 2011] and select the patterns that best describe the phenomenon studied.

**Shapelet-Based Classifiers** A shapelet is defined as follows [Ye et al., 2009]:

**Definition 3** (**Shapelet**). *Given a time series $T$ of length $m$, a shapelet $S$ of $T$ is a sampling of length $l \leq m$ of contiguous positions from $T$, that is, $S = t_p, ..., t_p + l - 1$, for $1 \leq p \leq m - l + 1$.*

An illustration of shapelet is given in Figure 2.3 on the same MTS example as the one used in section 2.2.

According to the results published, Ultra-Fast Shapelets UFS [Wistuba et al., 2015] and Generalized Random Shapelet Forest (gRSF) [Karlsson et al., 2016] are the current state-of-the-art shapelet models in MTS classification. They relax the major limiting factor of the time to find discriminative subsequences in multiple dimensions (shapelet discovery) by randomly selecting shapelets.

First, UFS models the input data as a set of distances from shapelets. UFS considers shapelets globally from a restricted pool of randomly selected shapelets per dimension.

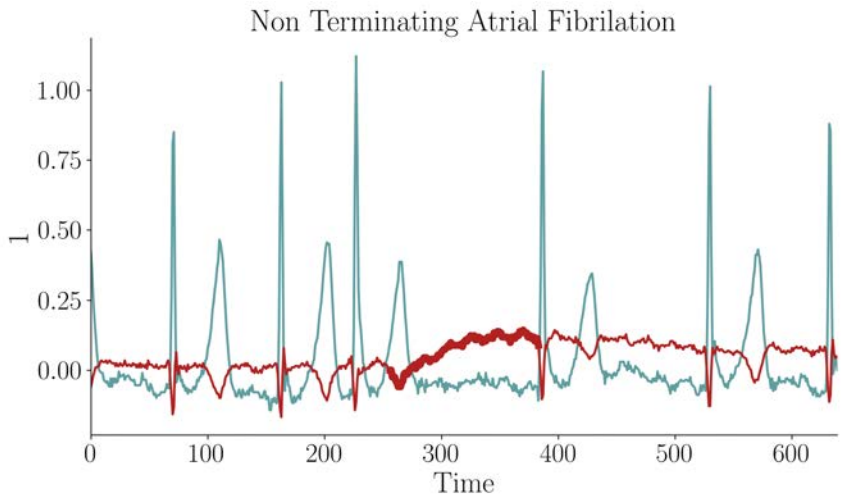



Figure 2.3 – The first MTS sample of the UEA Atrial Fibrilation test set with a shapelet highlighted on the red channel (timestamps 256 to 383).

Finally, the classification of the input data with the shapelets as features can be performed by any classifier.

Then, gRSF creates decision trees over randomly selected MTS instances and randomly selected shapelets. The decision trees, at each node, locally consider a predefined number of shapelets randomly selected along the dimensions.

gRSF shows better accuracy results than UFS and kNN-DTW on average (14 public MTS datasets) [Karlsson et al., 2016]. The accuracy results also exhibits that there is a statistically significant difference only between gRSF and kNN-DTW.

**Pattern-Based Classifiers** Since different types of patterns can be informative, multiple studies have proposed different classification methods based on pattern features, including itemset-based approaches [Cheng et al., 2007, 2008], sequence-based classification [Buza et al., 2010; Fradkin, 2014] and chronicle-based approaches [Dauxais et al., 2019]. An *itemset* can be defined as a group of symbols. Then, *sequences* are ordered group of symbols and *chronicles* are patterns with more diverse temporal relations among the symbols. As far as I have seen, most of the approaches mine frequent patterns. A pattern is *frequent* if it occurs in at least a predefined percentage (support) of the time series. In addition, the patterns can be mined in closed-form in order to limit their number. A *closed pattern* is a maximal set of patterns common to a set of objects.

The different patterns are defined as follows [Cheng et al., 2008; Dauxais et al., 2019;

Fradkin, 2014]:

**Definition 4** (**Itemset**). *Let $D$ be a training database, $I = \{i_1, i_2, ..., i_m\}$ be the set of distinct items, and $C = \{c_1, c_2, ..., c_k\}$ be the set of class labels. Assume $D$ contains a set of $n$ training instances $D = \{x_i, y_i\}_{i=1}^{n}$, where $x_i \subseteq I$ is a set of items and $y_i \in C$ is a class label. An itemset $\alpha = \{\alpha_1, \alpha_2, ..., \alpha_l\}$ is a subset of $I$.*

**Definition 5** (**Sequence**). *A sequence $S = \{s_i\}$, $i = \{1, ..., k\}$ matches a sequential pattern $P = \{p_j\}$, $j = \{1, ..., m\}$ iff $\exists$ $\{i_1, ..., i_m\}$ with $p_j \subseteq s_{i_j}$ for $j = \{i, ..., m\}$, such that $\forall$ $1 \leq j$, $k \leq m : p_j \prec p_k$ implies $i_j < i_k$.*

**Definition 6** (**Chronicle**). *Let $\mathbb{E}$ be a set of event types and $\mathbb{T}$ be a temporal domain where $\mathbb{T} \in \overline{\mathbb{R}}$. A chronicle is a couple $(\mathcal{E}, \mathcal{T})$, where $\mathcal{E} = \{e_1, ..., e_n\}$, $e_i \in \mathbb{E}$ and $\forall (i, j)$, $1 \leq i < j \leq n$, $e_i \leq_{\mathbb{E}} e_j$, $\mathcal{T}$ is a temporal constraint set: $\mathcal{T} = \{e[a, b]e' \mid (e, e') \in \mathcal{E}, e \leq_{\mathbb{E}} e'\}$.*

Therefore, according to these definitions and compared to the shapelets introduced in the previous section, patterns allows the extraction of ordered/unordered group of symbols with potential gaps and temporal constraints among these symbols, whereas shapelets are entire subseries.

There are two types of approaches used for pattern-based classification [Fradkin, 2014]: indirect and direct. The *indirect approach* is composed of three stages. First, the candidate patterns are mined in an unsupervised manner. Then, feature selection is performed to reduce the number of patterns and finally, classification is undertaken based on the remaining features. The *direct approach* leverages the class label information in the pattern mining stage, therefore suppressing the feature selection stage by generating fewer patterns. I present in the following paragraphs the state-of-the-art classifiers on each type of approach and pattern.

According to the results published, the current state-of-the-art approaches for itemset-based classifiers are a classifier issued in [Cheng et al., 2007] (indirect) and DDPMine [Cheng et al., 2008] (direct). Firstly, the indirect approach [Cheng et al., 2007] uses FP-Close [Grahne et al., 2003] to generate closed itemsets [Pasquier et al., 1999] and develops a new feature selection method (MMRFS). MMRFS selects a feature if it is relevant to the class label and if it contains very low redundancy to the features already selected. Finally, it performs classification with a decision tree on the remaining features. Then, the direct approach DDPMine [Cheng et al., 2008] imposes a branch-and-bound search on the FP-growth [Han et al., 2000] mining process, which prunes the search space significantly.

It works in an iterative fashion and reduces the problem size incrementally by eliminating the instances which are covered by the selected features. Finally, the classification is performed with a support vector machine.

Concerning sequence-based classifiers, a classifier issued in [Buza et al., 2010] (indirect) and BIDE - Discriminative [Fradkin, 2014] (direct) are the current state-of-the-art methods according to the results published. In [Buza et al., 2010], the algorithm converts time series into symbolic sequences using Symbolic Aggregate approXimation [Lin et al., 2003]. Sequential patterns are then mined using an extended version of Apriori [Agrawal et al., 1994] with taxonomy. These patterns are used as features for construction of support vector machines and Bayesian networks. Then, BIDE-Discriminative modifies BIDE closed sequential pattern mining algorithm [Wang et al., 2004] to integrate class information for direct mining of sequences. It relies on pruning statistical measures of feature predictiveness. Finally, the classification is performed with a support vector machine.

Next, according to the results published, the current state-of-the-art chronicle-based classifier is DCM [Dauxais et al., 2019] (Indirect). As far as I have seen, there is no direct approach for chronicle-based classification. DCM extracts frequent multisets, which are chronicles without temporal constraints, and then it mines discriminant temporal constraints from these multisets. Finally, the classification is performed by applying an order on the extracted set.

Some experiments have compared DCM to BIDE-Discriminative [Dauxais et al., 2019]. The results show that discriminant chronicles as rules can produce similar and sometimes better accuracy than sequential patterns as features (3 public datasets). However, pattern-based classifiers are usually not considered into the benchmark of the state-of-the-art MTS classifiers so there is no other basis of comparison available. Indeed, in the case of time series data, the use of pattern mining algorithms requires a discretization of the dataset which is application-specific. In addition, pattern-based classifiers are often seen as less accurate than the state-the-art MTS methods. Nevertheless, I include the pattern-based classifiers in the MTS classification state-of-the-art as they can propose other characteristics (e.g. model comprehensibility, accessibility of the patterns) which could justify their adoption on certain applications. The chapter 8 presents a new pattern-based classifier suited for the dairy resource monitoring application.

### Bag-of-Words Classifiers

Bag-of-words models (LPS [Baydogan et al., 2016], mv-ARF [Tuncel et al., 2018], SMTS [Baydogan et al., 2014] and WEASEL+MUSE [Schäfer et al., 2017]) break up MTS into windows, convert the windows into a bag of discrete words, and build a histogram of feature counts as basis for classification. Therefore, compared to shapelet-based and pattern-based models, bag-of-words models extract information on the whole time series with words on each window.

Firstly, Symbolic representation for Multivariate Time Series (SMTS) trains a random forest [Breiman, 2001] on an elementary representation of the MTS (time index, values and derivatives) to partition the MTS into leaf nodes. Each leaf node is then labeled by a word and a bag-of-words representation is generated from the terminal nodes of the trees. Each symbol is considered to be a word and the relative frequency vector of the symbols from each tree are concatenated to form the bag-of-words representation. This representation is processed with a second random forest for classification.

Learned Pattern Similarity (LPS) extracts all possible windows with a predefined length from a MTS and trains regression trees to identify structural dependencies between the time series observations. The regression trees trained in this manner represent non-linear autoregressive models. Similar to SMTS, LPS next builds a bag-of-words representation based on the labels of the leaf nodes. Finally, a similarity measure is defined on the bag-of-words representation of the MTS for classification.

Next, multivariate AutoRegressive Forest (mv-ARF) trains, for each class, a forest of regression trees on lagged observations. Then, a bag-of-words representation is generated based on the concatenation of the different mean squared errors of the models for each MTS. Finally, any classifier can be applied to this representation.

Lastly, WEASEL+MUSE generates a bag-of-words representation by applying various sliding windows with different sizes on each discretized dimension (Symbolic Fourier Approximation) to capture features (unigrams, bigrams, dimension identification). Each feature is considered to be a word and the word counts are concatenated to form the bag-of-words representation. Following a feature selection with chi-square test, it classifies the MTS based on a logistic regression classifier.

WEASEL+MUSE shows better accuracy results compared to LPS, mv-ARF, SMTS and gRSF on average (20 public MTS datasets) [Schäfer et al., 2017]. None of these MTS classifiers exhibits a statistically significant accuracy difference.

### 2.2.3 Deep Learning Classifiers

The state-of-the-art deep learning MTS classifiers (FCN [Wang et al., 2017], ResNet [He et al., 2016], MLSTM-FCN [Karim et al., 2019]) use Convolutional Neural Networks (CNN) and/or Long-Short Term Memory (LSTM) Neural Networks.

Before presenting the state-of-the-art deep learning MTS classifiers, I introduce some notions about neural networks as background. A *neural network* is a composition of $L$ parametric functions referred to as layers, where each layer is considered a representation of the input domain [Goodfellow et al., 2016]. One layer $l_i$, such as $i \in \{1, ..., L\}$, contains neurons, which are small units that compute one element of the layer's output. The layer $l_i$ takes as input the output of its previous layer $l_{i-1}$ and applies a transformation to compute its own output. The behavior of these transformations is controlled by a set of parameters $\theta_i$ for each layer and an activation sublayer to shape the non-linearity of the network. These parameters are called weights and link the input of the previous layer to the output of the current layer based on matrix multiplication. This process is also referred to as feedforward propagation in the deep learning literature and is the constituent of multilayer perceptrons (MLPs). A neural network is usually called "deep" when it contains more than one layer between its input and output layer.

Following the good performance of CNN architectures in image recognition [Huang et al., 2017] and natural language processing [Devlin et al., 2019; Sutskever et al., 2014], CNNs have started to be adopted for time series analysis [Cristian Borges Gamboa, 2017]. *CNNs* are neural networks that use convolution in place of general matrix multiplication in at least one of their layers [Goodfellow et al., 2016]. A convolution can be seen as applying and sliding a filter over the time series. The use of different types, numbers and sequences of filters allow the learning of multiple discriminative features (feature maps) useful for the classification task. Thus, a CNN for classification is composed of at least one convolutional layer to extract features and a classifier (e.g. fully connected layers - MLP, global average pooling with a softmax layer). Generally, a convolutional block in the MTS classification setting is made up of three cascading layers: a convolutional layer to extract features based on a predefined number of convolution filters, a batch normalization layer [Ioffe et al., 2015] to enable faster convergence and better generalization of the network [Bjorck et al., 2018], and an activation layer to shape the non-linearity (e.g. ReLU [Nair et al., 2010]). The use of global average pooling instead of fully connected layers for classification improves the generalization ability of the network. Taking the average of each feature map is more robust to spatial translations of the input [Lin et al., 2014]. An illustration of a

CNN is shown in Figure 2.4.

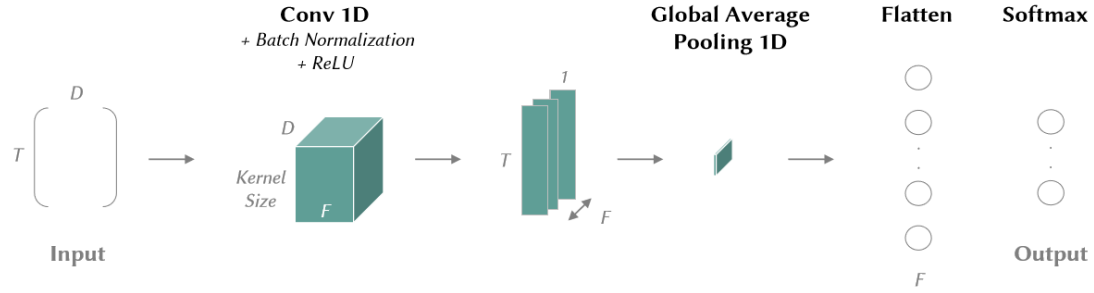



Figure 2.4 – Example of a convolutional neural network with one padded convolutional block followed by a global average pooling with a softmax layer for classification. Abbreviations: $D$ - number of observed variables, $F$ - number of filters, $T$ - time series length.

LSTM networks are a variant of recurrent neural networks (RNNs), which are designed for sequential data. When feedforward neural networks/MLPs are extended to include feedback connections, they are called RNNs [Goodfellow et al., 2016]. LSTM networks have been developed to address the vanishing/exploding gradient problem of RNNs [Bengio et al., 1994; Hochreiter, 1991; Pascanu et al., 2013], which are unable to learn long-term dependencies. RNNs use sequential processing over time and is trained through backpropagation. So, if the gradient vanishes, it means that the earlier states have no real effect on the later states as the weights have stopped to be updated. *LSTM networks* are able to learn temporal dependencies over arbitrary intervals [Graves, 2012] by keeping the recursive gradient close to one and perform classification (e.g. softmax layer) on the output from the LSTM layer. An LSTM has the ability to add, modify or remove information as it flows through the different steps. Each step is composed of three major gate mechanisms (input gate, forget gate and output gate).

The following paragraphs present the state-of-the-art deep learning MTS classifiers (according to the results published): FCN [Wang et al., 2017], ResNet [He et al., 2016] and MLSTM-FCN [Karim et al., 2019].

Firstly, Fully Convolutional Network (FCN) is composed of three convolutional blocks, each containing a convolutional layer followed by a batch normalization layer and a ReLU activation layer. After the convolution blocks, the features generated are fed into a global average pooling layer and the classification is performed with a softmax layer.

Then, ResNets extend the neural networks to very deep structures which aim to learn more complex features based on hierarchical feature learning. It introduces a new neural layer called the residual block. Compared to a convolutional block (such as in FCNs),

the difference is that a linear shortcut is added to link the output of a residual block to its input thus enabling the flow of the gradient directly through this connection, which renders the training of a deep neural network much easier by reducing the vanishing gradient effect. Wang et al. [2017] propose a ResNet for MTS classification reusing the structure of the FCN, it is composed of three residual blocks followed by a global average pooling and a softmax layer.

Finally, MLSTM-FCN consists of the concatenation of a CNN block with a LSTM block. The CNN block is used as feature extractor and is composed of 3 convolutional sub-blocks. As FCN, each block contains a convolutional layer followed by a batch normalization layer and a ReLU activation layer. In addition, the first two convolutional blocks end with a Squeeze-and-Excitation block [Hu et al., 2017]. The Squeeze-and-Excitation block adaptively recalibrates the input feature maps. After the convolution blocks, the features generated are fed into a global average pooling layer and following the concatenation of the CNN and LSTM blocks, the classification is performed with a softmax layer.

According to the results published in [Karim et al., 2019] and my experiments, MLSTM-FCN shows better accuracy results than FCN and ResNet on average on the UEA datasets [Bagnall et al., 2018]. In addition, MLSTM-FCN exhibits better accuracy results than WEASEL +MUSE on large datasets (relative to the one tested) on average (20 public MTS datasets) [Schäfer et al., 2017]. None of these MTS classifiers (FCN, ResNet, MLSTM-FCN and WEASEL+MUSE) exhibits a statistically significant accuracy difference.

## 2.3 Explainability

The issue of explainability in machine learning has received considerable attention in recent years. The number of works published concerning methods to support machine learning predictions is extensive and it is not the objective of this thesis to give an exhaustive list. Instead, I introduce the three commonly recognized categories (explainability by design, post-hoc model-specific explainability and post-hoc model-agnostic explainability) [Du et al., 2020] to which all of the explainability methods are belonging to and give some examples of methods for each category.

### 2.3.1 Explainability by Design

First, some machine learning models provide explainability by design. These self-explanatory models incorporate explainability directly to their structures. This category

includes, for example, decision trees, rule-based models and linear models. The decision nodes of a decision tree and the coefficients of a linear model convey the explanations to support their predictions. For instance, Lakkaraju et al. [2016] present an interpretable decision sets approach. Decision sets are sets of short, independent and non-overlapping rules that cover the whole feature space and classes. The rules are extracted based on a two-step approach where frequent itemsets are mined and then a set of rules is selected following the maximization of a joint objective that scores decision sets based on both how interpretable and accurate they are.

Moreover, certain neural networks can provide explainability by design. For example, the attention mechanism [Bahdanau et al., 2015] can give users the possibility to interpret which parts of the input are attended by the model through visualizing the attention weight matrix for individual predictions. In particular, the attention mechanism is a core component of the Transformer [Vaswani et al., 2017], a state-of-the-art network in neural machine translation. The Transformer has been introduced to replace the RNN-based sequence-to-sequence model in machine translation, model which is used to convert sequences of a certain type to sequences of another type. A RNN-based sequence-to-sequence model is generally composed of two RNNs (an encoder and a decoder) with an attention mechanism which passes the weighted sum of the hidden states of the encoder as context vector to the decoder. As presented in previous section, RNNs have problems dealing with long-term dependencies and the sequential nature of the model prevents its parallelization. The Transformer is a neural network also composed of a two-part architecture (encoder and decoder) but which learns dependencies based on multi-head attention mechanisms and feedforward neural networks. The multi-head attention mechanism consists in computing the attention mechanism multiple times, in parallel and independently, and the outputs are concatenated and linearly transformed. The attention mechanism used in the Transformer is a self-attention mechanism, sometimes called intra-attention, which relates different positions of a sequence in order to compute a representation of the sequence. Thus, Song et al. [2018] present a multivariate time series classifier based on a multi-head attention mechanism which incorporates temporal order into the sequence representation using both positional encoding and dense interpolation embedding techniques. It has been shown in [Voita et al., 2019] that only a small subset of the attention heads of the Transformer appears to be important. However, it remains unclear what relationship exists between attention weights and model predictions [Jain et al., 2019]. Following this study, Wiegreffe et al. [2019] have provided a suite of experiments that could be used in

order to make informed decisions about the quality of the models' attention mechanisms when used as explanation for model predictions.

### 2.3.2 Post-Hoc Model-Specific Explainability

Post-hoc model-specific explainability methods are specifically designed to extract explanations for a particular model. These methods applied on the trained model (post-hoc) usually derive explanations by examining internal model structures and parameters.

For example, a method has been designed to measure the contribution of each feature in random forest models [Palczewska et al., 2013]. It extends the computation of feature contribution [Kuz'min et al., 2011] to random forest models for classification and proposes three techniques (the analysis of median feature contributions, of clusters and of log-likelihoods) for discovering class-specific feature contribution "patterns" in the decision-making process of random forest models.

Some post-hoc model-specific explainability methods also exist for neural networks. First, perturbation-based methods compute the contribution of a feature by removing, masking or altering them, and measuring the difference with the original output following a forward pass on the new input. For instance, Zeiler et al. [2014] analyze the probability of the correct class given by a CNN as a function of the position of a grey patch occluding part of the input. However, perturbation-based methods tend to be very slow as the number of features grows [Zintgraf et al., 2017]. Then, the approaches based on back-propagation are seen as the state-of-the-art explainability methods for deep learning models [Ancona et al., 2018]. Methods based on back-propagation [Bach et al., 2015; Erhan et al., 2009; Selvaraju et al., 2019; Shrikumar et al., 2017, 2016; Springenberg et al., 2015; Sundararajan et al., 2017] calculate the gradient, or its variants, of a particular output with respect to the input using back-propagation to derive the contribution of features. The Gradient Explanation [Erhan et al., 2009] quantifies how much a change in each input feature in a small neighborhood around the input would change the prediction of the model. Guided Backpropagation [Springenberg et al., 2015] builds on the "DeConvNet" explanation method [Zeiler et al., 2014] and corresponds to the Gradient Explanation where negative gradient entries are set to zero while back-propagating through a ReLU unit. The Gradient $\odot$ Input [Shrikumar et al., 2016] method proposes to address "gradient saturation" by computing the signed partial derivatives of the output with respect to the input and multiplying them with the input itself, while the Integrated Gradients [Sundararajan et al., 2017] sums over scaled versions of the input. It is has been shown in [Ancona

et al., 2018] that for ReLU networks some other existing explanation methods ($\varepsilon$-Layer-wise Relevance Propagation [Bach et al., 2015] and DeepLift (Rescale) [Shrikumar et al., 2017]) are equivalent to the Gradient $\odot$ Input. Finally, Gradient-weighted Class Activation Mapping (Grad-CAM) [Selvaraju et al., 2019] has proven to be an adequate method to support convolutional neural networks predictions. Grad-CAM identifies the regions of the input data that are important for predictions in convolutional neural networks using the class-specific gradient information (attribution map). It generates the attribution maps based on the weighted combination of the feature map activations from the layer of interest, where the weights are the global average pool of the gradients over each feature map. The method has been shown to provide faithful explanations with regard to the model [Adebayo et al., 2018], it passes both the model parameter and data randomization tests.

### 2.3.3 Post-Hoc Model-Agnostic Explainability

Finally, post-hoc model-agnostic explainability methods provide explanations from any machine learning model. These methods treat the model as a black-box and do not inspect internal model parameters.

For example, the permutation feature importance method [Altmann et al., 2010] provides the importance of a specific feature to the overall performance of a model. It determines the importance of a feature by calculating how the model prediction accuracy deviates after permuting the values of that feature.

Some methods [Dhurandhar et al., 2018; Kim et al., 2016] generate contrastive explanations, i.e. positive and negative explanations that support the model's predictions, using some representations of the input data. In [Kim et al., 2016], MMD-critic complements prototypes explanations (representative samples) with critics (samples that do not quite fit the model). MMD-critic uses the maximum mean discrepancy (MMD) statistic as a measure of similarity between points and potential prototypes, and selects prototypes that maximize the statistic. In addition to prototypes, it selects criticism samples using a regularized witness function score. Then, CEM (Contrastive Explanations Method) [Dhurandhar et al., 2018] identifies what should be minimally and sufficiently present to justify the model's predictions and analogously what should be minimally and necessarily absent. CEM finds a minimal amount of features in the input that are sufficient in themselves to yield the same prediction, and a minimal amount of features that should be absent in the input to prevent the prediction from changing, based on close perturbations of the data

manifold leveraging a convolutional autoencoder.

Some other methods use an explainable surrogate model [Guidotti et al., 2019; Lakkaraju et al., 2017; Lundberg et al., 2017; Ribeiro et al., 2016, 2018] to support the model predictions. A *surrogate model* is a model that aims to mimic the predictions of the original model. LIME (Local Interpretable Model-agnostic Explanations) [Ribeiro et al., 2016], Anchors [Ribeiro et al., 2018] and LORE (LOcal Rule-based Explanations) [Guidotti et al., 2019] are local-only approaches. They focus on the behavior of the model in the neighborhood of a specific instance, without providing a single description of the logic of the model for all possible instances. These approaches assume that the decision boundary for the model can be arbitrarily complex over the whole data space, but that in the neighborhood of a data point the decision boundary can be captured by an explainable model. LIME describes the local behavior of the model using a linearly weighted combination of the input features, learned on perturbations of an instance. Then, Anchors, as a variant of LIME, uses a set of association rules instead of a linear model as explanation method. LORE learns a local decision tree classifier, generated by a genetic algorithm, on a synthetic neighborhood of an instance. It derives from the decision tree an explanation consisting of a decision rule, explaining the factual reasons of the decision and a set of counterfactuals. Next, BETA (Black Box Explanations through Transparent Approximations) [Lakkaraju et al., 2017] constructs global-only explanations of a model predictions. It constructs a small number of compact decision sets that aim to mimic the model in terms of assigning class labels to instances. Each of these decision sets captures the behavior of the model in non-overlapping parts of the feature space. Finally, SHAP (SHapley Additive exPlanations) [Lundberg et al., 2017] provides explanations at both global and local level. The surrogate explainable model is a linear model: an additive feature attribution method that uses simplified inputs (conditional expectations) assuming feature independence. The SHAP values can be averaged per class to obtain the average impact of each variable on model predictions at global level.

The multiple explainability methods presented in this section reflect the diversity of explanations generated to support model predictions, therefore the need for a framework in order to benchmark the machine learning methods explainability. A new framework to benchmark the performance-explainability of the machine learning methods of this thesis is presented in chapter 4.

Chapter 3

# Applications

## Contents


**3.1 Dairy Resource Monitoring . . . . . 42**
3.1.1 Background . . . . . 42
3.1.2 Machine Learning Solutions . . . . . 44
**3.2 Earthquake Early Warning . . . . . 45**
3.2.1 Background . . . . . 45
3.2.2 Machine Learning Solutions . . . . . 47


This chapter presents the background and the existing machine learning solutions concerning the two applications of this thesis: dairy resource monitoring and earthquake early warning.

## 3.1 Dairy Resource Monitoring

### 3.1.1 Background

Nowadays, data (e.g. temperature, activity, body weight, milk production) is collected in dairy farms through different types of sensors to support farmers' decision making in various aspects of management (e.g. reproduction, diseases, feeding, environment). Machine learning methods can help to exploit the value of this ever-growing volume of data.

Reproduction is a key factor for dairy farm performance as it directly impacts milk production. As shown in Figure 3.1, the standard scheme for a cow in dairy farm is one calf a year (365 days) with 305 days of milk production and 60 days of dry period. Gestation period is 9 months with an ovarian cycle of more or less 21 days. Following this scheme, insemination of cows should ideally occur 90 days after the previous calving to allow the next calving in 9 months. However, in practice, insemination can occur only during estrus or right after. *Estrus* is the period preceding the ovulation period and is the only period

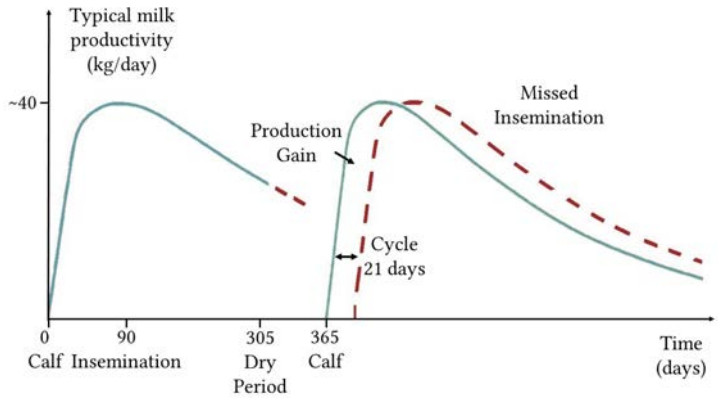



Figure 3.1 – Typical milk productivity evolution during one lactation for a dairy cow.

when the cow is susceptible to pregnancy. For each missed estrus identification, another 21 days is needed for the next attempt. Half of the lactation production is produced in the first 4 months with a declining productivity after the first 3 months. So, in case of missed insemination, a sensitive impact on milk production occurs. When compared with an average calving interval of 362 days, [Inchaisri et al., 2010] estimated that a longer average calving interval of 407 or 507 days caused an average milk production loss of, respectively, 4% or 13% per cow per year. In addition, as shown in [Bascom et al., 1998], the most prevalent reason for cow culling is reproduction issue (e.g. long intervals between 2 calves). Therefore, estrus detection for cow insemination is crucial in dairy farms for an optimal resource use.

Traditionally, estrus detection relies on visual observation of animal behaviors as activity usually increases markedly in cows during estrus [Gaillard et al., 2016]. However, less than 50% of estrus are detected visually [Peralta et al., 2005]. This low detection rate can be explained by five main reasons. First, primary sign of estrus (mounting) has decreased from 80% to 50% over the past 30 to 50 years [Dobson et al., 2008]; second, 35% of estrus are not associated with obvious behavioral signs - a phenomena defined as silent estrus [Palmer et al., 2010]; third, only 65% of cows have a normal ovarian cyclicity - postpartum anovulation shorter than 50 days and regular ovarian cycle of 20-25 days [Disenhaus et al., 2009]; fourth, duration of estrus has declined from 18 hours to 8 hours over the last 50 years [Reames et al., 2011]; and fifth, sexual behaviors mostly expressed from 1am to 7am [Kerbrat et al., 2004].

Different methods have been developed to aid visual detection. Hormonal induction aims to induce synchronous ovulation and thus allows fixed-time insemination without the need for estrus detection. This practice is widespread in the United States [Car-

aviello et al., 2006] but not in Europe due to public reluctance to accept animal products treated with hormones [Chastant-Maillard, 2006]. Therefore, alternative methods have been adopted. The gold standard is estrus estimation using automated progesterone analysis in milk. We observe a consensus around the use of progesterone profiles to identify estruses [Cutullic et al., 2011; Gilmore et al., 2011; Tenghe et al., 2015]. This option is now available for the farms but the cost of this solution limits its extensive implementation. As a result, activity and body temperature are considered having potential for automatic estrus detection [Fricke et al., 2017; Saint-Dizier et al., 2012; Senger, 1994]. A study shows that automatic activity measurement (with activity meter or accelerometer) is suitable for estrus detection and is likely to be gainful for most dairy farms [Rutten et al., 2014]. Some commercial detection solutions based on affordable activity sensors exist. However, their adoption rate remains moderate [Steeneveld et al., 2015]. First, these solutions lack of performance (false alerts, incomplete estrus coverage - behavioral estrus only). Second, the absence of key information behind detection alerts generate confusion and mistrust from farmers. Therefore, aside from an enhanced performance, key justifications for estrus alerts are also considered essential to the farmers. In order to train a model that covers both behavioral and silent estrus, the dataset of this thesis is labeled with progesterone dosage in milk. The next section presents the related work of automatic estrus detection solutions based on machine learning.

### 3.1.2 Machine Learning Solutions

There are multiple studies about the application of machine learning methods on estrus detection [Dolecheck et al., 2015; Krieter, 2005; Minegishi et al., 2019; Mitchell et al., 1996]. None of them uses the currently recognized method for behavioral and silent estrus identification as labels (progesterone profiles), so their estrus labeling methods are not exhaustive. Moreover, two studies use different variables (milk volume, milking order, days since last estrus) rather than the affordable activity or temperature measurements. Finally, none of them gives insights on algorithm predictions based on its explainability.

Mitchell et al. [1996] base the study on time series data of milk volume and milking order, using visual detection as the ground truth. Two learning schemes were tested - FOIL and C4.5. Algorithms detected 69% of estruses identified by visual method and a large number of false positives occurred (74%).

Krieter [2005] learns a multilayer perceptron on time series data of activity and the number of days since last estrus, using successful insemination as the ground truth. The

model showed a sensitivity, a specificity and an error rate of 77.5, 99.6 and 9.1% on 373 estrus. The *sensitivity*, also called *recall*, measures the proportion of actual positives that are correctly identified as such, and the *specificity* measures the proportion of actual negatives that are correctly identified as such.

Next, Dolecheck et al. [2015] base the study on time series data of activity, using visual detection as the ground truth (65.6% of all estruses). Three machine learning techniques were tested - random forest, linear discriminant and MLP. Algorithms showed 91%-100% accuracy on a limited dataset of 18 cows.

And lastly, Minegishi et al. [2019] learn a logistic regression on activity variables (7 days moving average/standard deviation, and the daily absolute maximum of 6-hour-window cumulative change of the data), using the combination of an automatic estrus detection solution (collar-mounted activity meter) and visual detection as the ground truth. Two herds have been studied (low-input conventional and organic) with seasonal breeding. The model shows 51-91% sensitivity and 92-99% specificity (threshold: 0.7) according to the herd and the season (total dataset: 1,462 estrus, 300 cows).

Therefore, there is a need to enhance the performance of estrus detection solutions based on affordable sensor data and to support the alerts with explanations. I propose three new machine learning solutions for estrus detection which are presented and discussed in the chapters 6, 8 and 9. The next section introduces the second application of this thesis: earthquake early warning.

## 3.2 Earthquake Early Warning

### 3.2.1 Background

Earthquakes are accompanied by a suite of associated hazards. In particular, strong shaking and large tsunamis can affect regions hundreds of kilometers long causing substantial loss of life and damage to the built environment. These large ground motions also trigger secondary hazards such as fires and landslides. In an effort to protect against these hazards, a number of Earthquake Early Warning (EEW) systems have been built around the world [Allen et al., 2019]. These critical systems, operating 24/7, are expected to automatically detect and characterize earthquakes as they happen, and to deliver alerts before the ground motion actually reaches sensitive areas so that protective measures could be taken.

An EEW system needs to be able to detect both medium ($5 \leq$ magnitude $< 6$, Richter

scale) and large earthquakes (6 $\leq$ magnitude, Richter scale). Depending on the distance from the origin of the earthquake, both of these can cause serious damages. Seismometers, which have long been the bulwark of seismology to detect earthquakes, have a difficulty to detect and characterize large earthquakes [Melgar et al., 2013] due to a saturation issue caused by their sensitivity to ground motion velocity. As a result, earthquakes over magnitude 7.5 tend to be underestimated. A promising solution to this issue [Melgar et al., 2015] emerged with novel high-precision Global Positioning System (GPS) sensors, with their millimeter to centimeter accuracy when measuring high ground motion velocity. However, GPS are unable to characterize medium earthquakes, as they are prone to containing significant signals from a variety of noise sources, mostly of atmospheric origin. Consequently, multi-sensor solutions (leveraging both GPS and seismometers) appear as a promising approach. As EEW can be assimilated as a classification problem, where the input is sensor data and the output is a class (normal activity/medium earthquake/large earthquake), recent machine learning approaches designed to combine large volumes of data from multiple data sources can be applied.

The following paragraphs define the key concepts of earthquake early warning as background to the presentation in the next section of the existing machine learning solutions.

An *earthquake* is the shaking of the surface of the Earth caused by seismic waves. Among these seismic waves, two types stand out: Primary waves (P-waves) and Secondary waves (S-waves). Both waves have the same origin - most commonly an abrupt movement of tectonic plates. However, P-waves travel through Earth's crust around 1.7 times faster than S-waves which propagate through Earth's interior. In addition, only S-waves are responsible for the severe damages. P-waves cause soft shaking due to their longitudinal shape (they move sideways), whereas S-waves are transverse waves (they move up and down). Therefore, an Earthquake Early Warning (EEW) system, which aims to provide an alert before the damaging effects reach sensitive areas, relies on the detection of the P-wave before the S-wave arrives. This gives communities, organizations and governments a time window of seconds to minutes to take protective actions.

Traditionally, inertial seismometers are used to detect primary waves. The inertial mass is designed to remain stationary following sudden movements while the frame and drum move with the ground to record waves. However, during large earthquakes, ground motion velocity causes the inertial mass to be displaced above the allowed span. This effect is called saturation. As a result, earthquakes over magnitude 7.5 (Richter scale) tend to be underestimated. On the other hand, GPS satellites are not affected by earthquakes,

so a GPS receiver station on Earth can be used to assess strong ground motion based on the station displacement. However, GPS is sensitive to a variety of noise sources, mostly of atmospheric origin, and is unable to characterize moderate earthquakes. Both sensors produce data in the form of 3D time series indicating the direction of a ground motion (east-west, north-south and up-down) at a frequency of around 20Hz.

P-waves follow a propagation model (IASP91 [Kennet, 1991]) which is used for labeling the time series (sequences of measurements) corresponding to an earthquake. Based on the distance below Earth's surface where each earthquake happened, the P-wave arrival time on each sensor (seismometers and GPS stations) is estimated according to its distance to the epicenter with the propagation model.

The next section presents the existing machine learning solutions to earthquake early warning which detect P-waves.

### 3.2.2 Machine Learning Solutions

Machine learning in seismology is still a developing field. There are a couple of studies [Li et al., 2018b; Perol et al., 2018; Yoon et al., 2015] using machine learning methods for earthquake characterization based on P-wave detection (EEW). However, none of them uses a combination of GPS and seismometers data so the whole spectrum of earthquakes with damaging potential is not appropriately covered. The detection is only based on seismometers data, so the saturation issue on large earthquakes is present. Moreover, the three studies adopt a binary classification approach (earthquakes vs. noise) with no distinction between medium and large earthquakes. Finally, two of these studies [Perol et al., 2018; Yoon et al., 2015] limit their scope to medium earthquakes.

Yoon et al. [2015] propose a waveform similarity-based method optimized by locality-sensitive hashing search using seismometers data from California. It presents a precision of 88.1% and a recall of 87.5%. Next, Li et al. [2018] have developed a generative adversarial network with a random forest on seismometers data from Southern California and Japan. It obtains an accuracy of 99.2%. And, Perol et al. [2018] train a convolutional neural network using seismometers data from Oklahoma and show a precision of 94.8% and a recall of 100%.

Consequently, an EEW machine learning-based solution that can be generalized to the whole spectrum of earthquakes with damaging potential is necessary and is presented in chapter 5. The next chapter introduces a performance-explainability framework to assess and benchmark the machine learning methods of this thesis.

Part II

# Performance-Explainability Analytical Framework

Chapter 4

# Analytical Framework

## Contents


**4.1 Introduction . . . . . 49**
**4.2 Framework . . . . . 50**
**4.3 Example . . . . . 58**
**4.4 In The Next Parts... . . . . 60**


## 4.1 Introduction

This part introduces the performance-explainability analytical framework that will be used in the following parts to benchmark the different machine learning methods. The next three parts present different classifiers (convolutional neural network classifier, ensemble methods, pattern-based classifier) with their own explainability approach (by design, post-hoc model-specific, post-hoc model-agnostic) on two applications (dairy resource monitoring and earthquake early warning). Therefore, an analytical framework is needed to compare and discuss the strengths/limitations of these machine learning approaches with regard to their performance and explainability.

The performance of a machine learning method can be assessed by the extent to which it correctly predicts unseen instances. A metric like the accuracy score commonly measures the performance of a classification model. However, there is no standard approach to assess explainability. First, there is no mathematical definition of explainability. A definition proposed by [Miller, 2019] states that the higher the explainability of a machine learning algorithm, the easier it is for someone to comprehend why certain decisions or predictions have been made. Second, as presented in the state-of-the-art section 2.3, there are several methods belonging to different categories (explainability by design, post-hoc model-specific explainability and post-hoc model-agnostic explainability) which provide their own form of explanations to support their respective predictions.

The requirements for explainable machine learning methods are dependent upon the

application and to whom the explanations are intended for [Bohlender et al., 2019; Tomsett et al., 2018]. In order to match these requirements and conduct experiments to validate the usefulness of the explanations by the end-users, there is a need to have a comprehensive assessment of the explainability of the existing methods. Doshi-Velez et al. [2017] claim that creating a shared language is essential for evaluation and comparison of machine learning methods, which is currently challenging without a set of explanation characteristics. As far as I have seen, there is no existing framework which defines a set of explanation characteristics that systematize the assessment of the explainability of existing machine learning methods.

Hence, in the next section, I propose a new framework to assess and benchmark the performance-explainability characteristics of the different machine learning methods of the thesis. The framework hypothesizes a set of explanation characteristics and, as emphasized in [Wolf, 2019], focuses on what people might need to understand about machine learning methods in order to act in concert with the model outputs. The framework does not claim to be exhaustive and excludes application-specific implementation constraints like time, memory usage and privacy. It could be a basis for the development of a comprehensive assessment of the machine learning methods with regard to their performance and explainability and for the design of new machine learning methods.

## 4.2 Framework

The framework aims to respond to the different questions an end-user may ask to take an informed decision based on the predictions made by a machine learning model: *What is the level of performance of the model? Is the model comprehensible? Is it possible to get an explanation for a particular instance? Which kind of information does the explanation provide? Can we trust the explanations? What is the target user category of the explanations?* The performance-explainability analytical framework that I propose is composed of the following components:

**Performance** - *What is the level of performance of the model?*

The first component of the framework characterizes the performance of a machine learning model. Different methods (e.g. holdout, k-fold cross-validation) and metrics (e.g. accuracy, F-measure, Area Under the ROC Curve) exist to evaluate the performance of a machine learning model [Witten et al., 2016]. However, there is no consensus on an evaluation procedure to assess the performance of a machine learn-

ing model. Recent work suggests that the definition of such an evaluation procedure necessitates the development of a measurement theory for machine learning [Flach, 2019]. Many of the problems stem from a limited appreciation of the importance of the *scale* on which the evaluation measures are expressed.

Then, in current practices, the choice of a metric to evaluate the performance of a machine learning model depends on the application. According to the application, a metric aligned with the goal of the experiments is selected, which prevents the performance comparison of machine learning models across applications.

Therefore, in the framework of this thesis, the performance component is defined as a first step towards a standard procedure to assess the performance of machine learning models. It corresponds to the relative performance of a model on a particular application. More specifically, it indicates the relative performance of the models according to the state-of-the-art model on a particular application and an evaluation setting. This definition allows the categorization of the models' performance on an application and an evaluation setting. In the case of different applications with a similar machine learning task, the performance component can give the list of models which outperformed current state-of-the-art models on their respective application. Thus, it points to certain models that could be interesting to evaluate on a new application, without providing guarantee that these models would perform the same on this new application. I propose an assessment of the performance in three categories:

- *Best*: best performance. It corresponds to the performance of the first ranked model on the application following an evaluation setting (datasets, evaluation method, models);
- *Similar*: performance similar to that of the state-of-the-art models. Based on the same evaluation setting, it corresponds to all the models which do not show a statistically significant performance difference with the second ranked model. For example, the statistical comparison of multiple classifiers on multiple datasets is usually presented on a critical difference diagram [Demšar, 2006];
- *Below*: performance below that of the state-of-the-art models. It corresponds to the performance of the remaining models with the same evaluation setting.

**Model Comprehensibility** - *Is the model comprehensible?*

The model comprehensibility corresponds to the ability for the user to understand how the model works and produces certain predictions. Comprehensibility is tightly linked to the model complexity; yet, there is no consensus on model complexity assessment [Guidotti et al., 2018]. Currently, two categories of models are commonly recognized: "white-box" models, i.e. easy-to-understand models, and "black-box" models, i.e. complicated-to-understand models [Lipton, 2016]. For example, many rule-based models and decision trees are regarded as "white-box" models while ensemble methods and deep learning models are "black-box" models. Not all rule-based models or decision trees are "white-box" models. Cognitive limitations of humans place restrictions on the complexity of the approximations that are understandable to humans. For example, a decision tree with a hundred levels cannot be considered as an easy-to-understand model [Lakkaraju et al., 2017]. However, the distinction between "white-box" models and "black-box" models is clear among the machine learning models of this thesis. The models are all "black-box" (ensemble methods, convolutional neural network) except one which is an easy-to-understand decision tree with a depth of 4. Therefore, I propose an assessment of the comprehensibility in two categories:

- *Black-Box*: "black-box" model;
- *White-Box*: "white-box" model.

**Granularity of the Explanations** - *Is it possible to get an explanation for a particular instance?*

The granularity indicates the level of possible explanations. Two levels are generally distinguished: global and local [Du et al., 2020]. Global explainability means that explanations concern the overall behavior of the model across the full dataset, while local explainability informs the user about a particular prediction. Some methods can provide either global or local-only explainability while other methods can provide both (e.g. decision trees). Therefore, I propose an assessment of the granularity in three categories:

- *Global*: global explainability;
- *Local*: local explainability;
- *Global & Local*: both global and local explainability.

**Information Type** - *Which kind of information does the explanation provide?*

The information type informs the user about the kind of information communicated. The most valuable information is close to the language of human reasoning, with causal and counterfactual rules [Pearl et al., 2018]. Causal rules can tell the user that certain observed variables are the causes of specific model predictions. However, machine learning usually leverages statistical associations in the data and do not convey information about the causal relationships among the observed variables and the unobserved confounding variables. The usual statistical associations discovered by machine learning methods highly depend on the machine learning task. Therefore, I first give a generic high-level definition of the information type and then I detail and illustrate it for the application case of this thesis (MTS classification). I propose a generic assessment of the information type in 3 categories from the least to the most informative:

- *Importance*: the explanations reveal the relative importance of each dataset variable on predictions. The importance indicates the statistical contribution of each variable to the underlying model when making decisions;
- *Patterns*: the explanations provide the small conjunctions of symbols with a predefined semantic (patterns) associated with the predictions;
- *Causal*: the most informative category corresponds to explanations under the form of causal rules;

In this thesis, the issue of Multivariate Time Series (MTS) classification is addressed. Thus, considering the MTS data type, the information can be structured around the features, i.e. the observed variables, and the time. I propose to decompose the 3 categories presented into 8 categories. In addition, I will illustrate each of these categories with an application in the medical field (see Figure 4.1), which is the same example as in the Introduction and State-of-the-Art of this thesis. It is worth noting that the explanations provided to illustrate each category are assumptive rather than validated, they are given as illustrative in nature.

- *Features* (*Importance*): the explanations reveal the relative importance of the features on predictions. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the channel 2 has a greater importance on the prediction than the channel 1;

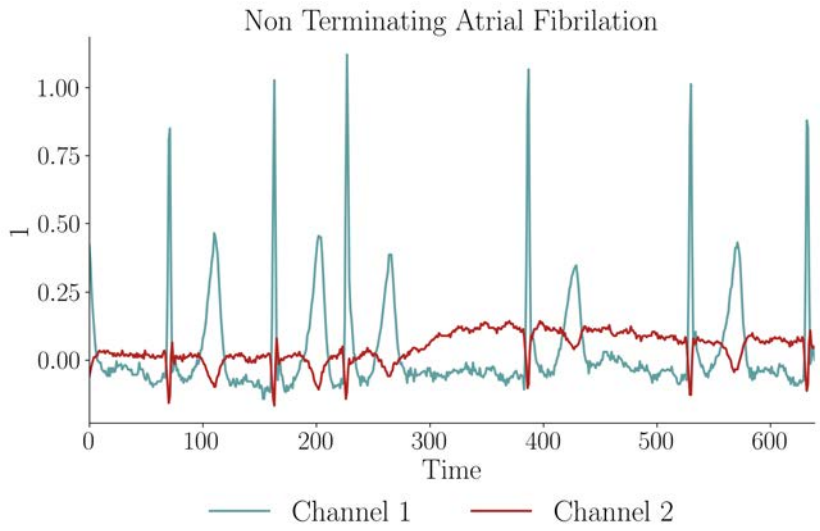



Figure 4.1 – The first MTS sample of the UEA Atrial Fibrilation test set. It belongs to the class *Non-Terminating Atrial Fibrilation* and is composed of two channels ECG on a 5 second period (128 samples per second).

- *Features + Time* (*Importance*): the explanations provide the relative importance of the features and timestamps on predictions. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the channel 2 has a greater importance on the prediction than the channel 1 and that the timestamps are in increasing order of importance on the prediction;
- *Features + Time + Values* (*Importance*): in addition to the relative importance of the features and timestamps on predictions, the explanations indicate the discriminative values of a feature for each class. For example in Figure 4.1, the explanations could give the same explanations as the previous category, plus, it could tell the user that the timestamps with the highest importance are associated with high values (values above 0.15) on the channel 2;
- *Uni Itemsets* (*Patterns*): the explanations provide patterns under the form of groups of values, also called itemsets, which occur per feature and are associated with the prediction. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the following itemsets are associated with the prediction: {channel 1: extremely high value (above 1); channel 1: low value (below -0.05)} and {channel 2: high value (above

0.15); channel 2: extremely low value (below -0.1)}. The first itemset can be read as: the prediction is associated with the occurence on the channel 1 of an extremely high value being above 1 and a low value being below -0.05 at another moment, without information on which one appears first;

- *Multi Itemsets* (*Patterns*): the explanations provide patterns under the form of multidimensional itemsets, i.e. groups of values composed of different features, which are associated with the prediction. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the following itemset is associated with the prediction: {channel 1: extremely high value (above 1); channel 2: high value (above 0.15)};
- *Uni Sequences* (*Patterns*): the explanations provide patterns under the form of ordered groups of values, also called sequences, which occur per feature and are associated with the prediction. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the following sequences are associated with the prediction: <channel 1: extremely high value (above 1); channel 1: low value (below -0.05)> and <channel 2: high values (above 0.15) with an increase during 1 second>. The first sequence can be read as: the prediction is associated with the occurrence on the channel 1 of an extremely high value being above 1 followed by a low value being below -0.05;
- *Multi Sequences* (*Patterns*): the explanations provide patterns under the form of multidimensional sequences, i.e. ordered groups of values composed of different features, which are associated with the prediction. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the following sequence is associated with the prediction: <channel 1: extremely high value (above 1); channel 2: high values (above 0.15) with an increase during 1 second>;
- *Causal*: the last category corresponds to explanations under the form of causal rules. For example, in order to support a model output from the MTS of the Figure 4.1, the explanations could tell the user that the following rule applies: if (channel 1: extremely high value (above 1)) & (channel 2: high values (above 0.15) with an increase during 1 second), then the MTS belongs to the class *Non-Terminating Atrial Fibrilation*.

**Faithfulness** - *Can we trust the explanations?*

The faithfulness corresponds to the level of trust an end-user can have in the explanations of model predictions, i.e. the level of relatedness of the explanations to what the model actually computes. An explanation extracted directly from the original model is faithful by definition. Some post-hoc explanation methods propose to approximate the behavior of the original "black-box" model with an explainable surrogate model. The explanations from the surrogate models cannot be perfectly faithful with respect to the original model [Rudin, 2019]. The fidelity criteria is used to quantify the faithfulness by the extent to which the surrogate model imitates the prediction score of the original model [Guidotti et al., 2018]. In the thesis, there is only one machine learning method using an explainable surrogate model among the 5 methods presented. Therefore, there is no need to distinguish between the degree of fidelity of the surrogate models in the framework. I propose an assessment of the faithfulness in two categories:

- *Imperfect*: imperfect faithfulness (use of an explainable surrogate model);
- *Perfect*: perfect faithfulness.

**User category** - *What is the target user category of the explanations?* The user category indicates the audience to whom the explanations are accessible. The user's experience will affect what kind of *cognitive chunks* they have, that is, how they organize individual elements of information into collections [Neath et al., 2003]. Thus, it could be interesting to categorize the user types and associate with the model to whom the explanations will be accessible to. The broader the audience, the better are the explanations. Therefore, an assessment in three categories is proposed:

- *Machine Learning Expert*;
- *Domain Expert*: domain experts (e.g. professionals, researchers);
- *Broad Audience*: non-domain experts (e.g. policy makers).

Table 4.1 summarizes the different aspects of the performance-explainability framework. In order to compare the methods visually using the proposed framework, the different aspects can be represented on a parallel coordinates plot. A parallel coordinate plot allows a 2-dimensional visualization of a high dimensional dataset and is suited for

Table 4.1 – Performance-explainability analytical framework. The table summarizes the different components of the framework, the corresponding name used in the figures of the thesis and the assessment method.

| **Component** | **Name** | **Assessment** |
|---|---|---|
| Performance | Performance | - *Below*: performance below that of the state-of-the-art models<br>- *Similar*: performance similar to that of the state-of-the-art models<br>- *Best*: best performance |
| Model Comprehensibility | Comprehensibility | - *Black-box*: "black-box" model<br>- *White-box*: "white-box" model |
| Granularity of the Explanations | Granularity | - *Global*: global explainability<br>- *Local*: local explainability<br>- *Global & Local*: both global and local explainability |
| Information Type | Information | - *Features*: ranking of the features<br>- *Features+Time*: ranking of the features and the timestamps<br>- *Features+Time+Values*: ranking of the features and the timestamps with the discriminative values of the features<br>- *Uni Itemsets*: set of values per feature<br>- *Multi Itemsets*: multidimensional set of values<br>- *Uni Sequences*: sequence of values per feature<br>- *Multi Sequences*: multidimensional sequence of values<br>- *Causal*: causal rules |
| Faithfulness | Faithfulness | - *Imperfect*: imperfect faithfulness<br>- *Perfect*: perfect faithfulness |
| User Category | User | - *Machine Learning Expert*<br>- *Domain Expert*<br>- *Broad Audience* |

the categorical data of this framework. The next section presents an example of parallel coordinates plots comparing two state-of-the-art MTS classifiers on the public UEA datasets [Bagnall et al., 2018].

## 4.3 Example

In this section, I give an example of comparison between two state-of-the art MTS classifiers with the analytical framework introduced.

The first MTS classifier belongs to the similarity-based category and is the one-Nearest Neighbor MTS classifier with Dynamic Time Warping distance ($\text{DTW}_I$) [Shokoohi-Yekta et al., 2017]. $\text{DTW}_I$ classifies MTS samples based on the label of their nearest sample. The similarity is calculated as the cumulative distances of all dimensions independently measured under DTW. For an individual MTS, the explanation supporting the prediction is the ranking of features and timestamps in decreasing order of their DTW distance with the nearest MTS. Figure 4.2 shows $\text{DTW}_I$ parallel coordinates plot on the UEA datasets. Based on predefined train/test splits and an arithmetic mean of the accuracies, $\text{DTW}_I$ underperforms the current state-of-the-art MTS classifiers on the UEA datasets (Performance: *Below*). However, $\text{DTW}_I$ model is comprehensible (Comprehensibility: *White-Box*) and provides faithful explanations (Faithfulness: *Perfect*) at local level (Granularity: *Local*). Nevertheless, the model $\text{DTW}_I$ conveys limited information (Information: *Features+Time*) that needs to be analyzed by a domain expert (User: *Domain Expert*).

The second MTS classifier belongs to the shapelet-based category and is the Ultra-Fast Shapelets (UFS) [Wistuba et al., 2015]. UFS models the input data as a set of distances from shapelets. Then, the classification of the input data with the shapelets as features can be performed by a decision tree. Figure 4.2 also shows UFS parallel coordinates plot on the UEA datasets. Based on the same evaluation method as $\text{DTW}_I$, UFS has the same performance level as current state-of-the-art MTS classifiers on the UEA datasets (Performance: *Similar*, $\text{DTW}_I$: *Below*). Concerning the explainability, similar to $\text{DTW}_I$, UFS model is comprehensible (Comprehensibility: *White-Box*) and provides faithful explanations (Faithfulness: *Perfect*) which are accessible by a domain expert (User: *Domain Expert*). However, UFS provides more informative explanations than $\text{DTW}_I$ (Information: *Uni Sequences*, $\text{DTW}_I$: *Features+Time*) at all granularity levels (Granularity: *Both Global & Local*, $\text{DTW}_I$: *Local*). A set of shapelets, i.e. a set of sequences per feature, can

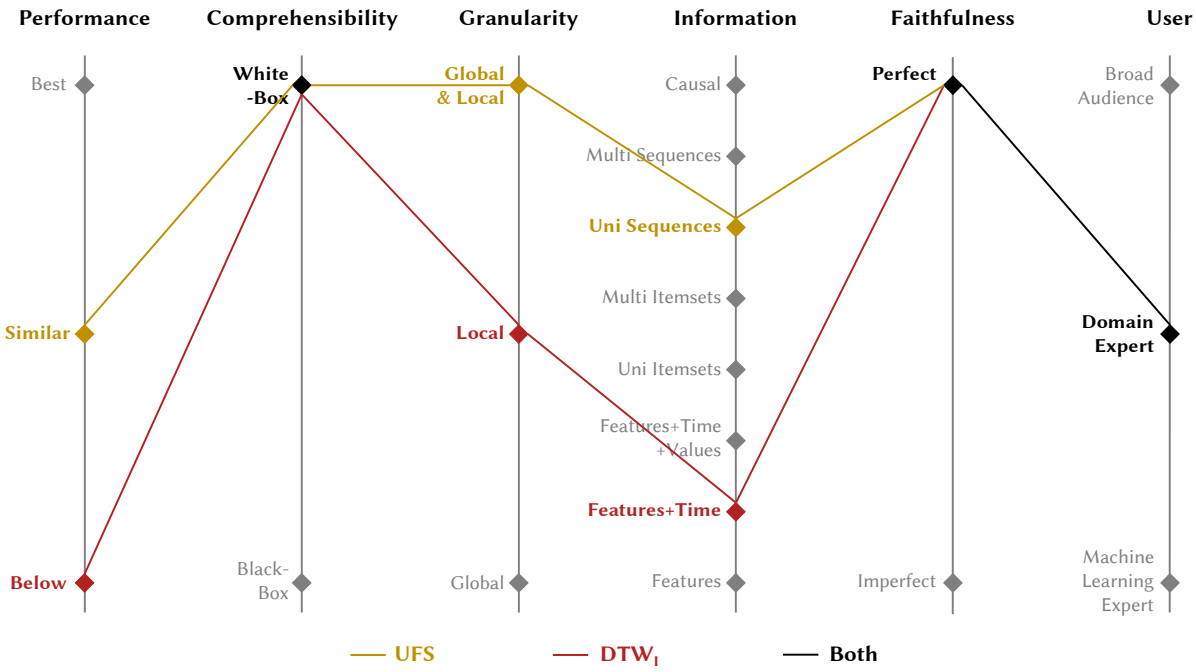



Figure 4.2 – Parallel coordinates plot of DTW$_I$ and UFS MTS classifiers on the UEA datasets. Performance evaluation method: predefined train/test splits and an arithmetic mean of the accuracies on the public UEA datasets. Models evaluated in the benchmark: DTW$_D$, DTW$_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.

be communicated to the domain expert to explain the overall behavior of the classifier or support the prediction of a particular instance.

Therefore, based on the performance-explainability analytical framework introduced in this chapter, it would be preferable to choose UFS instead of $\text{DTW}_I$ on average on the UEA datasets. In addition to its better level of performance, UFS provides more informative explanations and at a better granularity level.

## 4.4 In The Next Parts...

The performance-explainability framework presented is used to compare the machine learning methods to be presented in the following parts. First, the part III introduces new ensemble methods and discusses how a faithful explainable by design MTS classifier can compete with the level of information a post-hoc model-agnostic explainability method could provide, while maintaining performance. Next, a new pattern-based MTS classifier is presented in the chapter 8 and shows how to widen the audience with an easy-to-understand model compared to the ensemble methods of the part III. Finally, a new convolutional neural network MTS classifier is presented in the chapter 9 and shows how to improve the performance of the ensemble method of the part III, while offering faithfulness and more informative explanations than the post-hoc model-agnostic explainability method based on a post-hoc model-specific explainability method.

Part III

# Towards Faithful, Informative and Human-Friendly Explainability in Performant Ensemble Methods

This part is composed of three chapters about ensemble methods. Ensemble methods are the current state-of-the-art classifiers for traditional multivariate data classification and univariate time series classification, whereas there is no ensemble method among the state-of-the-art MTS classifiers. This part shows that ensemble methods can also be performant in the MTS classification setting, while providing some explainability.

The first chapter presents a new stacking ensemble method to earthquake early warning, which has been published in *AAAI 2020* [Fauvel et al., 2020a]. The performance criteria has shaped the design of this new stacking ensemble method but it provides faithful and local explanations. The second chapter introduces a new hybrid ensemble method for dairy resource monitoring, which has been published in *KDD 2019* [Fauvel et al., 2019b]. The post-hoc model-agnostic explainability method offers more informative explanations that could be useful to broader audiences compared to the first method. However, the enhanced explanations come at the cost of faithfulness, which is a prerequisite for numerous applications. Therefore, the third chapter presents an extension of the hybrid ensemble method which integrates faithfulness with explainability by design. This method competes with the level of information and the audience of the post-hoc model-agnostic explainability method while maintaining performance (available on ArXiv [Fauvel et al., 2020b]).

Chapter 5

# A Distributed Multivariate Time Series Ensemble Method to Earthquake Early Warning

## Contents


**5.1 Introduction . . . . . 64**
5.1.1 Contributions . . . . . 65
**5.2 DMSEEW . . . . . 66**
**5.3 Evaluation . . . . . 67**
5.3.1 Real-World Dataset . . . . . 68
5.3.2 Experimental Setting . . . . . 70
**5.4 Results . . . . . 71**
5.4.1 Sensor-Level Predictions . . . . . 72
5.4.2 Combined Predictions at Central Level . . . . . 72
**5.5 Performance-Explainability Analysis . . . . . 75**


The first machine learning method of the thesis aims to improve the accuracy of Earthquake Early Warning (EEW) systems (application background presented in section 3.2.1). The method is the result of a collaboration that I have jointly built with researchers from different institutions participating in a NSF project: they are researchers from the Department of Earth Sciences of the University of Oregon, USA, researchers in distributed computing from the Rutgers Discovery Informatics Institute, USA, and the KerData team at Inria, France, and researchers in machine learning from the LACODAM ("Large Scale Collaborative Data Mining") team at Inria, France. During this collaboration, I have had the chance to be a visiting researcher at the Rutgers Discovery Informatics Institute for a month in April 2019. This work has been published in *AAAI 2020* [Fauvel et al., 2020a] and received the "AAAI 2020 Outstanding Paper Award" in the category "AI for Social Impact".

## 5.1 Introduction

As presented in section 3.2.1, the detection of the whole spectrum of earthquakes with damaging potential, i.e. medium and large earthquakes, is an open problem in the field of EEW. To tackle this challenge, multi-sensor solutions leveraging GPS and seismometers complementary characteristics appear as a promising approach. Seismometers have a difficulty to detect and characterize large earthquakes due to a saturation issue caused by their sensitivity to ground motion velocity. Whereas, GPS stations are unable to characterize medium earthquakes, as they are prone to containing significant signals from a variety of noise sources, mostly of atmospheric origin. As EEW can be assimilated as a MTS classification problem, where the input is sensor data (3 dimensions: east-west, north-south and up-down) and the output is a class (normal activity/medium earthquake/large earthquake); machine learning approaches designed to combine large volumes of data from multiple data sources can be applied.

Integrating and processing high-frequency data streams from multiple sensors scattered over a large territory in a timely manner requires high-performance computing techniques and equipments. Thus, a machine learning earthquake detection solution has to be designed with experts in distributed computing and cyberinfrastructure to enable real-time alerts. A cyberinfrastructure is the set of logical and physical computational systems onto which a scientific application is deployed. Because of the large number of sensors and their high sampling rate, a traditional centralized approach which transfers all data to a single point may be impractical.

Current approaches to EEW in the literature make use of centralized data processing strategies: all sensors send their data, through a network, to a data center where processing will take place [Fischer et al., 2012]. This strategy implicitly depends on cyberinfrastructures supporting almost insignificant network latency and very high bandwidth, often provided by costly fiber networks. Furthermore a lot of trust is placed in the reliability of the network and telemetry paths [Allen et al., 2019]. In the case of large earthquakes, power failures and wiring disconnections can frequently lead to regional shutdowns, as happened after the magnitude-9 earthquake in Japan in 2011 [Hoshiba et al., 2012].

In this work, a distributed cyberinfrastructure is targeted for executing the proposed EEW system, meaning that data processing tasks can be performed in different parts of the infrastructure and at different locations. In particular, processing part of the data at the edge of the network [Satyanarayanan, 2017; Shi et al., 2016], i.e. as close as possible

to the sources of data, is favored in order to reduce the amount of data transferred to the main data center [Yang et al., 2010].

Machine learning in seismology is still a developing field. As presented in section 3.2.2, there are a couple of studies using machine learning methods for earthquake characterization based on P-wave detection (EEW). However, none of them used a combination of GPS and seismometers data so the whole spectrum of earthquakes with damaging potential is not appropriately covered. Additionally, none of them used a distributed approach. Consequently, a distributed EEW machine learning-based solution that can be generalized to the whole spectrum of earthquakes is designed.

### 5.1.1 Contributions

EEW MTS classification critical problem is particularly interesting to study from the performance and explainability perspectives of this thesis. The detection performance is a prerequisite in this application and explanations can support the decision-makers (communities, organizations and governments) in the short time window of seconds to minutes they have to take protective actions.

Therefore, in this chapter, MTS machine learning methods are used to address the most urgent challenges faced by EEW systems, i.e. integrating multiple data sources in real-time to cover the whole spectrum of potentially damaging earthquakes (medium and large). The solution relies on two complementary types of sensors (GPS stations and seismometers). A new machine learning technique specifically tailored to allow efficient computation on large-scale distributed cyberinfrastructures is introduced. The study will:

- Propose a new EEW approach to characterize the whole spectrum of earthquakes with damaging potential (both medium and large) using a real-world dataset collected and validated with geoscientists combining two complementary data sources;
- Present DMSEEW (Distributed Multi-Sensor Earthquake Early Warning), a new stacking ensemble method jointly designed with cyberinfrastructure experts, which enables real-time earthquake detection and robustness to partial infrastructure failures;
- Show that DMSEEW is more accurate than both the seismometer-only baseline approach, the combined sensors (GPS and seismometers) baseline approach that adopts the rule of relative strength and detects all large earthquakes with a precision of 100%;

- Detail how DMSEEW provides local and faithful explanations on its predictions by design.

## 5.2 DMSEEW

In this section, I present the Distributed Multi-Sensor Earthquake Early Warning algorithm (DMSEEW), a new two-step stacking ensemble method for earthquake detection. A stacking ensemble is a method which takes the predictions of sub-models as inputs and then attempts to learn how to best combine the input predictions to make a better output prediction. DMSEEW takes sensor-level class predictions (normal activity, medium earthquake or large earthquake) based on the data gathered by each individual sensor (GPS stations and seismometers). It then aggregates those sensor-level class predictions using a bag-of-words representation in order to calculate a final prediction for the earthquake category.

**Step 1 – Predicting the MTS Category at the Sensor-Level:** There are two types of sensors - GPS stations and seismometers, and one MTS classifier per sensor type is trained. The classifiers are trained using a dataset composed of time series of 3 dimensions (east-west, north-south and up-down) and fixed time length (60 seconds, defined in Section 5.3.1). This first step of the approach is illustrated in the upper part of Figure 5.1.

In order to predict the earthquake category at the individual sensor level, WEASEL +MUSE [Schäfer et al., 2017] MTS classifier is employed. As presented in section 2.2.2, WEASEL +MUSE creates a symbolic representation of the MTS (a Symbolic Fourier Approximation - SFA) on each dimension, then generates a set of features (multiple window lengths, unigrams, bigrams, dimension identification), and finally performs the classification based on a one-hot encoding representation of the MTS (bag-of-words, feature selection). WEASEL+MUSE fits the approach because (*i*) its symbolic representation filters out noise (related to GPS and seismometers sensors) from the dataset; (*ii*) it is phase invariant, i.e. features generated do not have to appear at the same time across different MTS belonging to the same class, which improves generalization; (*iii*) it keeps the interplay of dimensions since features generated by WEASEL+MUSE contain the identifier of the dimension, which allows the characterization of co-occurrence of events on different dimensions. As further discussed in the Section 5.4, WEASEL+MUSE outperforms other

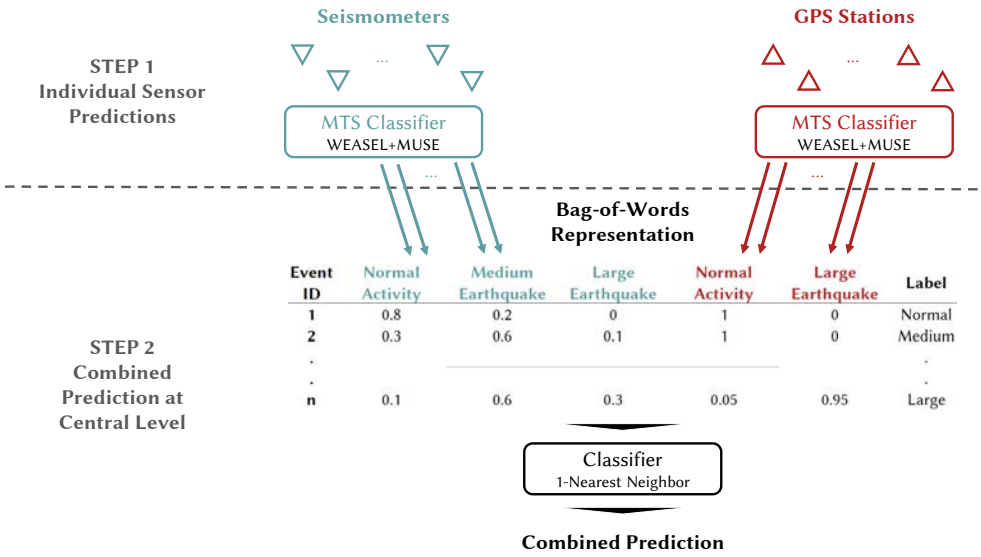



Figure 5.1 – Distributed Multi-Sensor Earthquake Early Warning Approach (DMSEEW).

MTS classifiers on both GPS and seismometers data.

**Step 2 – Detecting Earthquakes by Combining Sensor-level Predictions:** The class predictions from the different sensors (GPS stations and seismometers) are collected and a bag-of-words representation is performed. Each sensor-predicted class is considered to be a word and the relative frequency vector of the words from each earthquake is used to classify its category. This frequency vector is normalized by the number of instances (number of MTS per earthquake, i.e. number of sensors) to obtain the relative frequency vector. The last step consists of combining the bag-of-words of GPS stations and seismometers to characterize the whole spectrum of earthquakes with damaging potential. This second step of the approach is illustrated in the lower part of Figure 5.1. For example, 80% of seismometers and 100% of GPS stations for event 1 predict that the activity recorded is normal. Finally, a classifier is trained on this bag-of-words representation to perform the combined class prediction. As presented in section 5.4, 1-nearest neighbor outperforms other classifiers.

## 5.3 Evaluation

In this section, the methodology and datasets used for evaluating the work are introduced, as well as the preprocessing routines and experimental setting. In addition, the

real-world dataset collected and validated with geoscientists is rendered public.

### 5.3.1 Real-World Dataset

A real-world dataset [1] [Fauvel et al., 2019a] is employed. It is composed of GPS and seismometers data on normal activity/medium earthquakes/large earthquakes collected and validated with geoscientists. There are two main difficulties to construct such a dataset: *i*) large earthquakes are rare and *ii*) GPS data is not continuously recorded. The dataset has been built around the most complete GPS/seismometers dataset of large earthquakes (29 earthquakes worldwide) which occurred between 2001 and 2018 [Ruhl et al., 2019] with the corresponding metadata (time, magnitude, and location). A differentiated approach between GPS and seismometers data is adopted to augment the dataset, which is presented in the following two paragraphs. MTS length is set to 60 seconds for both GPS and seismometers. This value reflects the relevant time window to distinguish primary waves from noise across geographical regions, as recognized by the geoscience community.

First, the two main seismometers data repositories worldwide are the American Incorporated Research Institutions for Seismology (IRIS) and the Japanese National Research Institute for Earth science and Disaster Resilience (NIED). Earthquake origins are defined differently between the two repositories, preventing a direct comparison of P-wave arrival time on each seismometer. Therefore, in order to be able to adopt a homogeneous labeling method, the study is limited to the data available from IRIS (14 large earthquakes remaining over 29). Seismometers data corresponding to medium earthquakes are sampled from medium earthquakes occurring in the same region as large earthquakes ($-179 \leq$ longitude $\leq 25$, $-62 \leq$ latitude $\leq 73$). The number of medium earthquakes is calculated by the ratio of medium over large earthquakes during the past 10 years in the region. Then, a ratio above 30% between the number of MTS corresponding to earthquakes (medium + large) and total (earthquakes + normal activity) number of MTS is kept to prevent a class imbalance issue during the training phase. So, two normal activity MTS for each medium earthquake MTS (9 and 7 minutes before each medium earthquake) are collected to respect this ratio. IRIS data (normal activity, medium earthquakes) is collected with the international Federation of Digital Seismograph Networks (FDSN) client available in Python package ObsPy [2]. Based on geoscience expertise, the relevant region of seismome-

1. https://figshare.com/articles/Earthquake_Early_Warning_Dataset/9758555
2. https://docs.obspy.org/packages/obspy.clients.fdsn.html

ters is set to 1,000 kilometers around the earthquake epicenter.

Second, unlike seismometers data, GPS displacement data is not continuously recorded. Furthermore, GPS data outside of large earthquake periods can be considered as normal activity (noise). Hence, the approach based on GPS sensors characterizes only normal activity and large earthquakes. I collected GPS normal activity data from an archive of real-time GPS positions maintained by the University of Oregon [3] which stores a representative extract of GPS noise. Normal activity MTS are randomly sampled from the archive to match the number of seismometers events (255, normal activity and medium earthquakes) and to keep a ratio above 30% between the number of large earthquakes MTS and normal activity in order to avoid class imbalance issues.

The number of sensor records available varies between earthquakes according to the location and the magnitude of the earthquake. The full dataset composition is presented in Table 5.1.

Table 5.1 – Dataset composition.

| # 60s MTS | Seismometers (# Events) | GPS (# Events) |
|---|---|---|
| Normal Activity | 7,718 (170) | 1,424 (255) |
| Medium Earthquakes | 3,859 (85) | None |
| Large Earthquakes | 1,688 (14) | 648 (14) |
| Total | 13,265 (269) | 2,072 (269) |

**Preprocessing** First, seismometers data are available as digital signal, which is specific for each sensor. Therefore, each instrument digital signal is converted to its physical signal (acceleration) to obtain comparable seismometers data. Second, standardization (StandardScaler [Pedregosa et al., 2011]) of the GPS and seismometers data (fitted on train sets and applied on test sets) are performed to harmonize the different scales. Standardization procedure allows us to keep outliers, which are fundamental in P-wave detection, as compared to the normalization procedure. Finally, data aggregation by second (mean) is performed which permits a common time scale between sensors (frequency between sensors can differ) without deteriorating the P-wave signal.

3. http://tunguska.uoregon.edu/rtgnss/data/cwu/mseed/

### 5.3.2 Experimental Setting

In this section, the algorithms evaluated and the methods used to assess them are presented.

#### Algorithms

Different algorithms at sensor-level and central level are evaluated according to the data type in order to define the two blocks of the machine learning solution (Figure 5.1).

At sensor-level, there is a multivariate time series classification task. As detailed in section 2.2, MTS classifiers are composed of 3 categories: similarity-based, feature-based and deep learning methods. In this work, the best-in-class for each category is chosen: $\text{DTW}_D$, $\text{DTW}_I$, WEASEL+MUSE and MLSTM-FCN classifiers. Therefore, the following algorithms are compared:

- $\text{DTW}_D$ and $\text{DTW}_I$: I use the public implementation [4] based on the original paper [Shokoohi-Yekta et al., 2017];
- WEASEL+MUSE: I use the public implementation [5] with the recommended settings (SFA word lengths l in [2,4,6], windows length in [4:60], chi=2, bias=1, p=0.1, c=5 and a solver equals to L2R LR DUAL) [Schäfer et al., 2017];
- MLSTM-FCN: I test the public implementation [6] based on the original paper [Karim et al., 2019], using the recommended settings (128-256-128 filters, 250 training epochs, a dropout of 0.8 and a batch size of 128);

At central level, there is a classification task on a bag-of-words representation (relative frequency vector) for each earthquake based on individual class predictions of GPS stations and seismometers. I compare the state-of-the-art classifiers with the following implementations: K-Nearest Neighbors [7]; Elastic Net[7]; Support Vector Machine[7] with a radial basis function kernel due to the lower number of features than the number of samples in the dataset; Random Forest[7] and Extreme Gradient Boosting [8].

4. https://github.com/DavideNardone/MTSS-Multivariate-Time-Series-Software
5. https://github.com/patrickzib/SFA
6. https://github.com/titu1994/MLSTM-FCN
7. https://scikit-learn.org/stable/
8. https://xgboost.readthedocs.io/en/latest/python/

### Hyperparameters

Firstly, at sensor-level, classifier hyperparameters setting is presented in previous section with the public implementations of the algorithms used. Next, hyperparameters of classifiers at central level are set by hyperopt, a sequential model-based optimization using a tree of Parzen estimators search algorithm [Bergstra et al., 2013]. Hyperopt chooses the next hyperparameters decision from the previous choices and a tree-based optimization algorithm. Tree of Parzen estimators meet or exceed grid search and random search performance for hyperparameters setting [Bergstra et al., 2011]. I use the implementation available in the Python package hyperopt [9]. Optimization is undertaken to maximize accuracy score considering the multiclass study.

### Classification Performance

Classifiers are trained with a 3 class labeling on seismometers data (sensor-level, see section 5.3.1), a 2 class labeling on GPS data (sensor-level, see section 5.3.1) and a 3 class labeling on the bag-of-words representation (central level). A stratified k-fold cross-validation which kept the same proportion of earthquakes of different categories for each fold is performed. K is set to 3 considering the number of large earthquakes (14 earthquakes). The dataset split is presented in Table 5.2. Therefore, the results presented corresponds to the 3-fold performance on the test sets.

Table 5.2 – Cross-validation split.

| # Events | Fold 1 | Fold 2 | Fold 3 | Total |
|---|---|---|---|---|
| **Seismometers** | **90** | **90** | **89** | **269** |
| Normal Activity | 56 | 57 | 57 | 170 |
| Medium Earthquakes | 29 | 28 | 28 | 85 |
| Large Earthquakes | 5 | 5 | 4 | 14 |
| **GPS** | **90** | **90** | **89** | **269** |
| Normal Activity | 85 | 85 | 85 | 255 |
| Medium Earthquakes | None | None | None | None |
| Large Earthquakes | 5 | 5 | 4 | 14 |

## 5.4 Results

This section first presents the results at sensor-level (DMSEEW step 1). Then, the performance of the combined approach (DMSEEW steps 1 and 2) is compared to the

9. https://github.com/hyperopt/hyperopt

traditional seismometers baseline approach and the combined sensors (GPS and seismometers) baseline approach that adopts the rule of relative strength.

### 5.4.1 Sensor-Level Predictions

The accuracy results of the different MTS classifiers on seismometers (13,265 MTS, 3 classes) and GPS stations (2,072 MTS, 2 classes) are presented in Table 5.3.

Table 5.3 – Accuracy score on test sets of the MTS classifiers trained on GPS or seismometers data.

| **Accuracy (%)** | **DTW$_D$** | **DTW$_I$** | **MLSTM - FCN** | **WEASEL + MUSE** |
|---|---|---|---|---|
| Seismometers | 35.3 | 35.5 | 54.6 | **63.6** |
| GPS | 97.9 | 97.8 | 98.9 | **99.5** |

WEASEL+MUSE outperforms MLSTM-FCN and similarity-based classifiers (DTW$_D$ and DTW$_I$) on both GPS and seismometers data. The difference between WEASEL +MUSE and other classifiers is particularly important on seismometers data. We can infer that the noise reduction performed by the truncated Fourier Transform and discretization of WEASEL+MUSE led to a better exploitation of the P-wave information.

The detection results obtained from both GPS stations and seismometers data confirm the complementary performance of these sensors. GPS data distinguishes large earthquakes while the detection based on seismometers data performs poorly (F1-score on large earthquakes: GPS 99%, seismometers 28%). In the next section, I present the results of a combined prediction benefiting from the complementary performance of these sensors following the transfer of all sensor-level class predictions to a central computation facility.

### 5.4.2 Combined Predictions at Central Level

DMSEEW benefits from the complementary performances through the combination of sensor-level class predictions (GPS and seismometers) using a bag-of-words representation, followed by the training of a classifier on this representation. There is no state-of-the-art method covering the whole spectrum of earthquakes with damaging potential (medium and large). In order to evaluate the performance of DMSEEW, two baselines are defined at central level.

The first is the traditional seismometer approach which relies on seismometers data only. I compute the performance of the traditional seismometer approach by calculating

the arg max directly on the seismometers bag-of-words representation for each of the 269 events in the dataset. The second baseline corresponds to the combined sensors (GPS and seismometers) baseline approach that adopts the rule of relative strength in order to assess the value added by DMSEEW combining approach. It is defined based on the strengths of each sensor type: if the GPS bag-of-words representation indicates that the event is a large earthquake, a large earthquake is predicted. Else, I calculate the arg max between normal activity and medium earthquake on the seismometers bag-of-words representation. Table 5.4 shows the performance of the two baselines and DMSEEW. The traditional seismometer approach is indicated as "Baseline Seismometer" and the combined rule-based approach as "Baseline Combined". The DMSEEW scores correspond to average results on test sets of the 1-Nearest Neighbors (1NN) trained on the combined bag-of-words representation (GPS and seismometers representations). 1NN outperforms other classifiers (Elastic Net, Support Vector Machine, Random Forest, Extreme Gradient Boosting) on the 3-fold cross-validation.

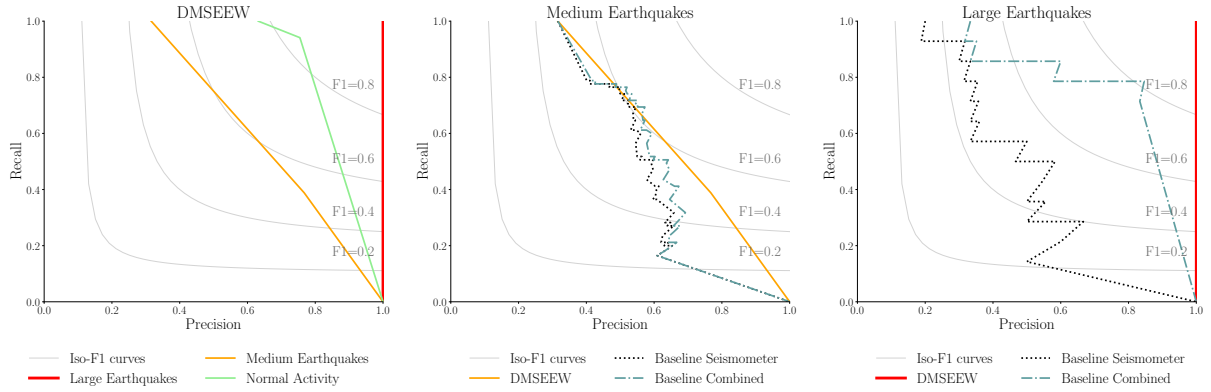



Figure 5.2 – Precision-recall curves of DMSEEW.

The dataset has a class imbalance (normal activity 63%/medium earthquakes 32%/large earthquakes 5%), but it does not affect the detection performance on the least represented class. DMSEEW detects all the large earthquakes (100.0% recall) without false alert (100.0% precision). The 1-NN of DMSEEW is always able to closely match an existing typical distribution of GPS predictions in case of large earthquakes, which allows the correct 1-NN classification. It is critical for an EEW system to detect all the large earthquakes with a precision of 100%. The decisions subsequent to a large earthquake alert imply major mitigation measures for the population possibly impacted. We observe

Table 5.4 – Performances on test sets of DMSEEW and the two baselines. Standard errors are presented in parentheses.

| | **Baseline Seismometer** | **Baseline Combined** | **DMSEEW** |
|---|---|---|---|
| **Normal Activity** | | | |
| Precision (%) | 76.2 (1.8) | **76.6 (1.5)** | 75.5 (1.8) |
| Recall (%) | **94.1 (3.2)** | **94.1 (3.2)** | **94.1 (3.2)** |
| F1 (%) | 84.2 (0.6) | **84.4 (0.7)** | 83.8 (0.9) |
| **Medium Earthquakes** | | | |
| Precision (%) | 65.9 (12.2) | 70.7 (10.5) | **76.7 (9.7)** |
| Recall (%) | 34.1 (11.8) | 34.1 (11.8) | **38.8 (7.3)** |
| F1 (%) | 45.0 (11.5) | 46.0 (12.0) | **51.6 (6.1)** |
| **Large Earthquakes** | | | |
| Precision (%) | 53.3 (17.9) | 63.2 (16.2) | **100.0 (0.0)** |
| Recall (%) | 57.1 (19.2) | 85.7 (13.3) | **100.0 (0.0)** |
| F1 (%) | 55.2 (11.6) | 72.7 (6.1) | **100.0 (0.0)** |
| **Total** | | | |
| Accuracy (%) | 73.2 (1.5) | 74.7 (1.8) | **76.9 (1.6)** |

in Table 5.4 that DMSEEW outperforms both baselines (accuracy score: 76.9% versus 74.7% and 73.2%). Moreover, DMSEEW outperforms both baselines on medium and large earthquakes detection. Figure 5.2 shows the precision-recall curves of DMSEEW versus both baseline on medium and large earthquakes (second and third plots). Firstly, DMSEEW obtains an average F1-score on test sets for medium earthquakes detection of 51.6% versus 45.0% for the baseline seismometer and 46.0% for the baseline combined. The higher F1-score of DMSEEW on medium earthquakes compared to both baselines is driven by higher performances on both precision and recall (precision: 76.7% versus 65.9% baseline seismometer versus 70.7% baseline rule-based, recall: 38.8% versus 34.1% baselines). Lastly, DMSEEW obtains an average F1-score on test sets for large earthquake detection of 100.0% versus 55.2% for the baseline seismometer and 72.7% for the baseline combined. The higher F1-score of DMSEEW on large earthquakes compared to both baselines is also driven by higher performances on both precision and recall (precision: 100.0% versus 53.3% baseline seismometer versus 63.2% baseline rule-based, recall: 100.0% versus 57.1% baseline seismometer versus 85.7% baseline rule-based). These performances confirm the interest of combining GPS stations and seismometers data to cover the whole spectrum of earthquakes with damaging potential (medium and large). In addition, it reveals the benefit of DMSEEW combined approach instead of the combined sensors (GPS and seismometers) baseline approach that adopts the rule of relative strength.

## 5.5 Performance-Explainability Analysis

This section introduces the new ensemble method DMSEEW into the analytical framework of the thesis (part II). The different aspects of the DMSEEW framework can be visualized in Figure 5.3.

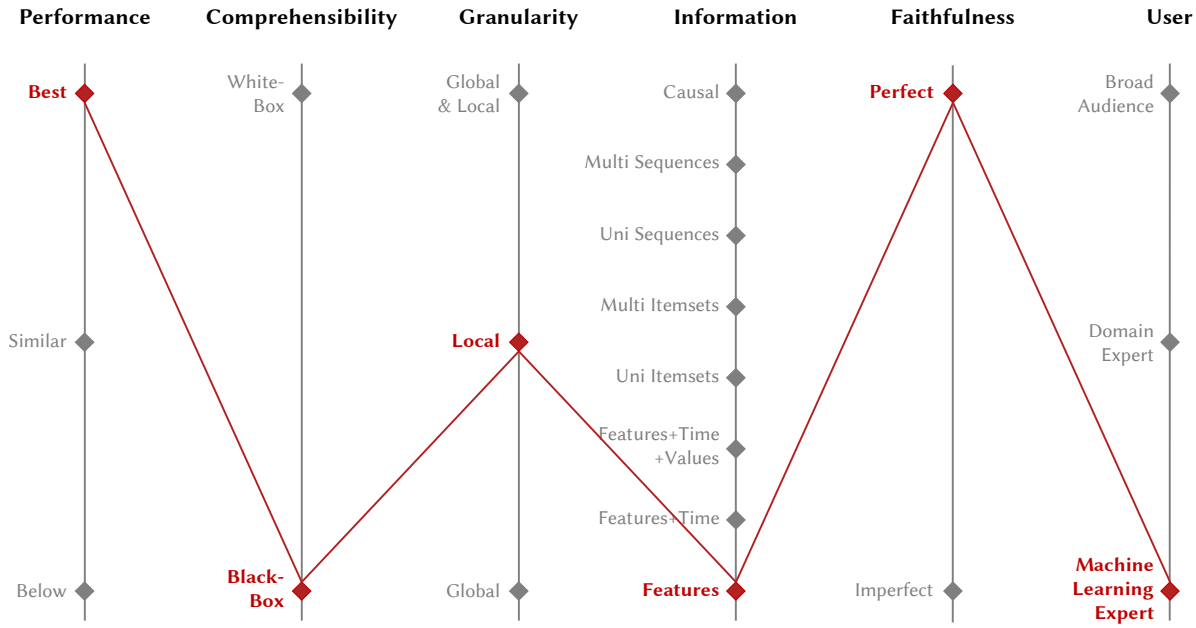



Figure 5.3 – Parallel coordinates plot of DMSEEW. Performance evaluation method: 3-fold cross-validation and an arithmetic mean of the accuracies on the Earthquake Early Warning Dataset. Models evaluated in the benchmark: $\text{DTW}_D$, $\text{DTW}_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.

DMSEEW is a novel stacking ensemble approach for characterizing the whole spectrum of earthquakes with damaging potential by combining both GPS and seismometer data. The evaluation on a real-world dataset collected with domain experts demonstrates that the distributed stacking ensemble approach improves the detection of both medium and large earthquakes compared to traditional seismometer-only approach and the combined sensors (GPS and seismometers) baseline approach that adopts the rule of relative strength. Therefore, in the framework presented in part II, following a 3-fold cross validation and an arithmetic mean of the accuracies, the performance is better than the state-of-the-art (Performance: *Best*).

Concerning the explainability, the stacking ensemble method is a “black-box” model

(Comprehensibility: *Black-Box*). However, DMSEEW provides faithful (Faithfulness: *Perfect*) and local (Granularity: *Local*) explanations even if the level of information is low (Information: *Features*). For each event, DMSEEW provides the relative sensor type importance by design. The bag-of-words central representation input to the 1-nearest neighbor classifier gives the proportion of class predictions per sensor type (GPS stations, seismometers). A higher proportion of GPS stations predicting a large earthquake indicates a higher importance of GPS stations on the prediction compared to seismometers. In any other case, seismometers carry a higher importance than GPS stations on the prediction. Nevertheless, these explanations are only accessible to a machine learning expert who can analyze the bag-of-words central representation (User: *Machine Learning Expert*).

The next chapter shows how a post-hoc model-agnostic explainability method on a "black-box" ensemble method can enhance the level of information of the explanations presented in this chapter and make it accessible to a broader audience.

**Summary**

- DMSEEW is a new stacking ensemble method which improves the detection of earthquakes with damaging potential on a real-world dataset.
- DMSEEW provides faithful and local explanations.
- However, the explanations are only accessible to machine learning experts and the level of information is limited to the relative sensor type importance.

Chapter 6

# A Hybrid Ensemble Method for Dairy Resource Monitoring

## Contents


**6.1 Introduction . . . . . 78**
6.1.1 Contributions . . . . . 79
**6.2 Local Cascade Ensemble . . . . . 80**
6.2.1 Local Cascade - LC . . . . . 80
6.2.2 Local Cascade Ensemble - LCE . . . . . 81
**6.3 Evaluation . . . . . 83**
6.3.1 Datasets . . . . . 83
6.3.2 Experimental Setting . . . . . 86
**6.4 Results . . . . . 88**
6.4.1 Public Datasets . . . . . 88
6.4.2 Real-World Application . . . . . 90
**6.5 Peformance-Explainability Analysis . . . . . 96**


The second machine learning method of the thesis aims to improve the detection of determining events for milk production in dairy farms, which is crucial for an optimal resource use (application background presented in section 3.1.1). It introduces a new hybrid ensemble method with a post-hoc model-agnostic explainability, which offers more informative explanations that could be useful to broader audiences compared to the first method DMSEEW presented in chapter 5. The method is the result of a collaboration between researchers from the PEGASE ("Physiology, Environment, Genetics for Animals and rearing systems") unit at the French National Institute for Agriculture, Food and Environment (INRAE) and machine learning researchers from the LACODAM ("Large Scale Collaborative Data Mining") team at Inria, France. This work has been published in *KDD 2019* [Fauvel et al., 2019b].

## 6.1 Introduction

As presented in section 3.1.1, there is a need to enhance the performance of estrus detection solutions based on affordable sensor data, and to support the alerts with explanations. Affordable activity and body temperature sensor data are considered having potential for automatic estrus detection [Saint-Dizier et al., 2012]. As detailed in section 3.1.2, there are multiple studies about the application of machine learning methods on estrus detection. However, none of them uses the currently recognized method for behavioral and silent estrus identification as labels (progesterone profiles), so their estrus labeling methods are not exhaustive. Moreover, two studies use different variables (milk volume, milking order, days since last estrus) rather than the affordable activity or temperature measurements. Finally, none of them gives insights on algorithm predictions based on its explainability.

In the experimental farm of INRAE, MTS collected from activity and temperature sensors can be labeled as either estrus or anestrus, the period of sexual inactivity between two periods of estrus, using the exhaustive estrus labeling method to cover both behavioral and silent estrus. Thus, estrus detection can be formulated as a binary MTS classification problem where both performance and explainability are required. However, data from sensors are 24hr aggregated, which corresponds to the relevant window for both estrus detection and, from an alert standpoint, farmers' needs; and according to animal scientists, data on the day of estrus and the day before estrus could be sufficient for estrus detection (time series length equals to two). Such a short time window prevents the use of most state-of-the-art MTS classifiers (feature-based classifiers, see section 2.2). Therefore, estrus detection is approached as a traditional multivariate data classification task in this chapter, where the time aspect is managed by setting the different timestamps as column variables in the dataset. Following the results presented in this chapter, we will see in the next chapters about this application (chapter 8 and chapter 9) that the estrus detection can be approached as a MTS classification task.

As detailed in section 2.1, the combination of different classifiers - an ensemble method - is often considered a good method to obtain a better generalizing classifier. In particular, a hybrid ensemble method is encouraged. Therefore, a new hybrid ensemble method for multivariate data classification, called Local Cascade Ensemble (LCE), is proposed in this chapter. Concerning the explainability, the problem requires explanations at both global (estrus/anestrus) and local (behavioral/silent) levels, which are two levels of infor-

mation interesting for the support of farmers' decision-making as detailed in section 6.4.2. LCE "black-box" classifier does not provide explainability by design or have a post-hoc model-specific explainability method, so it can only rely on post-hoc model-agnostic explainability methods. As presented in section 2.3.3, the state-of-the-art method meeting these requirements (both global and local, model-agnostic) is SHapley Additive exPlanations (SHAP) [Lundberg et al., 2017]. Therefore, SHAP is adopted in order to explain the output of LCE. This technique is inspired by game theory, which is used to determine how much each player in a collaborative game has contributed to its success. In this study, SHAP values measure how much an activity or temperature variable impacts estrus predictions. A higher absolute SHAP value of a variable compared to other variables means that this variable has a higher predictive or discriminative power in detection algorithm. SHAP interaction values, an extension of SHAP values based on Shapley interaction index [Fujimoto et al., 2006], capture pairwise interaction effects.

Consequently, a new hybrid ensemble method (LCE) with a post-hoc model-agnostic explainability method (SHAP) to enhance estrus detection is proposed.

### 6.1.1 Contributions

The research of this chapter tackles the challenge of improving the monitoring of resource use in dairy farms with machine learning methods. Based on affordable data (activity, temperature), it aims to enhance estrus detection, especially on the currently undetected silent estrus, and provides explanations to support farmers' decisions. Thus, with a real-world data analysis and an exhaustive estrus labeling (behavioral, silent) approach, the study in this chapter will:

- Present Local Cascade Ensemble (LCE), a new hybrid ensemble method for multivariate data classification;
- Show that LCE outperforms the state-of-the-art classifiers on the public UCI datasets [Dua et al., 2017] and significantly outperforms a commercial reference in estrus detection;
- Evaluate the relevance of deploying a combination of 2 affordable sensors (activity and temperature) in estrus detection;
- Identify the key drivers supporting estrus alerts at global (estrus/anestrus) and local (behavioral/silent) levels based on a post-hoc model-agnostic explainability method (SHAP) and propose an approach to reduce solution mistrust.

## 6.2 Local Cascade Ensemble

As introduced in section 2.1.1, a new hybrid ensemble method is proposed, LCE, which combines a boosting-bagging approach to handle the bias-variance trade-off and a divide-and-conquer approach to learn different parts of the training data. LCE is an improved hybrid (explicit and implicit) version of an implicit cascade generalization approach [Sesmero et al., 2015]: Local Cascade (LC) [Gama et al., 2000]. Among the implicit approaches, LC is one of the easiest to augment with explicit techniques. LC uses a decision tree as a divide-and-conquer method, which is compatible with the explicit bagging/boosting approaches. This criteria has motivated the choice of LC algorithm as the starting point for the hybrid ensemble method. In this section, I first introduce LC and then I explain LCE. Figure 6.1 illustrates the different algorithms.

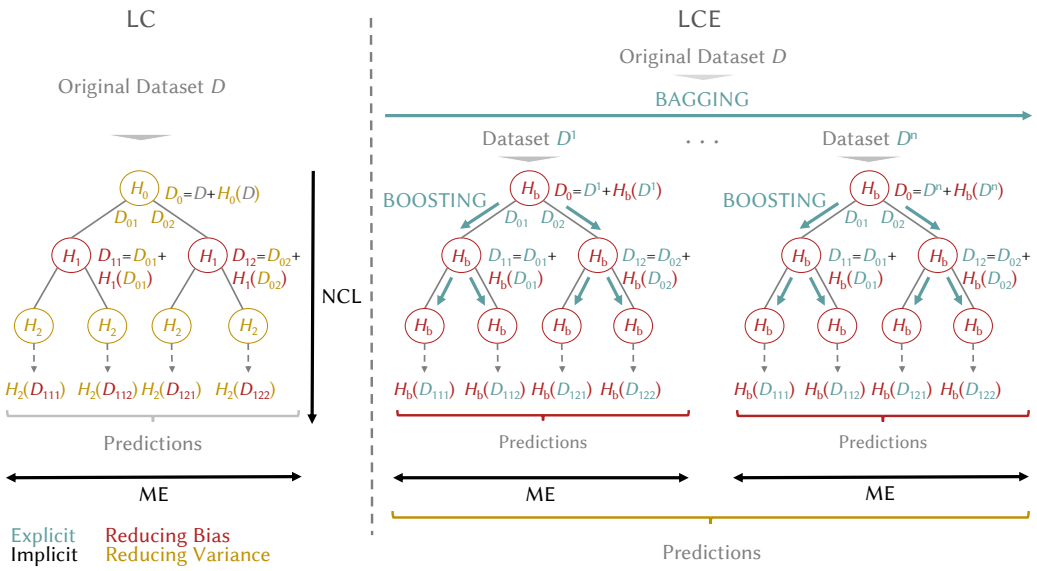



Figure 6.1 – Local Cascade LC versus Local Cascade Ensemble LCE. Abbreviations: $H_i$ - base classifier trained on a dataset at a tree depth of i, $D_i$ - dataset at a tree depth of i augmented with the class probabilities of the base classifier $H_i$, NCL - Negative Correlation Learning, ME - Mixture of Experts.

### 6.2.1 Local Cascade - LC

LC is a combined implicit approach (negative correlation learning and mixture of experts) based on a cascade generalization [Sesmero et al., 2015]. Cascade generalization uses a set of classifiers sequentially and at each stage adds new attributes to the original

dataset. The new attributes are derived from the class probabilities given by a classifier, called a base classifier (e.g. class probabilities $H_0(D)$, $H_1(D_{01})$ in Figure 6.1). The bias-variance trade-off is obtained by negative correlation learning: at each stage of the sequence, classifiers with different behaviors are selected. It is recommended in cascade generalization to begin with a low variance algorithm to draw stable decision surfaces ($H_0$ in Figure 6.1) and then use a low bias algorithm to fit more complex ones ($H_1$ in Figure 6.1). LC applies cascade generalization locally following a divide-and-conquer strategy based on mixture of experts. The objective of this approach is to capture new relationships that cannot be discovered globally. The LC divide-and-conquer method is a decision tree. When growing the tree, new attributes (class probabilities from a base classifier) are computed at each decision node and propagated down the tree. In order to be applied as a predictor, local cascade stores, in each node, the model generated by the base classifier.

### 6.2.2 Local Cascade Ensemble - LCE

The contribution of LCE intervenes in the explicit manner of handling the bias-variance trade-off whereas LC approach is implicit, alternating between base classifiers behaviors (bias reduction, variance reduction) at each level of the tree. LCE is a hybrid ensemble method which combines an explicit boosting-bagging approach to handle the bias-variance trade-off and an implicit decision tree divide-and-conquer approach as LC. Firstly, LCE reduces bias across decision tree divide-and-conquer approach through the use of boosting-based classifiers as base classifiers ($H_b$ in Figure 6.1). A boosting-based classifier iteratively changes the data distribution with its reweighting scheme which decreases the bias. In addition, boosting is propagated down the tree by adding the class probabilities of the base classifier as new attributes to the dataset. Class probabilities indicate the ability of the base classifier to correctly classify a sample. At the next tree level, class probabilities added to the dataset are exploited by the base classifier as a weighting scheme to focus more on previously misclassified samples. Then, the overfitting generated by the boosted decision tree is mitigated by the use of bagging. Bagging provides variance reduction by creating multiple predictors from random sampling with replacement of the original dataset (see $D^1 \dots D^n$ in Figure 6.1). Trees are aggregated with a simple majority vote.

The hybrid ensemble method LCE allows to balance bias and variance while benefiting from the improved generalization ability of explicitly creating different training sets (bagging, boosting). Furthermore, LCE implicit divide-and-conquer method ensures that

classifiers are learned on different parts of the training data.

I present LCE pseudocode in Algorithm 1. A function (LCE_Tree) builds a tree and the second one (LCE) the forest of trees through bagging.

**Algorithm 1** LCE: Local Cascade Ensemble

**Require:** A dataset $D$, a set of classifiers $H$, maximum depth of a tree $max_depth$, number of trees $n_trees$

**function** LCE($D$, $H$, $n_trees$, $max_depth$)
  $F \leftarrow \emptyset$
  **for** each $i$ in $[1, n_trees]$ **do**
    $S \leftarrow$ A bootstrap sample from $D$
    $t \leftarrow$ LCE_Tree($S$, $H$, $max_depth$, 0)
    $F \leftarrow F \cup t$
  **return** $F$
**function** LCE_Tree($D$, $H$, $max_depth$, $depth$)
  **if** $max_depth$ or uniform class **then**
    **return** leaf
  **else**
    $D' \leftarrow$ Concatenate$(D, H_{depth}(D))$
    Split $D'$ on attribute maximizing Gini criterion
    $depth \leftarrow depth + 1$
    **for** $D'^{(j)} \in \mathcal{P}(D')$ **do**
      $Tree_j =$ LCE_Tree($D'^{(j)}$, $H$, $max_depth$, $depth$)
    **return** tree containing a decision node, storing classifier $H_{depth}(D)$ and descendant subtrees $Tree_j$

There are 2 stopping criteria during a tree building phase: when a node has an unique class or when the tree reaches the maximum depth. The range of tree depth is set from 0 to 3 in LCE instead of 0 to 5 in LC. This hyperparameter is used to control overfitting. The choice of low bias boosting base classifiers justifies the maximum depth adjustment to 3. In this study, the set of low bias base classifiers is limited to the best performing state-of-the-art boosting algorithm (extreme gradient boosting - XGB [Chen et al., 2016]).

In addition, two rules implemented in LC to reduce variance are removed: the maximum base classifier error rate and the minimum class representation in a node. The first rule requires the stopping of propagation down the tree to prevent overfitting if the base classifier, in a node, had an error rate below a certain threshold (0.5). LCE approach suggests a variance reduction through bagging, and not during a tree construction; so this rule is not kept. In order to restrict the attention to well populated classes, the second rule requires considering a class in a node if the number of examples belonging to this

class is greater than $N$ times (3) the number of attributes. The second rule is not kept for the same reason.

## 6.3 Evaluation

In this section, I introduce the datasets and methodology used for evaluating LCE approach.

### 6.3.1 Datasets

LCE is first evaluated on a public multivariate data classification benchmark and then on the real-world application dataset.

#### Public Datasets

In the experiments, LCE is benchmarked on the UCI datasets [Dua et al., 2017]. I have randomly selected one dataset per category available on the repository and obtained 26 UCI datasets. The categories are defined according to the dataset topic (life sciences, physical sciences, computer science/engineering, social sciences, business and game), the number of instances (less than 100, 100 to 1,000 and greater than 1,000) and the number of dimensions (less than 10, 10 to 100 and greater than 100). The characteristics of each dataset are presented in Table 6.1. There is no train/test split provided on the repository so a 3-fold cross-validation is performed.

#### Real-World Dataset

The dataset is offline. From 2014 to 2018, an experiment was conducted at the INRAE Méjusseaume dairy farm (48°06' N, 1°47' W, Brittany, France). This experiment enrolled 162 Holstein cows housed in free stalls representing 214 lactations. The first 3 years dataset (125 different cows) is used for cross validation and the last year is used for external validation (61 cows). In the external validation dataset, 24 cows are also in the cross validation dataset but within a new lactation and 37 are different.

Each cow was equipped with a collar-mounted activity meter (HeatPhone - Medria Technologies, Châteaubourg, France) and a temperature sensor in first stomach (Thermobolus - Medria Technologies, Châteaubourg, France). Based on its good performance

Table 6.1 – The UCI datasets. Abbreviations: CS - Computer Science, Dims - Dimensions.

| **Datasets** | **Type** | **Instances** | **Dims** | **Classes** | **LCE Parameters** | |
|---|---|---|---|---|---|---|
| | | | | | **Trees** | **Depth** |
| Absenteeism at Work | Business | 740 | 19 | 19 | 100 | 2 |
| Banknote Authentification | CS/Engineering | 1372 | 4 | 2 | 5 | 1 |
| Breast Cancer Coimbra | Life Sciences | 116 | 9 | 2 | 60 | 0 |
| CNAE-9 | Business | 1,080 | 856 | 9 | 20 | 2 |
| Congressional Voting | Social Sciences | 435 | 16 | 2 | 1 | 1 |
| Drug Consumption (quantified) | Social Sciences | 1,185 | 12 | 7 | 5 | 2 |
| Electrical Grid Stability | Physical Sciences | 10,000 | 13 | 2 | 40 | 1 |
| Gas Sensor | CS/Engineering | 58 | 432 | 4 | 100 | 0 |
| HTRU2 | Physical Sciences | 17,898 | 8 | 2 | 60 | 2 |
| Iris | Life Sciences | 150 | 4 | 3 | 20 | 2 |
| Leaf | CS/Engineering | 340 | 13 | 30 | 5 | 0 |
| LSVT Voice Rehabilitation | Life Sciences | 126 | 310 | 2 | 5 | 0 |
| Lung Cancer | Life Sciences | 32 | 56 | 3 | 60 | 1 |
| Mice Protein Expression | Life Sciences | 1,080 | 77 | 8 | 60 | 1 |
| Musk V1 | Physical Sciences | 476 | 166 | 2 | 5 | 2 |
| Musk V2 | Physical Sciences | 6,598 | 166 | 2 | 5 | 2 |
| p53 Mutants | Life Sciences | 31,159 | 5,408 | 2 | 10 | 1 |
| Page Blocks Classification | CS/Engineering | 5473 | 10 | 5 | 80 | 2 |
| Parkinson Disease | CS/Engineering | 756 | 753 | 2 | 5 | 2 |
| Semeion Handwritten Digit | CS/Engineering | 1,593 | 256 | 10 | 20 | 2 |
| Ultrasonic Flowmeter | CS/Engineering | 181 | 43 | 4 | 60 | 1 |
| User Knowledge Modeling | CS/Engineering | 403 | 5 | 5 | 40 | 2 |
| Wholesale Customers | Business | 440 | 6 | 2 | 40 | 0 |
| Wine | Physical Sciences | 178 | 13 | 3 | 100 | 0 |
| Wine Quality | Business | 1,599 | 11 | 6 | 100 | 2 |
| Yeast | Life Sciences | 1,484 | 8 | 10 | 80 | 2 |

compared to other solutions [Chanvallon et al., 2014] and its international market presence, we can hold that Medria estrus detection system is a reasonable basis of comparison. Medria estrus system has been designed to detect behavioral estrus. In the following sections, Medria is called the commercial solution (CS). The dataset consists of visual estrus alerts, Medria estrus alerts and Medria numeric variables with a 5-minute frequency (*rumination*, *ingestion*, *rest*, *standing up*, *over activity*, *other activity*, *temperature*, and *temperature corrected*). *Temperature corrected* takes into account the cooling effect of water ingestion by the cows. Concerning the visual estrus alerts, visual observation was conducted by farm staffs. Staff also checked the commercial solution alerts before inputting their visual records, thus these visual estrus alerts are shown as Visual&CS in the study. The preprocessing applied on the data collected is a 24hr aggregation (activity: sum, temperature: mean). It is assumed that the treatment operated by Medria on raw data to generate variables is stable during the experiment.

The novel approach addresses both estrus categories detection (behavioral and silent). Therefore, estrus are labeled by measuring the progesterone concentration in whole milk, the current reference for an exhaustive estrus identification. This time-effective and non-

invasive method for the cow induces commonly accepted errors (progesterone measurements, profiles analysis [Adriaens et al., 2018]). Silent estrus is defined by the absence of obvious behavioral sign and represent around 35% of all estrus [Kerbrat et al., 2004; Palmer et al., 2010; Ranasinghe et al., 2010]. An estrus is therefore marked as behavioral estrus when either a visual detection or a CS alert occurred. An estrus is considered silent when neither visual detection or a CS alert occurred. The proportion of silent estrus in the dataset is aligned with the literature: the cross-validation dataset is composed of 671 estruses with 37% of silent estrus and the external validation dataset is composed of 321 estruses with 44% of silent estrus.

Days preceding estrus are a valuable source of information for estrus detection, it is set as a hyperparameter. Every value in the range from 1 to 21 days, the length of a regular ovarian cycle, are tested. Past days of variables are added as feature columns.

**Feature Selection** Feature selection is performed in this study because of the sensitivity of the method chosen to explain the detection algorithm (SHAP) to high correlations among features. A subset selection is conducted on pairs of collinear features based on the Pearson correlation coefficient (threshold 0.8). One pair of features is above the threshold (0.9: *temperature corrected*, *temperature*). Since *temperature* is affected by the cooling effect of water ingestion, the variable *temperature corrected* is selected. From this point onwards, *temperature corrected* is named temperature. After this feature selection, no Pearson pairwise correlation in the case of the 21 past days dataset is above the threshold.

**Dataset Structure** A 5-fold cross validation is performed. The dataset split is presented in Table 6.2.

Table 6.2 – Dataset split. Abbreviations: Ext Val - External Validation

| | **Fold 1** | **Fold 2** | **Fold 3** | **Fold 4** | **Fold 5** | **All** | **Ext Val** |
|---|---|---|---|---|---|---|---|
| Estrus | 126 | 136 | 118 | 141 | 153 | 671 | 321 |
| Silent % | 33 | 40 | 24 | 40 | 46 | 37 | 44 |

The split has kept the same number of days in estrus in each fold (1,144 days). This choice is made to avoid overfitting on a particular animal. The impact of a split keeping the same number of animals per fold is discussed in the detection performance section. Moreover, there is no structural imbalance on silent estrus percentage across the folds.

### 6.3.2 Experimental Setting

I present in this section algorithms and methods used in the experiments.

#### Algorithms

I tested the hybrid ensemble LCE (explicit - implicit approaches) versus the initial implicit approach (LC) and the state-of-the-art algorithm for each explicit approach (bagging: random forest [Breiman, 2001], boosting: extreme gradient boosting [Chen et al., 2016]). As presented in section 2.1, the state-of-the-art multivariate data classifiers are also composed of k-Nearest Neighbors (kNN), regularized logistic regressions, Support Vector Machines (SVMs) and neural networks. In this work, the best-in-class for each category is chosen: kNN, elastic net [Zou et al., 2005], SVM [Cortes et al., 1995] and MultiLayer Perceptron (MLP) [Haykin, 2009] classifiers. Among categories of neural networks, given the size of the real-world dataset (18,000 samples), small MLPs are chosen.

- k-nearest neighbors - KNN: I use the implementation neighbors.KNeighborsClassifier in the scikit-learn package for Python [1]
- Elastic net - EN: I use the implementation linear model SGDClassifier in the scikit-learn package for Python[1]
- Support Vector Machine - SVM: I use the implementation svm.SVC in the scikit-learn package for Python[1]
- Random Forest - RF: I use the implementation ensemble.RandomForestClassifier in the scikit-learn package for Python[1]
- Extreme Gradient Boosting - XGB: I use the implementation in the xgboost package for Python [2]
- Local cascade - LC: algorithm has been reimplemented in Python 3.6 based on the description of the paper since no public version available.
- LCE: algorithm implemented in Python 3.6
- Multilayer Perceptron - MLP: I use the implementation available in the package Keras for Python [3] and limit the neural network architecture to 3 layers

1. https://scikit-learn.org/stable/
2. https://xgboost.readthedocs.io/en/latest/python/
3. https://keras.io/

Finally, concerning the explainability method, I use the SHAP implementation available in the Python package shap [4].

### Hyperparameters

Hyperparameters of classifiers are set by hyperopt, a sequential model-based optimization using a tree of Parzen estimators search algorithm [Bergstra et al., 2011]. Hyperopt chooses the next hyperparameters decision from the previous choices and a tree-based optimization algorithm. Tree of Parzen estimators meet or exceed grid search and random search performance for hyperparameters setting. I use the implementation available in the Python package hyperopt [5] and hyperas wrapper for Keras.

### Metrics

**Public Datasets** For each dataset, the classification accuracy is computed. Then, the average rank and the number of wins/ties are presented to compare the different classifiers on the same datasets. Finally, I present the critical difference diagram [Demšar, 2006], the statistical comparison of multiple classifiers on multiple datasets based on the non-parametric Friedman test, to show the overall performance of LCE. I use the implementation available in R package scmamp [6].

**Real-World Application** Optimization is undertaken to maximize F1-score. The choice of this metric is driven by 2 reasons. First, no assumption is made about the dairy management style; farmers can favor a higher estrus detection rate (higher recall) or fewer false alerts (higher precision) according to their needs. Second, there is a class imbalance (33% of estrus days) which renders irrelevant the accuracy metric. Following a 5-fold cross-validation 60/20/20 train/validation/test split, the best classifier is selected based on the highest F1-score on validation sets. Then, as recommended by [Dietterich, 1998], a $5 \times 2$ cross validation t-test is used for statistical significance of machine learning algorithms on one dataset.

The experiments use progesterone profiles as ground truth for exhaustive estrus identification. The levels of progesterone allow us to identify a time window of 3 days for

4. https://github.com/slundberg/shap
5. https://github.com/hyperopt/hyperopt
6. https://www.rdocumentation.org/packages/scmamp/versions/0.2.55/topics/plotCD

estrus with a duration of less than 24 hours, in the standard scheme. Adopting a conservative approach, it is decided to aggregate by the maximum of the daily predictions on estrus/anestrus period to calculate the classification performance. In addition, we can observe that for high thresholds (threshold $> 0.95$), classifiers performances are unstable with a significant decrease in estrus detection rate (recall below 70%). In addition, for low thresholds (threshold $< 0.1$), classifiers are equivalent to a random classifier. So, it is decided to adopt a F1-score calculation based on the average of F1-score on threshold range 0.1-0.95. This calculation does not modify the classifier selection results or the comparison result with the commercial solution. Nevertheless, it corresponds to the plausible range of calibration for dairy management and shows a detection performance closer to real conditions.

## 6.4 Results

In this section I first present the performance results of LCE on the public UCI datasets. Then, this section shows that LCE significantly outperforms a commercial reference in estrus detection while offering explanations to support farmers' decisions at both global (estrus/anestrus) and local (behavioral/silent) levels.

### 6.4.1 Public Datasets

Table 6.3 shows the classification results of the classifiers on the 26 UCI datasets. The best accuracy for each dataset is denoted in boldface. We observe that the top 3 classifiers are the ensemble methods: LCE obtains the best average rank (2.2), followed by RF in second position (rank: 2.4) and XGB in third position (rank: 2.7).

First of all, LCE obtains the best average rank with the first position on 38% of the datasets (10 wins/ties). Based on the categorization of the UCI datasets presented in section 6.3.1, we do not observe any influence of the number of instances, dimensions or classes on the performance of LCE relative to other classifiers. Nevertheless, LCE exhibits varying performances across the different dataset types. LCE shows its best performance on physical sciences (rank: 1.8, 19% of all datasets) and life sciences (rank: 1.9, 27% of all datasets) datasets while having its worst performance on computer science/engineering (rank: 2.4, 31% of all datasets), business (rank: 2.5, 15% of all datasets) and social sciences (rank: 3.5, 8% of all datasets) datasets. However, none of the classifiers shows a better

average rank than LCE on the business and social sciences datasets.

Then, we observe that the second ranked classifier RF obtains the same number of wins/ties as LCE. RF exhibits better performance than LCE on the computer science/engineering datasets (rank 1.8, 31% of all datasets) which represents half of RF wins/ties. 80% of the wins/ties of RF on the computer science/engineering category are obtained on small datasets (train size $< 1000$). We can infer that the bagging only (variance reduction) of RF can provide better generalization than LCE bagging-boosting combination on small datasets (wins/ties on small datasets - 54% of the datasets: LCE 6, RF 6). The third ranked classifier XGB gets 5 wins/ties. There is no influence of the different dataset categories on XGB wins/ties relative to LCE. Therefore, I conclude that LCE bagging and boosting combination to handle the bias-variance trade-off exhibits better generalization on average than the bagging only (RF) and boosting only (XGB) algorithms on these 26 UCI datasets.

Table 6.3 – Accuracy results on the UCI datasets.

| **Datasets** | **LCE** | **LC** | **XGB** | **RF** | **MLP** | **SVM** | **EN** |
|---|---|---|---|---|---|---|---|
| Absenteeism at Work | 42.7 | 27.6 | **44.2** | 42.0 | 28.3 | 28.7 | 31.7 |
| Banknote Authentification | 99.3 | 98.9 | 99.6 | 99.1 | 89.5 | **100.0** | 98.8 |
| Breast Cancer Coimbra | **71.4** | 65.5 | 64.6 | 64.5 | 48.4 | 55.2 | 57.5 |
| CNAE-9 | 86.2 | 51.0 | 84.1 | 91.6 | **95.6** | 30.4 | 92.2 |
| Congressional Voting | **97.0** | 94.0 | 96.8 | 96.6 | 79.5 | 87.8 | 91.7 |
| Drug Consumption (quantified) | 34.6 | 27.9 | 37.8 | 38.5 | **40.3** | **40.3** | 39.3 |
| Electrical Grid Stability | **100.0** | 99.9 | **100.0** | **100.0** | 88.5 | 79.3 | 96.8 |
| Gas Sensor | 74.4 | 63.3 | 74.6 | **89.6** | 78.7 | 61.5 | 70.4 |
| HTRU2 | **97.9** | 97.8 | **97.9** | 97.8 | 96.8 | 91.1 | 97.6 |
| Iris | **96.7** | 90.2 | **96.7** | **96.7** | 44.4 | 95.4 | 83.0 |
| Leaf | 52.5 | 48.7 | 61.6 | **71.7** | 8.5 | 35.2 | 56.0 |
| LSVT Voice Rehabilitation | **81.0** | 57.1 | 77.0 | **81.0** | 66.7 | 66.7 | 66.7 |
| Lung Cancer | 41.1 | 47.2 | 34.4 | 37.2 | 37.2 | 36.7 | **52.8** |
| Mice Protein Expression | **56.7** | 40.1 | 43.1 | 53.1 | 13.9 | 14.4 | 42.9 |
| Musk V1 | 73.3 | 63.5 | **76.1** | 72.5 | 57.4 | 56.5 | 72.3 |
| Musk V2 | 78.8 | 74.5 | 78.4 | 77.5 | 84.6 | **84.7** | 76.3 |
| p53 Mutants | 96.6 | 82.7 | 94.8 | 95.6 | **99.5** | 86.5 | 81.7 |
| Page Blocks Classification | **97.3** | 90.8 | 96.5 | 96.0 | 90.4 | 91.1 | 94.2 |
| Parkinson Disease | 82.7 | 74.2 | 82.5 | **83.2** | 58.2 | 74.6 | 41.4 |
| Semeion Handwritten Digit | 90.3 | 43.2 | 90.0 | **92.2** | 92.1 | 36.4 | 75.8 |
| Ultrasonic Flowmeter | **59.0** | 40.2 | 45.2 | 49.6 | 24.4 | 29.8 | 45.1 |
| User Knowledge Modeling | **85.6** | 80.4 | **85.6** | **85.6** | 29.8 | 80.4 | 74.6 |
| Wholesale Customers | 91.8 | 88.6 | **92.5** | 91.6 | 77.0 | 67.7 | 83.0 |
| Wine | 92.8 | **96.1** | 91.1 | 92.8 | 35.4 | 42.7 | 75.4 |
| Wine Quality | 55.5 | 49.2 | 54.5 | **56.9** | 42.1 | 41.9 | 45.9 |
| Yeast | 57.1 | 35.3 | 59.2 | **59.6** | 28.9 | 58.9 | 53.2 |
| Average Rank | 2.2 | 5.0 | 2.7 | 2.4 | 5.3 | 5.2 | 4.7 |
| Wins/Ties | 10 | 1 | 7 | 10 | 3 | 3 | 1 |

Next, LC algorithm gets the fifth rank with one win/tie. There is no particular influence of the different dataset categories on LC performance. So, the outperformance of LCE compared to LC on the 26 UCI datasets confirms the better generalization ability

of a hybrid (explicit and implicit) versus an implicit only approach. The comparison in Table 6.4 aims to underline the superior performance of LCE compared to LC on the UCI datasets. In order to be comparable, the low bias base classifier in LC is XGB. The depth of a tree is set to 1 for LCE and LC. The results correspond to the average accuracy on test sets with the corresponding standard error. Results show a comparable accuracy variability of LCE compared to LC when the number of trees is set to 1 (standard error of 4.6% versus 4.8%). However, LCE on 1 tree exhibits a higher accuracy than LC (71.8% versus 65.9%). Additionally, through bagging, we observe LCE variability reduction as well as an increase of accuracy (71.8±4.6 with 1 tree versus 74.9±4.1 with 60 trees versus 65.9±4.8 with LC). Therefore, this comparison affirms the superiority of the explicit bias-variance trade-off approach compared to the implicit approach of LC on the UCI datasets.

Concerning the other classifiers, EN obtains only one win/tie but gets a better rank on average than SVM (3 wins/ties) and MLP (3 wins/ties).

Table 6.4 – Average accuracy score of LCE versus LC on test sets of the UCI datasets with the corresponding standard error.

| **Trees** | **1** | **5** | **10** | **20** | **40** | **60** | **80** |
|---|---|---|---|---|---|---|---|
| **LCE** | 71.8 ±4.6 | 74.1 ±4.3 | 73.6 ±4.4 | 72.8 ±4.4 | 73.2 ±4.5 | 74.9 ±4.1 | 73.9 ±4.2 |
| **LC** | | | | $65.9 \pm 4.8$ | | | |

Finally, I analyze a statistical test to evaluate the performance of LCE compared to other classifiers. Figure 6.2 presents the critical difference plot with alpha equals to 0.05 from results shown in Table 6.3. The values correspond to the average rank and the classifiers linked by a bar do not have a statistically significant difference. The plot confirms the top 3 ranking as presented before (LCE: 1, RF: 2, XGB: 3), without showing a statistically significant difference between each other. We also observe that the ensemble methods accuracies are statistically different from other classifiers.

### 6.4.2 Real-World Application

This section is structured into two parts: performance and explainability. The detection performance part compares LCE to other detection methods (classifiers, commercial solution) and evaluates the relevance of deploying 2 sensors. Then, the key drivers (variables impact, temporal interactions) behind the estrus detection alerts are identified at global

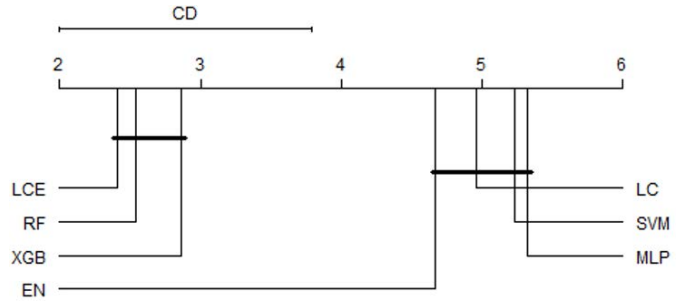



Figure 6.2 – Critical difference plot of the classifiers on the UCI datasets with alpha equals to 0.05.

(estrus/anestrus) and local (behavioral/silent) levels based on algorithm explainability (SHAP) and an approach to reduce the solution mistrust is proposed.

### Performance

Before presenting the detection performance, I discuss the composition of the dataset and how it could bias the results.

We can observe in the dataset that more estrus occur in the first lactation (59%, external validation: 60% of all estrus), and that cows in the first lactation (primiparous) experience a higher proportion of silent estrus compared to cows in higher lactations - multiparous (41% vs 31%, external validation: 50% vs 34%). I do not consider that it could bias the study. In the literature, the effect of parity is unclear on the estrus detection performance. Some authors reported greater estrus intensity for older cows [De Silva et al., 1981; Gwazdauskas et al., 1983] while others reported a greater activity for primiparous [Peralta et al., 2005; Van Vliet et al., 1996; Yániz et al., 2006] or no difference [Lopez et al., 2004; Van Eerdenburg et al., 2002].

Then, classification results on test sets are presented in Figure 6.3. The best classifier on validation sets is LCE with the following hyperparameters: 3 past days, depth equals to 1 and 70 trees. We do not observe an overfit of LCE, the performance observed on test sets (F1-score: 68.9) is stable compared to the one of the validation sets (F1-score: 68.1) and the one of the external validation set (F1-score: 71.4).

Furthermore, the performance of LCE responds to the objective of an increase in performance in both estrus detection rate and fewer false alerts compared to the commercial solution (CS). At the same precision, LCE recall is constantly higher than commercial solution recall. At a precision of 78%, the precision rate of the commercial solution in this

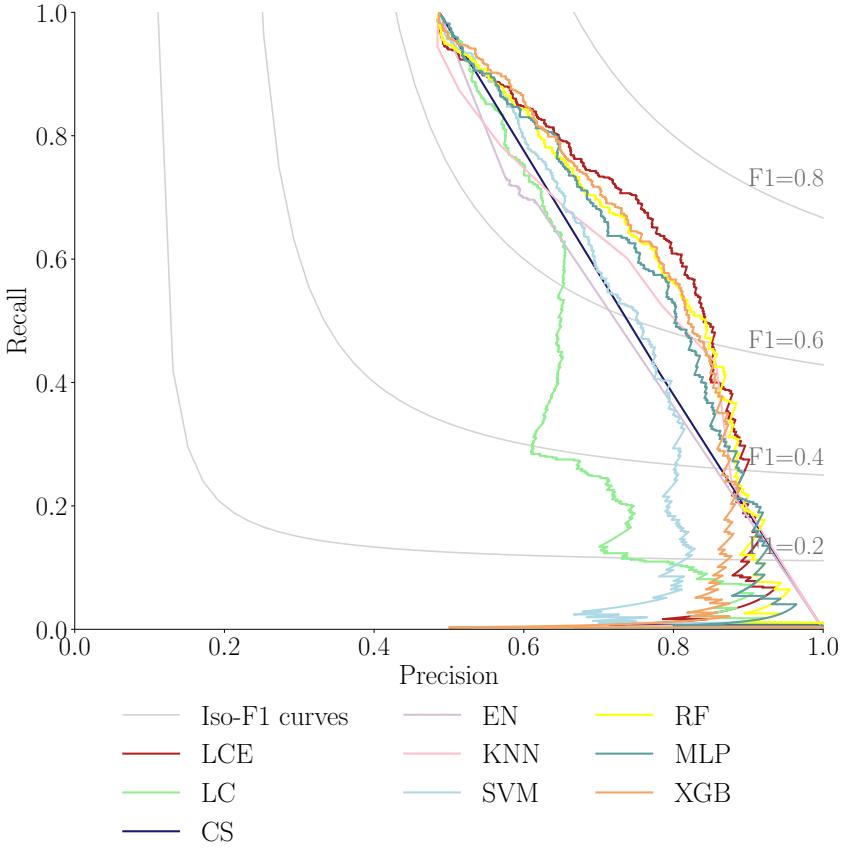



Figure 6.3 – Precision-recall curves on test sets of the classifiers versus the commercial solution.

study, LCE detects 22% more estrus.

**Comparative Analysis** I compare the error rate correlation of LCE to those of other detection methods. This comparison allows us to:

- gain insights into the shortcomings of the commercial solution and LCE detections;
- identify limitations of the approach for deployment.

A low correlation indicates that classifiers err in different regions of the instance space. Table 6.5 presents Pearson correlations of LCE prediction errors with other detection methods (classifiers and commercial solution) on test sets. In order to be comparable, the threshold of each classifier is set with the same precision as the commercial solution (78%).

Table 6.5 – Pearson pairwise correlations of LCE prediction errors with other detection methods on test sets.

| KNN | EN | SVM | MLP | RF | XGB | LC | CS |
|---|---|---|---|---|---|---|---|
| 0.61 | 0.19 | 0.57 | 0.69 | 0.73 | 0.8 | 0.41 | 0.37 |

First, the commercial solution shows an intrinsic different behavior from that of LCE (correlation: 0.37). This low correlation is mainly explained by the null performance of the commercial solution on silent estrus detection across the herd. On 67% of the cows, composed of a slighlty higher proportion of silent estrus compared to average (40% versus 37%), predictions correlation of the commercial solution with LCE is $0.21 \pm 0.03$.

Next, the low correlation between LCE and LC (0.41) confirms the value added by the explicit bias-variance trade-off of the LCE approach. This low correlation is explained by the low recall (11%) of the LC for a precision of 78%. The stable decision surface drawn by naive Bayes at the root of the LC decision tree substantially limits the range of performance of the algorithm on the dataset (recall drops with a precision higher than 66%). We observe this performance drop for precision above 66% in Figure 6.3.

Finally, the classifier with the closest behavior to LCE is XGB (0.8). However, the correlation difference remains substantial and is explained by some divergence among few cows. The divergence, an error rate correlation below 0.6, concerns 12% of the cows comprising a proportion of silent estrus aligned with average (35%). Therefore, the bias-variance approach enhances XGB performance on standard cases (cows with 35% of silent estrus). Nevertheless, there is a poor performance of LCE on 11% of the cows exhibiting a high proportion of silent estrus (F1-score $< 55\%$, silent estrus proportion: 54%). Silent

estrus are not equally distributed among cows. In the dataset, 16% of the cows represent 40% of the silent estrus. LCE performance per cow is exposed to the animal estrus type proportion. It is confirmed by the LCE performance drop when assessed on the activity and temperature dataset generated by a stratified 5-fold on animals (66.3±3.4). LCE performance per cow variability according to the animal estrus type proportion is a limitation of the solution for deployment; meanwhile it is also a driver for detection improvement. It would be interesting to further investigate by incorporating additional animal individual features.

**One or Two Sensors?** In order to answer this question, I compare the detection performance on test sets of LCE on the temperature, the activity and both variables. I also compare LCE detection results to the commercial solution and visual method.

First, the results confirm the potential of data science techniques for automatic estrus detection versus visual detection as concluded by [Dolecheck et al., 2015]. We observe that LCE for both behavioral and silent estrus detection, trained on activity and temperature data, manifests significantly better performance (F1-score and lower variability) than Visual&CS ($68.9 \pm 2.4$ versus $60.4 \pm 4.6$, $P < 0.05$). The Visual&CS performance is aligned with the state-of-the-art [Peralta et al., 2005]; the detection rate is slightly below 50% (47%).

Second, there is a better performance (higher F1-score and lower variability) with the algorithm trained on activity and temperature than activity or temperature alone (68.9±2.4 versus 67.0±3.0 versus 55.9±2.3). The performance difference is only significant when compared to the algorithm trained using the temperature. We can infer that, in the conditions of the experiment, only activity sensor should be deployed: the performance is not significantly lower than that trained with two sensors (activity and temperature).

Nonetheless, temperature information cannot be excluded. There is a markedly lower variability of the algorithm based on temperature across folds which allows the algorithm based on activity and temperature to reduce its variability. It means that the algorithm based on temperature is consistent on different data. It implies a possible higher discriminative and generalizing power. It would be interesting to further study the potential of temperature data for estrus detection with a broader data heterogeneity (cows breed, environment). The next step would consist of a partnership with an automatic detection solution provider to have access to a more diverse dataset.

### Explainability

In this section, I firstly present the relative impact of variables in LCE predictions and their temporal interactions. Then, I propose an approach to give insights on estrus detection to the farmers based on these elements.

Figure 6.4 shows the average impact of each variable on algorithm predictions for estrus and anestrus by decreasing order (global level).

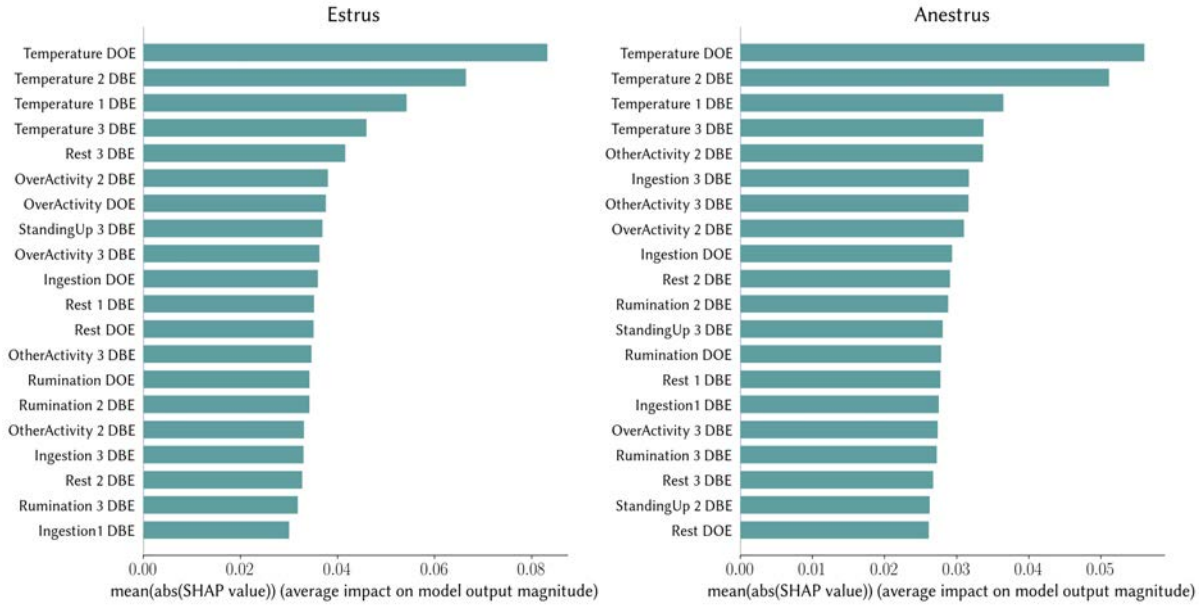



Figure 6.4 – Average impact of the attributes on algorithm predictions for estrus and anestrus. Abbreviations: DBE - Day Before Estrus, DOE - Day Of Estrus.

These results confirm the discriminative power of the temperature and its potential for improving estrus detection performance. The variable with the strongest impact to algorithm predictions is the temperature on the day of estrus for both estrus and anestrus classes.

Next, we can observe that the ranking of all activity variables are different with a significant rank change between estrus and anestrus. Therefore, the relative impact of each activity variable in LCE predictions differs between estrus and anestrus. Over activity on the day of estrus, a typical characteristic of most estrus (65%), appears as the third most impactful variable after temperature for estrus and does not appear on the top 20 of variables for anestrus.

By taking the same impact ranking approach locally for behavioral versus silent estrus, we also observe a significant change on the ranking of activity variables (75% of rank change). Rumination 2 days before estrus is a key variable in silent estrus detection. It is the third most impactful activity variable for silent estrus and appears at the 19th position for behavioral estrus.

Finally, temporal relations among variables differ between behavioral and silent estrus. SHAP interaction values reveal that algorithm predictions are more impacted by activity variables further to the day of estrus for silent estrus than behavioral estrus. For example, the variable of highest interaction with rumination on the day of estrus is the rest 3 days before estrus for silent estrus versus the rest 2 days before estrus for behavioral estrus. This observation holds true for over activity, standing up and ingestion (two third of activity variables).

Therefore, in order to support LCE estrus alerts and ease solution adoption, I propose an approach based on LCE explainability (activity sensor only). First, communicate to the farmer the relatedness of the estrus detection to historical cases through a confidence indicator and the amplitude of differences in the 3 most impactful activity variables (rest 3 days before estrus, over activity 2 days before estrus and over activity on the day of estrus). The confidence indicator corresponds to the weighted average of absolute SHAP values differences by the ranking of impact variables for estrus from the reference presented above (global level). Second, in case of estrus, inform the farmer about the type of estrus (behavioral/silent) with a confidence level and which temporal interactions are satisfied (local level). The information about the type of estrus aims to reassure farmers when they are not able to verify the estrus alert by visual behavioral signs, therefore reduce potential mistrust. Confidence level is calculated like the previous one but using ranking of variables impact of silent estrus as a reference. In addition, temporal interactions are communicated in decreasing order of variable impact.

## 6.5 Peformance-Explainability Analysis

This section introduces the new ensemble method LCE with SHAP post-hoc model-agnostic explainability method into the analytical framework of the thesis (part II). The different aspects of the LCE framework are summarized in the Table 6.6 and can be visualized in Figure 6.5.

The study of this chapter firstly shows that LCE outperforms the state-of-the-art

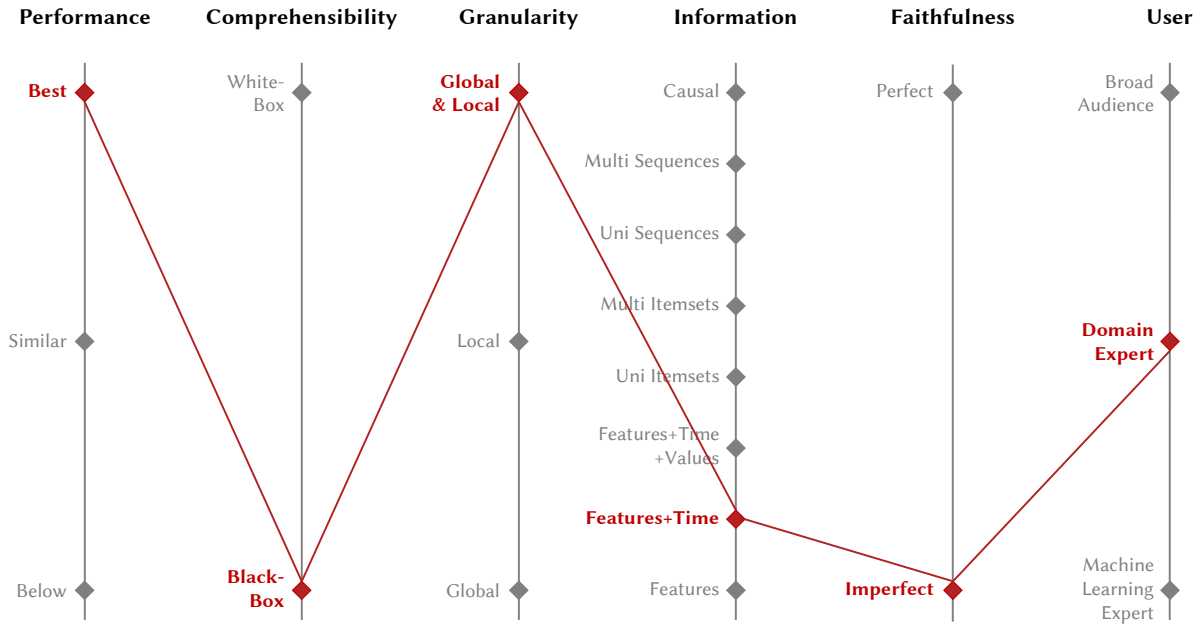



Figure 6.5 – Parallel coordinates plot of LCE with SHAP post-hoc model-agnostic explainability method on the dairy resource monitoring application. Performance evaluation method: 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM and XGB.

classifiers on the public UCI datasets. Then, it exhibits the significant performance improvement of LCE on estrus detection on a real-world dataset compared to a commercial reference, a result driven by silent estrus detection. Therefore, in the framework presented in part II, following a 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset, the performance of LCE is better than the state-of-the-art (Performance: *Best*).

Table 6.6 – Performance-explainability results of LCE with SHAP post-hoc model-agnostic explainability method and DMSEEW. Abbreviations: ML - Machine Learning.

| | **DMSEEW** | **LCE + SHAP** |
|---|---|---|
| Performance | Best[1] | Best[2 3] |
| Comprehensibility | Black-Box | Black-Box |
| Granularity | Local | Both Global & Local |
| Information | Features | Features+Time |
| Faithfulness | Perfect | Imperfect |
| User | ML Expert | Domain Expert |

[1] 3-fold cross-validation and an arithmetic mean of the accuracies on the Earthquake Early Warning Dataset. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.
[2] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM and XGB.
[3] 3-fold cross-validation and an arithmetic mean of the accuracies on the UCI datasets. Models evaluated in the benchmark: EN, LC, LCE, MLP, RF, SVM and XGB.

Concerning the explainability, as the DMSEEW stacking ensemble method presented in the previous chapter, the hybrid ensemble method is a "black-box" model (Comprehensibility: *Black-Box*). However, LCE with SHAP post-hoc model-agnostic explainability method is more informative than DMSEEW explainability by design (Information: *Features+Time*, DMSEEW: *Features*) and accessible to a wider audience (User: *Domain Expert*, DMSEEW: *Machine Learning Expert*). SHAP provides the features relative importance on predictions at local level (silent versus behavioral estrus), which can be averaged for each class to obtain it globally (estrus versus anestrus). Moreover, the time aspect is managed in this chapter by setting the different timestamps as column variables of the dataset. Therefore, SHAP provides the features and the timestamps relative importance on predictions at both local and global level (Granularity: *Both Global & Local*, DMSEEW: *Local*). This kind of information allows the communication to the end-user of the relatedness of the predictions to historical cases and the possibility of visually verifying the estrus (behavioral versus silent). These explanations are accessible to a domain expert (e.g. a farmer) who is able to analyze the confidence indicator with the amplitude of differences in the 3 most impactful variables. Nevertheless, unlike DMSEEW explainability

by design, the faithfulness of the SHAP explanations is imperfect (Faithfulness: *Imperfect*, DMSEEW: *Perfect*). SHAP approximates the original model using an explainable model. The explanation model is a linear model, an additive feature attribution method that uses simplified inputs (conditional expectations) assuming feature independence.

The next chapter shows how to extend LCE hybrid ensemble method for MTS classification to combine performance with faithful and more informative explanations.

**Summary**

- LCE is a new hybrid ensemble method which significantly improves the detection of estrus on a real-world dataset compared to a commercial reference. The superior performance of LCE also holds on the public UCI datasets.
- LCE with SHAP post-hoc model-agnostic explainability method provides informative explanations accessible to domain experts at all granularity levels. The explanations provide to the user the features and the timestamps relative importance on predictions.
- However, the faithfulness of explanations is imperfect.

Chapter 7

# An Explainable by Design Ensemble Method for Multivariate Time Series Classification

## Contents


**7.1 Introduction . . . . . 101**
7.1.1 Contributions . . . . . 102
**7.2 XEM . . . . . 103**
7.2.1 Dataset Transformation . . . . . 103
7.2.2 Classification . . . . . 104
7.2.3 Explainability by Design . . . . . 105
7.2.4 Properties . . . . . 106
7.2.5 Time Complexity . . . . . 107
7.2.6 Implementation . . . . . 108
**7.3 Evaluation . . . . . 108**
7.3.1 Public Datasets . . . . . 108
7.3.2 Algorithms . . . . . 111
7.3.3 Hyperparameters . . . . . 112
7.3.4 Metrics . . . . . 112
**7.4 Results . . . . . 112**
7.4.1 Performance . . . . . 112
7.4.2 Explainability . . . . . 118
7.4.3 Discussion . . . . . 123
**7.5 Performance-Explainability Analysis . . . . . 124**


The third machine learning method of the thesis is an extension of LCE to MTS classification studied on public MTS datasets. The method is the result of a collaboration between researchers from the PEGASE ("Physiology, Environment, Genetics for Animals

and rearing systems") unit at the French National Institute for Agriculture, Food and Environment (INRAE) and machine learning researchers from the LACODAM ("Large Scale Collaborative Data Mining") team at Inria, France. It is available on ArXiv [Fauvel et al., 2020b].

## 7.1 Introduction

In chapter 6, driven by the evaluation setting of the dairy resource monitoring application (potential time series length equals to 2), the MTS classification task has been approached as a traditional multivariate data classification and LCE has been developed. As a hyperparameter of the evaluation, the time series length has finally been set to 4 (see section 6.4.2), which allows the use of most state-of-the-art MTS classifiers. Thus, the new methods in chapter 8 (pattern-based method) and chapter 9 (deep learning method) will approach the dairy resource monitoring application as a MTS classification problem. However, a short time series length of 4 on the dairy resource monitoring application still prevents the use of interval-based MTS classifiers like WEASEL+MUSE or the LCE extension to MTS classification proposed in this chapter. Therefore, the study of this chapter limits the evaluation of the new MTS classifier to the public MTS datasets benchmark (UEA archive [Bagnall et al., 2018]).

Ensemble methods are the current state-of-the-art classifiers for traditional multivariate data classification (Random Forest [Breiman, 2001], Extreme Gradient Boosting [Chen et al., 2016]) and univariate time series classification (HIVE-COTE [Lines et al., 2016]), whereas there is no ensemble method among the state-of-the-art MTS classifiers. The study in chapter 6 has shown that LCE outperforms the state-of-the-art multivariate data classifiers on the public UCI datasets. Consequently, this chapter proposes to extend LCE to MTS classification and benchmark it against the state-of-the-art MTS classifiers on the public UEA datasets.

Concerning explainability, LCE relies on post-hoc model-agnostic methods. The main line of work in post-hoc model-agnostic explainability methods approximates the decision surface of a model using an explainable one. However, the explanations from the surrogate models cannot be perfectly faithful with respect to the original model [Rudin, 2019], which is a prerequisite for numerous applications. Thus, the extension of LCE to MTS classification, XEM, integrates explainability by design in order to ensure perfectly faithful explanations.

In addition, XEM has some interesting properties out of its explainability by design. In particular, XEM is robust with varying MTS input data quality (different MTS length, missing data and noise), which often arises in continuous data collection systems.

None of the state-of-the-art MTS classifiers ($DTW_D$, $DTW_I$, WEASEL + MUSE, MLSTM-FCN) reconciles performance and perfectly faithful explainability. The best performing MTS classifiers, WEASEL+MUSE and MLSTM-FCN, cannot provide perfectly faithful explanations as they can only rely on post-hoc model-agnostic explainability methods. Similarity-based methods provide by design faithful explainability by revealing the distance between two MTS for each timestamp. However, they are often less accurate than other MTS classification methods. Moreover, none of the state-of-the-art MTS classifiers handles the three varying data quality aspects (different TS length, missing data, noise). Table 7.1 presents an overview of the challenges addressed by state-of-the-art MTS classifiers and how is positioned the new ensemble method XEM.

Table 7.1 – The state-of-the-art MTS classifiers - overview.

| | **Similarity Based** | | **Deep Learning** | **Feature Based** | **Ensemble** |
|---|---|---|---|---|---|
| | **ED** | **DTW** | **MLSTM FCN** | **WEASEL+ MUSE** | **XEM** |
| **Input** | | | | | |
| *Varying TS Length* | | ✓ | ✓ | ✓ | ✓ |
| *Missing Data* | | | | | ✓ |
| *Noise* | | | | ✓ | ✓ |

### 7.1.1 Contributions

This chapter presents in detail and thoroughly examines the behavior of XEM. The study will:

- Present XEM, a new eXplainable by design Ensemble method for MTS classification;
- Show that XEM outperforms the state-of-the-art MTS classifiers on the UEA datasets [Bagnall et al., 2018];
- Illustrate the explainability by design feature of XEM, which provides the time window used to classify the whole MTS;
- Show that XEM manifests robust performance when faced with challenges arising from continuous data collection (different MTS length, missing data and noise);

## 7.2 XEM

A subset of the MTS can be characteristic of the event we aim to predict and can be adequate for the prediction. Therefore, I propose a new MTS classifier, XEM, that generates features allowing the direct identification of the MTS time window that is important for prediction. These features correspond to the confidence levels of the classifier LCE on each MTS subsequence of a predefined length. The subsequence where the classifier is the most confident is used for classification and provided to the end-user as faithful explanation to support the MTS prediction. As in [Baydogan et al., 2014], I have chosen a tabular classifier because it fulfills two needs simultaneously: first, the need to handle the relationship between the variables; second, the need to handle really small time series according to the predefined time window length of interest. Most MTS classifiers fail to meet the second need. More specifically, I have adopted LCE tabular classifier, the best performing tabular classifier on the public UCI datasets (see section 6.4.1). As a method aggregating features which are the output of multiple predictors, XEM can be categorized as an ensemble method. In the following sections, I first present how dividing the time series into time windows is used to help XEM classify MTS based on their discriminative part and then how it provides explainability.

### 7.2.1 Dataset Transformation

In order to classify a MTS, the whole series is not always needed: only some parts may be relevant for the classification task, while others can be considered as noise and may even degrade classification performance. Therefore, I introduce a parameter to XEM defining the time window size, i.e. the size of the subsequence of the MTS expected to be sufficient to assign a label to the MTS. Suitable methods to set this hyperparameter are discussed in section 7.2.2. XEM is trained on subsequences of MTS, which require a transformation of the dataset. This transformation is presented in Figure 7.1. Using a sliding window, all subsequences corresponding to the time window size (MTS length-window size+1 subsequences) are generated. The time aspect is managed by setting the different timestamps as column dimensions. Each subsequence is considered as a new sample, labeled as the original MTS. For example in Figure 7.1, 4 subsequences (samples) are generated from the first MTS, composed of 2 timestamps (time window size) with 2 dimensions each (4 attributes columns). The 4 subsequences are calculated as: 5 (MTS length) - 2 (time window size) + 1. I present in the next section how to compute the

classification performance with the transformed dataset and how this configuration allows a better model explainability.

**Input Dataset**: $nT^*(d+3)$

| ID | Timestamp | Attribute1 | Attribute2 | Class |
|---|---|---|---|---|
| 1 | 1 | 10 | 12 | 1 |
| 1 | 2 | 6 | 14 | 1 |
| 1 | 3 | 20 | 7 | 1 |
| 1 | 4 | 24 | 25 | 1 |
| 1 | 5 | 30 | 32 | 1 |
| . | | | | . |
| . | | | | . |
| n | 1 | 3 | 8 | 3 |
| n | 2 | 9 | 12 | 3 |
| n | 3 | 7 | 18 | 3 |
| n | 4 | 19 | 23 | 3 |
| n | 5 | 12 | 31 | 3 |

**Dataset Transformed**: $n(T\text{-}win_size+1)^*(dwin_size+2)$

| ID | Attribute1 | Attribute2 | Attribute1 $^{-1}$ | Attribute2 $^{-1}$ | Class |
|---|---|---|---|---|---|
| 1 | 6 | 14 | 10 | 12 | 1 |
| 1 | 20 | 7 | 6 | 14 | 1 |
| 1 | 24 | 25 | 20 | 7 | 1 |
| 1 | 30 | 32 | 24 | 25 | 1 |
| . | | | | | . |
| . | | | | | . |
| n | 9 | 12 | 3 | 8 | 3 |
| n | 7 | 18 | 9 | 12 | 3 |
| n | 19 | 23 | 7 | 18 | 3 |
| n | 12 | 31 | 19 | 23 | 3 |

Figure 7.1 – The dataset transformation (from MTS to a traditional multivariate flat dataset). Abbreviations: ID - MTS identifier, Timestamp - one element of the time series, AttributeX - value produced by the sensor X at each timestamp, d - number of dimensions, n - number of MTS, T - time series length, win_size - time window size. In this example: T=5, d=2 and win_size=2.

### 7.2.2 Classification

As seen in the previous section, XEM is trained on subsequences of MTS which sizes are controlled by the time window size parameter. Then, XEM assigns class probabilities to all subsequences of the MTS. For example, on the upper part of the Figure 7.2, XEM assigns class probabilities for each of the 4 subsequences of a MTS. Finally, XEM determines the class of a MTS based on the subsequence on which it is the most confident. For each MTS, the maximum class probability over the different subsequences is selected to determine the whole MTS classification output. For example, on the lower part of Figure 7.2, we

can observe that XEM assigns the class 1 to the first MTS (ID=1) based on the highest class probability (0.95 versus 0.6 and 0.7) obtained with the classification of the third subsequence of the MTS. In the case where XEM is the most confident for a subsequence of a MTS which is not discriminative, it means that the time window size value is not suited for the classification problem and it would lead to poor classification accuracy of XEM on the training set. A time window size better suited for the classification problem would lead to better accuracy on the training set and would therefore be selected. In the evaluation, without having prior knowledge on the time window size which would suit the classification tasks, the time window size is set by hyperparameter optimization (see section 7.3.3). The transformation presented and the performance evaluation procedure allow any classifier to perform MTS classification. Therefore, I compare in section 7.4.1 the performance of XEM to the best two state-of-the-art classifiers applying the same transformation as LCE and to the state-of-the-art MTS classifiers.

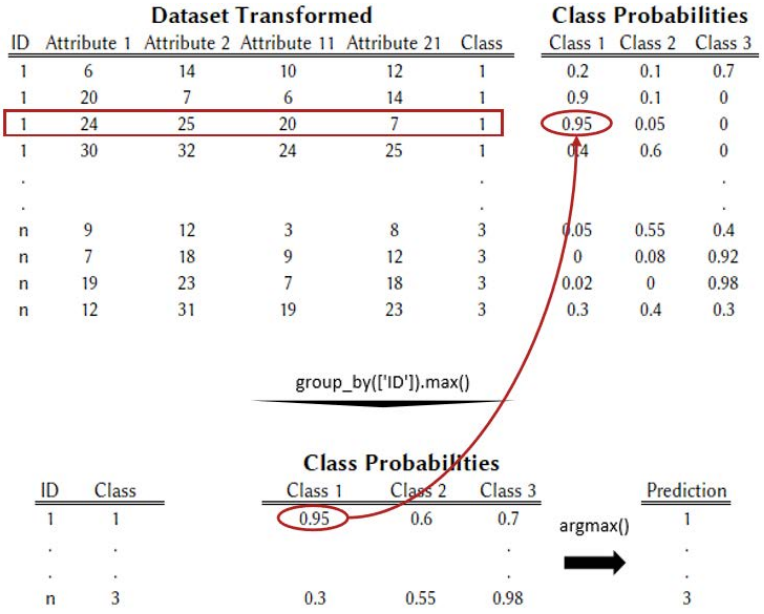



Figure 7.2 – XEM prediction computation on the example from Figure 7.1 and illustration of the explainability on the first MTS (ID=1).

### 7.2.3 Explainability by Design

XEM provides local explainability by design through the identification of the time window used to classify a MTS. Following the dataset transformation performed (see

section 7.2.1), we obtain the class probabilities for every subsequences from XEM. As mentioned, a subset of the MTS can be characteristic of the event we aim to predict and can be adequate for the prediction. Therefore, the prediction for a MTS is based on the subsequence that has the highest class probability - the subsequence on which XEM is the most confident. The explainability of XEM is illustrated with the previous section example in Figure 7.2. We observe that for the first MTS (ID=1), after performing a grouping by MTS ID and taking the maximum, class 1 has the highest probability (0.95). We can trace back to the subsequence from which XEM is predicting this class probability (third subsequence), and show it to the user. This subsequence can help the user to understand why the MTS classifier attributed a particular label to the whole MTS (explainability). The explainability property of XEM is further illustrated in section 7.4.2 on a synthetic and two UEA datasets.

### 7.2.4 Properties

In addition to its explainability by design, XEM has other interesting properties: phase invariance, interplay of dimensions, different MTS length compatibility, missing data management, noise robustness and scalability.

- *Phase Invariance*: XEM is not sensitive to the position of the discriminative subsequence in the MTS due to the selection of the subsequence which has the highest class probability to classify the whole MTS. This property improves the generalization ability of the algorithm: in the possible cases when the sequences of events in a MTS change, the classification result is not modified. For example, the classification result would be the same if the discriminative subsequence appears at the beginning or at the end of the MTS;
- *Interplay of Dimensions*: XEM exploits the relationships among the dimensions through the use of boosting-based classifier as base classifier. It allows XEM to exploit complex interactions among the dimensions at different timestamps to perform classification;
- *Different MTS Length Compatibility*: XEM handles it in two different ways. If a MTS length is inferior to the maximum length of the MTS in a dataset multiplied by the window size selected, XEM uses padding of 0 values. Otherwise, no padding is necessary, less samples are generated per MTS but the performance evaluation procedure presented in 7.2.2 remains valid;

- *Missing Data Management*: XEM naturally handles missing data through its tree-based learning [Breiman et al., 1984]. Similar to extreme gradient boosting [Chen et al., 2016], XEM excludes missing values for the split and uses block propagation. During a node split, block propagation sends all samples with missing data to the side minimizing the error. This property is evaluated in the experiments in section 7.4.1;
- *Noise Robustness*: the bagging component of XEM provides noise robustness through variance reduction by creating multiple predictors from random sampling with replacement of the original dataset. This property is discussed in the experiments in section 7.4.1;
- *Scalability*: as a tree-based ensemble method, XEM is scalable. Its time complexity is detailed in section 7.2.5.

Most of the properties of XEM are coming from LCE. The properties shared between LCE and XEM are interplay of dimensions, missing data management, noise robustness and scalability.

### 7.2.5 Time Complexity

LCE time complexity is determined by the time complexity of multiple decision trees learning and extreme gradient boosting. The time complexity of building a single tree is $O(n(wd)D_t)$, where $n$ is the number of samples after the dataset transformation, $w$ is the time window size, $d$ is the number of dimensions and $D_t$ is the maximum depth of the tree. So the time complexity of creating multiple decision trees with bagging is $O(N_t n(wd)D_t)$, where $N_t$ is the number of trees. Extreme gradient boosting has a time complexity of $O(N_b D_b \|x\|_0 \log(n))$ where $N_b$ is the number of trees, $D_b$ is the maximum depth of the trees and $\|x\|_0$ is the number of non-missing entries in the data. Therefore, LCE has a time complexity of $O(N_t nwdD_t 2^{D_t} N_b D_b \|x\|_0 \log(n))$, where $2^{D_t}$ represents the maximum number of nodes in a binary tree. Table 7.2 shows the time complexity of LCE in comparison with the state-of-the-art ensemble methods of the benchmark (RF, XGB, LC).

XEM time complexity is the same as LCE plus the dataset transformation which is linear in the number of samples.

Table 7.2 – Time complexities of the ensemble methods. Abbreviations: $d$ - number of dimensions, $d'$ - number of dimensions in RF subset of dimensions, $D$ - maximum depth of a tree, $n$ - number of samples, $N$ - number of trees, $T_{Base}$ - time complexity of a base classifier, $\|x\|_0$ - number of non-missing entries in the data.

| Algorithm | Time Complexity |
|---|---|
| **RF** | $O(Nnd'D)$ |
| **XGB** | $O(N\log(n)\|x\|_0 D)$ |
| **LC** | $O(ndD2^D T_{Base})$ |
| **LCE** | $O(NndD2^D T_{Base})$ |

### 7.2.6 Implementation

XEM pseudocode is presented in Algorithm 2. XEM implementation is the same as LCE plus the dataset tranformation. A function (XEM_Tree) builds a tree and the second one (XEM) builds the forest of trees through bagging, after having transformed the dataset. There are 2 stopping criteria during a tree building phase which are the same as LCE: when a node has an unique class or when the tree reaches the maximum depth. We set the range of tree depth from 0 to 2 in XEM as in LCE. This hyperparameter is used to control overfitting. Low bias boosting-based classifier as base classifier justifies the maximum depth of 2. The set of low bias base classifiers is limited to the best performing state-of-the-art boosting algorithm (extreme gradient boosting - XGB [Chen et al., 2016]).

## 7.3 Evaluation

In this section, I present the evaluation method. As explained in section 7.2.2, the dataset transformation performed and the performance calculation to extend LCE for MTS classification can be done for any classifier. Therefore, XEM performance is compared to the best two classifiers from LCE evaluation (RF and XGB, see section 6.4.1) applying the same transformation as LCE and to the state-of-the-art MTS classifiers.

### 7.3.1 Public Datasets

XEM is benchmarked on the 30 currently available UEA MTS datasets [Bagnall et al., 2018]. For each dataset, the train/test split provided in the archive is kept. The characteristics of each dataset are presented in Table 7.3.

**Algorithm 2** XEM

**Require:** A dataset $D$, a set of classifiers $H$, time window size $win_size$, maximum depth of a tree $max_depth$, number of trees $n_trees$

1: **function** XEM($D$, $H$, $win_size$, $n_trees$, $max_depth$)
2: $D' \leftarrow$ Dataset_Transformation($D$, $win_size$)
3: $F \leftarrow \emptyset$
4: **for** each $i$ in $[1, n_trees]$ **do**
5: $S \leftarrow$ A bootstrap sample from $D'$
6: $t \leftarrow$ XEM_Tree($S$, $H$, $max_depth$, 0)
7: $F \leftarrow F \cup t$
8: **return** $F$
9: **function** XEM_Tree($D$, $H$, $max_depth$, $depth$)
10: **if** $max_depth$ or uniform class **then**
11: **return** leaf
12: **else**
13: $D' \leftarrow$ Concatenate$(D, H_{depth}(D))$
14: Split $D'$ on attribute maximizing Gini criterion
15: $depth \leftarrow depth + 1$
16: **for** $D'^{(j)} \in \mathcal{P}(D')$ **do**
17: $Tree_j =$ XEM_Tree($D'^{(j)}$, $H$, $max_depth$, $depth$)
18: **return** tree containing one decision node, storing classifier $H_{depth}(D)$ and descendant subtrees $Tree_j$

Table 7.3 – The UEA MTS datasets. Abbreviations: AS - Audio Spectra, C - Classes, D - Dimensions, ECG - Electrocardiogram, EEG - Electroencephalogram, HAR - Human Activity Recognition, L - Length, MEG - Magnetoencephalography.

| **Datasets** | **Type** | **Train** | **Test** | **L** | **D** | **C** | **XEM Parameters** | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | **Time Window (%)** | **Trees** | **Depth** |
| Articulary Word Recognition | Motion | 275 | 300 | 144 | 9 | 25 | 40 | 5 | 1 |
| Atrial Fibrilation | ECG | 15 | 15 | 640 | 2 | 3 | 20 | 1 | 0 |
| Basic Motions | HAR | 40 | 40 | 100 | 6 | 4 | 20 | 1 | 0 |
| Character Trajectories | Motion | 1,422 | 1,436 | 182 | 3 | 20 | 80 | 10 | 2 |
| Cricket | HAR | 108 | 72 | 1,197 | 6 | 12 | 40 | 20 | 0 |
| Duck Duck Geese | AS | 60 | 40 | 270 | 1,345 | 5 | 100 | 20 | 0 |
| Eigen Worms | Motion | 128 | 131 | 17,984 | 6 | 5 | 100 | 20 | 1 |
| Epilepsy | HAR | 137 | 138 | 206 | 3 | 4 | 20 | 1 | 1 |
| Ering | HAR | 30 | 30 | 65 | 4 | 6 | 20 | 1 | 2 |
| Ethanol Concentration | Other | 261 | 263 | 1751 | 3 | 4 | 20 | 1 | 2 |
| Face Detection | EEG/MEG | 5,890 | 3,524 | 62 | 144 | 2 | 100 | 5 | 2 |
| Finger Movements | EEG/MEG | 316 | 100 | 50 | 28 | 2 | 60 | 5 | 2 |
| Hand Movement Direction | EEG/MEG | 320 | 147 | 400 | 10 | 4 | 80 | 20 | 2 |
| Handwriting | HAR | 150 | 850 | 152 | 3 | 26 | 20 | 10 | 2 |
| Heartbeat | AS | 204 | 205 | 405 | 61 | 2 | 80 | 10 | 0 |
| Insect Wingbeat | AS | 30,000 | 20,000 | 200 | 30 | 10 | 100 | 10 | 1 |
| Japanese Vowels | AS | 270 | 370 | 29 | 12 | 9 | 40 | 5 | 1 |
| Libras | HAR | 180 | 180 | 45 | 2 | 15 | 40 | 60 | 1 |
| LSST | Other | 2,459 | 2,466 | 36 | 6 | 14 | 60 | 10 | 2 |
| Motor Imagery | EEG/MEG | 278 | 100 | 3,000 | 64 | 2 | 100 | 20 | 1 |
| NATOPS | HAR | 180 | 180 | 51 | 24 | 6 | 40 | 10 | 0 |
| PenDigits | Motion | 7,494 | 3,498 | 8 | 2 | 10 | 80 | 80 | 2 |
| PEMSF | Other | 267 | 173 | 144 | 963 | 7 | 100 | 20 | 1 |
| Phoneme | AS | 3315 | 3353 | 217 | 11 | 39 | 80 | 1 | 2 |
| Racket Sports | HAR | 151 | 152 | 30 | 6 | 4 | 60 | 20 | 0 |
| Self Regulation SCP1 | EEG/MEG | 268 | 293 | 896 | 6 | 2 | 100 | 5 | 2 |
| Self Regulation SCP2 | EEG/MEG | 200 | 180 | 1152 | 7 | 2 | 100 | 20 | 2 |
| Spoken Arabic Digits | AS | 6,599 | 2,199 | 93 | 13 | 10 | 80 | 10 | 1 |
| Stand Walk Jump | ECG | 12 | 15 | 2,500 | 4 | 3 | 20 | 1 | 1 |
| U Wave Gesture Library | HAR | 120 | 320 | 315 | 3 | 8 | 60 | 1 | 0 |

### 7.3.2 Algorithms

As presented in section 2.2, MTS classifiers are composed of 3 categories: similarity-based, feature-based and deep learning methods. In this work, the best-in-class for each category is chosen: $DTW_D$, $DTW_I$, WEASEL+MUSE and MLSTM-FCN classifiers. XEM is also compared to the best two classifiers from LCE evaluation (RF and XGB, see section 6.4.1) applying the same transformation as LCE. Thus, the MTS classifiers evaluated are:

- $DTW_D$ with and without normalization: the one nearest neighbor classifier with DTW distance based on multi-dimensional points instead of treating each dimension separately. I report the results published in the UEA archive [Bagnall et al., 2018];
- $DTW_I$ with and without normalization: the one nearest neighbor classifier based on the sum of DTW distance for each dimension. I report the results published in the UEA archive [Bagnall et al., 2018];
- ED with and without normalization: the one nearest neighbor classifier with Euclidean distance. I report the results published in the UEA archive [Bagnall et al., 2018];
- MLSTM-FCN [Karim et al., 2019]: I use the implementation available [1] and run it with the setting recommended by the authors in the paper (128-256-128 filters, 250 training epochs, a dropout of 0.8 and a batch size of 128);
- RFM: Random Forest for Multivariate time series classification. I use the public implementation [2] with the transformation presented in section 7.2.1;
- WEASEL+MUSE [Schäfer et al., 2017]: I use the implementation available [3] and run it with the setting recommended by the authors in the paper (SFA word lengths l in [2,4,6], windows length in [4:max(MTS length)], chi=2, bias=1, p=0.1, c=5 and a solver equals to L2R LR DUAL);
- XGBM: Extreme Gradient Boosting for Multivariate time series classification. I use the implementation in the xgboost package for Python [4] with the transformation presented in section 7.2.1.

1. https://github.com/houshd/MLSTM-FCN
2. https://scikit-learn.org/stable/
3. https://github.com/patrickzib/SFA
4. https://xgboost.readthedocs.io/en/latest/python/

### 7.3.3 Hyperparameters

The ranges of XEM hyperparameters are the same as LCE: number of trees ($n_trees$) $[1, 5, 10, 20, 40, 60, 80, 100]$ and maximum depth ($max_depth$) $[0, 1, 2]$. As explained in section 7.2, the time window size hyperparameter ($win_size$) is added to XEM. This parameter is expressed as a percentage of the total size of the MTS and the range of time window size percentages is $[20\%, 40\%, 60\%, 80\%, 100\%]$.

The hyperparameters are set by hyperopt, a sequential model-based optimization using a tree of Parzen estimators search algorithm [Bergstra et al., 2011]. Hyperopt chooses the next hyperparameters decision from both the previous choices and a tree-based optimization algorithm. Tree of Parzen estimators meet or exceed grid search and random search performance for hyperparameters setting. I use the implementation available in the Python package hyperopt [5].

### 7.3.4 Metrics

For each dataset, the classification accuracy is computed. Then, I present the average rank and the number of wins/ties to compare the different classifiers on the same datasets. Finally, I present the critical difference diagram [Demšar, 2006], the statistical comparison of multiple classifiers on multiple datasets based on the non-parametric Friedman test, to show the overall performance of XEM. I use the implementation available in R package scmamp [6].

## 7.4 Results

### 7.4.1 Performance

The classification results of the 11 MTS classifiers are presented in Table 7.4. A blank in the table indicates that the approach ran out of memory or the accuracy is not reported [Bagnall et al., 2018]. The best accuracy for each dataset is denoted in boldface. We observe that XEM obtains the best average rank (3.0), followed by RFM in second position (rank: 3.7) and MLSTM-FCN in third position (rank: 3.9).

XEM gets the first position in one third of the datasets. Using the categorization of the

5. https://github.com/hyperopt/hyperopt
6. https://www.rdocumentation.org/packages/scmamp/versions/0.2.55/topics/plotCD

Table 7.4 – Accuracy results on the UEA MTS datasets. Abbreviations: MF - MLSTM-FCN, n - Normalized, RM - RFM, WM - WEASEL+MUSE, XG - XGBM, XM - XEM.

| **Datasets** | **XM** | **XG** | **RM** | **MF** | **WM** | **ED** | **DTW$_I$** | **DTW$_D$** | **ED (n)** | **DTW$_I$ (n)** | **DTW$_D$ (n)** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Articulary Word Recognition | **99.3** | 99.0 | 99.0 | 98.6 | **99.3** | 97.0 | 98.0 | 98.7 | 97.0 | 98.0 | 98.7 |
| Atrial Fibrilation | **46.7** | 40.0 | 33.3 | 20.0 | 26.7 | 26.7 | 26.7 | 20.0 | 26.7 | 26.7 | 22.0 |
| Basic Motions | **100.0** | **100.0** | **100.0** | **100.0** | **100.0** | 67.5 | **100.0** | 97.5 | 67.6 | **100.0** | 97.5 |
| Character Trajectories | 97.9 | 98.3 | 98.5 | **99.3** | 99.0 | 96.4 | 96.9 | 99.0 | 96.4 | 96.9 | 98.9 |
| Cricket | 98.6 | 97.2 | 98.6 | 98.6 | 98.6 | 94.4 | 98.6 | **100.0** | 94.4 | 98.6 | **100.0** |
| Duck Duck Geese | 37.5 | 40.0 | 40.0 | **67.5** | 57.5 | 27.5 | 55.0 | 60.0 | 27.5 | 55.0 | 60.0 |
| Eigen Worms | 52.7 | 55.0 | **100.0** | 80.9 | 89.0 | 55.0 | 60.3 | 61.8 | 54.9 |  | 61.8 |
| Epilepsy | 98.6 | 97.8 | 98.6 | 96.4 | **99.3** | 66.7 | 97.8 | 96.4 | 66.6 | 97.8 | 96.4 |
| Ering | **20.0** | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 |
| Ethanol Concentration | 37.2 | 42.2 | **43.3** | 27.4 | 31.6 | 29.3 | 30.4 | 32.3 | 29.3 | 30.4 | 32.3 |
| Face Detection | 61.4 | **62.9** | 61.4 | 55.5 | 54.5 | 51.9 | 51.3 | 52.9 | 51.9 |  | 52.9 |
| Finger Movements | 59.0 | 53.0 | 56.0 | **61.0** | 54.0 | 55.0 | 52.0 | 53.0 | 55.0 | 52.0 | 53.0 |
| Hand Movement Direction | **64.9** | 54.1 | 50.0 | 37.8 | 37.8 | 27.9 | 30.6 | 23.1 | 27.8 | 30.6 | 23.1 |
| Handwriting | 28.7 | 26.7 | 26.7 | 54.7 | 53.1 | 37.1 | 50.9 | **60.7** | 20.0 | 31.6 | 28.6 |
| Heartbeat | 76.1 | 69.3 | **80.0** | 71.4 | 72.7 | 62.0 | 65.9 | 71.7 | 61.9 | 65.8 | 71.7 |
| Insect Wingbeat | 22.8 | **23.7** | 22.4 | 10.5 |  | 12.8 |  | 11.5 | 12.8 |  |  |
| Japanese Vowels | 97.8 | 96.8 | 97.0 | **99.2** | 97.8 | 92.4 | 95.9 | 94.9 | 92.4 | 95.9 | 94.9 |
| Libras | 77.2 | 76.7 | 78.3 | **92.2** | 89.4 | 83.3 | 89.4 | 87.2 | 83.3 | 89.4 | 87.0 |
| LSST | **65.2** | 63.3 | 61.2 | 64.6 | 62.8 | 45.6 | 57.5 | 55.1 | 45.6 | 57.5 | 55.1 |
| Motor Imagery | **60.0** | 46.0 | 55.0 | 53.0 | 50.0 | 51.0 | 39.0 | 50.0 | 51.0 |  | 50.0 |
| NATOPS | 91.6 | 90.0 | 91.1 | **96.1** | 88.3 | 85.0 | 85.0 | 88.3 | 85.0 | 85.0 | 88.3 |
| PenDigits | 97.7 | 95.1 | 95.1 | **98.7** | 96.9 | 97.3 | 93.9 | 97.7 | 97.3 | 93.9 | 97.7 |
| PEMSF | 94.2 | **98.3** | **98.3** | 65.3 |  | 70.5 | 73.4 | 71.1 | 70.5 | 73.4 | 71.1 |
| Phoneme | **28.8** | 18.7 | 22.2 | 27.5 | 19.0 | 10.4 | 15.1 | 15.1 | 10.4 | 15.1 | 15.1 |
| Racket Sports | **94.1** | 92.8 | 92.1 | 88.2 | 91.4 | 86.4 | 84.2 | 80.3 | 86.8 | 84.2 | 80.3 |
| Self Regulation SCP1 | 83.9 | 82.9 | 82.6 | **86.7** | 74.4 | 77.1 | 76.5 | 77.5 | 77.1 | 76.5 | 77.5 |
| Self Regulation SCP2 | **55.0** | 48.3 | 47.8 | 52.2 | 52.2 | 48.3 | 53.3 | 53.9 | 48.3 | 53.3 | 53.9 |
| Spoken Arabic Digits | 97.3 | 97.0 | 96.8 | **99.4** | 98.2 | 96.7 | 96.0 | 96.3 | 96.7 | 95.9 | 96.3 |
| Stand Walk Jump | 40.0 | 33.3 | **46.7** | **46.7** | 33.3 | 20.0 | 33.3 | 20.0 | 20.0 | 33.3 | 20.0 |
| U Wave Gesture Library | 89.7 | 89.4 | 90.0 | 85.7 | **90.3** | 88.1 | 86.9 | **90.3** | 88.1 | 86.8 | **90.3** |
| Average Rank | 3.0 | 4.8 | 3.7 | 3.9 | 4.1 | 7.5 | 6.3 | 5.3 | 7.9 | 6.7 | 5.7 |
| Wins/Ties | 10 | 4 | 6 | 11 | 4 | 0 | 1 | 3 | 0 | 1 | 2 |

datasets published in the archive website [7], we do not see any influence from the different train set sizes, MTS lengths, dimensions and number of classes on XEM performance relative to the other classifiers on the UEA datasets. Nonetheless, XEM exhibits weaker performance on average on human activity recognition (rank: 3.6, 30% of all datasets) and motion classification (rank: 5.0, 13% of all datasets) datasets.

Then, we observe that the better generalization of LCE bagging-boosting combination compared to bagging only (RF) and boosting only (XGB) is also valid on the MTS datasets (average rank: XEM 3.0, RFM 3.7, XGBM 4.8). The adaptation of ensemble methods to the MTS datasets (see section 7.2.1) is well performing: the three ensemble methods obtain the highest number of wins/ties (ensemble methods for MTS: 17 - 57% of all datasets, MLSTM-FCN: 11 - 37% of all datasets, WEASEL+MUSE: 4 - 13% of all datasets). The 6 wins/ties of RFM are obtained on small datasets (train size $< 500$). As seen in section 6.4.1, we can infer that the bagging only (variance reduction) of RFM can provide better generalization than XEM bagging-boosting combination on small datasets (wins/ties on small datasets - 77% of the datasets: XEM 8, RFM 6). On the time window sizes used, we observe that the choice of XEM time window is a trade-off between its bagging and boosting components. XEM and XGBM use the same time window size on 70% of the datasets. When the time window size is different, XEM obtains a better accuracy than XGBM on 90% of the cases. Moreover, XEM employs the same time window size as RFM on half of the UEA datasets. On the other half of the datasets, RFM adopts a slightly bigger time window size than XEM. RFM uses a bigger time window in 75% of the time with an average time window difference of 29% between XEM and RFM. The different choice of XEM time window size leads to a better accuracy on 75% of the cases compared to RFM. These observations prove that XEM bias-variance trade-off can refine the time window size of boosting only and bagging only to obtain a better generalization ability on average.

Specifically, with regard to the hyperparameter *win_size* of XEM, Figure 7.3 shows the average relative drop in performance across the datasets when using the other time window sizes than the one used in the best configuration given in Table 7.3. In order to evaluate the relative impact with respect to the range of performance, I have defined four categories of datasets: datasets with XEM original accuracy$<$ 50%, datasets with 50% $\leq$ accuracy $<$ 90% and datasets with accuracy $\geq$ 90%. First, as expected, we can observe that the average relative impact of using suboptimal time window sizes is higher when

7. http://www.timeseriesclassification.com/dataset.php

XEM level of performance is low (average relative drop in accuracy: 15.1% when XEM accuracy $< 50\%$ versus 4.5% when XEM accuracy $\geq 90\%$). Then, the average relative drop in accuracy when using suboptimal time window sizes is not negligible but remains limited in all the cases. This drop is below 16% on average on the category where XEM has the lowest level of accuracy (15.1% $\pm$ 5.3%) and below 10% on average across all the datasets (9.9% $\pm$ 1.8%).

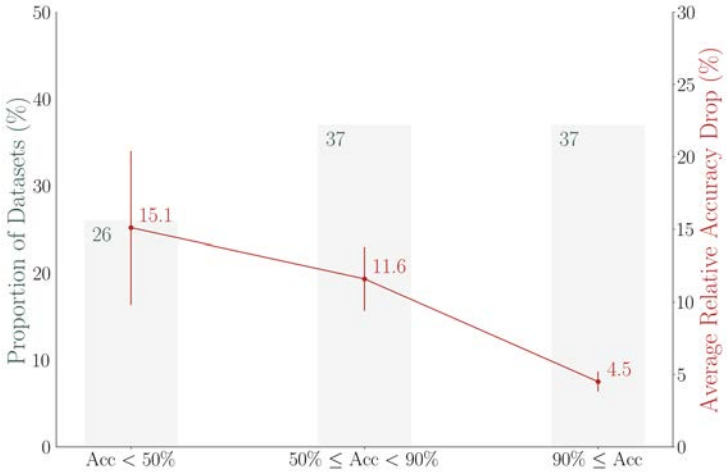



Figure 7.3 – XEM average relative accuracy drop across the UEA datasets when using other time window sizes than the one used in the best configuration given in Table 7.3. The performance drop is presented across three categories of datasets, defined according to XEM levels of accuracy shown in Table 7.4. Abbreviation: Acc - Accuracy.

Concerning state-of-the-art MTS classifiers, we observe a performance difference between the third (MLSTM-FCN) and fourth (WEASEL+MUSE) classifiers on datasets sizes. MLSTM-FCN outperforms WEASEL+MUSE (rank: 2.6 versus 4.6 for WEASEL + MUSE) on the largest datasets (train size $\geq 500$, 23% of all datasets) whereas WEASEL + MUSE slightly outperforms MLSTM-FCN (rank 3.6 versus 3.8 for MLSTM-FCN) on the smallest datasets (train size $< 500$, 77% of all datasets). XEM shows the same performance as MLSTM-FCN on the largest datasets (rank 2.6) while outperforming WEASEL+MUSE on the smallest datasets (rank: 3.1 versus 3.6 for WEASEL+MUSE). Therefore, XEM is better than state-of-the-art MTS classifiers on both small and large UEA datasets. Last, similarity-based methods obtain the lowest wins/ties counts. Euclidean distance is never in the first position on the UEA datasets. The wins/ties of DTW ($DTW_D$ normalized: 2, $DTW_D$: 3) stem from their outperformance on human activity recognition datasets.

Next, a statistical test is performed to evaluate the performance of XEM compared

to other MTS classifiers. I present in Figure 7.4 the critical difference plot with alpha equals to 0.05 from results shown in Table 7.4. The values correspond to the average rank and the classifiers linked by a bar do not have a statistically significant difference. The plot confirms the top 3 ranking as presented before (XEM: 1, RFM: 2, MLSTM-FCN: 3), without showing a statistically significant difference between each other. As seen in the evaluation on the UCI datasets, the plot also confirms that there is no statistically significant difference between the ensemble methods XEM/RFM/XGM on the MTS datasets. We can notice that XEM is the only classifier with a significant performance difference compared to $\text{DTW}_D$ normalized.

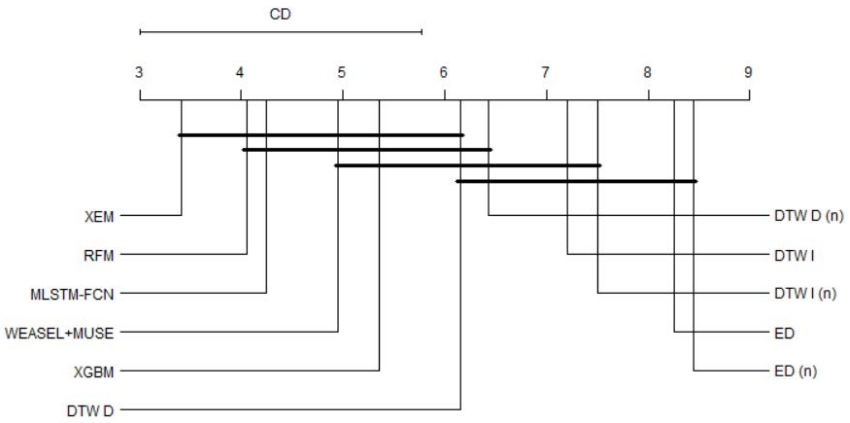



Figure 7.4 – Critical difference plot of the MTS classifiers on the UEA datasets with alpha equals to 0.05.

### Effect of Missing Data

None of the state-of-the-art MTS classifiers handles missing data. Missing data are interpolated, which adds a parameter to the problem. XEM naturally handles missing data through its tree-based learning [Breiman et al., 1984]. Similar to extreme gradient boosting [Chen et al., 2016], XEM excludes missing values for the split and uses block propagation. Block propagation sends all samples with missing data to the side minimizing the error.

I present in this section an experiment to illustrate the performance of XEM in the case of missing data. Three datasets have been selected from the most representing type of the UEA datasets (human activity recognition, 30% of the datasets); it is also a type on which XEM does not obtain the best performance comparing to the other classifiers (rank: 3.6). The three datasets are chosen according to the performance of XEM to show the evolution

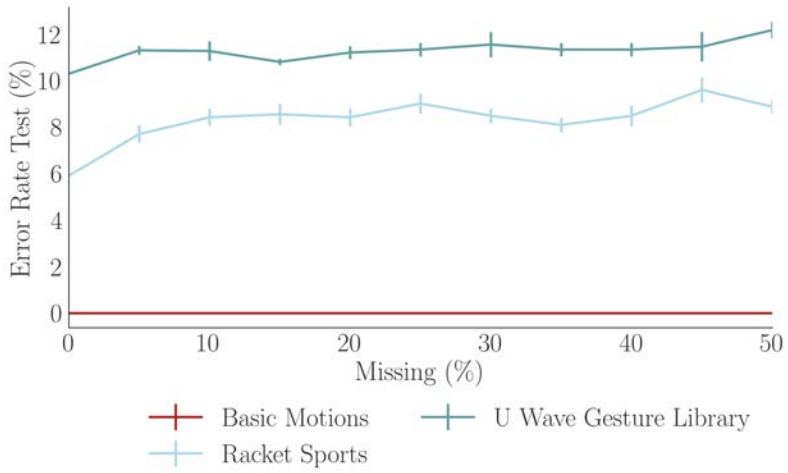



Figure 7.5 – Evolution of XEM error rates with standard errors according to the proportion of missing values on three Human Activity Recognition datasets.

of accuracies according to different starting points: Basic Motions (XEM accuracy: 100%, no error), Racket Sports (94.1%, ]0,10] percent of error) and U Wave Gesture Library (89.7%, ]10,100] percent of error). Then, I randomly removed an increasing proportion of the values for each time series ($[5\%, 10\%, ..., 50\%]$) of the datasets before transformation (see section 7.2.1). The error rates on test sets over 10 replications are presented in Figure 7.5.

First, we observe that missing data does not have an effect on XEM performance (100% accuracy) on the dataset Basic Motions. On the other two datasets, the error rates of XEM increase progressively with the proportion of missing data. The error rate induced by missing data never exceeds 5% on these 2 datasets when half the data is missing (accuracy difference from 0% to 50% missing data: Racket Sports +3.7% and U Wave Gesture Library +1.9%). Finally, XEM performance is stable: the error rates remain roughly the same across the 10 replications on all proportions of missing values (mean of standard error across Racket Sports/U Wave Gesture Library: 0.34%).

### Effect of Gaussian Noise

In this section, I evaluate the robustness of XEM to Gaussian noise compared to the second and third ranked MTS classifiers according to the number of wins/ties. Therefore, the performance of XEM is compared to RFM and MLSTM-FCN, with RFM proven to be robust to noise based on bagging [Breiman, 1996].

Following the same logic as the section on missing values, an experiment on the same three datasets is performed. These three datasets are from the most representing type of the UEA datasets (human activity recognition, 30% of the datasets) and from different XEM accuracy categories: Basic Motions (XEM accuracy: 100%, no error), Racket Sports (94.1%, ]0,10] percent of error) and U Wave Gesture Library (89.7%, ]10,100] percent of error). Then, after z-normalization of these datasets on each dimension (standard deviation of 1), an increasing Gaussian noise is added with a standard deviation of 0 to 1 to each dimension, which is equivalent to noise levels of 0% to 100%. The average error rates with standard errors on these three datasets are presented in Figure 7.6.

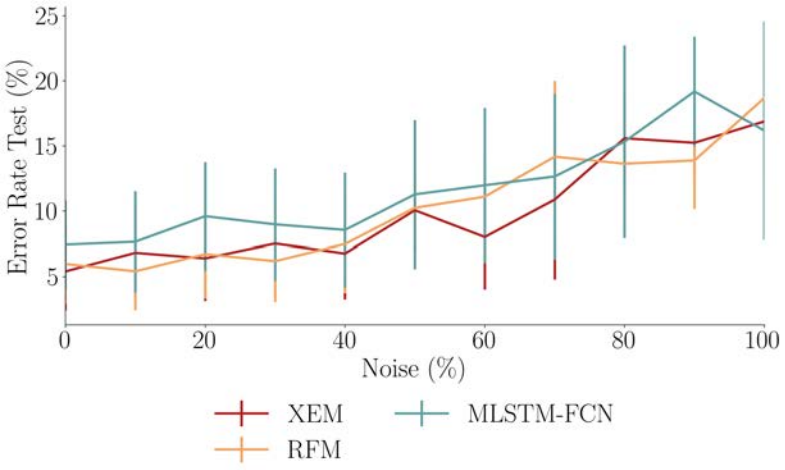



Figure 7.6 – Evolution of the top three MTS classifiers average error rates with standard errors on three Human Activity Recognition datasets (Basic Motions, Racket Sports, U Wave Gesture Library) according to the level of noise.

We observe that XEM fully exploits its bagging component and is as robust to noise as RFM. XEM shows lower error rates than RFM on 60% of the noise levels, without having a greater variability across the datasets (average standard error: XEM 3.7% versus RFM 3.5%). Moreover, XEM is more robust to noise than MLSTM-FCN. XEM exhibits lower error rates than MLTSM-FCN on 80% of the noise levels with a lower variability across the datasets (average standard error: XEM 3.7% versus MLSTM-FCN 5.3%).

### 7.4.2 Explainability

XEM provides explainability through the identification of the time window used to classify the whole MTS. There is no quantitative approach to assess a model explainabil-

ity. Therefore, a qualitative approach is adopted to analyze XEM explainability. First, I illustrate the explainability of XEM on a synthetic dataset, then I show which windows have been used on the UEA datasets of section 7.4.1 and I illustrate it on two UEA datasets (Atrial Fibrilation and Racket Sports).

**Synthetic Dataset** First of all, I show that XEM uses and identifies the expected time window to perform the classification on a MTS synthetic dataset. I design a dataset composed of 20 MTS (50%/50% train/test split) with a length of 100, 2 dimensions and 2 balanced classes. The difference between the 10 MTS belonging to the negative class and the one belonging to the positive class stems from a 20% time window of the MTS. As illustrated in Figure 7.7, negative class MTS are sine waves and positive class MTS are sine waves with a square signal on 20% of the dimension 1 (see timestamps between 60 and 80).

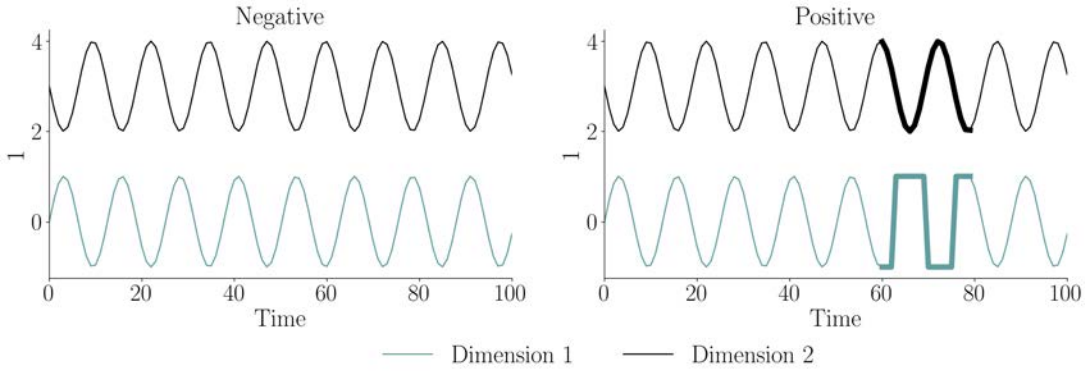



Figure 7.7 – The two MTS types of the synthetic dataset.

The classification results show that XEM with a time window size parameter set to 20% is enough to correctly classify the 10 MTS of the test set (accuracy: 100% - *n_trees*: 10, *max_depth*: 1). Moreover, the classification results for the positive class MTS are based on the 20% time window with a square signal on dimension 1. We observe that the maximum class probability for the MTS of positive class is 100% and this probability is reached for samples on the range [62,100] (maximum class probability on the range [0,61]: 92.6%). This range is the expected range. As explained in section 7.2.1, all the samples of the dataset obtained with a 20% sliding window have a piece of the square signal for the timestamps in the range [62,100], which is the information sufficient to correctly classify the MTS in the positive class. Therefore, by taking all the samples of the dataset with the maximum class probability, XEM allows the identification of the full parts of the MTS

which are characteristic of a class (e.g. the square signal on 20% of the dimension 1 in Figure 7.7).

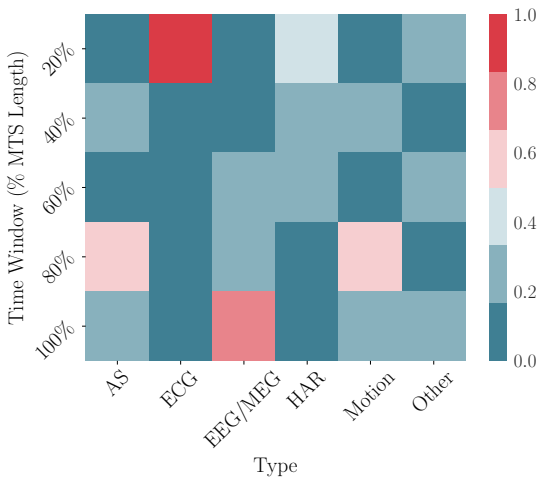



Figure 7.8 – Heatmap of the proportion of the time window size percentages used by XEM per UEA classification type.

**Time Window Size Percentages on the UEA** I then present the XEM explainability results on the UEA datasets. I begin with illustrating in Figure 7.8 the distribution of the time window size percentage used by XEM on the UEA archive per dataset type. We observe that XEM has a tendency to use particular time window size percentages per dataset type. Most of audio spectra, EEG/MEG and motion datasets have been classified on a time window size $> 60\%$ of the MTS lengths. Meanwhile, most ECG and human activity recognition datasets have been classified on a time window size $\leq 60\%$ of the MTS lengths. Therefore, we can induce that the information provided by the whole MTS is useful to discriminate between the different classes on the audio spectra, EEG/MEG and motion datasets. Concerning the ECG and human activity recognition datasets, we can infer that the discriminative information is located in a particular part of the MTS.

**Atrial Fibrilation Dataset** For example, XEM obtains its best performance on the two ECG datasets using a time window size of 20%. Therefore, we can assume that the information necessary for XEM to classify the MTS in ECG datasets are really condensed compared to the entire MTS available. I illustrate it in Figure 7.9 by highlighting the 20% time window of the first MTS sample per class in the Atrial Fibrilation test set to gain insights on XEM classification result. Atrial Fibrilation dataset is composed of two channels ECG on a 5 second period (128 samples per second). MTS are labeled in 3 classes: *non-terminating atrial fibrilation*, *atrial fibrilation terminates one minute after*

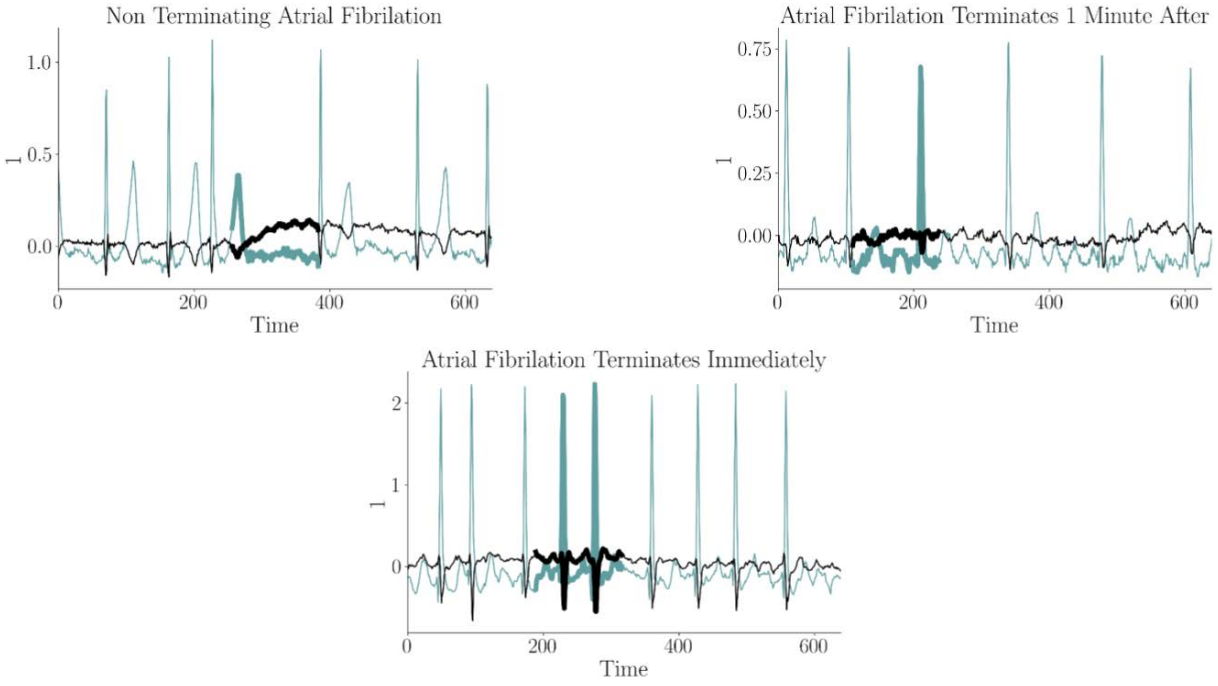



Figure 7.9 – First MTS sample per class of Atrial Fibrilation test set with the XEM time window used for classification.

and *atrial fibrilation terminates immediately*. XEM correctly predicts the 3 MTS based on the one second time window (20%) highlighted in Figure 7.9. There is a unique window for each MTS with the highest class probability (class *non-terminating atrial fibrilation*: 94.6%, *atrial fibrilation terminates one minute after*: 97.7%, *atrial fibrilation terminates immediately*: 97.4%). We can observe in the *non terminating atrial fibrilation* MTS that the time window highlighted reveals an abnormal constant increase on channel 2 (black line) during one second whereas the other channel keeps the same motif as other windows. On the *atrial fibrilation terminates one minute after* MTS, we observe a smaller decrease in channel 2 than in other windows and a low peak in channel 1. These particular 20% time windows inform the user about XEM classification outcome, thus providing important information to domain experts.

**Racket Sports Dataset** The second category of datasets where XEM obtains its best results on a time window size $\leq 60\%$ of the MTS lengths is human activity recognition. As previously done with Atrial Fibrilation, I illustrate it in Figure 7.10 by highlighting the 60% time window of the first MTS sample per class in the Racket Sports test set to gain insights on XEM classification result. Racket Sports dataset is composed of 6 dimensions,

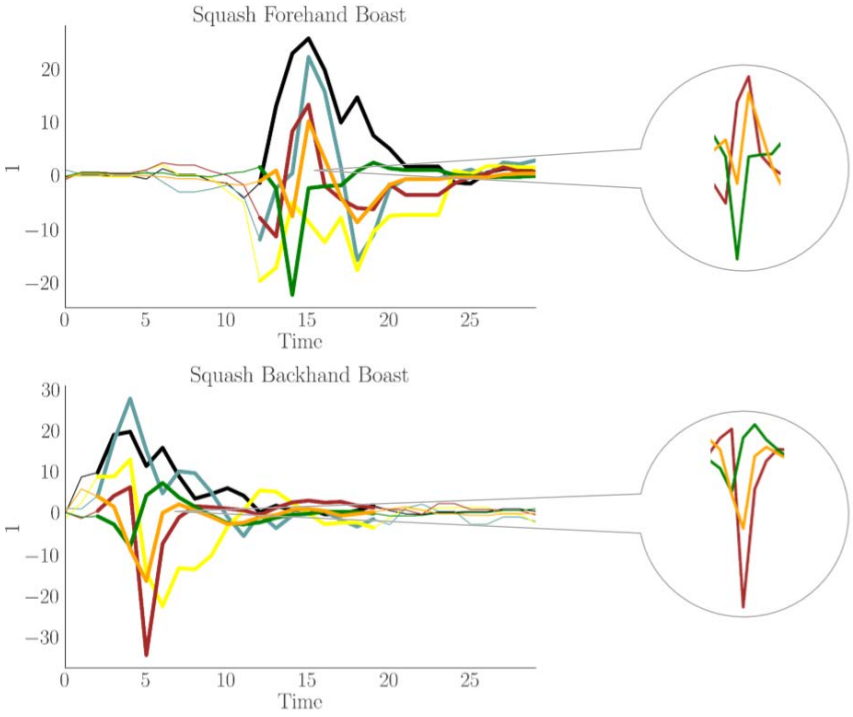



Figure 7.10 – First MTS sample per class of Squash Racket Sports test set with the XEM time window used for classification.

x/y/z coordinates for both the gyroscope and accelerometer of an android phone, on a 3 second period (10 samples per second). MTS are labeled in 4 classes: *badminton smash*, *badminton clear*, *squash forehand boast* and *squash backhand boast*. I illustrate the explainability of XEM on the two classes relative to the squash: *squash forehand boast* and *squash backhand boast*. XEM correctly predicts the 2 MTS based on the 1.8 seconds time window (60%) highlighted in Figure 7.10. There is a unique window for each MTS with the highest class probability (*squash forehand boast*: 90.3%, *squash backhand boast*: 86.7%). We can observe that for these 2 MTS the window highlighted well correspond to the period of the full movement. Then, we can see a simultaneous steep peak on red and orange dimensions with a steep decrease on green dimension for *squash forehand boast*. Whereas, we can see a simultaneous steep decrease on red and orange dimensions without a particular variation on the green dimension for *squash backhand boast*. These particular 60% time windows inform the user about XEM classification outcome, thus providing important information to domain experts.

These two examples show how XEM outperforms other MTS classifiers (rank 1 on Atrial Fibrilation and Racket Sports) while offering explainability on its predictions.

### 7.4.3 Discussion

I have presented the new explainable ensemble method for multivariate time series classification XEM. I have shown that XEM outperforms state-of-the-art MTS classifiers on the UEA datasets. In addition, XEM provides explainability by design and manifests robust performance when faced with challenges arising from continuous data collection (different MTS length, missing data and noise).

However, the ensemble method has some limitations. First, XEM predicts the class of a MTS based on a single window, the one on which it is the most confident, without considering the predictions on the other windows. Some datasets can contain MTS with different windows close to the characteristics of different classes. Therefore, XEM can have high class probabilities on different windows and when the window on which XEM is the most confident is characteristic of another class than the expected one, XEM incorrectly classifies the MTS. To illustrate it, I present in Figure 7.11 two MTS of the UEA Libras test set. XEM performed poorly on this dataset and obtained the rank 10/11 (see section 7.4.1). The Libras dataset contains 15 classes of 24 instances each, where each class references a hand movement type in the Brazilian sign language Libras. The hand movement is represented as a bi-dimensional curve performed by the hand in a period of time. We can observe in Figure 7.11 that the two MTS belonging to the same class have comparable evolution across time but XEM classifies them into two different classes. The first MTS is correctly classified based on the time window [23,40] with a class probability equals to 93.5%. We can assume that the evolution on this window is characteristic of the class 6. The second MTS also contains a comparable window on the range [23,40] but is incorrectly classified based on another window (range [0,17]) with a class probability of 94.5%. Therefore, XEM is the most confident on a window characteristic of another class (class 4). XEM did not considered the predictions on the other windows to take its decision. More particularly, XEM did not considered the expected window [23,40] to take its decision, where it also gets a high class probability of 86.3%. So, it would be interesting to improve the XEM by considering in the final decision the predictions on the different windows of a MTS.

Moreover, XEM provides explainability through identifying the time window used to classify the whole MTS. However, the black-box hybrid ensemble method LCE is not an explainable classifier. So, XEM explainability relies on human visual analysis of the selected window to identify the pattern characteristic of a MTS class. It would be valuable to integrate into XEM explainability a post-hoc approach to mine the pattern characteristic

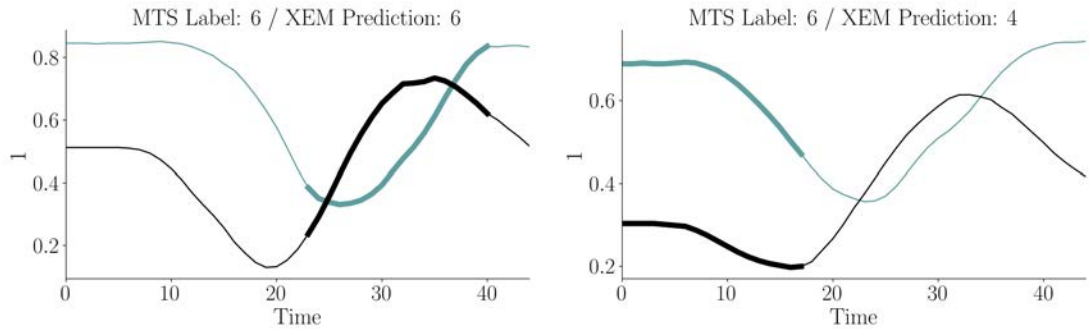



Figure 7.11 – Two MTS samples of Libras test set belonging to the same class with XEM predictions and the time windows used for classification (40%).

of the time window selected for each MTS.

Finally, I assume in XEM that a unique window size is suitable to discriminate the different classes. Nonetheless, we can imagine that different classes can be characterized by signals of different lengths. Therefore, it would also be interesting to improve XEM by integrating the possibility of multiple windows sizes.

## 7.5 Performance-Explainability Analysis

This section introduces the new ensemble method XEM into the analytical framework of the thesis (part II). The different aspects of the XEM framework are summarized in the Table 7.5 and can be visualized in Figure 7.12.

The study of this chapter shows that XEM exhibits better accuracy than the state-of-the-art MTS classifiers on the UEA datasets. Therefore, in the framework presented in part II, following predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets, the performance level of XEM is better than the state-of-the-art (Performance: *Best*). The category of performance for XEM is the same as the ensemble methods of the previous chapters (DMSEEW: *Best*, LCE: *Best*).

Concerning the explainability, as an ensemble method, XEM is a "black-box" classifier (Comprehensibility: *Black-Box* - DMSEEW: *Black-Box*, LCE: *Black-Box*). However, XEM conveys more informative explanations than DMSEEW and LCE with SHAP post-hoc model-agnostic explainability (Information: *Uni Sequences* - DMSEEW: *Features*, LCE + SHAP: *Features+Time*). XEM provides the time window used to classify the whole MTS as explanations to the end-user, i.e. the subseries on each feature. In addition, similar to DMSEEW, XEM's explainability by design provides faithful explanations (Faithfulness: *Perfect* - DMSEEW: *Perfect*, LCE + SHAP: *Imperfect*). As LCE with SHAP post-hoc

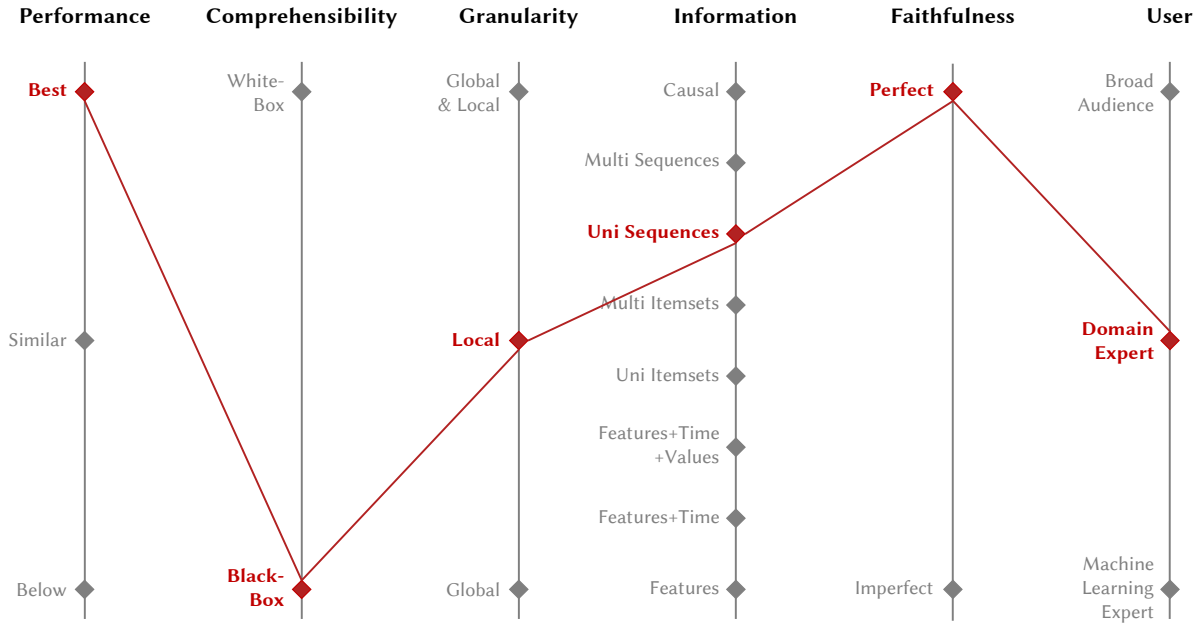



Figure 7.12 – Parallel coordinates plot of XEM on the public UEA MTS datasets. Performance evaluation method: predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets. Models evaluated in the benchmark: $\text{DTW}_D$, $\text{DTW}_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, RFM, SMTS, UFS, WEASEL+MUSE, XEM and XGBM.

model-agnostic method, the explanations are accessible to a domain expert (User: *Domain Expert* - DMSEEW: *Machine Learning Expert*, LCE + SHAP: *Domain Expert*). Nevertheless, unlike LCE with SHAP post-hoc model-agnostic explanations and like DMSEEW, XEM can only give explanations for a particular instance and not at global level (Granularity: *Local* - DMSEEW: *Local*, LCE + SHAP: *Both Global & Local*).

Table 7.5 – Performance-explainability results of the ensemble methods. Abbreviations: ML - Machine Learning.

| | **DMSEEW** | **LCE + SHAP** | **XEM** |
|---|---|---|---|
| Performance | Best[1] | Best[2 3] | Best[4] |
| Comprehensibility | Black-Box | Black-Box | Black-Box |
| Granularity | Local | Both Global & Local | Local |
| Information | Features | Features+Time | Uni Sequences |
| Faithfulness | Perfect | Imperfect | Perfect |
| User | ML Expert | Domain Expert | Domain Expert |

[1] 3-fold cross-validation and an arithmetic mean of the accuracies on the Earthquake Early Warning Dataset. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.
[2] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM and XGB.
[3] 3-fold cross-validation and an arithmetic mean of the accuracies on the UCI datasets. Models evaluated in the benchmark: EN, LC, LCE, MLP, RF, SVM and XGB.
[4] Predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, RFM, SMTS, UFS, WEASEL+MUSE, XEM and XGBM.

To further improve the level of information and granularity of XEM explanations, it would be interesting to analyze the time windows characteristic of each class in the training set in order to determine if they contain some common multidimensional patterns. Such patterns could also broaden the audience as they would synthesize the important information in the discriminative time windows.

The next chapter shows how to make the explanations accessible to a broader audience with a comprehensible pattern-based classifier.

**Summary**

- XEM is a new hybrid ensemble method for MTS classification which exhibits better accuracy than the state-of-the-art MTS classifiers on the UEA datasets.
- XEM provides faithful and informative explanations accessible to domain experts for each instance. The explanations provide to the user the time window used to classify the whole MTS.
- However, the audience remains limited and the model is not comprehensible.

Part IV

# Towards Individual Methods Combining Performance with Faithful, Informative and Human-Friendly Explainability

Chapter 8

# A Pattern-Based Approach with Human-Friendly Explainability by Design

## Contents


**8.1 Introduction . . . . . 129**
8.1.1 Contributions . . . . . 130
**8.2 XPM . . . . . 131**
8.2.1 Detection Algorithm . . . . . 131
8.2.2 Explainability by Design . . . . . 133
**8.3 Evaluation . . . . . 134**
8.3.1 Real-World Dataset . . . . . 134
8.3.2 Algorithms . . . . . 136
8.3.3 Hyperparameters . . . . . 137
8.3.4 Metrics . . . . . 138
**8.4 Results . . . . . 138**
8.4.1 Performance . . . . . 138
8.4.2 Explainability . . . . . 143
**8.5 Performance-Explainability Analysis . . . . . 146**


As the second machine learning method in chapter 6 (LCE + SHAP), the fourth machine learning of the thesis aims to improve the detection of determining events for milk production in dairy farms, which is crucial for an optimal resource use (application background presented in section 3.1.1). It aims to enhance the explainability of LCE + SHAP with an easy-to-understand model providing faithful and more informative explanations, which are accessible to a broader audience. The method is the result of a collaboration between researchers from the PEGASE ("Physiology, Environment, Genetics for Animals and rearing systems") unit at the French National Institute for Agriculture, Food and Environment (INRAE) and machine learning researchers from the LACODAM ("Large Scale Collaborative Data Mining") team at Inria, France.

## 8.1 Introduction

As presented in chapter 6, LCE is the only existing machine learning solution trained using the method of progesterone dosage in milk as the reference to obtain an exhaustive estrus labeling which covers both behavioral and silent estrus. LCE significantly improves the detection of estrus compared to a commercial reference and provides explanations at all granularity levels with SHAP, which are two levels of information interesting for the support of farmers' decision-making. However, LCE is a complicated-to-understand model ("black-box"). In addition, SHAP post-hoc model-agnostic explainability method does not provide perfectly faithful explanations and the level of information remains limited. The results presented in chapter 6 have shown that considering more than one day preceding the estrus can be valuable for estrus detection, i.e. a time series length larger than two. Therefore, this chapter studies a new easy-to-understand MTS classifier for estrus detection providing faithful and more informative explanations than LCE.

The use of patterns with an easy-to-understand classifier has great potential for explainability, and there is no pattern-based classifier among the state-of-the-art MTS classifiers. Patterns are small conjunctions of symbols with a predefined semantic. They are informative and can provide faithful explanations at all granularity levels when used as features of an easy-to-understand classifier. Moreover, as stated in [Fournier-Viger et al., 2017], patterns mined on a frequency criterion are understandable by humans. A pattern-based detection can broaden the audience and avoid assumptions about the user experience compared to other explainable time series classification methods based on subseries (e.g. shapelet methods [Karlsson et al., 2016]). The subseries correspond to the parts used for detection but do not provide information about the relevant relations among the elements of the subseries like patterns do (e.g. time gap). The diversity of pattern types available [Han et al., 2011] (e.g. itemsets, sequences, chronicles) allows the selection of patterns that best describe the phenomenon studied. So, I hypothesize that a pattern mining approach for the detection of the whole coverage of estrus based on affordable sensor data can provide by design some faithful, informative and widely accessible explanations at all granularity levels when used with an easy-to-understand classifier.

This study aims to provide explanations in ways that end-users can understand the alerts based on the patterns they could observe in animals. Therefore, the mining of frequent patterns seems suited. In addition, as stated in [Cheng et al., 2007], frequent patterns are high quality features and have good model generalization ability. Therefore,

the mining of frequent itemsets and sequences is considered. I excluded to mine more elaborated patterns which could add temporal relations among symbols (e.g. chronicles - definition in section 2.2.2) due to their limited interest on the short time series considered (e.g. 4 days with one timestamp per day). Next, sequential patterns as ordered groups of values provide more informative patterns than itemsets. In addition, a sequence-based classifier can exhibit better detection performance than an itemset-based classifier as shown in the experiments in section 8.4.1. So, I limit the discussion to sequence-based classifiers.

The current state-of-the-art of sequence-based classifiers are not adopted for explainability reasons. As presented in section 2.2.2, the adoption of support vector machines and Bayesian networks to perform classification in [Buza et al., 2010] limits the comprehensibility of how the patterns are used in the model output. Then, [Fradkin, 2014] classifies based on discriminant sequential patterns. But, the discriminant sequence mining task extracts patterns that occur also in other classes than those which are initially discriminated. Thus, the classification task can lead to unclear explanations supporting predictions in case of communication to the user of discriminative patterns of other classes than the one the model is predicting. So, in this chapter, I mine frequent sequences without considering the class information. In addition, as stated in [Fradkin, 2014], direct methods can reduce the number of patterns generated but can also lead to significantly worse performance compared to indirect methods. Therefore, a new indirect sequence-based classifier based on frequent sequences for estrus detection is proposed in this chapter.

### 8.1.1 Contributions

The research of this chapter aims to develop an explainable by design pattern-based estrus detection solution to detect both types of estrus (behavioral/silent) with the combination of data gathered by affordable sensors. This study will:

- Propose XPM, a new eXplainable Pattern-based method for MTS classification, to detect both types of estrus with combined real-world affordable data (activity, body temperature);
- Show that XPM performs slightly better than a commercial reference in estrus detection, driven by the detection of silent estrus;
- Present the limited set of patterns needed for these detections and how it can be used to support farmers' decision-making.

## 8.2 XPM

### 8.2.1 Detection Algorithm

Estrus detection can be formulated as a classification problem, where the input is sensor data and the ouput is a class (estrus/anestrus). More specifically, the problem is an instance of multivariate time series classification. There are a set of co-evolving time series (7 dimensions), recorded simultaneously by 2 sensors (activity meter, thermobolus) which form a multivariate time series (MTS). Time series are 24hr aggregated, which is sufficient for an estrus alert system. In addition, the 24hr aggregation allows the mining of patterns that are not affected by the intraday sequence of animals activities, which is irrelevant to estrus. As illustrated in Figure 8.1, the new indirect and eXplainable Pattern-based approach for MTS classification (XPM) is composed of the following steps:

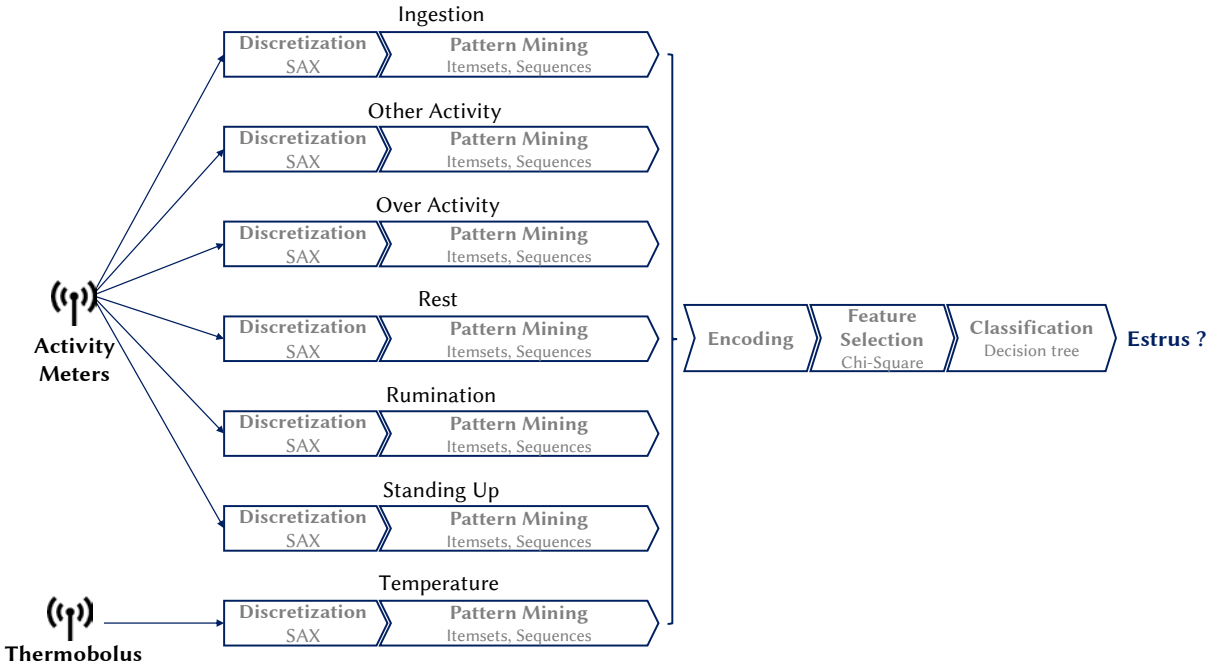



Figure 8.1 – Pipeline of the pattern-based classifier XPM.

- *Discretization*: SAX [Lin et al., 2002] is applied on each dimension. SAX transforms a time series into a string using an alphabet of predefined size. Each symbol of the alphabet corresponds to an interval of a dimension values, therefore SAX symbols can be interpreted (e.g. alphabet of size 3: {low, medium, high}). Figure 8.2 gives

a discretization example on a rumination time series of length 7 with an alphabet of size 8 $\{---, --, -, =-, =+, +, ++, +++\}$. Based on the intervals defined by SAX, the discretization output is: $+++, +++, --, +, =-, ++, ++$. Alphabet size per dimension is a hyperparameter. The alphabet size is limited to [1,10] to obtain readable patterns and 3 sizes of alphabet are set according to the type of dimensions (alphabet 1: continuous dimension - temperature, alphabet 2: integer dimensions - other and over activity, alphabet 3: binary dimensions - the remaining dimensions). Alphabets are defined on the training set and applied on validation/test sets;

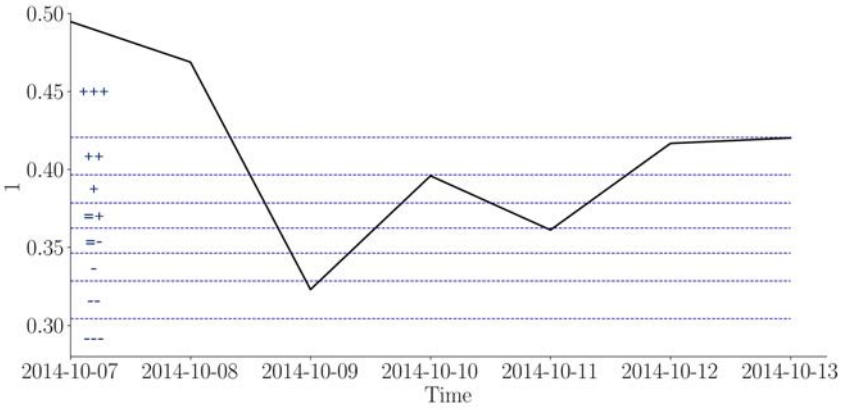



Figure 8.2 – SAX discretization example on a rumination time series of length 7 with an alphabet of size 8.

- *Pattern Mining*: two types of patterns are extracted - frequent itemsets with Eclat algorithm [Zaki, 2000] and frequent closed sequences with BIDE algorithm [Wang et al., 2004]. Frequent itemsets are groups of symbols occuring in at least a predefined percentage (support) of the time series. For example, the itemset $\{=-, +, ++\}$ present in the rumination time series of Figure 8.2 is frequent if it occurs in at least 20% of the time series of the training set. Frequent sequences correspond to frequent ordered groups of symbols. The type of pattern (itemsets, sequences) and the support are hyperparameters. Support is restricted to [10%,50%] for itemsets and [3%,9%] for sequences to not only mine high frequency patterns of length 1;
- *Encoding*: a matrix encodes which patterns (as columns) are present in which MTS (as rows) to form the input data of the classifier (see Figure 8.3);
- *Feature Selection*: before classification, a feature selection is performed to keep a limited and explainable set of patterns. A filter method is used to select the k-best

**Encoding**: m*(p+2)

| MTS ID | Pattern 1 | Pattern 2 | Pattern 3 | ... | Pattern p-2 | Pattern p-1 | Pattern p | Label |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | | 1 | 0 | 0 | Anestrus |
| 2 | 1 | 0 | 0 | | 0 | 0 | 1 | Estrus |
| 3 | 0 | 0 | 0 | | 0 | 1 | 0 | Anestrus |
| 4 | 1 | 0 | 1 | | 0 | 0 | 1 | Estrus |
| . | | | | | | | | . |
| . | | | | | | | | . |
| . | | | | | | | | . |
| m-2 | 0 | 1 | 0 | | 0 | 1 | 0 | Anestrus |
| m-1 | 1 | 0 | 0 | | 0 | 0 | 1 | Estrus |
| m | 0 | 1 | 0 | | 1 | 1 | 0 | Anestrus |

Figure 8.3 – Encoding matrix example. m: number of MTS samples, p: number of patterns mined

patterns according to a score (Chi-Square). The Chi-Square is chosen because it is suited for feature selection on booleans data relative to classes. The number of patterns is a hyperparameter and I limit its range to [10,40];

- *Classification*: finally, the MTS are classified using a decision tree, i.e. an easy-to-understand model which provides explanations at all granularity levels. The explainability provided by the decision tree classifier is detailed in section 8.2.2.

### 8.2.2 Explainability by Design

The explainability of the approach stems from the communication to the farmers of the presence and/or absence of a limited number of patterns determinant of estrus detection. Patterns are communicated to the farmers following a decision tree to classify estrus. I present in this section how to read a pattern and a decision tree. Figure 8.4 shows an example of a one node with two leaves decision tree trained on a dataset of 600 MTS (300 estrus/300 anestrus). The node is composed of the pattern ++++ on the dimension over activity. In an alphabet of size 9 $\{----, - - -, --, -, =, +, ++, +++, ++++\}$, ++++ refers to the interval of highest values relative to the dimension over activity. Therefore, we can observe that when a high over activity (pattern ++++ on over activity) is observed in a MTS (pattern present), which is the case for 199 MTS, most of the MTS correspond to estrus (184 over 199, error rate: 8%). In this case, the decision tree predicts the class estrus: the most represented class in the leaf. Blue filled nodes mostly contain estrus and grey filled nodes mostly anestrus. When the pattern ++++ on over activity is not observed in a MTS, the decision tree predicts anestrus but with a higher error rate (116/401 = 29%). Additional patterns could refine the decision

and reduce the error of the tree. The explainability results are presented in section 8.4.2.

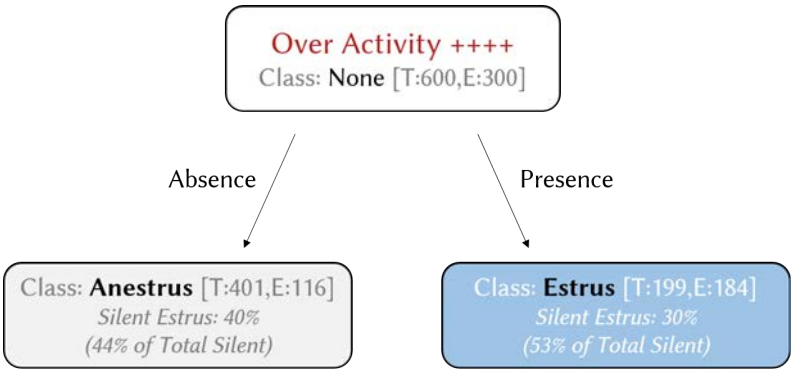



Figure 8.4 – Explainability - decision tree example with one node and two leaves. Abbreviations: T - total number of samples, E - number of estrus

## 8.3 Evaluation

In this section, I introduce the dataset and methodology used for evaluating the approach.

### 8.3.1 Real-World Dataset

The evaluation of XPM is performed on the same real-world dataset as LCE in chapter 6. The dataset is presented in section 6.3.1.

#### Labeling

The study covers both estrus types (behavioral and silent). Therefore, estrus are labeled by measuring the progesterone concentration in whole milk, the costly gold standard for an exhaustive estrus identification [Martin et al., 2013]. Milk samples were collected from each cow twice a week on Tuesdays and Thursdays and were immediately frozen at -20°C until the dosage. The enzyme-linked immunosorbent assay technique (kit ELISA Ridgeway Science Ltd) has been used. Then, with preserved and frozen milk, the separation of basic concentrations of progesterone to estrus period has been determined based on the quantile method [Cutullic et al., 2011; Petersson et al., 2006]. Figure 8.5 shows an example of the output from the quantile method applied on the progesterone profile

of a cow on a one month time period, and how this output is used to label the corresponding time series, in this case of 4-day length (the final configuration - detailed in section 8.4.1). Two types of labeling are adopted: first, a non-overlapping labeling on the cross-validation dataset to find patterns on clearly identified periods, which improves the relevancy of patterns used subsequently for classification; second, a sliding window labeling on the external validation dataset to evaluate the performance of the approach on real-world conditions, i.e. as a daily monitoring solution. The non-overlapping labeling consists of an equal number of estrus and anestrus time series to avoid class imbalance, with the estrus MTS ending in the middle of the estrus period and the anestrus MTS preceding the estrus one. On both labeling, the label of the last day is used to label the whole time series. Finally, behavioral and silent estrus are labeled based on the same methodology as the one used to evaluate LCE in section 6.3.1.

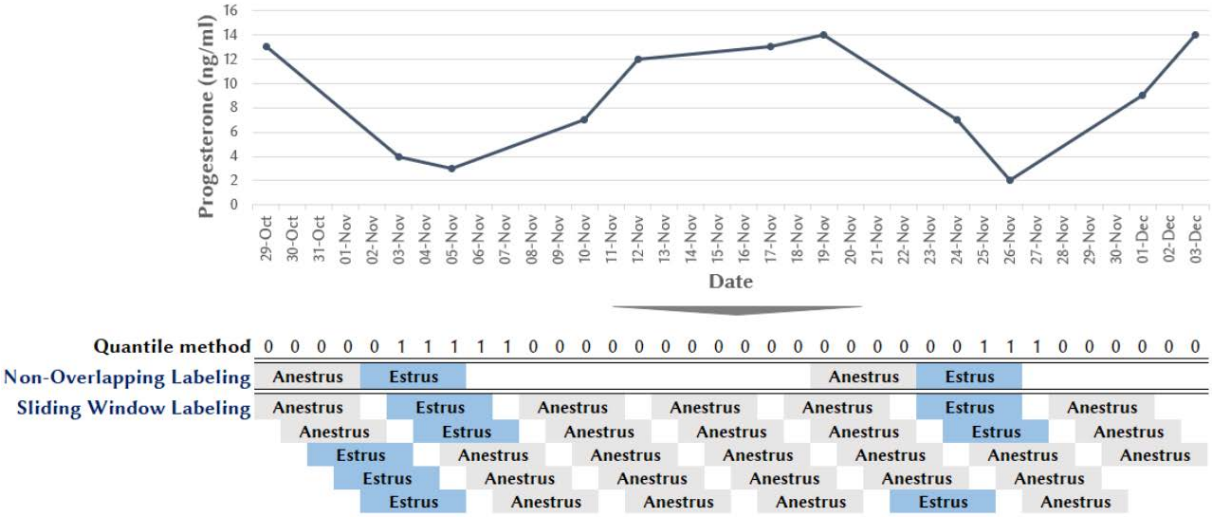



Figure 8.5 – Example of the output from the quantile method [Petersson et al., 2006], applied on the progesterone profile of a cow on a one month time period, with the corresponding non-overlapping and sliding window labeling in the case of time series length of 4 days. Quantile method output: 1 - day of an estrus period, 0 - day of a anestrus period.

### Composition

The composition of the cross-validation and external validation datasets are presented in Table 8.1. The cross-validation dataset was split into five folds. The split has kept the same number of estrus in each fold (100 days). This split does not lead to an overfit on

a particular animal. The study in chapter 6 shows that a dataset split on animals has a negative impact on detection performance (-2.6% impact on F1-score).

Table 8.1 – Dataset split. Abbreviation: Ext Val - external validation

| | **Folds** | | | | | | **Ext Val** |
|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **All** | |
| Estrus | 99 | 100 | 100 | 100 | 100 | 499 | 321 |
| Silent % | 32 | 38 | 34 | 30 | 36 | 34 | 44 |
| Lactation 1 | 64 | 55 | 61 | 58 | 61 | 299 | 193 |
| Silent % | 36 | 36 | 41 | 26 | 43 | 36 | 50 |
| Lactation 2+ | 35 | 45 | 39 | 42 | 39 | 200 | 128 |
| Silent % | 26 | 40 | 23 | 36 | 26 | 31 | 34 |

### 8.3.2 Algorithms

I evaluate in section 8.4.1 the performance of XPM on the real-world dataset in comparison with a reference commercial solution, the current state-of-the-art estrus detection algorithm and some variants of XPM:

- CS: commercial reference solution (see section 6.3.1);
- LCE: the current best performing machine learning approach for estrus detection (see section 6.4.2), which also adopts an exhaustive estrus labeling and affordable sensor data (activity, temperature);
- XPM: the proposed approach (see section 8.2.1);
- XPM-Derivatives: XPM on a dataset augmented by the derivatives of each dimension. Górecki et al. (2013) show that using derivatives can be helpful in time series classification. Derivatives correspond to the value difference of a dimension compared to the previous day. Figure 8.6 illustrates a dataset augmented by the derivatives of each dimension.
- XPM-Individual: XPM with animal-specific alphabets in order to evaluate the impact on performance of predicting estrus according to each cow previous activity and temperature. The discretization is performed on the value difference compared to the historical mean for each animal.

The pattern-based classifier XPM is implemented in Python 3.6. In the different steps of XPM pipeline, the following implementations are used:

| Animal ID | Timestamp | Attribute 1 | Attribute 1 Der | . . | Attribute 7 | Attribute 7 Der |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | - | | 39.24 | - |
| 1 | 2 | 1 | 0 | | 38.9 | -0.34 |
| 1 | 3 | 0 | -1 | | 38.78 | -0.12 |
| 1 | 4 | 0 | 0 | | 39.1 | 0.32 |
| 1 | 5 | 1 | 1 | | 39.19 | 0.09 |
| . | . | . | . | . . | . | . |
| 1 | T-1 | 1 | 0 | | 39.23 | 0.08 |
| 1 | T | 0 | -1 | | 39.34 | 0.11 |

Figure 8.6 – MTS sample with derivatives for one animal of the dataset. For each timestamp, the 7 attributes of the dataset with their derivatives are represented. Abbreviations: Der - derivatives, ID - identifier, T - time series length.

- *Discretization*: a SAX public implementation [1];
- *Pattern Mining*: public implementations of Eclat [2] and BIDE[2];
- *Feature Selection*: SelectKBest [3] [Pedregosa et al., 2011] public implementation;
- *Classification*: DecisionTreeClassifier [4] [Pedregosa et al., 2011] public implementation.

### 8.3.3 Hyperparameters

The hyperparameters of XPM are:

- Alphabet sizes: 3 alphabets are defined, according to the types of dimensions (alphabet 1: continuous dimension - temperature, alphabet 2: integer dimensions - other and over activity, alphabet 3: binary dimensions - the remaining dimensions), with sizes in [1,10];
- Time series length: it ranges from 4 to 21 days - the length of a regular ovarian cycle ([4,21]);
- Patterns: type of patterns (itemsets, sequences);
- Number of patterns: number of patterns kept during the feature selection ([10,40]);
- Support: minimum frequency of patterns (itemsets: [10%,50%], sequences: [3%,9%]);
- Decision tree hyperparameters: depth of the tree [1,ln(number of patterns)], minimum number of samples at a leaf [2,number of patterns].

1. https://github.com/seninp/saxpy
2. http://www.philippe-fournier-viger.com/spmf
3. sklearn.feature_selection.SelectKBest
4. sklearn.tree.DecisionTreeClassifier

Most of these hyperparameters (alphabet sizes, number of patterns, time series length, patterns, support) are determined by grid search on the validation sets of the cross-validation. On the same validation sets, decision tree hyperparameters are determined by the hyperopt algorithm [Bergstra et al., 2013].

### 8.3.4 Metrics

I do not make assumptions on dairy herd management, I do not give a preference to reduce false positives (false estrus alert) or false negatives (estrus not detected) so I have optimized the F1-score, the harmonic mean between precision and recall. Adopting a conservative approach, I decided to aggregate the model daily predictions by the maximum of the daily predictions on estrus/anestrus period to calculate the classification performance. Based on a 5-fold cross-validation 60/20/20 train/validation/test split, the algorithm is selected on the best F1-score on validation sets. I present two levels of performance. First, I show the F1-score with precision set to the same as CS (78%, threshold: 0.4) to allow comparison between the approaches. The threshold corresponds to the value from which the pattern-based classifier class probabilities predict estrus. The second level is the F1-score across all possible calibrations (threshold range: 0.3-0.75) which corresponds to the average performance of the solution. We can observe that for high thresholds (threshold $> 0.75$), the pattern-based classifier performance is unstable with a significant decrease in estrus detection rate (recall below 70%). In addition, for low thresholds (threshold $< 0.3$), the classifier is equivalent to a random classifier. So, I decided to adopt a F1-score calculation based on the average of F1-score on threshold range 0.3-0.75, which corresponds to the plausible range of calibration for dairy management and shows a detection performance closer to real conditions.

## 8.4 Results

In this section I first present the detection performance of the approach and then the explainability provided.

### 8.4.1 Performance

In this section, the performance of the pattern-based classifier XPM on the cross-validation and on the external validation datasets are presented. Then, I compare the

results to previous studies and finally, I discuss the performance of an individual approach.

Firstly, I have evaluated XPM approach on the cross-validation dataset. The dataset is composed of non-overlapping periods, with an equal number of estruses and anestruses to avoid class imbalance. The non-overlapping time windows labeling allows the extraction of patterns on clearly identified period, which improves the relevancy of patterns used subsequently for classification (see section 8.3.1). Table 8.2 presents the F1-score on test sets of XPM approach on the cross-validation dataset. We can observe a better F1-score with less variability across folds of XPM approach based on sequences than itemsets on both behavioral and silent estruses (total: 74.5% $\pm$ 2.2 versus 73.6% $\pm$ 3.9, behavioral: 78.2% $\pm$ 3.1 versus 77.4% $\pm$ 4.2, silent: 57.7% $\pm$ 1.3 versus 56.4% $\pm$ 2.9). Thus, according to the experiments, the most informative patterns (sequences) have to be selected for estrus detection. Moreover, as presented in section 8.3.2, I have evaluated what would be the impact of adding as dimensions to the dataset the derivatives of each dimension (XPM-Derivatives). A dataset with derivatives adds the possibility for a pattern to give information for example about the activity level of a cow, but also about the variation of activity compared to the day before. We can observe in Table 8.2 that mining sequences on both raw dimensions and derivatives improves F1-score compared to sequences on raw dimensions (total: XPM-Derivatives 75.4% versus XPM 74.5%). Based on the cross-validation, we obtain the best configuration for the XPM-Derivatives approach (alphabet 1 size: 7, alphabet 2 size: 9, alphabet 3 size: 8, time series length: 4 days, patterns: sequences, feature selection: 20, support: 3%). We can notice that the predefined parameter interval for feature selection ([10,40]) did not affect the selection of the classifier, considering the set of 20 patterns on the best configuration. To support the decision of a non-overlapping labeling, I have also evaluated the approach on a cross-validation dataset generated by a sliding window labeling. XPM with sequential patterns shows poor performance in the sliding window configuration: a F1-score average on test sets of 53.1%, which underperforms the commercial solution (F1-score 54.3%).

Then, I have evaluated the pattern-based classifier on the external validation dataset. The estrus detection solution alerts the farmers on a daily basis so the external validation dataset is generated with a sliding window for results presentation (see section 8.3.1). The performance presented in XPM-Derivatives with Sequences corresponds to the classifier with the best configuration obtained on cross-validation. In order to have comparable results to the commercial solution, I set the value from which the pattern-based classifier class probabilities predicts estrus to 0.4. It is the threshold for which the classifier shows

Table 8.2 – Comparison[1] of estrus detection F1-score[2] with 95% confidence interval and statistical significance[3] on test sets and external validation. Abbreviation: CS - Commercial Solution.

| | **Cross-Validation** | | | **External Validation** | |
|---|---|---|---|---|---|
| | **XPM with Itemsets** | **XPM with Sequences** | **XPM-Derivatives with Sequences** | **XPM-Derivatives with Sequences** | **CS** |
| Total | 73.8 ± 3.6 | 75.2 ± 1.5 | 78.1 ± 0.9 | 62.4 | 60.9 |
| | (73.6 ± 3.9) | (74.5 ± 2.2) | (75.4 ± 1.6) | - | - |
| Behavioral | 76.1 ± 3.9 | 78.8 ± 2.5 | 79.1 ± 1.3 | 56.3 | 81.8* |
| | (77.4 ± 4.2) | (78.2 ± 3.1) | (77.6 ± 1.6) | - | - |
| Silent | 56.2 ± 2.5 | 58.4 ± 1.2 | 55.2 ± 1.5 | 39.2 | 0.0*** |
| | (56.4 ± 2.9) | (57.7 ± 1.3) | (57.1 ± 1.8) | - | - |
| Lactation 1 | 73.3 ± 3.5 | 74.3 ± 1.7 | 76.4 ± 1.3 | 58.9 | 57.3 |
| | (72.9 ± 3.8) | (73.6 ± 2.3) | (74.4 ± 2.3) | - | - |
| Lactation 2+ | 74.3 ± 4.3 | 76.1 ± 2.2 | 80.4 ± 2.2 | 66.9 | 65.3 |
| | (74.1 ± 4.5) | (75.1 ± 2.7) | (76.4 ± 1.8) | - | - |

[1] Methods compared: XPM based on itemsets/sequences/sequences with derivatives and the commercial solution.
[2] Two levels of performance are presented. The first line corresponds to the performance based on the same total precision as CS (78%, S&D threshold set to 0.4). The second line with parenthesis shows the average performance across all possible calibrations (threshold range: 0.3-0.75).
[3] The P-value represents the 5*2-fold cross-validation paired t-test result of CS compared to XPM-derivatives with sequences (*P<0.05, ***P<0.01)

the same precision as the commercial solution on the training set (80%). The performance on the other thresholds of the range (0.3-0.75) is the same so I do not show it in the Table 8.2. First, we can observe that XPM approach has a slightly better F1-score than the commercial solution (62.4% versus 60.9%, no statistically significant difference), based on a better estrus coverage (recall: XPM-Derivatives with Sequences 53.0% versus CS 49.1%, precision: XPM-Derivatives with Sequences 75.9% versus CS 80.0%). In particular, we can observe a slightly better F1-score of XPM approach on the first lactation (58.9% versus 57.3%, no statistically significant difference), which is crucial for dairy farm viability. The higher performance of the approach is driven by the detection of silent estrus (F1-score: 39.2% versus 0.0%, P<0.01). I infer that the lower detection performance of the pattern-based classifier on behavioral estrus than the commercial solution (F1-score: 56.3% versus 81.8%, P<0.05) is due to the non-discriminative patterns mined. I mined patterns based on a frequency criteria without considering the type of estrus. Therefore, as illustrated in section 8.4.2, selected patterns are mostly as frequent in behavioral as in silent estrus, which prevents to fully characterize behavioral estrus. The performance results also manifest the value of combining affordable sensor data in XPM approach (data on activity and temperature instead of data on activity alone). We can observe that XPM-Derivatives with Sequences exhibits a better F1-score when trained on both activity and temperature,

while having a performance really close to CS when trained only on activity (activity and temperature: 62.4%, activity: 61.1%). The role of the pattern related to temperature on estrus detection is presented in the next section. Nonetheless, we can observe a drop in the performance of XPM-Derivatives with Sequences on both types of estrus on external validation compared to cross-validation (total: 62.4% versus 78.1%, behavioral: 56.3% versus 79.1%, silent: 39.2% versus 55.2%), mainly due to a drop in recall. The pattern coverage, superior to 85% on both cross-validation and external validation, is not responsible for this performance drop between cross-validation and external validation. The pattern coverage is the proportion of MTS which contains at least one pattern. However, we can observe a difference on the average number of patterns used for estrus/anestrus classification in cross-validation versus external validation (cross-validation: estrus - 7 patterns/anestrus - 2 patterns, external validation: estrus - 4 patterns/anestrus - 4 patterns). I infer that the lower average number of patterns in estrus MTS on external validation reduces the possibilities of XPM approach to classify estrus at the same level of performance as cross-validation. Therefore, the limited number of samples of the cross-validation dataset (998), due to the non-overlapping time windows setting, lead to an overfit which impacts the generalization ability of the pattern-based classifier. The next step would consist of a partnership with an automatic detection solution provider to have access to a heterogeneous dataset. It would allow the evaluation of the performance of XPM approach on a broader dataset to see if the frequent patterns mined in the cross-validation dataset are also present with the same proportion in the external validation dataset and what would be the impact on the performance.

We can compare the performance of XPM approach to one existing study (LCE - presented in chapter 6). The methodology employed is the same except that the cross-validation dataset was generated using a sliding window. LCE shows better F1-score than XPM-Derivatives with Sequences, while having stable results between cross-validation and external validation (cross-validation: 68.9% $\pm$ 2.4, external validation: 71.6% $\pm$ 0.4). So, LCE ensemble method has a better generalization ability than the pattern-based classifier. Nonetheless, the ensemble approach cannot support its predictions with perfectly faithful explanations as it relies on a post-hoc model-agnostic explainability method (SHAP), which can prevent LCE adoption as faithfulness is critical to reduce solution mistrust from farmers. In addition, SHAP provides the relative importance of observed variables and timestamps as explanations, which is less informative than supporting the alerts based on the patterns that the farmers could observe in animals as XPM approach does

(see section 8.4.2). Therefore, it would be interesting to work on an approach combining the performance of the ensemble method and the explainability of the pattern-based approach of this study. Next, we cannot compare the detection results of XPM approach to the ones from other existing studies using affordable activity measurements because the labeling method used is not exhaustive (visual detection). Dolecheck et al. (2015) base the study on time series data of activity, using visual detection as the ground truth (65.6% of all estruses). The cows were housed with open access to freestalls and estrus were synchronized. Three machine learning techniques are tested on a limited dataset of 18 estruses (18 cows): random forest, linear discriminant and a neural network (multilayer perceptron). Minegishi et al. (2019) learn a logistic regression on activity variables (7 days moving average/standard deviation, and the daily absolute maximum of 6-hour-window cumulative change of the data), using the combination of an automatic estrus detection solution (collar-mounted activity meter) and visual detection as the ground truth. Two herds have been studied (low-input conventional and organic) with seasonal breeding (total dataset: 1,462 estruses, 300 cows). Ma et al. (2020) trained a neural network (long short-term memory network along with a convolutional neural network) as estrus detection solution based on activity, feeding and rumination data (40 cows with 6 estruses labeled visually). Therefore, I have limited the baselines to the commercial solution CS and LCE.

Lastly, I have experimented an individual approach to evaluate the impact on performance of predicting estrus according to each cow previous activity and temperature (XPM-Individual). The alphabet sizes remain the same (alphabet 1 size: 7, alphabet 2 size: 9, alphabet 3 size: 8), but each cow has its own alphabets. We can observe a lower F1-score on cross-validation test sets of the individual approach compared to the commercial solution (52.4% versus 54.3%). I explain the poor performance of the individual approach by the insufficient data I was able to collect in order to train one classifier per cow. An estrus detection system needs to operate from the first lactation and the first estrus phases. Therefore, I have used a unique classifier trained on the encoding of the patterns with the alphabets depending on each cow. The limitation is that the same patterns are treated similarly by the classifier, while having different meanings across the animals. Thus, I infer that this limitation affects the performance of the individual approach. It would be interesting to evaluate an individual approach integrating a priori information on each animal (e.g. genetic information) to avoid the individual data collection phase to train a classifier.

### 8.4.2 Explainability

In this section I first present the decision tree of patterns used for estrus detection. Then, I compare the patterns selected to the results from previous studies and finally, I show how these patterns provide explainability to the farmers on estrus alerts.

Figure 8.7 is the decision tree corresponding to the best configuration determined by cross-validation and presented in section 8.4.1 - XPM-Derivatives with Sequences (alphabet 1 size: 7, alphabet 2 size: 9, alphabet 3 size: 8, time series length: 4 days, feature selection: 20, support: 3%). Firstly, we can observe that the presence of a steep peak in over activity (root node: pattern $\langle =, ---- \rangle$ on over activity derivative) leads to the identification of 49% of estruses of the training set (244 over 499 estruses) with a low error rate (3% - 7 anestruses over 251 samples). The simultaneous presence of a pattern confirming the first one with a steep decrease followed by a rise in rest (pattern $\langle ---, = + \rangle$ on rest derivative) leads to a refinement of the detection and identifies 38% of all estruses (191 over 499 estruses) with an error rate of 0.5% (1 anestrus over 192 samples). In the case of the absence of this confirming pattern (pattern $\langle --- , = + \rangle$ on rest derivative), the presence of a steep rise of temperature leads to the identification of estrus with a low error rate of 2.5% (1 anestrus over 40 samples). Then, in the absence of the two patterns relative to over activity (pattern $\langle =, ---- \rangle$ on over activity derivative and pattern $\langle -, ++++ \rangle$ on over activity), a low rumination (pattern $\langle --- \rangle$ on rumination) with a low rest (pattern $\langle --- \rangle$ on rest) confer the estrus (error rate of 21% - 9 anestruses over 43 samples).

Concerning the different types of estrus (silent/behavioral), some patterns among the 20 patterns are characteristic of behavioral estrus. Three patterns are three times more frequent in behavioral estrus than in silent estrus MTS: pattern $\langle -, ++++ \rangle$ on over activity, pattern $\langle --, = \rangle$ on over activity derivative and pattern $\langle -, = \rangle$ on over activity derivative. One of them (pattern $\langle -, ++++ \rangle$ on over activity - a prolonged high over activity) is used in the decision tree as a splitting dimension and the subsequent leaf contains, as expected, one of the lowest silent estrus proportion of the leaves (12.5%). We can observe that most of the patterns used in the decision tree are present in the same proportion in behavioral as in silent estrus MTS. There is no pattern among the 20 patterns that is characteristic of silent estrus (pattern at least twice more frequent in silent than in behavioral estrus). Therefore, the pattern-based approach mostly relies on the identification of patterns that are as much associated to behavioral estrus as to silent estrus (pattern $\langle =, ---- \rangle$ on over activity derivative, pattern $\langle + \rangle$ on temperature

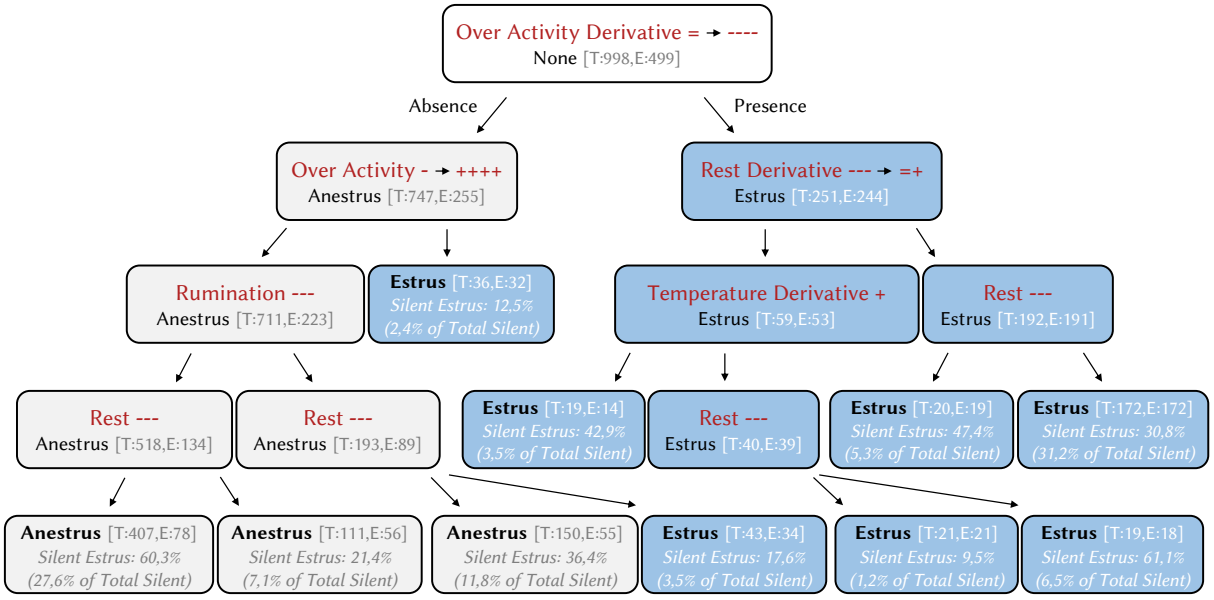



Figure 8.7 – Decision tree on cross validation dataset. Abbreviations: T - total number of samples, E - number of estrus.

derivative, pattern $\langle - - - \rangle$ on rest, pattern $\langle - - -, = + \rangle$ on rest derivative). It allows me to correctly classify more than half of the estrus type not detected by the commercial solution (51% of silent estrus). In order to obtain some patterns characteristic of silent estrus and enhance their detection performance, it would be interesting to work on a three class classification setting (anestrus/behavioral estrus/silent estrus) with discriminative patterns.

The patterns observed corroborate some observations from previous studies. First, over activity is significantly higher in estrus than in anestrus. According to the device used, activity measured in steps and neck movements increases on the day of estrus from 69 to 170% in [Mayo et al., 2019]. Jónsson et al. (2011) show that during estrus, the period of time cows spent lying decreases as a result of increased activity and restlessness. Concerning the low rumination pattern, a study from [Reith et al., 2012] reveals that rumination reduces on the day of estrus from 7.2 to 5.9 h/d. Mayo et al. (2019) confirm this reduction by publishing a $-2$ to $-16$% change in rumination time on the day of estrus for both neck and ear-based technologies. Additionally, the period of 4 days of XPM best configuration is aligned with the study of [Zebari et al., 2018] which demonstrates that on the day of behavioral estrus, the number of the steps are higher compared to 3 days before and 3 days after estrus.

With regard to the explainability results of LCE (see section 6.4.2), the temperature appeared as the most discriminative variable according to the SHAP method [Lundberg et al., 2017]. SHAP values are calculated by the average marginal contribution of a feature value towards the prediction over all possible coalitions. Therefore, SHAP values inform the inclusion/exclusion of a variable's impact on prediction. In the pattern-based approach, temperature is part of the top variables used for classification (pattern $\langle + \rangle$ on temperature derivative) but it is not the first splitting variable of the decision tree. The interpretation of the importance of variables differs between these explainability methods as SHAP is based on the inclusion/exclusion of the variables and the pattern-based approach is based on the frequency of the values taken by the variables. Nonetheless, we can draw the conclusion that temperature plays a key role in estrus detection.

The explainability of the approach stems from the communication to the farmers of the presence and/or absence of a limited number of patterns determinant of estrus detection. Patterns are communicated to the farmers following the decision tree shown in Figure 8.7. On a daily basis, XPM solution informs the farmers about the cows in estrus, the type of estrus detected with its associated probability and the patterns which

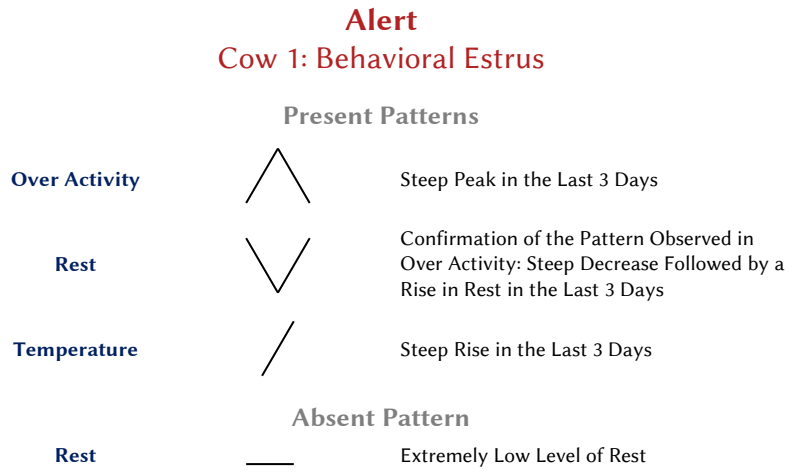



Figure 8.8 – Example of a behavioral estrus alert with the corresponding patterns detected.

have generated the alerts. In case of behavioral estrus, these patterns allow the farmers to confirm the patterns visually sensed. In addition, in case of silent estrus, XPM solution allows the farmers to save time looking for non visually verifiable behavioral signs and tells them that a potential insemination would be performed on a silent estrus. For example, Figure 8.8 illustrates the level of information that a farmer could receive with an estrus alert. The interface contains the animal identifier, the type of estrus and the patterns detected. XPM solution predicts that cow 1 is in behavioral estrus on the day of the alert based on the detection in the last 3 days of a steep peak of over activity, a steep decrease followed by a rise in rest, a steep rise in temperature and the absence of an extremely low level of rest. As indicated in the decision tree of XPM approach, the combination of these 3 patterns and the absence of the pattern low rest (rest $\langle - - - \rangle$) always lead to an estrus (0% error rate - 0 anestrus over 21 samples), with a vast majority of behavioral estruses (<10% of silent estruses).

## 8.5 Performance-Explainability Analysis

This section introduces the new pattern-based MTS classifier XPM into the analytical framework of the thesis (part II). The different aspects of XPM framework are summarized in Table 8.3 and can be visualized in Figure 8.9.

The study of this chapter shows that XPM exhibits comparable performance as a commercial reference in estrus detection on a real-world dataset. Moreover, XPM performs

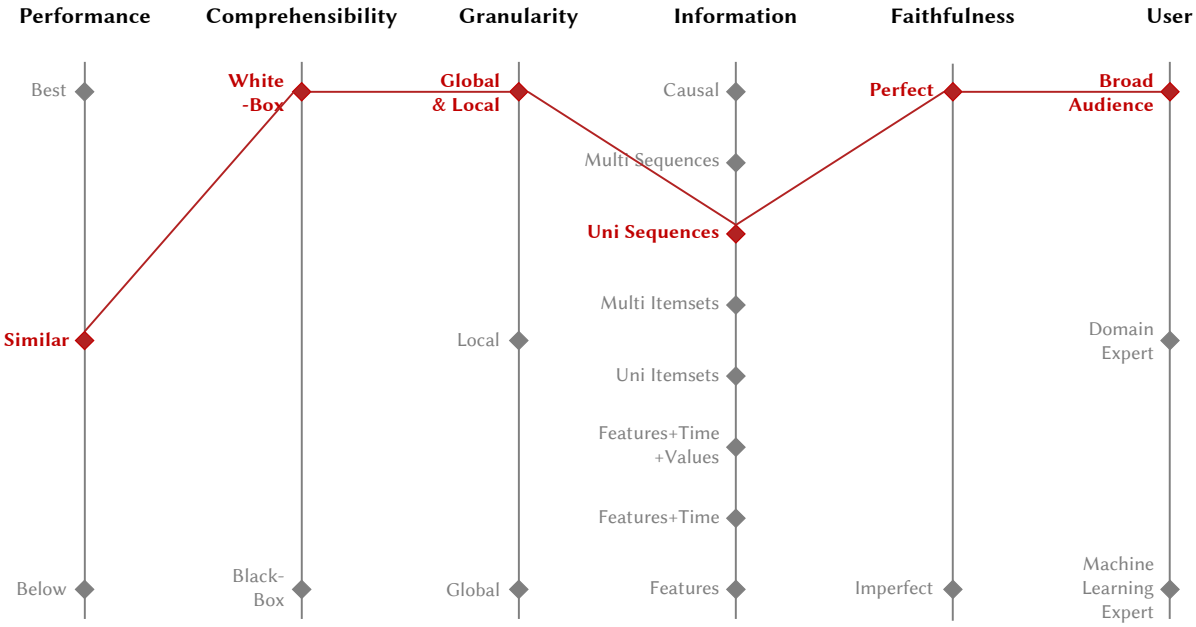



Figure 8.9 – Parallel coordinates plot of XPM on the dairy resource monitoring application. Performance evaluation method: 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM, XGB and XPM.

significantly less than the new hybrid ensemble method LCE (see section 8.4.1). Therefore, in the framework presented in part II, following a 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset, the performance level of XPM is the same as the current state-of-the-art (Performance: *Similar*, LCE: *Best*).

Concerning the explainability, unlike the ensemble methods presented in the part III, XPM is comprehensible (Comprehensibility: *White-Box*, ensemble methods: *Black-Box*). In addition, it provides faithul (Faithfulness: *Perfect* - DMSEEW: *Perfect*, LCE + SHAP: *Imperfect*, XEM: *Perfect*) and informative (Information: *Uni Sequences* - DMSEEW: *Features*, LCE + SHAP: *Features+Time*, XEM: *Uni Sequences*) explanations that are accessible to a broad audience (User: *Broad Audience* - DMSEEW: *Machine Learning Expert*, LCE + SHAP: *Domain Expert*, XEM: *Domain Expert*). The explanations consist in the presence and/or absence of a limited number of sequential patterns. These explanations are accessible to the farmers, but also to a broader audience not familiar with estrus detection based on the patterns they could observe in animals. Finally, XPM offers explanations both on its overall behavior and on a particular instance (Granularity: *Both Global & Local* - DMSEEW: *Local*, LCE + SHAP: *Both Global & Local*, XEM: *Local*).

Table 8.3 – Performance-explainability results of the ensemble methods and XPM. Abbreviations: ML - Machine Learning, G&L - Global & Local.

| | **Ensemble Methods** | | | |
|---|---|---|---|---|
| | **DMSEEW** | **LCE + SHAP** | **XEM** | **XPM** |
| Performance | Best[1] | Best[2,3] | Best[4] | Similar[5] |
| Comprehensibility | Black-Box | Black-Box | Black-Box | White-Box |
| Granularity | Local | Both G&L | Local | Both G&L |
| Information | Features | Features + Time | Uni Sequences | Uni Sequences |
| Faithfulness | Perfect | Imperfect | Perfect | Perfect |
| User | ML Expert | Domain Expert | Domain Expert | Broad Audience |

[1] 3-fold cross-validation and an arithmetic mean of the accuracies on the Earthquake Early Warning Dataset. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.

[2] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM and XGB.

[3] 3-fold cross-validation and an arithmetic mean of the accuracies on the UCI datasets. Models evaluated in the benchmark: EN, LC, LCE, MLP, RF, SVM and XGB.

[4] Predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, RFM, SMTS, UFS, WEASEL+MUSE, XEM and XGBM.

[5] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM, XGB and XPM.

The next chapter shows how to reconcile performance with faithful and informative

explainability at all granularity levels on the dairy resource monitoring application. It presents a new convolutional neural network with a post-hoc model-specific explainability method.

**Summary**

- The new pattern-based classifier XPM demonstrates similar performance as a commercial reference in estrus detection on a real-world dataset.
- However, XPM is comprehensible. In addition, it provides faithful and informative explanations at all granularity levels to a broad audience. The explanations provide to the user the presence and/or absence of a limited number of sequential patterns to support estrus alerts.

Chapter 9

# A Convolutional Neural Network Combining Performance with Faithful and Informative Post-Hoc Model-Specific Explainability

## Contents


**9.1 Introduction . . . . . 151**
9.1.1 Contributions . . . . . 153
**9.2 XCM . . . . . 153**
9.2.1 Architecture . . . . . 153
9.2.2 Explainability . . . . . 157
**9.3 Evaluation . . . . . 158**
9.3.1 Datasets . . . . . 158
9.3.2 Algorithms . . . . . 159
9.3.3 Hyperparameters . . . . . 160
9.3.4 Metrics . . . . . 160
**9.4 Results . . . . . 161**
9.4.1 Performance . . . . . 161
9.4.2 Explainability . . . . . 164
9.4.3 Real-World Application . . . . . 169
**9.5 Performance-Explainability Analysis . . . . . 173**


As the second machine learning method in chapter 6 (LCE + SHAP) and the fourth machine learning method in chapter 8 (XPM), the fifth machine learning method of the thesis continues to improve the detection of determining events for milk production in dairy farms (application background presented in section 3.1.1). It aims to enhance the performance of LCE and the information level of SHAP explanations, while providing

faithful explanations as XPM. The method is the result of a collaboration that I have built between researchers from the College of Biosystems Engineering and Food Science of the Zhejiang University, China, researchers from the PEGASE ("Physiology, Environment, Genetics for Animals and rearing systems") unit at the French National Institute for Agriculture, Food and Environment (INRAE) and machine learning researchers from the LACODAM ("Large Scale Collaborative Data Mining") team at Inria, France. During this collaboration, I have had the chance to be a visiting researcher at Zhejiang University for 2 months from mid-November 2019 to mid-January 2020. It is available on ArXiv [Fauvel et al., 2020c].

## 9.1 Introduction

As far as I have seen, the machine learning solutions introduced in chapter 6 (LCE + SHAP) and chapter 8 (XPM) are the only existing ones trained using progesterone dosage in milk as the reference to obtain an exhaustive estrus labeling which covers both behavioral and silent estrus. In addition, these two methods provide explanations at all granularity levels. As presented in chapter 6, the real-world application requires explanations at both global (estrus/anestrus) and local (behavioral versus silent estrus) levels, which are two levels of information interesting for the support of farmers' decision-making. LCE significantly improves the detection of estrus compared to a commercial reference. However, SHAP, as a post-hoc model-agnostic explainability method, cannot provide perfectly faithful explanations, and faithfulness is a prerequisite to reduce solution mistrust from the farmers. Next, XPM provides faithful and informative explanations to a broad audience. However, it demonstrates similar performance as a commercial reference in estrus detection, and the insufficient performance of current estrus detection commercial solutions is the first reason of their moderate adoption rate [Steeneveld et al., 2015]. Therefore, this chapter studies a new machine learning method for estrus detection enhancing the performance of LCE and the information level of SHAP explanations, while providing faithful explanations as XPM and at all granularity levels.

As mentioned in chapter 8, estrus detection is addressed as a MTS classification problem. The most accurate state-of-the-art MTS classifier on average on the UEA datasets [Bagnall et al., 2018] is a deep learning approach (MLSTM-FCN [Karim et al., 2019]). Chapter 7 presents that XEM ensemble method outperforms MLSTM-FCN, driven by its better performance on the small datasets (relatively to the public UEA archive - training

set size $< 500$), which represent 77% of the benchmark datasets. However, interval-based MTS classifiers like XEM could not be evaluated on the estrus detection application due to the short time windows considered (4 days with one timestamp per day). Hence, I propose a new explainable deep learning approach for multivariate time series classification that would perform well on both the large and small datasets, and particularly on the estrus detection application.

The current state-of-the-art deep learning approach for MTS classification - MLSTM-FCN, consists of the concatenation of a Long Short-Term Memory (LSTM) block with a Convolutional Neural Network (CNN) block composed of 3 convolutional sub-blocks. This approach contains an important number of trainable parameters which could be a significant reason of its poor performance on small datasets. In addition, MLSTM-FCN cannot provide perfectly faithful explanations as it can only rely on post-hoc model-agnostic explainability methods. CNNs along with post-hoc model-specific saliency methods like Gradient-weighted Class Activation Mapping - Grad-CAM [Selvaraju et al., 2019] have the potential to have a compact architecture while enabling faithful explanations at all granularity levels. As presented in section 2.3.2, Grad-CAM provides explanations for an individual instance and can offer global explainability by averaging the attribution maps values per class. A recent CNN, MTEX-CNN [Assaf et al., 2019], proposes to use 2D and 1D convolution filters in sequence to extract key MTS information, i.e. information relative to the observed variables and time, respectively. However, as confirmed by my experiments, the features related to time which are extracted from the output features of the first stage (relative to each observed variable) cannot fully incorporate the timing information from the input data, and subsequently yield poor performance compared to the state-of-the-art MTS classifiers. In addition, the significant number of trainable parameters of MTEX-CNN affects its generalization ability on small datasets. Finally, MTEX-CNN requires upsampling processes on feature maps when applying Grad-CAM, which can lead to an imprecise identification of the regions of the input data that are important for predictions.

Therefore, I propose an end-to-end new compact and eXplainable Convolutional neural network for Multivariate time series classification (XCM), which performs the extraction of information relative to the observed variables and timestamps in parallel and directly from the input data. XCM architecture enables faithful and precise identification of the observed variables and timestamps of the input data that are important for predictions based on the post-hoc model-specific method Grad-CAM.

### 9.1.1 Contributions

In this chapter, the study will:

- Present XCM, a new eXplainable Convolutional neural network for MTS classification;
- Show that XCM outperforms the state-of-the-art MTS classifiers on both the large and small UEA datasets [Bagnall et al., 2018];
- Illustrate the performance and explainability of XCM on a small synthetic dataset and the smallest public UEA dataset. The results demonstrate that XCM enables a more precise identification of the regions of the input data that are important for predictions compared to the current faithfully explainable deep learning MTS classifier MTEX-CNN;
- Show that XCM outperforms LCE and the current most accurate state-of-the-art algorithm on the dairy resource monitoring real-world application while enhancing explainability by providing faithful and more informative explanations.

## 9.2 XCM

In this section I present a new eXplainable Convolutional neural network for Multivariate time series classification (XCM) which extracts in parallel information relative to time and observed variables, before performing the prediction based on the concatenated features. The first part details the architecture of the network and the second part explains how XCM can provide explanations by identifying the observed variables and timestamps of the input data that are important for predictions.

### 9.2.1 Architecture

My approach aims to design a new compact and explainable CNN architecture that performs well on both the large and small UEA datasets. As illustrated in Figure 9.1, a recent explainable CNN, MTEX-CNN [Assaf et al., 2019], proposes to use 2D and 1D convolution filters in sequence to extract key MTS information, i.e. information relative to the observed variables and time, respectively. However, CNN architectures like MTEX-CNN have significant limitations. The use of 2D and 1D convolution filters in sequence means that the features related to time (features maps from 1D convolution filters) are

extracted from the processed features related to observed variables (features maps from 2D convolution filters). Therefore, features related to time cannot fully incorporate the timing information from the input data, and can only partially reflect the necessary information to discriminate between the different classes. Thus, my approach XCM extracts both features related to time (1D convolution filters) and observed variables (2D convolution filters) directly from the input data, which leads to more discriminative features by incorporating all the relevant information and ultimately to a better classification performance on average than the 2D/1D sequential approach (see results in section 9.4.1). Then, a CNN architecture using fully connected layers to perform classification, especially with the size of the first layer depending on the time series length as in MTEX-CNN, is prone to overfitting and can lead to the explosion of the number of trainable parameters. So, the output feature maps of XCM are processed with a 1D global average pooling before being input to a softmax layer for classification. The use of 1D global average pooling followed by a softmax layer for classification reduces the number of parameters and improves the generalization ability of the network compared to fully connected layers. Global average pooling consists in summarizing each feature map by its average. Such operation improves the generalization ability of the network as it does not have parameter to train and it provides robustness to spatial translations of the input [Lin et al., 2014]. In the possible cases when the sequences of events in a MTS change, the robustness to spatial translation ensures that the classification result is not modified. Finally, the use of non-fully padded convolution filters as in MTEX-CNN can lead to an imprecise identification of the regions of the input data that are important for predictions as Grad-CAM is sensitive to upsampling processes. Therefore, the 2D and 1D convolution filters of XCM are fully padded. As detailed in the next section, the output feature maps can then be analyzed with Grad-CAM explainability method without altering the precision of the explanations through upsampling processes. Figure 9.2 illustrates XCM and the following paragraphs detail the architecture.

Firstly, XCM extracts information relative to the observed variables with 2D convolution filters (upper green part in Figure 9.2). This upper part is composed of one 2D convolutional block which is then converted to one feature map to reduce the number of parameters with a $1 \times 1$ convolution filter. The convolutional block contains a 2D convolution layer followed by a batch normalization layer [Ioffe et al., 2015] and a ReLU activation layer [Nair et al., 2010]. I set the kernel size of the 2D convolution filters to $Window\,Size \times 1$, where $Window\,Size$ is a hyperparameter which specifies the time win-

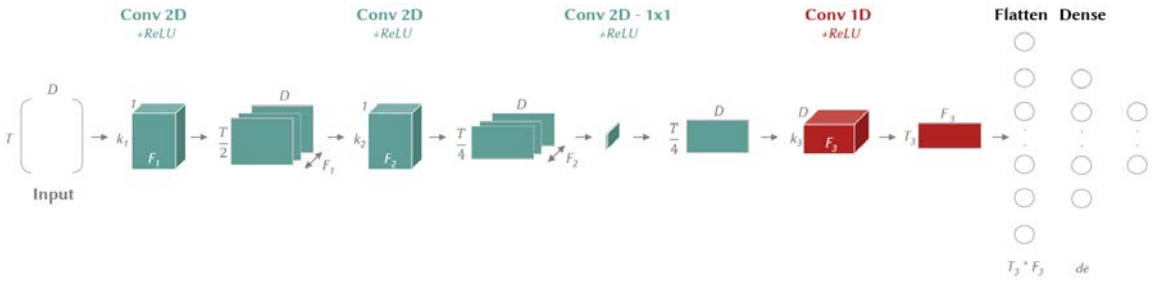



Figure 9.1 – MTEX-CNN architecture. Abbreviations: $D$ - number of observed variables, $de$ - dense layer size, $F$ - number of filters, $k$ - kernel size, $T$ - time series length.

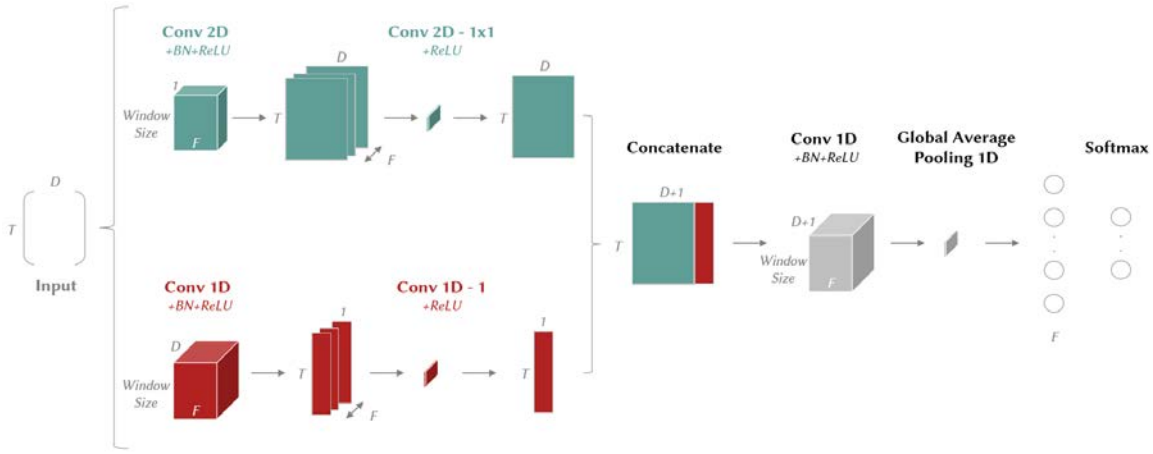



Figure 9.2 – XCM architecture. Abbreviations: $BN$ - Batch Normalization, $D$ - number of observed variables, $F$ - number of filters, $T$ - time series length, *Window Size* - kernel size which corresponds to the time window size.

dow size, i.e. the size of the subsequence of the MTS expected to be interesting to extract discriminative features, and $\times 1$ means for each observed variable. Thus, these 2D convolution filters (number: $F$ in Figure 9.2) allow the extraction of features per observed variable. The features are extracted using a sliding window (strides equal to 1) and I use padding instead of half padding to keep the dimension of the feature maps the same as the input data. The padding allows us to avoid using upsampling and interpolation methods on the features maps when building the *attribution maps*, i.e. the heatmaps of dimensions $T \times D$ that identify the regions of the input data that are important for predictions (detailed in the next section). Then, batch normalization brings normalization at layer level, it enables faster convergence and better generalization of the network [Bjorck et al., 2018]. And, the ReLU activation layer induces non-linearity in the network. Next, the output feature maps are fed into a module ($1 \times 1$ convolution filter) [Szegedy et al., 2015] which reduces the number of parameters. It projects the feature maps into one following a channel-wise pooling.

In parallel, XCM extracts information relative to time with 1D convolution filters (lower red part in Figure 9.2). This lower part is the same as the upper part, except that the 2D convolution filters are replaced by 1D. I set the kernel size of the 1D convolution filters to $Window\,Size \times D$, where $Window\,Size$ is the same hyperparameter as 2D convolution filters and $D$ is the number of observed variables of the input data. The 1D convolution filters slide over the time axis only (stride equals to 1) and capture the interaction between the different time series. Following the use of padding, the output feature map of this lower part has a dimension of $T \times 1$, with $T$ the time series length of the input data. The use of padding, similar to 2D convolution filters, allows us to avoid using upsampling of the features maps on the dimension related to the information extracted (time - $T$) when building the *attribution maps* (detailed in the next section).

In the following step, the output feature maps from these two parts are concatenated and form a feature map of dimensions $T \times (D + 1)$. I apply the same 1D convolution block (1D convolution layer - $F$ filters, kernel size $Window\,Size \times (D + 1)$, stride 1 and padding + batch normalization + ReLU activation layer) as presented in the previous paragraph to slide over the time axis and capture the interaction between the features extracted. Finally, I add a 1D global average pooling on the output feature maps and perform classification with a softmax layer. As previously introduced, the use of global average pooling instead of fully connected layers improves the generalization ability of the network.

In order to assess the potential advantage of concatenating the 2D and 1D convolution blocks instead of having them in sequence, independently from the choice of the classification layers (fully connected layers as in MTEX-CNN versus 1D global average pooling with a softmax layer in XCM), I include in the experiments in section 9.4.1 a variant of XCM (XCM-Seq). XCM-seq is the same as XCM except that the 2D and 1D convolution blocks are in sequence. The next section presents how the architecture of XCM allows the communication of explanations supporting the model predictions with Grad-CAM.

### 9.2.2 Explainability

The new CNN architecture of XCM has been designed to enable the precise identification of the observed variables and timestamps that are important for predictions by the use of Gradient-weighted Class Activation Mapping (Grad-CAM) [Selvaraju et al., 2019]. Grad-CAM identifies the regions of the input data that are important for predictions in convolutional neural networks using the class-specific gradient information. More specifically, Grad-CAM can output two types of attribution maps from XCM architecture: one related to observed variables and another one related to time. Attribution maps are heatmaps of the same size as the input data where some colors indicate features that contribute positively to the activation of the target output [Ancona et al., 2018]. These attribution maps constitute the explanations provided to support XCM model predictions and are available at sample level. The following paragraphs explain how Grad-CAM is used on XCM.

In order to build the first attribution map related to observed variables, Grad-CAM is applied to the output feature maps of the 2D convolution layer which uses convolution filters per observed variable (first block in the upper green part in Figure 9.2). To obtain the class-discriminative attribution map, $L^c_{2D} \in \mathbb{R}^{T \times D}$ with $T$ the time series length and $D$ the number of observed variables, I first compute the gradient of the score for class $c$ ($y_c$) with respect to feature map activations $A^k$ of the convolutional layer, i.e. $\frac{\partial y_c}{\partial A^k}$ with $k \in [1, ..., F]$ the identifier of the feature map. These gradients flowing back are global-average-pooled over the time series length ($T$) and observed variables ($D$) dimensions (indexed by i and j respectively) to obtain the weight of each feature map. Thus, as regards the feature map $k$, the weight is calculated as:

$$w^c_k = \frac{1}{T \times D} \sum_i \sum_j \frac{\partial y_c}{\partial A^k_{ij}}$$

I then use the weights to compute a weighted combination between all the feature maps for that particular class, and use a ReLU to keep only the positive attributions to the predictions.

$$L_{2D}^{c} = ReLU\left(\sum_{k} w_k^c \, A^k\right)$$

The second attribution map, $L_{1D}^{c}$, relates to time and is built on the same principle. Grad-CAM is applied to the output feature maps of the 1D convolution layer which uses convolution filters sliding over the time axis (first block in the lower red part in Figure 9.2). With respect to the feature maps activations $M$ and the class $c$, $L_{1D}^{c}$ is calculated as:

$$q_k^c = \frac{1}{T}\sum_{i} \frac{\partial y_c}{\partial M_i^k}$$

$$L_{1D}^{c} = ReLU\left(\sum_{k} q_k^c \, M^k\right)$$

Thus, $L_{1D}^{c}$ has $T \times 1$ as dimensions. I then upsample it to match the input data dimensions $T \times D$ with a bilinear interpolation in order to obtain the attribution map. This operation does not alter the time attribution results as the padding on the 1D convolution filters ensured that the feature extraction over the time dimension has kept the time series length. Therefore, the upsampling only replicates the results over the observed variables. Example of observed variables and time attribution maps on a synthetic and a public dataset are presented in section 9.4.2.

Before discussing the performance and explainability results of XCM, I present in the next section the evaluation setting.

## 9.3 Evaluation

In this section, I introduce the methodology and datasets used for evaluating the approach.

### 9.3.1 Datasets

XCM is benchmarked on the 30 currently available public UEA MTS datasets [Bagnall et al., 2018]. For each dataset, I keep the train/test split provided in the archive.

### 9.3.2 Algorithms

I implemented XCM with Keras in Python 3.6 and I adopted the following setting:

- 2D convolution layers: 128 feature maps, kernel size: $Window\,Size \times 1$ (see hyper-parameters in section 9.3.3 for the time window size), strides $1 \times 1$, padding *same* and ReLU activation;
- 1D convolutions layers: 128 feature maps, kernel size: time window size (see hyper-parameters in section 9.3.3), strides 1, padding *same* and ReLU activation.

The state-of-the-art MTS classifiers can be categorized into four families: similarity-based, feature-based, deep learning methods and ensemble methods. In this work I choose to benchmark XCM to the best-in-class for each similarity-based, feature-based, deep learning and ensemble method category ($\text{DTW}_I$, WEASEL+MUSE, MLSTM-FCN and XEM classifiers). I also include MTEX-CNN in the benchmark, which is the network from which the new architecture is derived and which has not been evaluated on the public UEA datasets. Therefore, XCM algorithm is compared to the following MTS classifiers:

- $\text{DTW}_D$ without and with normalization (n): the one nearest neighbor classifier with DTW distance based on multi-dimensional points instead of treating each dimension separately. I report the results published in the UEA archive [Bagnall et al., 2018];
- $\text{DTW}_I$ without and with normalization (n): the one nearest neighbor classifier based on the sum of DTW distance for each dimension. I report the results published in the UEA archive [Bagnall et al., 2018];
- ED without and with normalization (n): the one nearest neighbor classifier with Euclidean distance. I report the results published in the UEA archive [Bagnall et al., 2018];
- MLSTM-FCN [Karim et al., 2019]: I report the results presented in chapter 7. This study uses the implementation available [1] and runs it with the setting recommended by the authors in the paper (128-256-128 filters, 250 training epochs, a dropout of 0.8 and a batch size of 128);
- MTEX-CNN [Assaf et al., 2019]: I have implemented the algorithm with Keras in python 3.6 based on the description of the paper. I use the setting recommended by the authors (Stage 1: two convolution layers with half padding and ReLU activation, kernel sizes $8 \times 1$ and $6 \times 1$, strides $2 \times 1$, feature maps 64 and 128, dropout 0.4.

1. https://github.com/houshd/MLSTM-FCN

Stage 2: one convolution layer with ReLU activation, strides 2, kernel size 2, feature maps 128, dropout 0.4. Dense layer dimension 128 and L2 regularization 0.2);

- WEASEL+MUSE [Schäfer et al., 2017]: I report the results presented in chapter 7. This study uses the implementation available [2] and runs it with the setting recommended by the authors in the paper (SFA word lengths l in [2,4,6], windows length in [4:max(MTS length)], chi=2, bias=1, p=0.1, c=5 and a solver equals to L2R LR DUAL);
- XCM-Seq: XCM variant with 2D and 1D convolution blocks in sequence (see description in section 9.2.1). I use the same setting as XCM;
- XEM: I report the results presented in chapter 7.

All the networks in the experiments (XCM, XCM-Seq and MTEX-CNN) are trained with 100 epochs. The models are compiled with the categorical cross-entropy loss and the Adam optimization.

### 9.3.3 Hyperparameters

The first hyperparameter of XCM is *Batch Size* and the range is $[1, 8, 32]$. The second hyperparameter of XCM is *Window Size* (the time window size - kernel size). It is expressed as a percentage of the total size of the MTS and the range of time window size percentages is $[20\%, 40\%, 60\%, 80\%, 100\%]$. For each dataset, hyperparameters are set by grid search based on the best average accuracy of XCM following a stratified 5-fold cross-validation on the training set.

### 9.3.4 Metrics

For each dataset, the classification accuracy is computed. Then, the average rank and the number of wins/ties are presented to compare the different classifiers on the same datasets. Finally, I present the critical difference diagram [Demšar, 2006], the statistical comparison of multiple classifiers on multiple datasets based on the non-parametric Friedman test, to show the overall performance of XCM. I use the implementation available in R package scmamp.

2. https://github.com/patrickzib/SFA

## 9.4 Results

In this section I first present the performance results of XCM on the public UEA datasets. Then, I illustrate how XCM can reconcile performance and explainability on a synthetic and a public dataset. Finally, I end this section by showing that XCM outperforms the current state-of-the-art algorithm on the dairy resource monitoring real-world application while providing faithful and more informative explanations.

### 9.4.1 Performance

The accuracy results on the public UEA test sets of XCM and the other MTS classifiers are presented in Table 9.1. A blank in the table indicates that the approach ran out of memory or the accuracy is not reported. The best accuracy for each dataset is denoted in boldface.

Firstly, we can observe that XCM obtains the best average rank (2.7), followed by XEM in second position (rank: 3.4) and MLSTM-FCN in third position (rank: 4.1). Using the categorization of the datasets published in the archive website [3], we do not see any influence from the different train set sizes, MTS lengths, number of dimensions, number of classes and dataset types on XCM performance relative to the other classifiers on the UEA datasets.

More specifically, XCM exhibits better performance than XEM and MLSTM-FCN on both large (rank: 2.3, XEM rank: 3.1, MLSTM-FCN rank: 2.7 - train size $\geq$ 500, 23% of the datasets) and small datasets (rank: 2.9, XEM rank: 3.5, MLSTM-FCN rank: 4.5 - train size $<$ 500, 77% of the datasets). We can assume that the more compact architecture of XCM compared to the other deep learning classifiers provides a better generalization ability on the UEA datasets (average rank on the number of trainable parameters: XCM 1.7, MLSTM-FCN: 1.9, MTEX-CNN: 2.0). Furthermore, the results confirm the superiority of XCM approach based on the extraction in parallel and directly from the input data of features relative to the observed variables and time compared to the sequential approaches. XCM outperforms both XCM-Seq and MTEX-CNN on average on the UEA datasets (rank: 2.7, XCM-Seq: 5.6, MTEX-CNN: 8.1).

With regard to the hyperparameter $Window\,Size$ of XCM, Figure 9.3 shows the average relative drop in performance across the datasets when using the other time window sizes than the one used in the best configuration given in Table 9.1. In order to evaluate

3. http://www.timeseriesclassification.com/dataset.php

Table 9.1 – Accuracy results on the UEA MTS datasets. Abbreviations: BS - Batch Size, ED - ED, DW - DTW, MC - MTEX-CNN, MF - MLSTM-FCN, Win % - Window Size, WM - WEASEL+MUSE, X - XCM, XM - XEM, XS - XCM-Seq

| **Datasets** | **X** | **XS** | **MC** | **XM** | **MF** | **WM** | **ED** | **DW I** | **DW D** | **ED (n)** | **DW I (n)** | **DW D (n)** | **X Params** BS | Win % |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Articulary Word Recognition | 98.3 | 92.7 | 92.3 | **99.3** | 98.6 | **99.3** | 97.0 | 98.0 | 98.7 | 97.0 | 98.0 | 98.7 | 32 | 80 |
| Atrial Fibrilation | **46.7** | 33.3 | 33.3 | **46.7** | 20.0 | 26.7 | 26.7 | 26.7 | 20.0 | 26.7 | 26.7 | 22.0 | 1 | 60 |
| Basic Motions | **100** | **100** | **100** | **100** | **100** | **100** | 67.5 | **100** | 97.5 | 67.6 | **100** | 97.5 | 32 | 20 |
| Character Trajectories | **99.5** | 98.8 | 97.4 | 97.9 | 99.3 | 99.0 | 96.4 | 96.9 | 99.0 | 96.4 | 96.9 | 98.9 | 32 | 80 |
| Cricket | **100** | 93.1 | 90.3 | 98.6 | 98.6 | 98.6 | 94.4 | 98.6 | **100** | 94.4 | 98.6 | **100** | 32 | 20 |
| Duck Duck Geese | **70.0** | 52.5 | 65.0 | 37.5 | 67.5 | 57.5 | 27.5 | 55.0 | 60.0 | 27.5 | 55.0 | 60.0 | 8 | 80 |
| Eigen Worms | 43.5 | 45.0 | 41.9 | 52.7 | 80.9 | **89.0** | 55.0 | 60.3 | 61.8 | 54.9 |  | 61.8 | 32 | 40 |
| Epilepsy | **99.3** | 93.5 | 94.9 | 98.6 | 96.4 | **99.3** | 66.7 | 97.8 | 96.4 | 66.6 | 97.8 | 96.4 | 32 | 20 |
| Ering | 13.3 | 13.3 | 13.3 | **20.0** | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 13.3 | 32 | 20 |
| Ethanol Concentration | 34.6 | 31.6 | 30.8 | **37.2** | 27.4 | 31.6 | 29.3 | 30.4 | 32.3 | 29.3 | 30.4 | 32.3 | 32 | 80 |
| Face Detection | **63.9** | 63.8 | 50.0 | 61.4 | 55.5 | 54.5 | 51.9 | 51.3 | 52.9 | 51.9 |  | 52.9 | 32 | 60 |
| Finger Movements | 60.0 | 60.0 | 49.0 | 59.0 | **61.0** | 54.0 | 55.0 | 52.0 | 53.0 | 55.0 | 52.0 | 53.0 | 32 | 40 |
| Hand Movement Direction | 44.6 | 40.1 | 18.9 | **64.9** | 37.8 | 37.8 | 27.9 | 30.6 | 23.1 | 27.8 | 30.6 | 23.1 | 32 | 80 |
| Handwriting | 41.2 | 38.6 | 24.6 | 28.7 | 54.7 | 53.1 | 37.1 | 50.9 | **60.7** | 20.0 | 31.6 | 28.6 | 32 | 60 |
| Heartbeat | **77.6** | 74.1 | 72.2 | 76.1 | 71.4 | 72.7 | 62.0 | 65.9 | 71.7 | 61.9 | 65.8 | 71.7 | 32 | 80 |
| Insect Wingbeat | 10.5 | 10.5 | 10.5 | **22.8** | 10.5 |  | 12.8 |  | 11.5 | 12.8 |  |  | 32 | 20 |
| Japanese Vowels | 98.6 | 94.6 | 95.1 | 97.8 | **99.2** | 97.8 | 92.4 | 95.9 | 94.9 | 92.4 | 95.9 | 94.9 | 32 | 80 |
| Libras | 84.4 | 79.4 | 81.1 | 77.2 | **92.2** | 89.4 | 83.3 | 89.4 | 87.2 | 83.3 | 89.4 | 87.0 | 32 | 80 |
| LSST | 61.2 | 54.2 | 31.5 | **65.2** | 64.6 | 62.8 | 45.6 | 57.5 | 55.1 | 45.6 | 57.5 | 55.1 | 32 | 100 |
| Motor Imagery | 54.0 | 53.0 | 50.0 | **60.0** | 53.0 | 50.0 | 51.0 | 39.0 | 50.0 | 51.0 |  | 50.0 | 8 | 40 |
| NATOPS | **97.8** | 93.9 | 88.3 | 91.6 | 96.1 | 88.3 | 85.0 | 85.0 | 88.3 | 85.0 | 85.0 | 88.3 | 32 | 40 |
| PenDigits | 99.1 | 96.7 | 87.8 | 97.7 | 98.7 | 96.9 | 97.3 | 93.9 | 97.7 | 97.3 | 93.9 | 97.7 | 8 | 60 |
| PEMSF | 75.7 | 80.9 | 11.6 | **94.2** | 65.3 |  | 70.5 | 73.4 | 71.1 | 70.5 | 73.4 | 71.1 | 32 | 80 |
| Phoneme | 22.5 | 11.9 | 2.6 | **28.8** | 27.5 | 19.0 | 10.4 | 15.1 | 15.1 | 10.4 | 15.1 | 15.1 | 32 | 40 |
| Racket Sports | 89.5 | 86.8 | 82.9 | **94.1** | 88.2 | 91.4 | 86.4 | 84.2 | 80.3 | 86.8 | 84.2 | 80.3 | 32 | 80 |
| Self Regulation SCP1 | **87.8** | 81.6 | 78.5 | 83.9 | 86.7 | 74.4 | 77.1 | 76.5 | 77.5 | 77.1 | 76.5 | 77.5 | 32 | 80 |
| Self Regulation SCP2 | 54.4 | **55.0** | 50.0 | **55.0** | 52.2 | 52.2 | 48.3 | 53.3 | 53.9 | 48.3 | 53.3 | 53.9 | 32 | 80 |
| Spoken Arabic Digits | **99.5** | 99.4 | 98.6 | 97.3 | 99.4 | 98.2 | 96.7 | 96.0 | 96.3 | 96.7 | 95.9 | 96.3 | 32 | 80 |
| Stand Walk Jump | 40.0 | 46.7 | **53.3** | 40.0 | 46.7 | 33.3 | 20.0 | 33.3 | 20.0 | 20.0 | 33.3 | 20.0 | 32 | 60 |
| U Wave Gesture Library | 89.3 | 81.9 | 81.2 | 89.7 | 85.7 | **90.3** | 88.1 | 86.9 | **90.3** | 88.1 | 86.8 | **90.3** | 32 | 100 |
| Average Rank | 2.7 | 5.6 | 8.1 | 3.4 | 4.1 | 4.6 | 8.0 | 6.7 | 5.5 | 8.3 | 7.2 | 6.1 |  |  |
| Wins/Ties | 12 | 2 | 2 | 13 | 4 | 5 | 0 | 1 | 3 | 0 | 1 | 2 |  |  |




Figure 9.3 – XCM average relative accuracy drop across the UEA datasets when using other time window sizes than the one used in the best configuration given in Table 9.1. The performance drop is presented across four categories of datasets, defined according to XCM levels of accuracy shown in Table 9.1. Abbreviation: Acc - Accuracy.

the relative impact with respect to the range of performance, I have defined four categories of datasets: datasets with XCM original accuracy$<$ 50%, datasets with 50% $\leq$ accuracy $<$ 75%, datasets with 75% $\leq$ accuracy $<$ 90% and datasets with accuracy $\geq$ 90%. First, as expected, we can observe that the average relative impact of using suboptimal time window sizes is higher when XCM level of performance is low (average relative drop in accuracy: 13.1% when XCM accuracy $<$ 50% versus 3.0% when XCM accuracy $\geq$ 90%). Then, the average relative drop in accuracy when using suboptimal time window sizes is not negligible but remains limited in all the cases. This drop is below 15% on average on the category where XCM has the lowest level of accuracy (13.1% $\pm$ 3.2%) and below 10% on average across all the datasets (7.0% $\pm$ 1.3%).

Finally, a statistical test is performed to evaluate the performance of XCM compared to the other MTS classifiers. I present in Figure 9.4 the critical difference plot with alpha equals to 0.05 from results shown in Table 9.1. The values correspond to the average rank and the classifiers linked by a bar do not have a statistically significant difference. The plot confirms the top 3 ranking as presented before (XCM: 1, XEM: 2, MLSTM-FCN: 3), without showing a statistically significant difference between each other. We can notice that XCM is the only classifier with a significant performance difference compared to $\text{DTW}_D$.

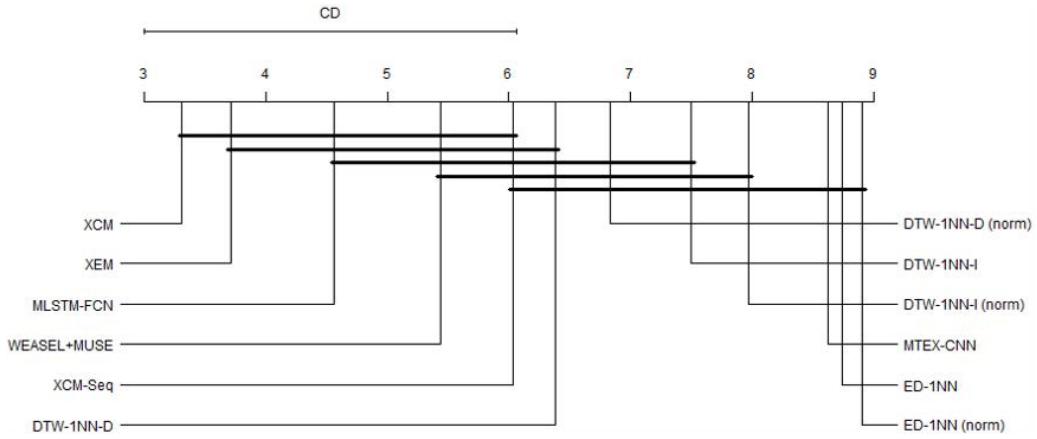



Figure 9.4 – Critical difference plot of the MTS classifiers on the UEA datasets with alpha equals to 0.05.

### 9.4.2 Explainability

In this section, I illustrate how XCM approach reconciles performance and explainability, and show that XCM enables a more precise identification of the regions of the input data that are important for predictions compared to the current faithfully explainable deep learning MTS classifier MTEX-CNN. The precision of the explanations provided by Grad-CAM, i.e. the fraction of explanations that are relevant to a prediction, can vary across CNN architectures as Grad-CAM is sensitive to the upsampling processes on feature maps to match the input data dimensions. There is no quantitative approach to assess a model explainability. Therefore, I adopt a qualitative approach to analyze XCM explainability. I illustrate XCM explainability on a synthetic dataset and a public one from the UEA archive. I choose the same synthetic and public dataset (Atrial Fibrilation) as in XEM study in chapter 7. Thus, it offers a basis of comparison with the current best performing MTS classifier, XEM, which provides explainability by design through the identification of the time window used to classify the whole MTS. Therefore, I compare XCM explainability on the synthetic and public datasets to XEM and MTEX-CNN. As presented in [Fawaz et al., 2019], it would also be interesting to extend the explainability analysis to the other state-of-the-art deep learning approaches (FCN, ResNet) and compare the results based on Class Activation Mapping (CAM) [Zhou et al., 2016]. CAM can also be applied to XCM as XCM uses global average pooling followed by a softmax layer for classification. However, in the current configuration, FCN/ResNet/XCM use 1D convolution filters in their last layer so I do not include a study with CAM in this section

as CAM results would not provide information relative to both the observed variables and time. In future work, it would be interesting to analyze the performance of these networks on MTS using 2D convolution filters in the last layer and the associated CAM results.

**Synthetic Dataset**

Firstly, I show that XCM uses and identifies the expected time window to perform the classification on a MTS synthetic dataset and compare the results to XEM and MTEX-CNN. The synthetic dataset is composed of 20 MTS (50%/50% train/test split) with a length of 100, 2 dimensions, 2 balanced classes. The difference between the 10 MTS belonging to the negative class and the one belonging to the positive class stems from a 20% time window of the MTS. Negative class MTS are sine waves and, as illustrated in the plot on the top part of Figure 9.5, positive class MTS are sine waves with a square signal on 20% of the dimension 1 (see timestamps between 60 and 80).

First, XEM, MTEX-CNN and XCM (*Batch Size*: 1, *Window Size*: 20%) correctly predict the 10 MTS of the test set (accuracy 100%). We can observe that XCM and MTEX-CNN obtain the same performance whereas XCM has around 10 times fewer parameters than MTEX-CNN (trainable parameters: XCM 17k, MTEX-CNN 232k).

Moreover, XEM, MTEX-CNN with Grad-CAM and XCM with Grad-CAM all correctly identify the discriminative time window. However, as shown in Figure 9.5, the attribution maps of MTEX-CNN and XCM with the same explainability method (Grad-CAM) are slightly different. Figure 9.5 shows one MTS sample belonging to the class positive, and the time and observed variables attribution maps supporting the correct MTEX-CNN and XCM predictions. Attribution maps are heatmaps of the same size as the input data. The more intense the red, the stronger the features (observed variables, time) positively contribute to the prediction. We can observe that the attribution maps drawn from XCM are more precise than the one from MTEX-CNN. On the time attribution map, high attribution values (above 0.6) for XCM begin on timestamp 63 and end on timestamp 76 (expected: $[60, 80]$), whereas for MTEX-CNN they begin later (timestamp 68). Concerning the attribution map of the observed variables, as expected we see that high attributions values on the discriminative dimension (dimension 1) appears at the same timestamps as high attribution values on the time attribution map for XCM (timestamps 63 and 76). Nonetheless, the observed variables attribution map of MTEX-CNN shows high attribution values on a window larger than the discriminative one (timestamps range $[34, 83]$). As MTEX-CNN attribution maps exhibit a red color gradient, the precision of

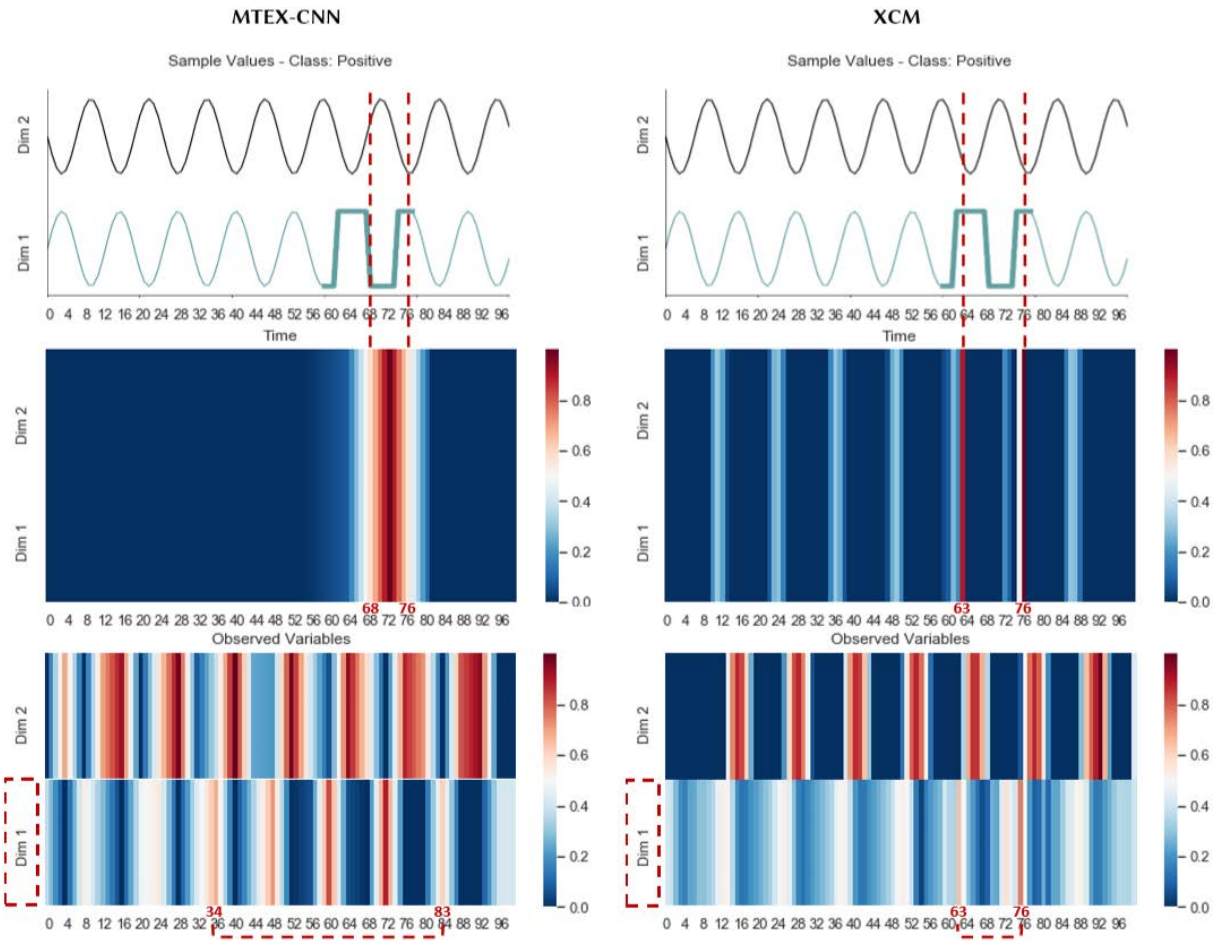



Figure 9.5 – Observed variables and time attribution maps supporting the correct MTEX-CNN and XCM predictions of a MTS from the synthetic dataset belonging to the class *Positive*. Abbreviation: Dim - Dimension.

identification of the regions of the input data on MTEX-CNN attribution maps could be enhanced by setting a higher threshold than 0.6 for the attribution values. However, the red color gradient is due to the upsampling processes needed to match the 2D/1D output features maps of MTEX-CNN to the size of the input data when applying Grad-CAM. Grad-CAM is applied at local level, which means that we would need to potentially set a different threshold for each instance and that would render MTEX-CNN explainability method impractical. So, based on the same attribution threshold (0.6), XCM allows a more precise identification of the regions of the input data that are important for predictions than MTEX-CNN. Both MTEX-CNN and XCM have periodically high attribution values on the dimension 2 of the observed variables attribution maps. It could be surprising as the sinusoidal signal on this dimension is the same across all MTS, however the fact that this information is uniformly high or low renders it irrelevant for explanations. Therefore, considering that XCM-Seq attributions maps are the same as XCM ones, we can assume that the use of half padding on the different convolution layers to reduce the number of parameters in MTEX-CNN, so the use of upsampling to retrieve the input data dimensions on the attribution maps, can lead to a less precise identification of the regions of the input data that are important for predictions.

Finally, on the synthetic dataset and as shown in section 7.4.2, XEM explanations correctly identify the time window $[60, 80]$, while XCM is able to provide the discriminative dimension in addition to the correct identification of the discriminative time window.

#### Atrial Fibrilation UEA Dataset

Next, I illustrate XCM explainability on a public dataset. I choose the Atrial Fibrilation dataset [Bagnall et al., 2018], i.e. the smallest one of the public UEA archive, to show how XCM as a convolutional neural network can also reconcile performance and explainability on small datasets. In addition, this choice is supported by the fact that it is one of the datasets used to illustrate XEM explainability, so it offers a basis of comparison.

First, as shown in Table 9.1, XEM and XCM are the most accurate models on the Atrial Fibrilation dataset (XCM: 46.7%, XEM: 46.7%, MTEX-CNN: 33.3%). XCM exhibits better performance than MTEX-CNN while having around 10 times fewer parameters (XCM: 100k, MTEX-CNN 1.3m).

Then, same as on the synthetic dataset, we can notice that XCM provides more delimited periods with high attribution values compared to MTEX-CNN, which uses half padding on the different convolution layers. Figure 9.6 shows the first MTS sample from

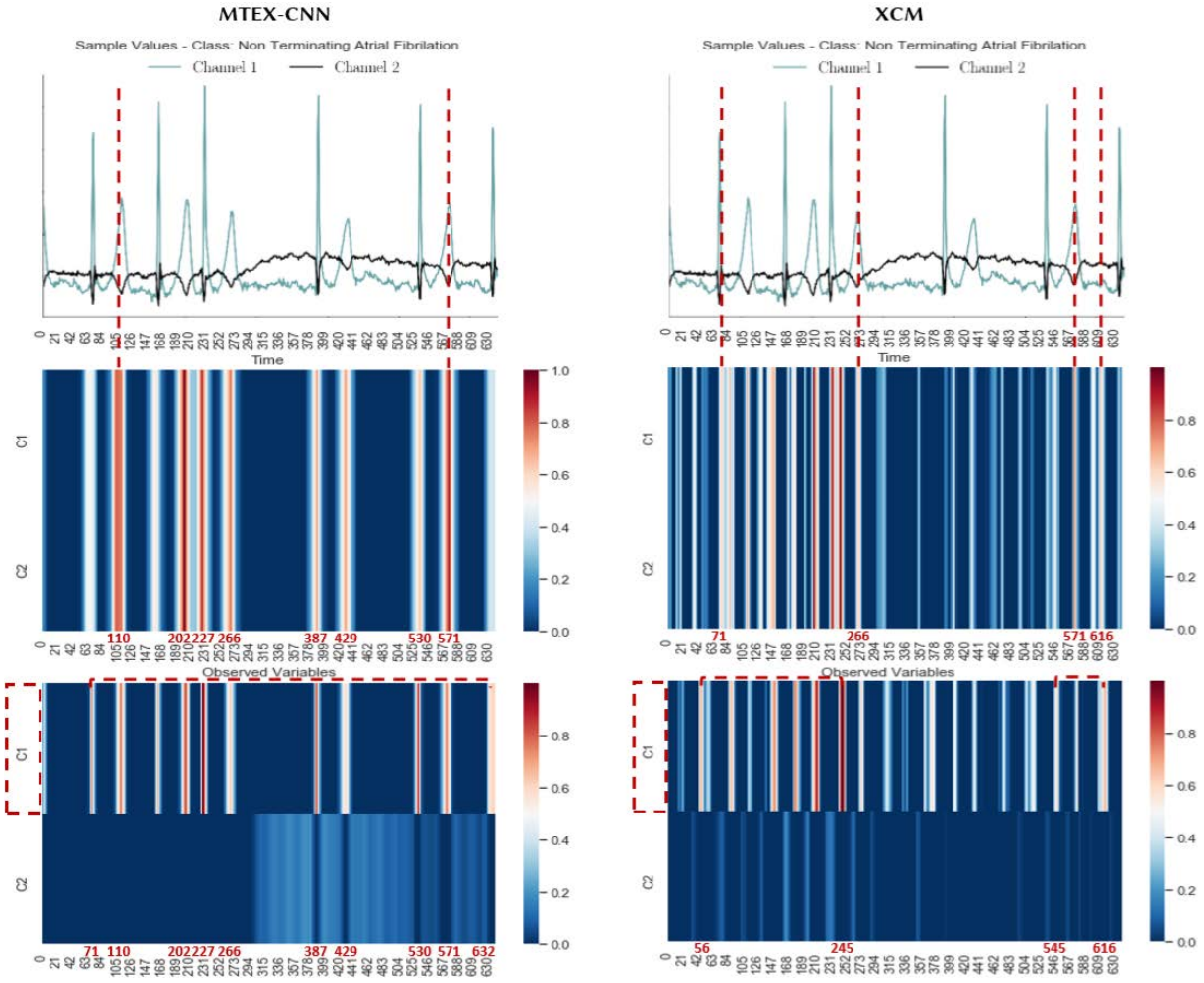



Figure 9.6 – Observed variables and time attribution maps supporting the correct MTEX-CNN and XCM predictions of the first MTS from Atrial Fibrilation test set, which belongs to the class *Non Terminating Atrial Fibrilation*. Abbreviation: C - Channel.

Atrial Fibrilation test set that belongs to the class *Non Terminating Atrial Fibrilation*, and the attribution maps of MTEX-CNN and XCM supporting their predictions. This MTS is composed of two dimensions (two channels ECG) with a length of 640 (5 second period with 128 samples per second). All the models (XEM, MTEX-CNN and XCM) correctly predict the sample class. An additional difficulty in the analysis of explainability results compared to that of the synthetic case is that we do not have the expected discriminative parts of the MTS that should be used to classify the whole MTS. However, we can see in Figure 9.6 that MTEX-CNN time attribution map does not identify particular discriminative time windows as it displays high attribution values (above 0.6) all along the time series on the timestamps corresponding to peaks $\{110, 202, 227, 266, 387, 429, 530, 571\}$, whereas we can identify two periods which strongly contribute to the prediction on XCM attribution map (71-266 and 571-616). Concerning the attribution map of the observed variables, we see that both MTEX-CNN and XCM use channel 1 as discriminative variable; and the same observation from the time attribution map holds on channel 1. Thus, the region of the input data identified by Grad-CAM to support XCM prediction is more precise than that of MTEX-CNN, in the sense that the intersection of XCM attribution maps highlights a subset of the peaks on channel 1 as important, which reduces the amount of information that needs to be evaluated by the end-user.

Finally, compared to XEM which uses a unique time window to classify the whole MTS, XCM can leverage multiple subseries. For example, as presented in section 7.4.1, XEM obtains its best performance on Atrial Fibrilation (accuracy 46.7%) using a 20% time window. On the MTS in Figure 9.6, XEM identifies the time window $[256, 383]$ as discriminative to classify the whole MTS (see Figure 7.9). Thus, XCM uses some timestamps contained in the same time window as XEM plus some others at the end. This could be a reason of XCM better performance than XEM on the UEA datasets.

### 9.4.3 Real-World Application

In this section, I evaluate the new XCM with Grad-CAM approach on the dairy resource monitoring application of this thesis. The evaluation consists in a comparison of the XCM approach with the state-of-the-art MTS classifiers and the previous machine learning methods of the thesis. The next paragraphs present the evaluation setting and the performance-explainability results.

### Evaluation Setting

**Dataset** The same real-world dataset as LCE in chapter 6 and XPM in chapter 8 is employed. In this chapter, I also evaluate the impact of adding the *ration* and the *weight* variables on performance. The delivery of the ration occurs twice daily (9am and 5pm) and contains on average 65% of maize silage, 10% of dehydrated alfalfa pellets and 25% of concentrate. Cows are also weighted twice a day (morning and afternoon). The evaluation method is the same as in chapter 6 and chapter 8: a 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores. The dataset split is presented in Table 9.2.

Table 9.2 – Dataset Split. Abbreviation: Ext Val - External Validation.

| | **Folds** | | | | | | **Ext Val** |
|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **All** | |
| Estrus | 121 | 129 | 113 | 139 | 149 | 651 | 307 |
| Silent % | 32 | 40 | 24 | 40 | 44 | 37 | 43 |

**Algorithms** Performance is a prerequisite in this application. Therefore, I compare XCM with Grad-CAM to the current best performing state-of-the-art MTS classifier MLTSM-FCN with SHAP and the best performing machine learning method on this application from the previous chapters (LCE with SHAP - see chapter 6). MLSTM-FCN can only rely on post-hoc model-agnostic explainability methods to support its predictions. I choose SHAP as its granularity of explanation is comparable to Grad-CAM (both global and local). SHAP provides the relative importance of the observed variables and timestamps on predictions.

### Results

**Performance** The detection results are presented in Table 9.3. We can observe that XCM (F1-score 69.7%) shows better performance on the test sets than the current state-of-the-art deep learning approach MLSTM-FCN (F1-score 63.1%), the best performing machine learning method from the previous chapters LCE (F1-score 67.5%) and the reference commercial solution (F1-score 55.3%) by increasing the average F1-score and obtaining the lowest variability across folds (1.5%). In addition, the same observation from the test sets holds on the external validation (68.1 $\pm$ 0.3, MLSTM-FCN: 64.8 $\pm$ 0.4, LCE: 67.9 $\pm$ 0.2 and CS: 59.6 $\pm$ 0), even if the performance between XCM and LCE is close.

XCM generalizes well by maintaining its level of performance on the external validation set (test: 69.7%, external validation: 68.1%).

Table 9.3 – Average F1-score with corresponding standard error on the activity and temperature dataset. Abbreviations: CS - Commercial Solution, Ext Val - External Validation.

| | **XCM** | **MLSTM-FCN** | **LCE** | **CS** |
|---|---|---|---|---|
| Test | **69.7 ± 1.5** | 63.1 ± 1.5 | 67.5 ± 1.5 | 55.3 ± 5.1 |
| Ext Val | **68.1 ± 0.3** | 64.8 ± 0.4 | 67.9 ± 0.2 | 59.6 ± 0 |

Furthermore, I have evaluated the impact of adding the *ration* and *weight* variables on estrus detection. The results of XCM estrus detection on the different datasets are presented in Table 9.4. We can observe that XCM obtains a slightly better performance on the test sets when the *ration* is included in the datasets (AT-R: 70.3%, AT-RW: 70.0% versus AT: 69.7%). However, this observation does not hold on the external validation. XCM does not exhibit a better detection performance on the external validation set when the ration is included (AT-R: 68.0%, AT-RW: 67.8% versus AT: 68.1%). Therefore, based on my experiments, the interest of adding the *ration* or the *weight* to enhance estrus detection is not demonstrated. Nonetheless, the performance difference is not significant, it would be interesting to further study the potential of ration data for estrus detection with a broader data heterogeneity (cows breed, environment).

Table 9.4 – Average F1-score of XCM with corresponding standard error on the different datasets. Abbreviation: AT - Activity and Temperature, Ext Val - External Validation, R - Ration, W - Weight.

| | **XCM** | | | |
|---|---|---|---|---|
| | **AT** | **AT-R** | **AT-W** | **AT-RW** |
| **Test** | 69.7 ± 1.5 | **70.3 ± 1.3** | 69.6 ± 1.0 | 70.0 ± 1.0 |
| **Ext Val** | **68.1 ± 0.3** | 68.0 ± 1.6 | **68.1 ± 0.3** | 67.8 ± 0.2 |

**Explainability** Concerning the explainability, Figure 9.7 shows an example of the time and observed variables attribution maps supporting the correct XCM prediction of a MTS sample belonging to the class *Estrus*. I plot the MTS sample with an heatmap to ease the readability. The intersection of attribution maps and sample values inform us that the prediction was made mainly based on the presence of a high over activity of the animal on the day of estrus (attribution values above 0.6 on *Day 0* and on the variable *over*

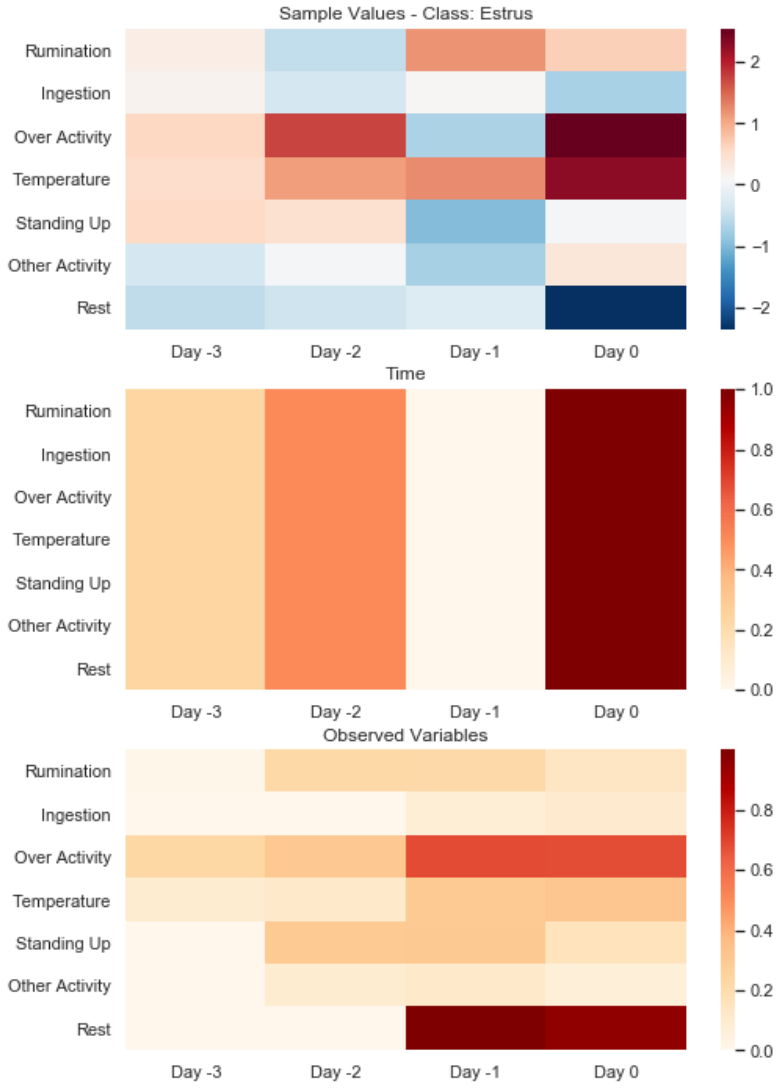



Figure 9.7 – Observed variables and time attribution maps supporting the correct XCM prediction of a MTS from the real-world test set, which belongs to the class *Estrus*.

*activity*, which has a high value). This behavior is aligned with the literature on estrus detection [Gaillard et al., 2016], it is the behavior associated with most of the estrus.

Thus, in addition to giving the relative importance of observed variables and time as LCE and MLSTM-FCN with SHAP, XCM with Grad-CAM provides more informative explanations by supplying the corresponding sample values. Finally, unlike LCE and MLSTM-FCN with SHAP post-hoc model-agnostic method, XCM with Grad-CAM approach provides faithful explanations, which is a prerequisite to reduce solution mistrust from the farmers. Therefore, XCM with Grad-CAM outperforms the current best performing state-of-the-art MTS classifier (MLSTM-FCN) with SHAP and exhibits a slightly better performance than the previous best machine learning method of the thesis (LCE + SHAP) on the real-world application, while enhancing explainability by providing faithful and more informative explanations.

## 9.5 Performance-Explainability Analysis

This section introduces the new CNN MTS classifier XCM with Grad-CAM post-hoc model-specific explainability method into the analytical framework of the thesis (part II). The different aspects of XCM framework are summarized in Table 9.5 and can be visualized in Figure 9.8.

The study of this chapter firstly shows that XCM outperforms the state-of-the-art MTS classifiers and the best performing method from the previous chapters on the public UEA datasets (XEM). Then, it exhibits the superior performance of XCM on the real-world application of estrus detection compared to a commercial reference and the best performing method from the previous chapters (LCE). Therefore, in the framework presented in part II, following a 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset, the performance level of XCM is better than the state-of-the-art (Performance: *Best*).

Concerning the explainability, similar to the ensemble methods, XCM is not comprehensible (Comprehensibility: *Black-Box* - ensemble: *Black-Box*, XPM: *White-Box*). However, XCM provides with Grad-CAM post-hoc model-specific explainability method some faithful (Faithfulness: *Perfect* - DMSEEW: *Perfect*, LCE + SHAP: *Imperfect*, XEM: *Perfect*, XPM: *Perfect*) and informative (Information: *Features+Time+Values* - DMSEEW: *Features*, LCE + SHAP: *Features+Time*, XEM: *Uni Sequences*, XPM: *Uni Sequences*) explanations at all granularity levels (Granularity: *Both Global & Local* - DMSEEW: *Lo-*

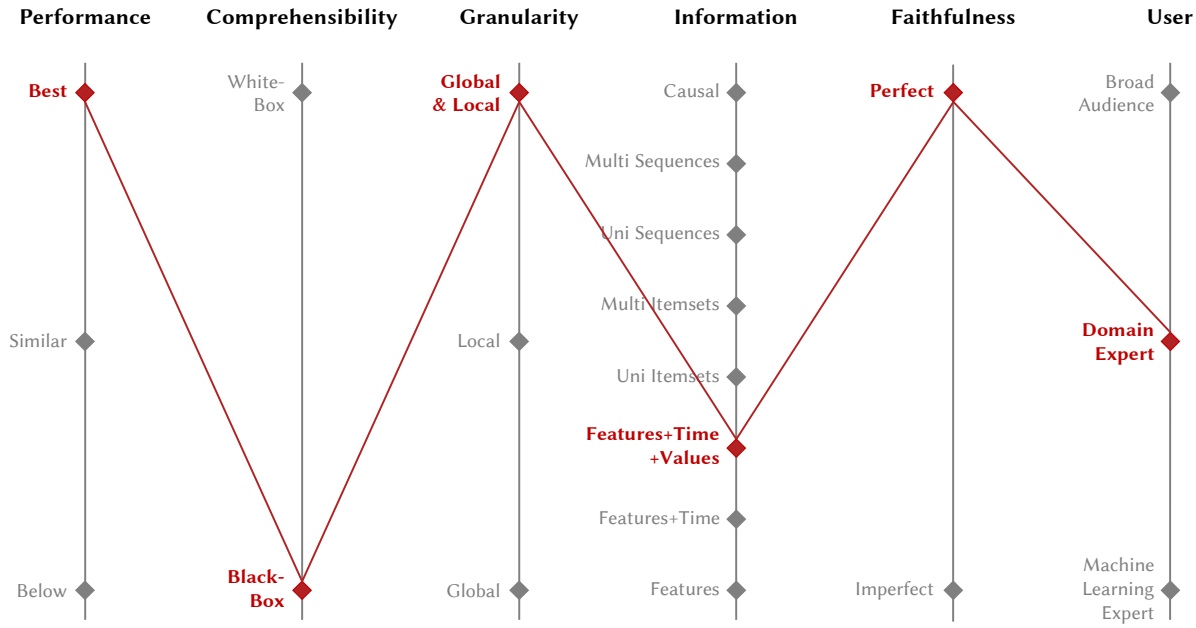



Figure 9.8 – Parallel coordinates plot of XCM with Grad-CAM on the dairy resource monitoring application. Performance evaluation method: 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, FCN, kNN, LC, LCE, MLP, MLSTM-FCN, MTEX-CNN, ResNet, RF, SVM, XCM, XGB and XPM.

*cal*, LCE + SHAP: *Both Global & Local*, XEM: *Local*, XPM: *Both Global & Local*). The explanations inform the end-user about the important observed variables and timestamps with the discriminative values to the predictions. These explanations are accessible to domain experts (e.g. farmers), who are able to analyze the relation between feature high attribution values and feature discriminative values (User: *Domain Expert* - DMSEEW: *Machine Learning Expert*, LCE + SHAP: *Domain Expert*, XEM: *Domain Expert*, XPM: *Broad Audience*).

Table 9.5 – Performance-explainability results of the machine learning methods of the thesis. Abbreviations: Broad - Broad Audience, Domain - Domain Expert, F - Features, G&L - Global & Local, ML - Machine Learning, T - Time, V - Values.

| | **Ensemble Methods** | | | | |
|---|---|---|---|---|---|
| | **DMSEEW** | **LCE + SHAP** | **XEM** | **XPM** | **XCM + Grad-CAM** |
| Performance | Best[1] | Best[2 3] | Best[4] | Similar[5] | Best[6 7] |
| Comprehensibility | Black-Box | Black-Box | Black-Box | White-Box | Black-Box |
| Granularity | Local | Both G&L | Local | Both G&L | Both G&L |
| Information | F | F + T | Uni Sequences | Uni Sequences | F + T + V |
| Faithfulness | Perfect | Imperfect | Perfect | Perfect | Perfect |
| User | ML | Domain | Domain | Broad | Domain |

[1] 3-fold cross-validation and an arithmetic mean of the accuracies on the Earthquake Early Warning Dataset. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, SMTS, UFS and WEASEL+MUSE.

[2] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM and XGB.

[3] 3-fold cross-validation and an arithmetic mean of the accuracies on the UCI datasets. Models evaluated in the benchmark: EN, LC, LCE, MLP, RF, SVM and XGB.

[4] Predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, RFM, SMTS, UFS, WEASEL+MUSE, XEM and XGBM.

[5] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, kNN, LC, LCE, MLP, RF, SVM, XGB and XPM.

[6] 5-fold cross-validation plus external validation and an arithmetic mean of the F1-scores on the INRAE proprietary dataset. Models evaluated in the benchmark: Commercial Solution, EN, FCN, kNN, LC, LCE, MLP, MLSTM-FCN, MTEX-CNN, ResNet, RF, SVM, XCM, XGB and XPM.

[7] Predefined train/test splits and an arithmetic mean of the accuracies on the UEA datasets. Models evaluated in the benchmark: $DTW_D$, $DTW_I$, FCN, gRSF, LPS, MLSTM-FCN, mv-ARF, ResNet, RFM, SMTS, UFS, WEASEL+MUSE, XCM, XEM and XGBM.

We observe that the new CNN MTS classifier XCM with Grad-CAM post-hoc model-specific explainability method does not obtain the best categories on all the components of the performance-explainability framework compared to the other machine learning methods presented in the thesis. Especially, the level of information conveyed by XCM approach is lower than the one of XEM and the one of the pattern-based MTS classifier XPM. Plus, XCM approach explanations are accessible by a lower audience than the ones from the pattern-based classifier XPM.

Nevertheless, XCM with Grad-CAM post-hoc model-specific explainability method is the best machine learning method on the estrus detection application according to the performance-explainability framework. First, performance is the primary criteria for the end-users and XCM is the best performing machine learning method on estrus detection (second: LCE). The insufficient performance of current estrus detection commercial solutions is the first reason of their moderate adoption rate [Steeneveld et al., 2015]. As detailed in chapter 7, the other well performing interval-based MTS classifiers DMSEEW and XEM could not be evaluated on the estrus detection application due to the short time windows considered. Both XCM and LCE approaches outperform the current commercial reference; and the performance difference between these approaches is not significant. However, unlike LCE with SHAP, XCM approach provides faithful explanations, which is a prerequisite to reduce solution mistrust from the farmers. Concerning the limited audience of XCM explanations, it is not a prohibitive factor as the end-users of the automatic estrus detection solutions are mainly the farmers, i.e. domain experts. XCM provides to the farmers the important observed variables and timestamps with the discriminative values to the predictions at both global (estrus versus anestrus) and local (behavioral versus silent) levels. It would be interesting to further enhance the explanations of XCM with Grad-CAM by synthesizing the information with patterns.

**Summary**

- The new CNN MTS classifier XCM exhibits better performance in estrus detection on a real-world dataset than a commercial solution reference and the machine learning methods of the previous chapters. The superior performance of XCM also holds on the public UEA datasets.
- XCM approach provides faithful and informative explanations at all granularity levels. The explanations inform the user about the important observed variables and timestamps with the discriminative values to the predictions.
- It would be interesting to further enhance the explanations of XCM with Grad-CAM by synthesizing the information with patterns.

Chapter 10

# Conclusion

## 10.1 Summary of the Contributions

The goal of this thesis is to enhance the performance and explainability of multivariate time series machine learning methods, and derive new insights from the new methods developed about two real-world applications.

First, I introduced a new performance-explainability framework to assess and benchmark machine learning methods. The framework details a set of characteristics that systematize the performance-explainability assessment of existing machine learning methods.

Then, I presented three new ensemble methods. The first one, DMSEEW, is a new stacking ensemble method for earthquake early warning. The performance criteria has shaped the design of this new stacking ensemble method but it provides faithful and local explanations. DMSEEW improves the detection of earthquakes with damaging potential. In particular, it detects all the large earthquakes with a precision of 100% on a real-world dataset, which is critical for an earthquake early warning system. The second one, LCE, is a new hybrid ensemble method for dairy resource monitoring. It relies on a post-hoc model-agnostic explainability method, SHAP, which offers more informative explanations that could be useful to broader audiences compared to the first method. However, the enhanced explanations come at the cost of faithfulness, which is a prerequisite for numerous applications. Therefore, the third method, XEM, is an extension of the hybrid ensemble method which integrates faithfulness with explainability by design. XEM competes with the level of information and the audience of the post-hoc model-agnostic explainability method while maintaining performance. XEM outperforms the current state-of-the-art MTS classifiers on the public UEA datasets.

Next, I proposed a new pattern-based MTS classifier for dairy resource monitoring, XPM; an easy-to-understand model with explanations accessible to a wider audience compared to the ensemble methods.

Lastly, I presented a new convolutional neural network for MTS classification, XCM,

which outperforms XEM on the public UEA datasets. In addition, XCM architecture enables faithful and precise identification of the observed variables and timestamps of the input data that are important for predictions using the gradient-based post-hoc model-specific method Grad-CAM. XCM with Grad-CAM exhibits better performance than the ensemble method LCE with SHAP on the dairy resource monitoring application, while enhancing explainability by providing faithful and more informative explanations. Furthermore, XCM detects around 20% more determining events for milk production (estrus) than a commercial solution reference on a real-world dataset, while having the same precision.

## 10.2 Perspectives

### 10.2.1 A Consensus About a Performance-Explainability Framework

There has been a trend in recent years to leverage machine learning methods to automate decision-making processes. However, for many tasks, the adoption of such methods cannot rely solely on their prediction performance; rather, explanations are required. Some of the main tasks that demand explainability are: model validation, model debugging and knowledge discovery. Model validation uses explanations to examine whether a machine learning model has employed the true evidences instead of biases which can exist in training data. Model debugging leverages the explanations to debug and analyze the misbehavior of models when models give wrong and unexpected predictions. Concerning knowledge discovery, the derived explanations allow humans to obtain new insights from machine learning model through comprehending their decision making process. Furthermore, explainability can be legally required as stated in the European Union's General Data Protection Regulation, which became enforceable on 25 May 2018 [1]. Therefore, in addition to their prediction performance, machine learning methods have to be assessed on how they can supply their decisions with explanations.

The requirements for explainable machine learning methods are dependent upon the application and to whom the explanations are intended for [Bohlender et al., 2019; Tomsett et al., 2018]. In order to match these requirements and then conduct experiments to validate the usefulness of the explanations by the end-users, there is a need to have a

1. https://ec.europa.eu/info/law/law-topic/data-protection_en

comprehensive assessment of the explainability of the existing methods. Doshi-Velez et al. [2017] claim that creating a shared language is essential for evaluation and comparison of machine learning methods, which is currently challenging without a set of explanation characteristics.

I proposed a first performance-explainability framework in this thesis. It could be a basis for the development of a comprehensive assessment of the machine learning methods with regard to their performance and explainability. This framework could also be used to inform the design of new machine learning methods. Thus, a consensus about a set of characteristics and their respective definition would be required. With regard to the set of characteristics identified in this thesis (Performance, Model Comprehensibility, Granularity of the explanations, Information Type, Faithfulness, User Category), a few challenges arise concerning their definitions. First, there is no consensus on an evaluation procedure to assess the performance of a machine learning model. Recent work suggests that the definition of such an evaluation procedure necessitates the development of a measurement theory for machine learning [Flach, 2019]. Many of the problems stem from a limited appreciation of the importance of the *scale* on which the evaluation measures are expressed. Second, model comprehensibility is tightly linked to the model complexity; yet, there is no consensus on model complexity assessment [Guidotti et al., 2018]. The current distinction, "black-box" / "white-box" models [Lipton, 2016], does not allow a full categorization of the different machine learning methods. For example, many rule-based models and decision trees are regarded as "white-box" models. However, not all rule-based models or decision trees are "white-box" models. Cognitive limitations of humans place restrictions on the complexity of the approximations that are understandable to humans [Lakkaraju et al., 2017]. For instance, a decision tree with a hundred levels cannot be considered as an easy-to-understand model. There would be a need to further refine this binary distinction in order to be able to categorize the different machine learning models with regard to their comprehensibility. Third, the framework introduced in this thesis proposes to distinguish between perfectly and imperfectly faithful explanations. There are three commonly recognized categories (explainability by design, post-hoc model-specific explainability and post-hoc model-agnostic explainability) [Du et al., 2020] to which all of the explainability methods are belonging to. For example, the main line of work in post-hoc model-agnostic explainability methods consists in approximating the decision surface of a model using an explainable one. However, the explanations from surrogate models cannot be perfectly faithful with respect to the original model [Rudin, 2019]. The

fidelity criteria is used to quantify the faithfulness by the extent to which the surrogate model imitates the prediction score of the original model [Guidotti et al., 2018]. But, how can imperfectly faithful explanations be used to support decisions? Without an associated guide that could be used to make informed decisions using an imperfectly faithful explainability method, there would be a need to focus on methods providing perfectly faithful explanations, and more particularly on explainable by design machine learning methods and post-hoc model-specific explainability methods.

Finally, following the development of a performance-explainability assessment of machine learning methods, it would be interesting to systematize a way to match the end-user needs to this assessment and develop a suite of experiments to test the usefulness of the machine learning method deployed.

### 10.2.2 Pattern-Based Post-Hoc Model-Specific Explainability Methods for Black-Box Machine Learning Models

Performance and explainability are often opposed whereas we should analyze how they can be best combined together along their different aspects. A performance-explainability framework like the one introduced in this thesis could be a basis for such analysis and the design of new machine learning methods.

Performance is usually a prerequisite before discussing explainability. On most applications and in particular for the MTS classifiers of this thesis, black-box machine learning models like deep learning models and ensemble methods exhibit the best performance on average. For example, based on predefined train/test splits and an arithmetic mean of the accuracies, XEM ensemble method and XCM convolutional neural network introduced in this thesis are the best performing methods on the public UEA datasets.

Then, as illustrated with XEM and XCM methods, it is possible to reconcile performance and faithful explanations by integrating explainability by design or by using post-hoc model-specific explainability methods. XEM provides explainability by design through identifying the time window used to classify the whole MTS. XCM with Grad-CAM post-hoc model-specific explainability method provides the important observed variables and timestamps with the discriminative values to the predictions.

However, the granularity of these explanations can be only local (e.g. XEM) and the level of information remains limited. Therefore, in order to further improve the explainability of these black-box classifiers, it would be interesting to develop new post-hoc

model-specific explainability methods. More specifically, the use of pattern-mining algorithms as post-hoc model-specific explainability methods would be interesting to explore. For example, to further improve the granularity and the level of information of XEM explanations, it would be valuable to analyze the time windows characteristic of each class in the training set in order to determine if they contain some common multidimensional sequential patterns. Such patterns could also broaden the audience as they would synthesize the important information in the discriminative time windows. This observation holds on the attribution maps for XCM with Grad-CAM.

# Bibliography


Adebayo, J., J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim (2018), Sanity Checks for Saliency Maps, *in*: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*.

Adriaens, I., W. Saeys, T. Huybrechts, C. Lamberigts, L. François, K. Geerinckx, J. Leroy, B. De Ketelaere, and B. Aernouts (2018), A Novel System for On-Farm Fertility Monitoring Based on Milk Progesterone, *in*: *Journal of Dairy Science* 101.9, pp. 8369–8382.

Agrawal, R. and R. Srikant (1994), Fast Algorithms for Mining Association Rules in Large Databases, *in*: *Proceedings of the 20th International Conference on Very Large Data Bases*.

Allen, R. M. and D. Melgar (2019), Earthquake Early Warning: Advances, Scientific Challenges, and Societal Needs, *in*: *Annual Review of Earth and Planetary Sciences* 47, pp. 361–388.

Altmann, A., L. Tolosi, O. Sander, and T. Lengauer (2010), Permutation Importance: A Corrected Feature Importance Measure, *in*: *Bioinformatics* 26.10, pp. 1340–1347.

Ancona, M., E. Ceolini, C. Öztireli, and M. Gross (2018), Towards Better Understanding of Gradient-Based Attribution Methods for Deep Neural Networks, *in*: *International Conference on Learning Representations*.

Assaf, R., I. Giurgiu, F. Bagehorn, and A. Schumann (2019), MTEX-CNN: Multivariate Time Series EXplanations for Predictions with Convolutional Neural Networks, *in*: *2019 IEEE International Conference on Data Mining*.

Bach, S., A. Binder, G. Montavon, F. Klauschen, K. Müller, and W. Samek (2015), On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation, *in*: *PLOS ONE* 10.7, pp. 1–46.

Bagnall, A., J. Lines, and E. Keogh (2018), The UEA UCR Time Series Classification Archive, *in*.

Bahdanau, D., K. Cho, and Y. Bengio (2015), Neural Machine Translation by Jointly Learning to Align and Translate, *in*: *Proceedings of 2015 the International Conference on Learning Representations*.

Bascom, S. and A. Young (1998), A Summary of the Reasons Why Farmers Cull Cows, *in*: *Journal of dairy science* 81.8, pp. 2299–2305.

Baydogan, M. and G. Runger (2014), Learning a Symbolic Representation for Multivariate Time Series Classification, *in*: *Data Mining and Knowledge Discovery* 29.2, pp. 400–422.

Baydogan, M. G. and G. Runger (2016), Time Series Representation and Similarity Based on Local Autopatterns, *in*: *Data Mining and Knowledge Discovery* 30.2, 476–509.

Bengio, Y., P. Simard, and P. Frasconi (1994), Learning Long-Term Dependencies with Gradient Descent Is Difficult, *in*: *IEEE Transactions on Neural Networks* 5.2, pp. 157–166.

Bergstra, J., R. Bardenet, Y. Bengio, and B. Kégl (2011), Algorithms for Hyper-Parameter Optimization, *in*: *Proceedings of the 24th International Conference on Neural Information Processing Systems*.

Bergstra, J., D. Yamins, and D. Cox (2013), Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures, *in*: *Proceedings of the 30th International Conference on Machine Learning*.

Bjorck, N., C. Gomes, B. Selman, and K. Weinberger (2018), Understanding Batch Normalization, *in*: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*.

Bohlender, D. and M. Kóhl (2019), Towards a Characterization of Explainable Systems, *in*: *ArXiv*.

Breiman, L. (1996), Bagging Predictors, *in*: *Machine Learning* 24.2, pp. 123–140.

— (2001), Random Forests, *in*: *Machine Learning* 45.1, 5–32.

Breiman, L., J. Friedman, C.J. Stone, and R.A. Olshen (1984), *Classification and Regression Trees*, The Wadsworth and Brooks-Cole statistics-probability series, Taylor & Francis.

Buza, K. and L. Schmidt-Thieme (2010), Motif-Based Classification of Time Series with Bayesian Networks and SVMs, *in*: *Advances in Data Analysis, Data Handling and Business Intelligence*, pp. 105–114.

Caraviello, D., K. Weigel, P. Fricke, M. Wiltbank, M. Florent, N. Cook, K. Nordlund, N. Zwald, and C. Rawson (2006), Survey of Management Practices on Reproductive Performance of Dairy Cattle on Large US Commercial Farms, *in*: *Journal of Dairy Science* 89.12, pp. 4723–4735.

Chanvallon, A., S. Coyral-Castel, J. Gatien, J. Lamy, D. Ribaud, C. Allain, P. Clément, and P. Salvetti (2014), Comparison of Three Devices for the Automated Detection of Estrus in Dairy Cows, *in*: *Theriogenology* 82.5, pp. 734–741.

Chastant-Maillard, S. (2006), Is There a Future for Pharmaceutical Management of Cow Reproduction: European Perspective, *in*: *Proceedings of the WBC*.

Chen, T. and C. Guestrin (2016), XGBoost: A Scalable Tree Boosting System, *in*: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Cheng, H., X. Yan, J. Han, and C. Hsu (2007), Discriminative Frequent Pattern Analysis for Effective Classification, *in*: *Proceedings of the 23rd International Conference on Data Engineering*.

Cheng, H., X. Yan, J. Han, and P. Yu (2008), Direct Discriminative Pattern Mining for Effective Classification, *in*: *Proceedings of the 24th International Conference on Data Engineering*.

Cortes, C. and V. Vapnik (1995), Support-Vector Networks, *in*: *Machine Learning* 20.3, 273–297.

Cristian Borges Gamboa, J. (2017), Deep Learning for Time-Series Analysis, *in*: *ArXiv*.

Cussins Newman, J. (2019), Toward AI Security: Global Aspirations for a More Resilient Future, *in*: *Center for Long-Term Cybersecurity*.

Cutullic, E., L. Delaby, Y. Gallard, and C. Disenhaus (2011), Dairy Cows' Reproductive Response to Feeding Level Differs According to the Reproductive Stage and the Breed, *in*: *Animal* 5.5, pp. 731–740.

Dauxais, Y., D. Gross-Amblard, T. Guyet, and A. Happe (2019), Discriminant Chronicle Mining, *in*: *Advanced Knowledge and Data Mining*, pp. 89–118.

De Silva, A.W, G.W. Anderson, F.C. Gwazdauskas, M.L. McGilliard, and J.A. Lineweaver (1981), Interrelationships With Estrous Behavior and Conception in Dairy Cattle, *in*: *Journal of Dairy Science* 64.12, pp. 2409–2418.

Demšar, J. (2006), Statistical Comparisons of Classifiers over Multiple Data Sets, *in*: *Journal of Machine Learning Research* 7, pp. 1–30.

Devlin, J., M. Chang, K. Lee, and K. Toutanova (2019), BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding, *in*: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Dhurandhar, A., P. Chen, R. Luss, C. Tu, P. Ting, K. Shanmugam, and P. Das (2018), Explanations Based on the Missing: Towards Contrastive Explanations with Pertinent Negatives, *in*: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*.

Dietterich, T. (1998), Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, *in*: *Neural Computation* 10.7, pp. 1895–1923.

— (2000), Ensemble Methods in Machine Learning, *in*: *Multiple Classifier Systems*, pp. 1–15.

Disenhaus, C., E. Cutullic, F. Blanc, and J. Agabriel (2009), Breed Comparison of Post Partum Ovarian Activity in Cows, *in*: *Journal of Dairy Science* 93.7, pp. 2938–2951.

Dobson, H., S.L. Walker, M.J. Morris, J.E. Routly, and R.F. Smith (2008), Why Is It Getting More Difficult to Successfully Artificially Inseminate Dairy Cows?, *in*: *Animal* 2.8, pp. 1104–1111.

Dolecheck, K.A., W.J. Silvia, G.Jr Heersche, Y.M. Chang, D.L. Ray, A.E. Stone, B.A. Wadsworth, and J.M. Bewley (2015), Behavioral and Physiological Changes Around Estrus Events Identified Using Multiple Automated Monitoring Technologies, *in*: *Journal of Dairy Science* 98.12, pp. 8723–8731.

Doshi-Velez, F. and B. Kim (2017), Towards a Rigorous Science of Interpretable Machine Learning, *in*: *ArXiv*.

Du, M., N. Liu, and X. Hu (2020), Techniques for Interpretable Machine Learning, *in*: *Communications of the ACM*.

Dua, D. and C. Graff (2017), *UCI Machine Learning Repository*.

Ebrahimpour, R., N. Sadeghnejad, S. Arani, and N. Mohammadi (2012), Boost-Wise Pre-Loaded Mixture of Experts for Classification Tasks, *in*: *Neural Computing and Applications* 22.1, pp. 365–377.

Erhan, D., Y. Bengio, A. Courville, and P. Vincent (2009), Visualizing Higher-Layer Features of a Deep Network, *in*: *Proceedings of the ICML Workshop on Learning Feature Hierarchies*.

Esteva, A., A. Robicquetand B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean (2019), A Guide to Deep Learning in Healthcare, *in*: *Nature Medicine* 25, pp. 24–29.

Fauvel, K., D. Balouek-Thomert, D. Melgar, P. Silva, A. Simonet, G. Antoniu, A. Costan, V. Masson, M. Parashar, I. Rodero, and A. Termier (2019a), Earthquake Early Warning Dataset, *in*: *figshare*.

Fauvel, K., D. Balouek-Thomert, D. Melgar, P. Silva, A. Simonet, G. Antoniu, A. Costan, V. Masson, M. Parashar, I. Rodero, and A. Termier (2020a), A Distributed Multi-Sensor Machine Learning Approach to Earthquake Early Warning, *in*: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*.

Fauvel, K., É. Fromont, V. Masson, P. Faverdin, and A. Termier (2020b), XEM: An Explainable Ensemble Method for Multivariate Time Series Classification, *in*: *ArXiv*.

Fauvel, K., T. Lin, V. Masson, É. Fromont, and A. Termier (2020c), XCM: An Explainable Convolutional Neural Network for Multivariate Time Series Classification, *in*: *ArXiv*.

Fauvel, K., V. Masson, and É. Fromont (2020d), A Performance-Explainability Framework to Benchmark Machine Learning Methods: Application to Multivariate Time Series Classifiers, *in*: *Proceedings of the IJCAI-PRICAI Workshop on Explainable Artificial Intelligence*.

Fauvel, K., V. Masson, É. Fromont, P. Faverdin, and A. Termier (2019b), Towards Sustainable Dairy Management - A Machine Learning Enhanced Method for Estrus Detection, *in*: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Fawaz, H., G. Forestier, J. Weber, L. Idoumghar, and P. Muller (2019), Deep Learning for Time Series Classification: A Review, *in*: *Data Mining Knowledge Discovery* 33.4, pp. 917–963.

Fischer, J., J. Redlich, J. Zschau, C. Milkereit, and M. Picozzi (2012), A Wireless Mesh Sensing Network for Early Warning, *in*: *Journal of Network and Computer Applications* 35.2, pp. 538–547.

Flach, P. (2019), Performance Evaluation in Machine Learning: The Good, the Bad, the Ugly, and the Way Forward, *in*: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.

Fournier-Viger, P., J. Lin, B. Vo, T. Chi, J. Zhang, and H. Le (2017), A Survey of Itemset Mining, *in*: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7.4.

Fradkin, D.and F. Mörchen (2014), Mining Sequential Patterns for Classification, *in*: *Knowledge and Information Systems* 45.3, pp. 731–749.

Fricke, P., P. Carvalho, J. Giordano, A. Valenza, G. Lopes, and M. Amundson (2017), Expression and Detection of Estrus in Dairy Cows: the Role of New Technologies, *in*: *Animal* 8.s1, pp. 134–143.

Fujimoto, K., I. Kojadinovic, and J.L. Marichal (2006), Axiomatic Characterizations of Probabilistic and Cardinal-Probabilistic Interaction Indices, *in*: *Games and Economic Behavior* 55.1, pp. 72–99.

Gaillard, C., H. Barbu, M.T. Sørensen, J. Sehested, H. Callesen, and M. Vestergaard (2016), Milk Yield and Estrous Behavior During Eight Consecutive Estruses in Holstein Cows Fed Standardized or High Energy Diets and Grouped According to Live Weight Changes in Early Lactation, *in*: *Journal of Dairy Science* 99.4, pp. 3134–3143.

Gama, J. and P. Brazdil (2000), Cascade Generalization, *in*: *Machine Learning* 41.3, pp. 315–343.

Gilmore, H.S., F.J. Young, D.C. Patterson, A.R. Wylie, R.A. Law, D.J. Kilpatrick, C.T. Elliott, and C.S. Mayne (2011), An Evaluation of the Effect of Altering Nutrition and Nutritional Strategies in Early Lactation on Reproductive Performance and Estrous Behavior of High-Yielding Holstein-Friesian Dairy Cows, *in*: *Journal of Dairy Science* 94.7, pp. 3510–3526.

Goodfellow, I., Y. Bengio, and A. Courville (2016), *Deep Learning*, MIT Press.

Górecki, T. and M. Łuczak (2013), Using Derivatives in Time Series Classification, *in*: *Data Mining and Knowledge Discovery* 26.2, pp. 310–331.

Grahne, G. and J. Zhu (2003), Efficiently Using Prefix-Trees in Mining Frequent Itemsets, *in*: *ICDM Workshop on Frequent Itemset Mining Implementations*.

Graves, A. (2012), *Supervised Sequence Labelling with Recurrent Neural Networks*, Studies in Computational Intelligence, Springer.

Guidotti, R., A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, and F. Turini (2019), Factual and Counterfactual Explanations for Black Box Decision Making, *in*: *IEEE Intelligent Systems* 34.6, pp. 14–23.

Guidotti, R., A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi (2018), A Survey of Methods for Explaining Black Box Models, *in*: *ACM Computing Surveys*.

Gwazdauskas F.C., J.A. Lineweaver and M.L. McGilliard (1983), Environmental and Management Factors Affecting Estrous Activity in Dairy Cattle, *in*: *Journal of Dairy Science* 66.7, pp. 1510–1514.

Han, J., J. Pei, and M. Kamber (2011), *Data Mining: Concepts and Techniques*, The Morgan Kaufmann Series in Data Management Systems, Elsevier Science.

Han, J., J. Pei, and Y. Yin (2000), Mining Frequent Patterns without Candidate Generation, *in*: *ACM SIGMOD Record* 29.2, pp. 1–12.

Haykin, S. (2009), *Neural Neworks and Learning Machines*, Prentice Hall.

He, K., X. Zhang, S. Ren, and J. Sun (2016), Deep Residual Learning for Image Recognition, *in*: *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*.

Hochreiter, J. (1991), Untersuchungen zu dynamischen neuronalen Netzen, *in*.

Hoshiba, M. and T. Ozaki (2012), Earthquake Early Warning and Tsunami Warning of JMA for the 2011 off the Pacific Coast of Tohoku Earthquake, *in*: *Journal of the Seismological Society of Japan* 64.3, pp. 155–168.

Hu, J., L. Shen, G. Sun, and S. Albanie (2017), Squeeze-and-Excitation Networks, *in*: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 2011–2023.

Huang, G., Z. Liu, L. Van Der Maaten, and K. Weinberger (2017), Densely Connected Convolutional Networks, *in*: *2017 IEEE Conference on Computer Vision and Pattern Recognition*.

Inchaisri, C., R. Jorritsma, P.L.A.M. Vos, G.C. Van Der Weijden, and H. Hogeveen (2010), Economic Consequences of Reproductive Performance in Dairy Cattle, *in*: *Theriogenology* 74.5, pp. 835–846.

Ioffe, S. and C. Szegedy (2015), Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, *in*: *Proceedings of the 32nd International Conference on Machine Learning*.

Jacobs, RA., MI. Jordan, SJ. Nowlan, and GE. Hinton (1991), Adaptive Mixtures of Local Experts, *in*: *Neural Computation* 3.1, 79–87.

Jain, S. and B. Wallace (2019), Attention is not Explanation, *in*: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Jiang, Renhe, Xuan Song, Dou Huang, Xiaoya Song, Tianqi Xia, Zekun Cai, Zhaonan Wang, Kyoung-Sook Kim, and Ryosuke Shibasaki (2019), DeepUrbanEvent: A System for Predicting Citywide Crowd Dynamics at Big Events, *in*: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Jónsson, R., M. Blanke, N.K. Poulsen, F. Caponetti, and S. Højsgaard (2011), Oestrus Detection in Dairy Cows from Activity and Lying Data Using On-Line Individual Models, *in*: *Computers and Electronics in Agriculture* 76.1, pp. 6–15.

Karim, F., S. Majumdar, H. Darabi, and S. Harford (2019), Multivariate LSTM-FCNs for Time Series Classification, *in*: *Neural Networks* 116, pp. 237–245.

Karlsson, I., P. Papapetrou, and H. Boström (2016), Generalized Random Shapelet Forests, *in*: *Data Mining and Knowledge Discovery* 30.5, pp. 1053–1085.

Kennet, B. L. N. (1991), IASPEI 1991 Seismological Tables, *in*: *Terra Nova* 3, pp. 122–122.

Kerbrat, S. and C. Disenhaus (2004), A Proposition for an Updated Behavioural Characterisation of the Oestrus Period in Dairy Cows, *in*: *Applied Animal Behaviour Science* 87.3-4, pp. 223–238.

Kim, B., R. Khanna, and O. Koyejo (2016), Examples are not Enough, Learn to Criticize! Criticism for Interpretability, *in*: *Proceedings of the 30th International Conference on Neural Information Processing Systems*.

Kotsiantis, S. and P. Pintelas (2005), Combining Bagging and Boosting, *in*: *International Journal of Computational Intelligence* 1.8, pp. 372–381.

Krieter, J. (2005), Oestrus Detection in Dairy Cows Using Control Charts and Neural Networks, *in*: *Proceedings of 56th Annual Meeting of the European Association for Animal Production. Commission on Cattle Production*, Uppsala, Sweden.

Kuz'min, V., P. Polishchuk, A. Artemenko, and S. Andronati (2011), Interpretation of QSAR Models Based on Random Forest Methods, *in*: *Molecular Informatics* 30.6-7, pp. 593–603.

Lakkaraju, H., E. Kamar, R. Caruana, and J. Leskovec (2017), Interpretable and Explorable Approximations of Black Box Models, *in*: *Proceedings of the KDD Workshop on Fairness, Accountability, and Transparency in Machine Learning*.

Lakkaraju, Himabindu, Stephen H. Bach, and Jure Leskovec (2016), Interpretable Decision Sets: A Joint Framework for Description and Prediction, *in*: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Li, Jia, Y. Rong, H. Meng, Z. Lu, T. Kwok, and H. Cheng (2018a), TATC: Predicting Alzheimer's Disease with Actigraphy Data, *in*: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Li, Z., M. A. Meier, E. Hauksson, Z. Zhan, and J. Andrews (2018b), Machine Learning Seismic Wave Discrimination: Application to Earthquake Early Warning, *in*: *Geophysical Research Letters* 45.10, pp. 4773–4779.

Lin, J., E. Keogh, S. Lonardi, and B. Chiu (2003), A Symbolic Representation of Time Series, with Implications for Streaming Algorithms, *in*: *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*.

Lin, J., E. Keogh, S. Lonardi, and P. Patel (2002), Finding Motifs in Time Series, *in*: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Lin, M., Q. Chen, and S. Yan (2014), Network in Network, *in*: *ArXiv*.

Lines, J, S. Taylor, and A. Bagnall (2016), HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification, *in*: *Proceedings of the IEEE 16th International Conference on Data Mining*.

Lipton, Z. (2016), The Mythos of Model Interpretability, *in*: *Proceedings of the ICML Workshop on Human Interpretability in Machine Learning*.

Liu, Y. and X. Yao (1999), Ensemble Learning via Negative Correlation, *in*: *Neural Networks* 12.10, pp. 1399–1404.

Lopez H., L.D. Satter and M.C. Wiltbank (2004), Relationship Between Level of Milk Production and Estrous Behavior of Lactating Dairy Cows, *in*: *Animal Reproduction Science* 81.3, pp. 209–223.

Lundberg, S. and S. Lee (2017), A Unified Approach to Interpreting Model Predictions, *in*: *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Ma, N., L. Pan, S. Chen, and B. Liu (2020), NB-IoT Estrus Detection System of Dairy Cows Based on LSTM Networks, *in*: *IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*.

Martin, O., N.C. Friggens, J. Dupont, P. Salvetti, S. Freret, C. Rame, S. Elis, J. Gatien, C. Disenhaus, and F. Blanc (2013), Data-Derived Reference Profiles with corepresentation of Progesterone, Estradiol, LH, and FSH Dynamics during the Bovine Estrous Cycle, *in*: *Theriogenology* 79.2, pp. 331–343.

Masoudnia, S. and R. Ebrahimpour (2014), Mixture of Experts: a Literature Survey, *in*: *Artificial Intelligence Review* 42.2, pp. 275–293.

Masoudnia, S., R. Ebrahimpour, and S. Arani (2012), Incorporation of a Regularization Term to Control Negative Correlation in Mixture of Experts, *in*: *Neural Processing Letters* 36.1, pp. 31–47.

Mayo, L., W.J. Silvia, D.L. Ray, B. Jones, A.E. Stone, I.C. Tsai, J.D. Clark, J. Bewley, and G. Heersche (2019), Automated Estrous Detection Using Multiple Commercial Precision Dairy Monitoring Technologies in Synchronized Dairy Dows, *in*: *Journal of Dairy Science* 102.3, pp. 2645–2656.

Melgar, D., Y. Bock, D. Sanchez, and B. W. Crowell (2013), On Robust and Reliable Automated Baseline Corrections for Strong Motion Seismology, *in*: *Journal of Geophysical Research: Solid Earth* 118.3, pp. 1177–1187.

Melgar, D., B. W. Crowell, J. Geng, R. M. Allen, Y. Bock, S. Riquelme, E. M. Hill, M. Protti, and A. Ganas (2015), Earthquake Magnitude Calculation Without Saturation from the Scaling of Peak Ground Displacement, *in*: *Geophysical Research Letters* 42.13, 5197–5205.

Miller, T. (2019), Explanation in Artificial Intelligence: Insights from the Social Sciences, *in*: *Artificial Intelligence* 267, pp. 1–38.

Minegishi, K., B.J. Heins, and G.M. Pereira (2019), Peri-Estrus Activity and Rumination Time and its Application to Estrus Prediction: Evidence from Dairy Herds under organic Grazing and Low-Input Conventional Production, *in*: *Livestock Science* 221, pp. 144–154.

Mitchell, R.S, R.A. Sherlock, and L.A. Smith (1996), An Investigation into the Use of Machine Learning for Determining Oestrus in Cows, *in*: *Computers and Electronics in Agriculture* 15.3, pp. 195–213.

Nair, V. and GE. Hinton (2010), Rectified Linear Units Improve Restricted Boltzmann Machines, *in*: *Proceedings of the 27th International Conference on International Conference on Machine Learning*.

Neath, I. and A. Surprenant (2003), *Human Memory: An Introduction to Research, Data, and Theory*, Thomson/Wadsworth.

Palczewska, A., J. Palczewski, R. Robinson, and D. Neagu (2013), Interpreting Random Forest Models Using a Feature Contribution Method, *in*: *Proceedings of the 14th IEEE International Conference on Information Reuse Integration*.

Palmer, M.A., G. Olmos, L.A. Boyle, and J.F. Mee (2010), Estrus Detection and Estrus Characteristics in Housed and Pastured Holstein-Friesian Cows, *in*: *Theriogenology* 74.2, pp. 255–264.

Pascanu, R., T. Mikolov, and Y. Bengio (2013), On the Difficulty of Training Recurrent Neural Networks, *in*: *Proceedings of the 30th International Conference on Machine Learning*.

Pasquier, N., Y. Bastide, R. Taouil, and L. Lakhal (1999), Discovering Frequent Closed Itemsets for Association Rules, *in*: *Proceedings of the 7th International Conference on Database Theory*.

Pearl, J. and D. Mackenzie (2018), *The Book of Why: The New Science of Cause and Effect*, Basic Books.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011), Scikit-Learn: Machine Learning in Python, *in*: *Journal of Machine Learning Research* 12.85, pp. 2825–2830.

Peralta, O.A., R.E. Pearson, and R.L. Nebel (2005), Comparison of Three Estrus Detection Systems During Summer in a Large Commercial Dairy Herd, *in*: *Animal Reproduction Science* 87.1, pp. 59–72.

Perol, T., M. Gharbi, and M. Denolle (2018), Convolutional Neural Network for Earthquake Detection and Location, *in*: *Science Advances* 4.2.

Petersson, K.J., H. Gustafsson, E. Strandberg, and B. Berglund (2006), Atypical Progesterone Profiles and Fertility in Swedish Dairy Cows, *in*: *Journal of Dairy Science* 89.7, pp. 2529–2538.

Ranasinghe, R.M., T. Nakao, K. Yamada, and K. Koike (2010), Silent Ovulation, Based on Walking Activity and Milk Progesterone Concentrations, in Holstein Cows Housed in a Free-Stall Barn, *in*: *Theriogenology* 73.7, pp. 942–949.

Ransbotham, S., S. Khodabandeh, R. Fehling, B. LaFountain, and D. Kiron (2019), Winning With AI, *in*: *MIT Sloan Management Review and Boston Consulting Group*.

Reames, P., T. Hatler, S. Hayes, L. Ray, and W. Silvia (2011), Differential Regulation of Estrous Behavior and Luteinizing Hormone Secretion by Estradiol-17βin Ovariectomized Dairy Cows, *in*: *Theriogenology* 75.2, pp. 233–240.

Reith, S. and S.Hoy (2012), Relationship Between Daily Rumination Time and Estrus of Dairy Cows, *in*: *Journal of Dairy Science* 95.11, pp. 6416–6420.

Ribeiro, M., S. Singh, and C. Guestrin (2016), "Why Should I Trust You?": Explaining the Predictions of Any Classifier, *in*: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

— (2018), Anchors: High-Precision Model-Agnostic Explanations, *in*: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Rudin, C. (2019), Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead, *in*: *Nature Machine Intelligence* 1, 206–215.

Ruhl, C. J., D. Melgar, A. I. Chung, R. Grapenthin, and R. M. Allen (2019), Quantifying the Value of Real-Time Geodetic Constraints for Earthquake Early Warning Using

a Global Seismic and Geodetic Data Set, *in*: *Journal of Geophysical Research: Solid Earth* 124.4, pp. 3819–3837.

Rutten, C., W. Steeneveld, C. Inchaisri, and H. Hogeveen (2014), An Ex Ante Analysis on the Use of Activity Meters for Automated Estrus Detection: to Invest or not to Invest?, *in*: *Journal of Dairy Science* 97.11, pp. 6869–6887.

Saint-Dizier, M. and S. Chastant-Maillard (2012), Towards an Automated Detection of Oestrus in Dairy Cattle, *in*: *Reproduction in Domestic Animals* 47.6, pp. 1056–1061.

Satyanarayanan, M. (2017), The Emergence of Edge Computing, *in*: *Computer* 50.1, pp. 30–39.

Schäfer, P. and U. Leser (2017), Multivariate Time Series Classification with WEASEL + MUSE, *in*: *ArXiv*.

Schapire, RE. (1990), The Strength of Weak Learnability, *in*: *Machine Learning* 5.2, pp. 197–227.

Searchinger, T., R. Waite, C. Hanson, J. Ranganathan, P. Dumas, and E. Matthews (2018), *Creating a Sustainable Food Future*, World Resources Institute.

Selvaraju, R., A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra (2019), Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization, *in*: *International Journal of Computer Vision* 128.8, pp. 336–359.

Senger, P. (1994), The Estrus Detection Problem: New Concepts, Technologies, and Possibilities, *in*: *Journal of Dairy Science* 77.9, pp. 2745–2753.

Sesmero, M., A. Ledezma, and A. Sanchis (2015), Generating Ensembles of Heterogeneous Classifiers Using Stacked Generalization, *in*: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 5.1, pp. 21–34.

Seto, S., W. Zhang, and Y. Zhou (2015), Multivariate Time Series Classification Using Dynamic Time Warping Template Selection for Human Activity Recognition, *in*: *Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence*.

Shi, W., J. Cao, Q. Zhang, Y. Li, and L. Xu (2016), Edge Computing: Vision and Challenges, *in*: *IEEE Internet of Things Journal* 3.5, pp. 637–646.

Shokoohi-Yekta, M., B. Hu, H. Jin, J. Wang, and E. Keogh (2017), Generalizing DTW to the Multi-Dimensional Case Requires an Adaptive Approach, *in*: *Data Mining and Knowledge Discovery* 31, pp. 1–31.

Shrikumar, A., P. Greenside, and A. Kundaje (2017), Learning Important Features through Propagating Activation Differences, *in*: *Proceedings of the 34th International Conference on Machine Learning*.

Shrikumar, A., P. Greenside, A. Shcherbina, and A. Kundaje (2016), Not Just a Black Box: Learning Important Features Through Propagating Activation Differences, *in*: *ArXiv*.

Song, H., D. Rajan, J. Thiagarajan, and A. Spanias (2018), Attend and Diagnose: Clinical Time Series Analysis using Attention Models, *in*: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Springenberg, J.T., A. Dosovitskiy, T. Brox, and M. Riedmiller (2015), Striving for Simplicity: The All Convolutional Net, *in*: *Proceedings of the International Conference on Learning Representations (Workshop Track)*.

Steeneveld, W. and H. Hogeveen (2015), Characterization of Dutch Dairy Farms Using Sensor Systems for Cow Management, *in*: *Journal of Dairy Science* 98.1, pp. 709–717.

Sundararajan, M., A. Taly, and Q. Yan (2017), Axiomatic Attribution for Deep Networks, *in*: *Proceedings of the 34th International Conference on Machine Learning*.

Sutskever, I., O. Vinyals, and Q. Le (2014), Sequence to Sequence Learning with Neural Networks, *in*: *Proceedings of the 27th International Conference on Neural Information Processing Systems*.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015), Going Deeper with Convolutions, *in*: *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*.

Tenghe, A.M., A.C. Bouwman, B. Berglund, E. Strandberg, J.Y. Blom, and R.F. Veerkamp (2015), Estimating Genetic Parameters for Fertility in Dairy Cows from In-Line Milk Progesterone Profiles, *in*: *Journal of Dairy Science* 98.8, pp. 5763–5773.

Tomsett, R., D. Braines, D. Harborne, A. Preece, and S. Chakraborty (2018), Interpretable to Whom? A Role-Based Model for Analyzing Interpretable Machine Learning Systems, *in*: *Proceedings of the ICML Workshop on Human Interpretability in Machine Learning*, pp. 1–8.

Tuncel, K. and M. Baydogan (2018), Autoregressive Forests for Multivariate Time Series Modeling, *in*: *Pattern Recognition* 73, pp. 202–215.

Van Eerdenburg, F.J., D. Karthaus, M.A. Taverne, I. Merics, and O. Szenci (2002), The Relationship Between Estrous Behavioral Score and Time of Ovulation in Dairy Cattle, *in*: *Journal of Dairy Science* 85.5, pp. 1150–1156.

Van Vliet, J.H. and F.J. Van Eerdenburg (1996), Sexual Activities and Oestrus Detection in Lactating Holstein Cows, *in*: *Applied Animal Behaviour Science* 50.1, pp. 57–69.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin (2017), Attention Is All You Need, *in*: *Proceedings of the 31st International Conference on Neural Information Processing Systems*.

Vidal, Enrique, Francisco Casacuberta, and Hector Rulot Segovia (1985), Is the DTW "Distance" Really a Metric? An Algorithm Reducing the Number of DTW Comparisons in Isolated Word Recognition, *in*: *Speech Communication* 4, pp. 333–344.

Voita, E., D. Talbot, F. Moiseev, R. Sennrich, and I. Titov (2019), Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned, *in*: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Wang, J. and J. Han (2004), BIDE: Efficient Mining of Frequent Closed Sequences, *in*: *Proceedings of the 20th International Conference on Data Engineering*, pp. 79–90.

Wang, Z., W. Yan, and T. Oates (2017), Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline, *in*: *Proceedings of the 2017 International Joint Conference on Neural Networks*.

Wiegreffe, S. and Y. Pinter (2019), Attention is not not Explanation, *in*: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

WIPO (2019), *WIPO Technology Trends 2019 – Artificial Intelligence*, World Intellectual Property Organization.

Wistuba, M., J. Grabocka, and L. Schmidt-Thieme (2015), Ultra-Fast Shapelets for Time Series Classification, *in*: *ArXiv*.

Witten, I., E. Frank, M. Hall, and C. Pal (2016), *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Series.

Wolf, C. (2019), Explainability Scenarios: Towards Scenario-Based XAI Design, *in*: *Proceedings of the 24th International Conference on Intelligent User Interfaces*, pp. 252–257.

Wolpert, D. (1996), The Lack of A Priori Distinctions Between Learning Algorithms, *in*: *Neural Computation* 8.7, 1341—1390.

Yang, C., R. Raskin, M. Goodchild, and M. Gahegan (2010), Geospatial Cyberinfrastructure: Past, Present and Future, *in*: *Computers, Environment and Urban Systems* 34.4, pp. 264–277.

Yániz, J.L., P. Santolaria, A. Giribet, and F. López-Gatius (2006), Factors Affecting Walking Activity at Estrus During Postpartum Period and Subsequent fertility in Dairy Cows, *in*: *Theriogenology* 66.8, pp. 1943–1950.

Ye, L. and E. Keogh (2009), Time Series Shapelets: A New Primitive for Data Mining, *in*: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Yoon, C., O. O'Reilly, K. J. Bergen, and G. C. Beroza (2015), Earthquake Detection Through Computationally Efficient Similarity Search, *in*: *Science Advances*.

Zaki, M.J. (2000), Scalable Algorithms for Association Mining, *in*: *IEEE Transactions on Knowledge and Data Engineering* 12.3, pp. 372–390.

Zebari, H.M., S. Rutter, and E. Bleach (2018), Characterizing Changes in Activity and Feeding Behaviour of Lactating Dairy Cows During Behavioural and Silent Oestrus, *in*: *Applied Animal Behaviour Science* 206, pp. 12–17.

Zeiler, M.D. and R. Fergus (2014), Visualizing and Understanding Convolutional Networks, *in*: *Proceedings of the European conference on computer vision*.

Zhou, B., A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba (2016), Learning Deep Features for Discriminative Localization, *in*: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Zintgraf, L.M., T.S. Cohen, T. Adel, and M. Welling (2017), Visualizing Deep Neural Network Decisions: Prediction Difference Analysis, *in*: *5th International Conference on Learning Representations*.

Zou, H. and T. Hastie (2005), Regularization and Variable Selection via the Elastic Net, *in*: *Journal of the Royal Statistical Society. Series B (Statistical Methodology)* 67.2, pp. 301–320.

Appendix A

# Publications

- Fauvel, K., D. Balouek-Thomert, D. Melgar, P. Silva, A. Simonet, G. Antoniu, A. Costan, V. Masson, M. Parashar, I. Rodero, and A. Termier (2020), A Distributed Multi-Sensor Machine Learning Approach to Earthquake Early Warning, *in: Proceedings of the 34th AAAI Conference on Artificial Intelligence*
- Fauvel, K., V. Masson, and É. Fromont (2020), A Performance-Explainability Framework to Benchmark Machine Learning Methods: Application to Multivariate Time Series Classifiers, *in: Proceedings of the IJCAI-PRICAI Workshop on Explainable Artificial Intelligence*
- Fauvel, K., T. Lin, V. Masson, É. Fromont, and A. Termier (2020), XCM: An Explainable Convolutional Neural Network for Multivariate Time Series Classification, *in: ArXiv*
- Fauvel, K., É. Fromont, V. Masson, P. Faverdin, and A. Termier (2020), XEM: An Explainable Ensemble Method for Multivariate Data Classification, *in: ArXiv*
- Fauvel, K., D. Balouek-Thomert, D. Melgar, P. Silva, A. Simonet, G. Antoniu, A. Costan, V. Masson, M. Parashar, I. Rodero, and A. Termier (2019), Earthquake Early Warning Dataset, *in: figshare*
- Fauvel, K., V. Masson, É. Fromont, P. Faverdin, and A. Termier (2019), Towards Sustainable Dairy Management - A Machine Learning Enhanced Method for Estrus Detection, *in: Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*

**Amélioration de la Performance et de l'Explicabilité des Méthodes d'Apprentissage Automatique de Séries Temporelles Multivariées**

**Mots-clés :** Apprentissage Automatique, Explicabilité de l'Intelligence Artificielle, Portée Collective, Séries Temporelles Multivariées

**Résumé :** Cette thèse a pour objectif d'améliorer la performance et l'explicabilité des méthodes d'apprentissage automatique de séries temporelles multivariées, et d'établir à partir des méthodes développées de nouvelles connaissances concernant deux applications réelles.

Tout d'abord, une nouvelle grille d'analyse pour évaluer et comparer les méthodes d'apprentissage automatique est proposée. Cette grille d'analyse introduit un jeu de caractéristiques qui systématise l'évaluation des méthodes d'apprentissage automatique.

Puis, cette thèse présente trois nouvelles méthodes ensemblistes. La première, DMSEEW, est une nouvelle méthode ensembliste, basée sur le stacking, pour la détection avancée de séismes et qui fournit des explications exactes au niveau local. DMSEEW améliore la détection des séismes pouvant causer des dommages. Tout particulièrement, DMSEEW détecte tous les séismes forts avec une précision de 100% sur un jeu de données réel, ce qui est primordial pour une détection avancée des séismes. La seconde, LCE, est une nouvelle méthode ensembliste hybride pour la gestion des ressources dans les exploitations laitières. Elle se repose sur une méthode d'explicabilité applicable à n'importe quelle méthode d'apprentissage automatique (post-hoc modèle-agnostique), SHAP, qui offre des explications plus informatives et accessibles à une plus grande audience que celles de la première méthode. Cependant, cette amélioration des explications s'effectue au détriment de l'exactitude, ce qui est un prérequis pour de nombreuses applications. Aussi, la troisième méthode, XEM, est une extension de la méthode ensembliste hybride qui intègre une explicabilité intrinsèque afin d'assurer l'exactitude des explications. XEM rivalise avec le niveau d'information et l'accessibilité de la méthode post-hoc modèle-agnostique tout en maintenant la performance. XEM est plus performante que les méthodes de l'état de l'art en classification de séries temporelles multivariées sur les jeux de données publiques UEA.

Ensuite, un nouveau classifieur se reposant sur des motifs fréquents (XPM) est proposé pour la gestion des ressources dans les exploitations laitières, un classifieur facile à comprendre avec des explications accessibles à une audience plus large que celles des méthodes ensemblistes.

Enfin, cette thèse présente un nouveau réseau de neurones à convolution, XCM, se révelant plus performant que XEM sur les jeux de données publiques UEA. De plus, l'architecture de XCM permet, grâce à l'utilisation de la méthode Grad-CAM (post-hoc modèle-spécifique), une identification précise et exacte des variables observées et des timestamps des données d'entrée importants pour les prédictions. XCM avec Grad-CAM présente de meilleures performances que la méthode ensembliste LCE avec SHAP sur l'application relative à la gestion des ressources dans les exploitations laitières, tout en améliorant l'explicabilité avec des explications exactes et plus informatives. En outre, XCM détecte environ 20% d'évènements clés en plus qu'une solution commerciale de référence sur le jeu de données réel en exploitation laitière, tout en préservant la même précision.

# Enhancing Performance and Explainability of Multivariate Time Series Machine Learning Methods

**Keywords:** Explainable AI, Machine Learning, Multivariate Time Series, Social Impact

**Abstract:** This thesis aims to enhance the performance and explainability of multivariate time series machine learning methods, and derive new insights from the new methods developed about two real-world applications.

Firstly, a new performance-explainability framework to assess and benchmark machine learning methods is introduced. The framework details a set of characteristics that systematize the performance-explainability assessment of existing machine learning methods.

Then, this thesis presents three new ensemble methods. The first one, DMSEEW, is a new stacking ensemble method for earthquake early warning which provides faithful and local explanations. DMSEEW improves the detection of earthquakes with damaging potential. In particular, it detects all the large earthquakes with a precision of 100% on a real-world dataset, which is critical for an earthquake early warning system. The second one, LCE, is a new hybrid ensemble method for dairy resource monitoring. It relies on a post-hoc model-agnostic explainability method, SHAP, which offers more informative explanations that could be useful to broader audiences compared to the first method. However, the enhanced explanations come at the cost of faithfulness, which is a prerequisite for numerous applications. Therefore, the third method, XEM, is an extension of the hybrid ensemble method which integrates faithfulness with explainability by design. XEM competes with the level of information and the audience of the post-hoc model-agnostic explainability method while maintaining performance. XEM outperforms the current state-of-the-art MTS classifiers on the public UEA datasets.

Next, a new pattern-based MTS classifier for dairy resource monitoring is proposed (XPM), an easy-to-understand model with explanations accessible to a wider audience compared to the ensemble methods.

Lastly, this thesis presents a new convolutional neural network for MTS classification, XCM, which outperforms XEM on the public UEA datasets. In addition, XCM architecture enables faithful and precise identification of the observed variables and timestamps of the input data that are important for predictions using the gradient-based post-hoc model-specific method Grad-CAM. XCM with Grad-CAM exhibits better performance than the ensemble method LCE with SHAP on the dairy resource monitoring application, while enhancing explainability by providing faithful and more informative explanations. Furthermore, XCM detects around 20% more key events than a commercial solution reference on a dairy resource monitoring real-world dataset, while having the same precision.