UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

REPRESENTATION OF ASTRONOMICAL TIME SERIES USING INFORMATION RETRIEVAL THEORY

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN
CIENCIAS MENCIÓN COMPUTACIÓN

FRANCISCO JAVIER MUÑOZ PONCE

PROFESOR GUÍA:
JÉRÉMY BARBAY LEFEVRE

PROFESOR CO-GUÍA:
FRANCISCO FÖRSTER BURÓN

MIEMBROS DE LA COMISIÓN:
CLAUDIO GUTIÉRREZ GALLARDO
JOSÉ MANUEL SAAVEDRA RONDO
GUILLERMO CABRERA VIVES

SANTIAGO DE CHILE
2022

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS
MENCIÓN COMPUTACIÓN
AUTOR: **FRANCISCO JAVIER MUÑOZ PONCE**
PROF. GUÍA: JÉRÉMY BARBAY LEFEVRE
PROF. CO-GUÍA: FRANCISCO FÖRSTER BURÓN

REPRESENTACIÓN DE SERIES DE TIEMPO ASTRONÓMICAS UTILIZANDO TEORÍA DE RECUPERACIÓN DE LA INFORMACIÓN

Series de Tiempo son un tipo de dato ampliamente utilizado en muchos campos como ciencias, ingeniería, finanzas o industria, para la clasificación de objetos astronómicos, análisis de indicadores económicos o análisis de fenómenos meteorológicos entre otros. La mayoría de los trabajos propuestos en esta área son diseñados para Series de Tiempo regularmente muestreadas y no aplican cuando la Serie de Tiempo presenta muestreo irregular y variables multi-dimensionales, como ocurre usualmente en Astronomía con las Series de Tiempo multi-banda.

En esta tesis estudiamos métodos de representación de Series de Tiempo y los aplicamos al desafiante problema de clasificación en grandes conjuntos de datos astronómicos. Proponemos un nuevo método de representación de Series de Tiempo, llamado IBOPF (Irregular Bag-of-Pattern Features), el cual es una extensión del clásico BOPF, pero adaptado para Series de Tiempo irregulares y multivariadas. Adicionalmente, hemos extendido nuestro método para aplicaciones de múltiples cantidades estadísticas y múltiples niveles de resolución en un intento de incrementar el rendimiento de la representación, a estas extensiones las hemos llamado Extended IBOPF. IBOPF calcula los vectores de características utilizando Teoría de Recuperación de la Información, transformando la Serie de Tiempo en secuencias de palabras, las cuales son representadas en vectores compactos a través de métodos de selección de features o reducción de dimensión.

Para las evaluaciones experimentales utilizamos el set de datos PLaSTiCC (The Photometric LSST Astronomical Time Series Classification Challenge), un set de datos altamente desbalanceado con un set de entrenamiento no representativo. El método propuesto es comparado con el método del estado-del-arte AVOCADO en clasificación, búsqueda por similitud y tiempo computacional. En general, los resultados muestran que AVOCADO supera nuestro método propuesto en clasificación (0.82 y 0.65 acc., respectivamente), y búsqueda por similitud (0.67 y 0.34 mAP@10, respectivamente), pero nuestro método tiene menores tiempos computacionales (256 ± 66 y 4 ± 1 ms por Serie de Tiempo, respectivamente). Sobre los resultados concluimos que aunque es posible aplicar IBOPF a Series de Tiempo Irregulares y Multivariadas, se necesita realizar más estudios y ajustes para producir resultados competitivos, en donde hemos detallado algunas posibles líneas de investigación futura.

Abstract

Time Series data are used in many different fields such as science, engineering, finance or business for tasks like astronomical object classification, economic indicator analysis or Meteorologic phenomenon analysis, among others. Most of the works proposed in this area are designed for regular sampled Time Series and cannot be applied for irregularly sampled and multivariate data such as the Astronomical Time Series, also known as Multi-band Time Series.

We study Time Series representation methods and apply them to the challenging problem of classification in large astronomical datasets. We propose a new representation method, called Irregular Bag-of-Pattern Features (IBOPF), which is an extension of classical BOPF adapted for Irregular Multivariate Time Series. This proposed method is extended to use multiple statistical quantities and levels of resolutions in an attempt to produce more representative feature vectors, proposing what we call Extended IBOPF. IBOPF computes features using Information Retrieval theory, transforming Time Series into sequences of words, which are then represented in compact vector form using a feature selection or dimension reduction method.

For experimental evaluations we use The Photometric LSST Astronomical Time Series Classification Challenge (PLaSTiCC), a highly unbalanced dataset with non-representative train-set. The proposed approach is compared to the state-of-the-art method AVOCADO on classification, similarity search and computing times. In general, the results show that AVOCADO outperforms our proposed method in classification (0.82 and 0.65 acc., respectively) and similarity search (0.67 and 0.34 mAP@10, respectively), however, our method achieves smaller computing times on the same dataset (256 ± 66 ms and 4 ± 1 ms per Time Series, respectively). We see that while it is possible to apply IBOPF to Irregular Multivariate Time Series, the method needs further explorations and optimizations to produce competitive results, where we outline some potential future research directions.

A mis padres,
por su amor y apoyo incondicional.

Agradecimientos

Me gustaría agradecer al profesor Jérémy Barbay por su confianza, comprensión y guía, lo cual me ayudo durante todas las etapas de mi tesis. Al profesor Francisco Förster por sus conocimientos y consejos, sin los cuales no podría haber finalizado mi investigación. Gracias también a los miembros de la comisión por tomarse el tiempo de leer y corregir esta tesis.

Un agradecimiento especial a mis padres, hermanos y amigos, por su continuo apoyo y comprensión durante todo este proceso. No podría haberlo logrado sin ustedes, especialmente durante todos estos meses de pandemia.

Table of Content

1. Introduction **1**
1.1. Time Series . . . 1
1.1.1. Time Series in general . . . 1
1.1.2. Time Series in Astronomy . . . 2
1.2. Problem statement . . . 4
1.3. Hypothesis . . . 4
1.4. Objectives . . . 4
General Objective . . . 5
Specific Goals . . . 5
1.5. Methodology . . . 5
1.6. Contributions . . . 6
1.7. Outline . . . 7

2. Related work **8**
2.1. Key concepts on Time Series representation . . . 8
2.1.1. Time Series data . . . 8
2.1.2. Basis of representation method . . . 9
2.1.3. Data mining applications. . . . 9
2.2. Similarity measures . . . 10
2.2.1. Minkowski distance . . . 11
2.2.2. Cosine Similarity . . . 11
2.2.3. Dynamic Time Warping (DTW) distance . . . 12
2.2.4. Edit Distance . . . 13
2.2.5. Similarity measures for irregular-multivariate Time Series . . . 14
2.3. Efficient similarity search . . . 14
2.3.1. lower bounding distance . . . 15
2.3.2. indexing strategies . . . 16
2.3.2.1. One-dimensional data . . . 16
2.3.2.2. High-dimensional data . . . 16
2.3.2.3. Irregular sampled and multivariate data . . . 16
2.4. Time Series Representations . . . 17
2.4.1. Regular-Univariate data approaches . . . 17
2.4.1.1. Feature extraction . . . 17
2.4.1.2. Piecewise approximation . . . 18
2.4.1.3. Symbolic representation and Bag-of-Words . . . 21
2.4.2. Regular-Multivariate data approaches . . . 22
2.4.3. Irregular-Univariate data approaches . . . 23

2.4.4. Irregular-Multivariate data approaches 25
2.5. Discussion 27
2.5.1. Data mining application 27
2.5.2. Research directions 29

3. Information Retrieval theory for Time Series 30
3.1. Time Series Transformation 30
3.1.1. Document representation 30
3.1.2. Numerosity reduction 32
3.2. Vector space model 32
3.2.1. Bag-of-Words model 33
3.2.2. TF-IDF model 34
3.2.3. Sub-linear TF weights 35
3.2.4. Cosine normalization 35
3.2.5. Class based TF-IDF 35
3.3. Compact vectors 36
3.3.1. Latent Semantic Analysis (LSA) 36
3.3.2. One-way Analysis of Variance (ANOVA) 38
3.3.3. One-way Multivariate ANOVA (MANOVA) 40
3.4. Summary 42

4. A new representation method for astronomical Time Series 43
4.1. Irregular time Series 43
4.1.1. Pattern extraction 43
4.1.2. Pattern segmentation 45
4.2. Irregular Bag-of-Pattern Feature approach 45
4.3. Generalized approach for Multivariate cases 47
4.3.1. Compact representation using MANOVA 48
4.3.2. Compact representation using LSA 49
4.4. Extensions on Irregular Bag-of-Pattern Feature 50
4.4.1. Including Multi-Quantity representation 50
4.4.2. Including Multi-Resolution representation 54
4.5. Discussion 54
4.5.1. Proposed methods 55
4.5.2. Computational complexity 56
4.5.3. Data Mining application 56

5. Experimental Evaluation 57
5.1. Implementation 57
5.2. Dataset 58
5.3. Literature methods for comparison 61
5.4. Parameter explorations 61
5.4.1. Multi-Quantity search 63
5.4.2. Optimal Levels of resolution 65
5.5. Classification Comparison with the State-of-the-art 66
5.5.1. Optimal compact technique 66
5.5.2. Classification on full test set 68

5.5.3. Extra case study on classification 70
5.6. Similarity search experiments 73
5.7. Computing time experiments 73

6. Conclusion **75**
6.1. Contributions 75
6.2. Discussion 76
6.2.1. Proposed approach 76
6.2.2. Representation method performance 77
6.2.3. Classification comparissons 78
6.2.4. Similarity search 80
6.2.5. Computational complexity 80
6.3. Future work 81

Bibliography **84**

Annexed **93**
Additional classification figures 93

List of Tables

2.1. Summary of studied (dis)similarity measures, specifying the paper that defined the measure, the theoretical time complexity, if it is considered a metric or not, if it has a lower-bounding version ($Dist_{LB}$) for fast computing and for which data type it was designed. 14

2.2. Distribution of studied work on the four different groups of Time Series data defined by sampling (regular or irregular) and number of variables (univariate or multivariate). 17

2.3. Overview of representation methods for Regular-Univariate Time Series. Subgroups are: Feature Extraction (FE), Piecewise Segmentation (PS), Symbolic Regresentation (SR) and Bag-of-Words (BOW). 19

2.4. Overview of representation methods for Regular-Multivariate Time Series, identifying the original Article that proposed the method, the time required to build the representation, the distance measure used (if any), and some relevant characteristics. 20

2.5. Overview of representation methods for Irregular-Univariate Time Series, identifying the original Article that proposed the method, the time required to build the representation, the distance measure used (if any), and some relevant characteristics. 24

2.6. Overview of representation methods for Irregular-Multivariate Time Series, identifying the original article where the method was proposed, the time required to build the representation, the distance measure used (if any), and some relevant characteristics. 26

5.1. Summary of the object types included in the PLaSTiCC dataset. The table includes the random ID number, the full and short name of each object type, the number of objects on train set N_{train}, test set N_{test}, and the ratio between both sets N_{train}/N_{test}. A more complete summary can be found on the Unblinded release of PLaSTiCC [109] 59

5.2. Parameters required by our proposed method, describing each parameter name, symbol, criteria applied for its use and used value. 65

5.3. LightGBM classifier results on the full test-set, comparing the proposed method with the state-of-the-art method. The methods evaluated are AVOCADO as the state-of-the-art method, the proposed IBOPF-LSA method, and the combination of AVOCADO features with IBOPF-LSA features, named COMBINED. Flat-weighted metric is a classification metric using in the original AVOCADO's paper [16], which is computed from the multi-logloss metric on the training set. 69

5.4. Extra LightGBM classification experiments on subsets from the original PLaSTiCC dataset. The first subset consists of dropping the supernovae-type classes and the second subset consist of using only the supernovae-type classes. The same metrics are evaluated for both subsets. Flat-weighted metric is a classification metric used in the original AVOCADO's paper [16], which is computed from the multi-logloss metric on the training set. 72
5.5. Similarity search results using K-Nearest Neighbor approach with Euclidean distance. The input used for both methods is computed features plus metadata, as specified by AVOCADO [16]. The metric used to evaluate similarity search is the mean Average Precision at k (mAP@k) for $k = \{1, 5, 10, 20\}$. The table includes a macro average of mAP@k and separated mAP@k for each class on the dataset. 73
5.6. Computing times per representation method. For AVOCADO only 10 % of the whole test set was measured for computing times, which means that the average value could fluctuate. third column indicates the average time that takes to transform 1 time series and fourth column indicates the estimated time that could take to transform the whole test set, with 3.479.801 Time Series. 74

List of Figure

1.1. Example of irregular Multi-band Time Series with 6 passbands, each capturing data at different time instances. Each band captures data at different time instants of sources that can behave different, producing different Time Series which are mostly likely correlated. 2
1.2. Taxonomy used by the ALeRCE broker for classifying light curves, as an example of how complex the Astronomical objects are. ALeRCE covered this complex structure using 4 different models. 3
1.3. Diagram of the general methodology pipeline of proposed approach, where a model is trained from the train set, which is used to generate the feature vectors for each Time Series in the test set. 6
2.1. Example on how a confusion matrix works. On the left, a binary-classifier confusion matrix is shown, on the right, a multi-class classifier confusion matrix is shown. 10
2.2. Example on how boxplots are constructed. Where Q1, median (Q2) and Q3 are the Quartiles, and IQR is the Inter Quartile Range. The limits Q1 - 1.5IQR and Q2 + 1.5IQR defines outliers. These limits can be adjusted depending on the application, the expected distribution and the number of outliers desired. . . . 11
2.3. Example of two distance measures for the Time Series. Euclidean distance on the left, where the elements are compared one-by-one, represented by the vertical yellow lines. Dynamic Time Warping (DTW) on the right, where the best alignment is found, which can be one-to-one or one-to-many matching, this is represented by many different yellow lines going from one single point on one of the curves to many points on the other curve. As long as the matches never go back in the sequences, one point can have as many matches as required to find the best alignment. 13
2.4. Diagram of the general multistep query processing architecture. The Time Series in the dataset are preprocessed and indexed into a database, which uses a filter-and-refine techinque to solve the query. 15
3.1. Two examples of transforming a sub-sequence of a regular UTS to a SAX word using sliding window. Extracted sub-sequences are of length 30 (measure of time), the generated words have 4 characters each and the SAX alphabet is $\{A, B, C, D\}$. Resulting words are $AACD$ (top) and $DBAB$ (bottom). 31
3.2. Effects example of Numerosity Reduction (NR). For the same class event (simple peak), three Time Series are generated (Q_1, Q_2, Q_3), measuring the peak of the event at different time instants (left column). Their documents are constructed and the NR is applied (middle column). Finally, the documents are shown in the form of Histogram considering the vocabulary $\{AA, AB, BB, BA\}$. 33

3.3. Diagram of dimensionality reduction in LSA based on Truncated SVD. 37
4.1. Two-ways Sliding Window example. A fixed window width slides forward across a Time Series, reaches the end, and then slides backward. 44
4.2. Pattern segmentation example, where a sub-sequence was divided into 5 segments and for each one of them a mean value is computed. 46
4.3. Generalized Irregular Bag-of-Pattern Feature diagram 49
4.4. Example of early and late fusion schemes for statistical quantities. For the same sequence, divided into 5 segments, early-fusion and late-fusion are applied separately, generating two different word representations 50
4.5. Generalized Irregular Bag-of-Pattern Features diagram with Multi-Resolution extension 53
5.1. PLaSTiCC Light curve examples with GP-fit model, where the mean GP flux prediction is shown as a solid line surrounded by a shaded contour indicating the 1σ deviation 58
5.2. Number of observations per Time Series on train set grouped by DDF/WDF surveys. 60
5.3. Class distribution on the PLaSTiCC train set and augmented train set 61
5.4. Results of Multi-Quantity evaluations for the statistical quantities Mean (Me), Min (Mn), Max (Mx), Min-Max (mm), Trend (Tr) and Variance (Va). The notation follows the pattern s_1-s_2-... where each s_i is a statistical quantity, single or composed through early-fusion, and all s_i are combined through late-fusion 62
5.5. Results of Multi-Resolution evaluations for an incremental number of levels of resolution. On each level, different values for word length ω and window width T were tested and the best pair on each level is selected 64
5.6. Comparison of different combinations of IBOPF with compact methods. The metric used is balanced accuracy on classification tasks considering a random 10 % of the full test-set. The figure includes configurations that combines the proposed method with state-of-the-art method, these are named *IBOPF-sparse|AVOCADO (LSA)*, *IBOPF-sparse|AVOCADO (UMAP)*, and *IBOPF-LSA|AVOCADO*. The dashed horizontal lines represent the balanced accuracy achieved by AVOCADO alone. 67
5.7. 2-dimensions visualization of compact methods LSA, UMAP with Cosine distance and Supervised UMAP with Cosine distance, on train set (fit) and test set (transform). The transform is evaluated on a small subsample of the test set for fast computing. 68
5.8. 2-dimensions visualization of AVOCADO, IBOPF-LSA and their combination, using LSA and UMAP as the dimension reduction technique. The visualization is produced from features extracted from the train set. Colors represents the classes, where it is expected that similar color groups in clusters in the visualization. 70
5.9. Confusion Matrices of classification results using LightGBM Classifier on PLaSTiCC dataset. (a) The confusion matrix of our proposed method for the full test-set. (b) The confusion matrix of our proposed method for the subset with only Supernovae-type classes. (c) The confusion matrix of AVOCADO for the subset with only supernovae-type classes. The rest of confusion matrices, including the ROC-AUC curves, can be found on Appendix **??** 71
A.1. Confusion matrix for IBOPF-LSA on full dataset using LightGBM classifier . . 93

A.2. ROC-AUC curve for IBOPF-LSA on full dataset using LightGBM classifier . . 94
A.3. Confusion matrix for AVOCADO on full dataset using LightGBM classifier . . 94
A.4. ROC-AUC curve for AVOCADO on full dataset using LightGBM classifier . . 95
A.5. Confusion matrix for IBOPF-LSA combined with AVOCADO on full dataset using LightGBM classifier 95
A.6. ROC-AUC curve for IBOPF-LSA combined with AVOCADO on full dataset using LightGBM classifier 96
A.7. Confusion matrix for IBOPF-LSA on sub-dataset without Supernovae-type classes using LightGBM classifier 96
A.8. ROC-AUC curve for IBOPF-LSA on sub-dataset without Supernovae-type classes using LightGBM classifier 97
A.9. Confusion matrix for AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier 97
A.10. ROC-AUC curve for AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier 98
A.11. Confusion matrix for IBOPF-LSA combined with AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier 98
A.12. ROC-AUC curve for IBOPF-LSA combined with AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier 99
A.13. Confusion matrix for IBOPF-LSA on sub-dataset with only Supernovae-type classes using LightGBM classifier 99
A.14. ROC-AUC curve for IBOPF-LSA on sub-dataset with only Supernovae-type classes using LightGBM classifier 100
A.15. Confusion matrix for AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier 100
A.16. ROC-AUC curve for AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier 101
A.17. Confusion matrix for IBOPF-LSA combined with AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier 101
A.18. ROC-AUC curve for IBOPF-LSA combined with AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier 102

Chapter 1

Introduction

In this chapter, we introduce the reader to our work, describe our motivations (Section 1.1) and discuss the focus of this research, describing our Problem Statement (Section 1.2), Hypothesis (Section 1.3), Goals (Section 1.4) and Methodology (Section 1.5). Finally, we describe our contributions (Section 1.6) and the organization of this document (Section 1.7).

1.1. Time Series

Sciences like Astronomy, Meteorology or Medicine, among many other fields such as engineering, finance, business or industry, require the collection of data for solving problems, performing experiments or creating new theories. The data are usually collected at different times, presenting in some cases temporal variations with relevant information. To analyze and visualize this information, the data are sorted by time, producing a *Time Series*.

1.1.1. Time Series in general

In general, diverse data mining tasks can be performed on Time Series Data, for example, anomaly detection, classification, clustering or similarity search. In the past few decades, many works have been proposed to solve these data mining tasks using representation methods or indexing techniques [1, 4, 8, 41, 44, 68, 107], some of them addressing large dataset problems. However, most of the works focus on the simplest and most common variation of Time Series, called *regular Univariate Time Series* (from now on abbreviated as *regular UTS*).

When collecting data with temporal variations, it is desired to have control over the measurement intervals producing regular time intervals. A Time Series with regular time intervals can be simplified to a sequence, dropping the time component and simplifying the application of any data mining task. In general, the data collection aims to generate a *regular Time Series* and most of the work out there works with this kind of Time Series. Some complex variations involve multivariate data [14, 36, 81, 120] or Time Series with missing data [15], however, they mostly consider regular time intervals.

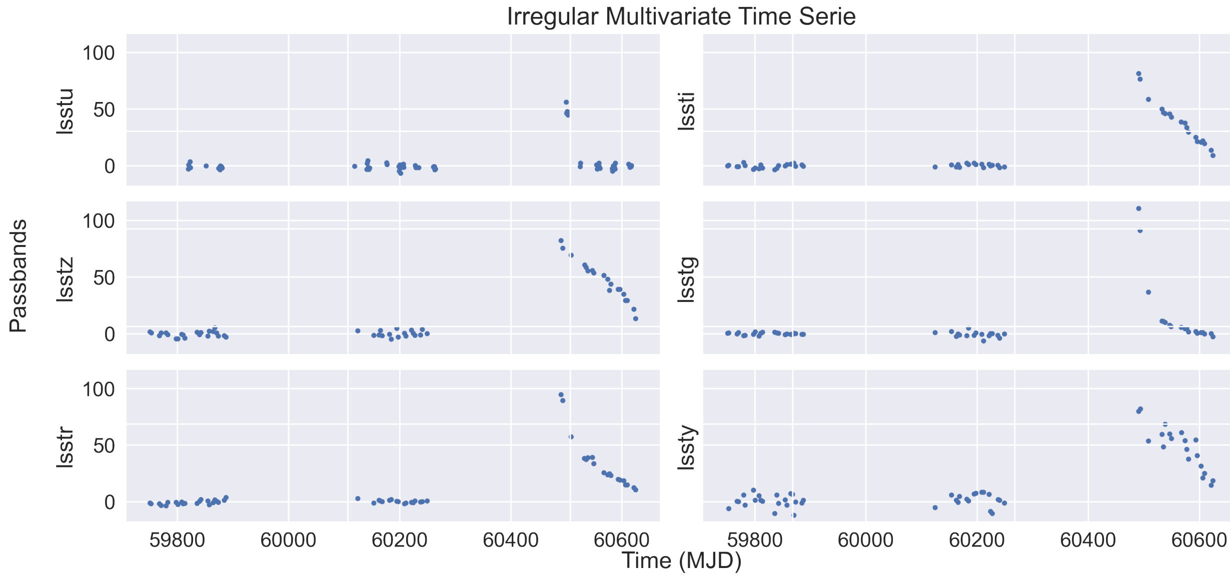


Figure 1.1: Example of irregular Multi-band Time Series with 6 passbands, each capturing data at different time instances. Each band captures data at different time instants of sources that can behave different, producing different Time Series which are mostly likely correlated.

1.1.2. Time Series in Astronomy

Astronomy is a science where every experiment is based on the collection of electromagnetic waves (light) coming from astronomical objects such as Stars or Galaxies, or events such as Supernovae, transit of exoplanets or gravitational microlensing. These observations are performed using telescopes and are conditioned by the earth rotation, earth translation, meteorological conditions or instrumental conditions, producing Time Series with irregular intervals between measures. Furthermore, due to the nature of electromagnetic waves, the full electromagnetic spectrum cannot be measured in one single variable. To address this, astronomers have defined the measured variables as frequency intervals called *bands* or *passbands*.

For every astronomical object or event, several bands can be observed producing a collection of highly correlated Univariate Time Series (UTS) of the same object, where each UTS can be measured at different time instants (non-simultaneous multivariate measures) or have a different number of samples. These kinds of Time Series are usually identified as Multivariate Time Series (MTS), but Astronomers usually refer to such Time Series as *Multi-band Time Series* [45, 48, 82, 112] (from now on abbreviated as MbTS). In the most general case, we will be working with irregular MbTS (equivalent to irregular MTS) which are the most complex variation of Time Series (see Figure 1.1). Although various literature work studied different data mining tasks in the astronomical Time Series considering single-band [10, 21, 50, 74, 88, 111] and multi-band [16, 49], there are still some challenges that need to be solved, which are described below.

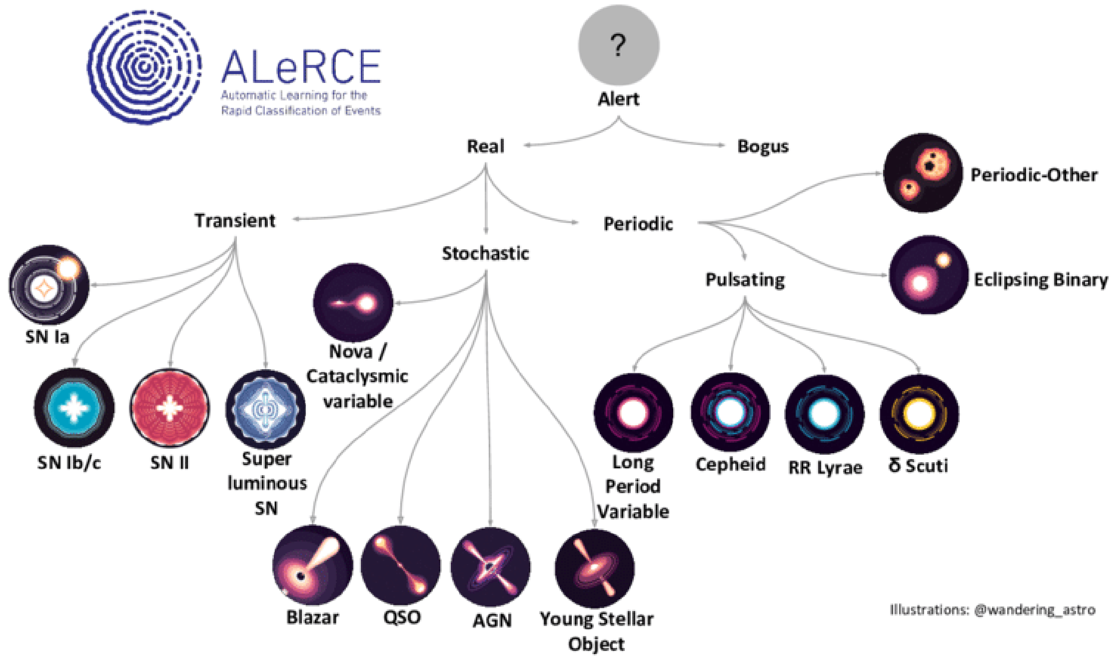


Figure 1.2: Taxonomy used by the ALeRCE broker [39] for classifying light curves, as an example of how complex the Astronomical objects are. ALeRCE covered this complex taxonomy structure using 4 different models.

In astronomy, the most used data mining tasks are classification, clustering and anomaly detection. Two approaches are used to solve these tasks: neural networks and (dis)similarity searches. The first has been widely explored in astronomy where, for example, Recurrent Neural Networks (RNN) [21, 88] with Long Short-Term Memory (LSTM) units [100] or its variation Phased LSTM units [31] is used over irregular UTS. The base idea of neural networks is to train a model to solve a particular data mining task based on features (i.e. supervised learning) or the learning of features (i.e. unsupervised learning). On the other hand, a (dis)similarity approach aims to solve the data mining task by comparing Time Series, looking for their similarities or dissimilarities. Only a few works can be found about solving data mining tasks on astronomical datasets using (dis)similarity approaches [74, 111], but limited to one-band Time Series.

The (dis)similarity approach is more powerful than the neural networks approach in the sense that it supports more applications. As an example, suppose we have a dataset with N irregular MbTS, and we want to get the k most similar objects to a query Time Series Q, or every object with a (dis)similarity value under a threshold ϵ compared to Q. A neural model like RNN cannot solve this kind of query. Furthermore, getting the k most similar objects, known as k-nearest neighbor, can be used on many other data mining tasks such as classification, clustering or anomaly detection.

The challenges related to the problem of (dis)similarity search in astronomical Time Series are: 1) the complex nature of irregular MbTS , 2) the complex taxonomy of astronomical objects, 3) the large size of the datasets, and 4) the need for fast processing. The first challenge is related to the irregular time intervals and the non-simultaneous multivariate measures. The second challenge is illustrated in Figure 1.2 where a complex ramification of classes is constructed. The third and fourth challenges are related to the recent expansion in the astronomical data collection. New telescopes and new technologies produce a large amount of data in short periods of time, and thus there is a need to process all this data faster. A series of related works about speeding up (dis)similarity approach algorithms for large datasets have been proposed in the past decades, mostly focused on the use of indexing structures. However, they ignore irregular Time Series and especially irregular MbTS. In this work we cover these challenges by designing a new representation method for irregular MbTS based on Information Retrieval (IR) theory. Challenges 1) and 2) are addressed by the representation method while challenges 3) and 4) can be addressed by the IR theory of indexing inverse documents. The experimental evaluation of our approach is based on efficiency, speed and performance on classification using Light Gradient Boosting Machine and similarity search using k-Nearest Neighbor.

1.2. Problem statement

There is a need in astronomy to solve data mining tasks such as Classification or Outlier Detection on irregular MbTS, while optimizing the space usage and computing times. In order to achieve this, three main factors have to be handled: a fast representation method that extracts relevant features from the irregular MbTS, a dimensionality reduction or feature selection technique that produces a compact representation, and a data mining algorithm that works on the given representation vector. Defining our problem formally, for a database B (train set), and a query Time Series Q (test set), we want to get the most likely class within B to which Q may belong.

1.3. Hypothesis

H_1 A representation method for irregular Multivariate Time Series (i.e. irregular MbTS) based on Information Retrieval theory, using symbolic approximation with different statistical quantities, discretization alphabets and/or resolution windows, yields faster means of extracting representative features (under classification or similarity search metrics) than state-of-the-art methods.

H_2 A representation method for irregular Multivariate Time Series (i.e irregular MbTS) based on Information Retrieval theory, using symbolic approximation with different statistical quantities, discretization alphabets and/or resolution windows, yields better classification accuracy or similarity search mean Average Precision at k (mAP@k) than state-of-the-art methods.

1.4. Objectives

General Objective

Design, develop and evaluate, in theory and practice, a representation method based on Information Retrieval theory that supports similarity search and classification algorithms.

Specific Goals

O_1 *Survey* existing techniques to work with Time Series under the (dis)similarity search approach, for Astronomy and other fields.

O_2 Identify the most relevant factors to be considered to develop a good representation method of irregular MbTS using Information Retrieval theory.

O_3 Design a base method using the explored theory that works on irregular MbTS.

O_4 Explore and design extensions and generalizations to this base method in order to produce a more representative feature vector.

O_5 Develop and implement the designed approach.

O_6 Evaluate the performance and efficiencies of implemented algorithms when solving classification tasks, including a reference method for comparison.

1.5. Methodology

Our approach is based on the Information Retrieval theory, and it is summarized in Figure 1.3. Here the approach starts by extracting sub-sequences from the Time Series in the dataset (for training), and for each sub-sequence it applies a pattern discretization process that produces a symbolic representation in the form of a word, transforming the whole time Series into a document of words. Then, for all of the documents in the dataset, Information Retrieval theory is applied so that a compact representation model is trained, which is then applied to a query Time Series to produce the features vector. This whole pipeline has a series of constraints in order to work on irregular Multivariate Time Series, which are addressed in Chapters 3 and 4. For the Sliding Window, an adapted version called *two-ways Sliding Window* is applied. For the pattern discretization, two general extensions are proposed, called *Multi-Quantity* and *Multi-Resolution*. For the compact representation model, two options are described, Latent Semantic Analysis (LSA) and Multivariate Analysis of Variance (MANOVA).

To evaluate our proposed methods, we use Light Gradient Boosting Machine (LightGBM) for classification and K-Nearest Neighbor (k-NN) for similarity search. Classification experiments are limited to LightGBM only since it has been shown to be one of the best methods to classify the PLaSTiCC dataset [16]. For comparison, we use the state-of-the-art method AVOCADO, where we also evaluate the computing times required to produce the feature vector using our proposed method and state-of-the-art method.

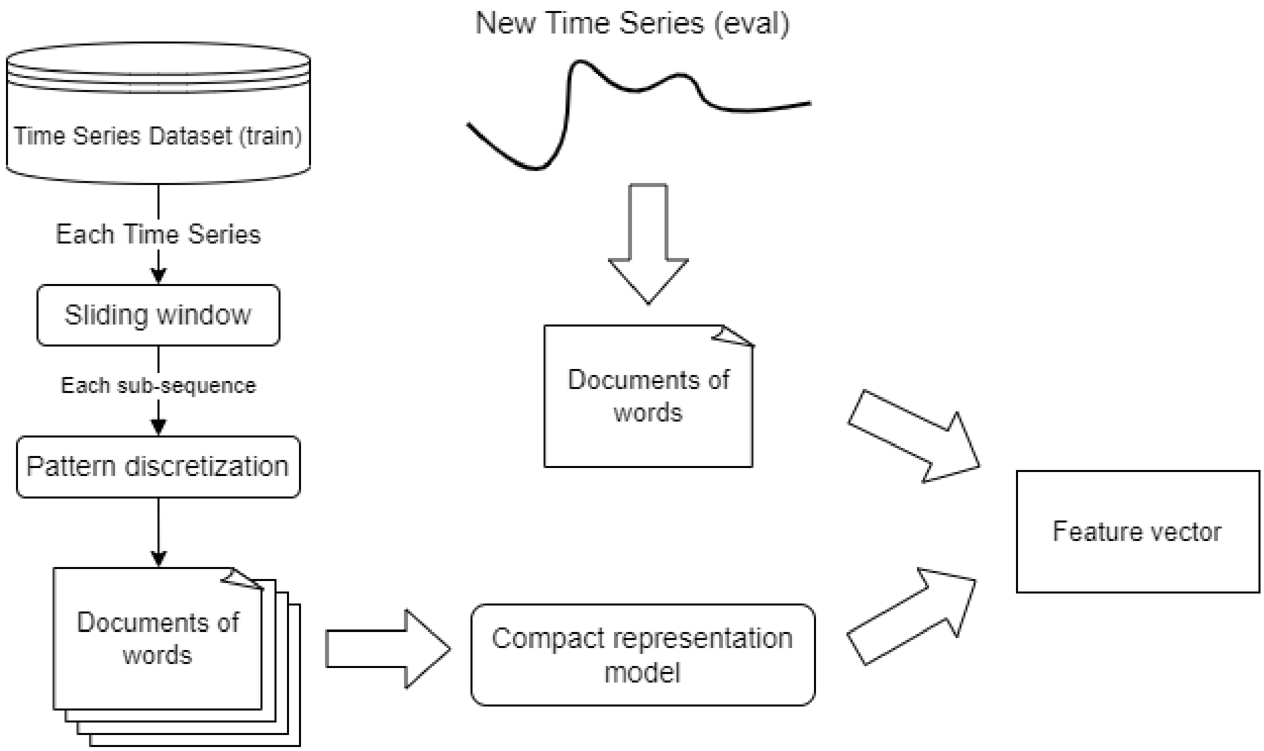


Figure 1.3: Diagram of the general methodology pipeline of proposed approach, where a model is trained from the train set, which is used to generate the feature vectors for each Time Series in the test set.

The source code for all of our implementations and evaluations can be found on Github as open-source software [1].

1.6. Contributions

The main contributions of this thesis are:

- **A Survey** of the existing methods and techniques designed to solve (dis)similarity search in Time Series from Astronomy and other fields.
- **Formalization of Information Retrieval theory applied to Time Series**, where we have detailed the vector space model more suited for irregular Time Series datasets and how they can be transformed into a compact form.
- **Proposition of Irregular Bag-of-Pattern Feature** which is an extension of an existing method called Bag-of-Pattern Features but applied to irregular Time Series.
- **Generalization of Irregular Bag-of-Pattern Features** where we adapt the method to work on irregular Multivariate time Series.

[1] https://github.com/frmunozz/irregular-bag-of-pattern

- **Multi-Quantity and Multi-Resolution extensions for Irregular Bag-of-Pattern Feature** as extensions for the adapted method which improve the performance of the resulting feature vector on classification tasks.
- **Experimental validation** of our proposed approach, with comparisons to the state-of-the-art method using *PLaSTiCC* dataset.
- **Open-source software** with our approach available for others to duplicate our experimental work and further applications.

1.7. Outline

This thesis work is organized in 6 Chapters as follows:

- **Chapter 1**: Presents the introduction, motivation, objectives and hypothesis of this thesis.
- **Chapter 2**: Presents a review of related works to this thesis, grouped in three categories: representation methods, (dis)similarity measures and indexing techniques for Time Series. In addition, a few key concepts are defined formally to allow the reader a better comprehension of the problem. At the end of the chapter, a brief discussion is made about the challenges and opportunities of research for the astronomical Time Series.
- **Chapter 3:** Describes and defines the most important concepts of Information Retrieval theory for Time Series data, focusing on vector space models and feature reduction.
- **Chapter 4:** Presents our Bag-of-Pattern Feature method designed for irregular Time Series. Including the generalization for irregular Multivariate Time Series and the extensions using multiples statistical quantities (Multi-Quantity) and multiple levels of resolutions (Multi-Resolution).
- **Chapter 5:** Presents our experimental evaluation performed on PLaSTiCC dataset. A parameter exploration was performed which validates the extensions for Multi-Quantity and Multi-Resolution. Experiments on classification tasks and computing times for our proposed method and an additional literature method (for comparison) are finally performed.
- **Chapter 6:** Presents the conclusions of this thesis and a perspective on directions for future work.

Chapter 2

Related work

In this chapter, we review the basic concepts and related work necessary to understand the problem and goals of this thesis, providing evidence for its relevance. First, we define formally what a time series is and its variations (Section 2.1), including some data-mining concepts. Then we describe the existing (dis)similarity measures (Section 2.2), indexing structures for efficient similarity search (Section 2.3), and the Time Series Representation approaches (Section 2.4) in Astronomy and other fields like Medicine or Finance. We conclude this chapter with a discussion about the reviewed works (Section 2.5).

2.1. Key concepts on Time Series representation

Before we start reviewing the literature, some key concepts need to be introduced related to Time Series data (Section 2.1.1) type and representation methods (Section 2.1.2).

2.1.1. Time Series data

To create a Time Series data, several measurements have to be made at different time instants. If the time intervals between consecutive measurements are always the same, it is called **regular sampling**. Otherwise, it is called **irregular sampling**. Furthermore, in some application domains, using one variable to measure an experiment is not enough for further analysis, and thus more variables are measured. When a Time Series has one single variable measured, it is called **Univariate Time Series (UTS)**, and if it has two or more, it is called **Multivariate Time Series (MTS)** [98]. In some applications domains (e.g., Astronomy), the MTS are also called *Multi-band Time Series* where the variables are identified as *bands* [76].

As we can see, four different groups are identified for Time Series data:

Regular-Univariate (RU) data: Time Series data with regular sampling and one variable measured (UTS)

Regular-Multivariate (RM) data: Time Series data with regular sampling and two or more variables measured (MTS)

Irregular-Univariate (IU) data: Time Series data with irregular sampling and one variable measured (UTS)

Irregular-Multivariate (IM) data: Time Series data with irregular sampling and two or more variables measured (MTS)

In the case of IM data, the multivariate measures do not have to be made simultaneously. This means that we can have different number of measures for each variable with different measuring times. This is a very complex challenge when the variables measured are correlated, and it is the main reason for the shortage of works on this application domain.

2.1.2. Basis of representation method

As previously mentioned, a representation method is basically a transformation from the raw-space Time Series to a feature-space sequence. Furthermore, the ideal representation method should address dimensionality reduction, fast construction, support similarity measures and efficient reconstruction. However, previous experiments could not provide conclusions for the existence of such a method [122]. For example, a method cannot achieve dimensionality reduction and efficient reconstruction at the same time, there is a trade-off between those two properties. The same goes for fast construction and efficient reconstruction. For the cases of Multivariate Time Series (MTS), the ideal representation method should also consider the correlation between variables, which yields to slower constructions.

For general purpose, we will include in our study any representation method that satisfied at least one of the properties of the ideal representation method.

2.1.3. Data mining applications.

Here we will describe some key concepts on data mining applications that are going to be used during this work.

For data mining applications, we focus our review on classification and similarity search, which are among the most commonly used tasks. There are very simple classifiers such as K-Nearest Neighbor, which takes a query Time Series Q and compute its distance to all Time Series in a labeled dataset R, finding the closest in terms of distance or similarity between vectors and assigning the respective label to Q. To optimize the query, specialized data structures can be used, such as M-Trees [26] or TS-Trees [6], among many others. Alternatively, there are more complex classifiers such as Decision trees, where a tree-like structure of decisions is used to classify a Time Series depending on the features values. The decision can be of the form: if feature A has a value less than 0.5 (or any other threshold), then take the left branch, otherwise, take the right branch. These methods are escalated to very complex and powerful tree-structures of decisions, even combining different approaches. For example, Light Gradient Boosting Machine (LightGBM) combines a Gradient Boosting Machine framework with decision trees to produce a fast and robust classifier.

To visualize the performance of a classifier over a dataset, we usually use a confusion

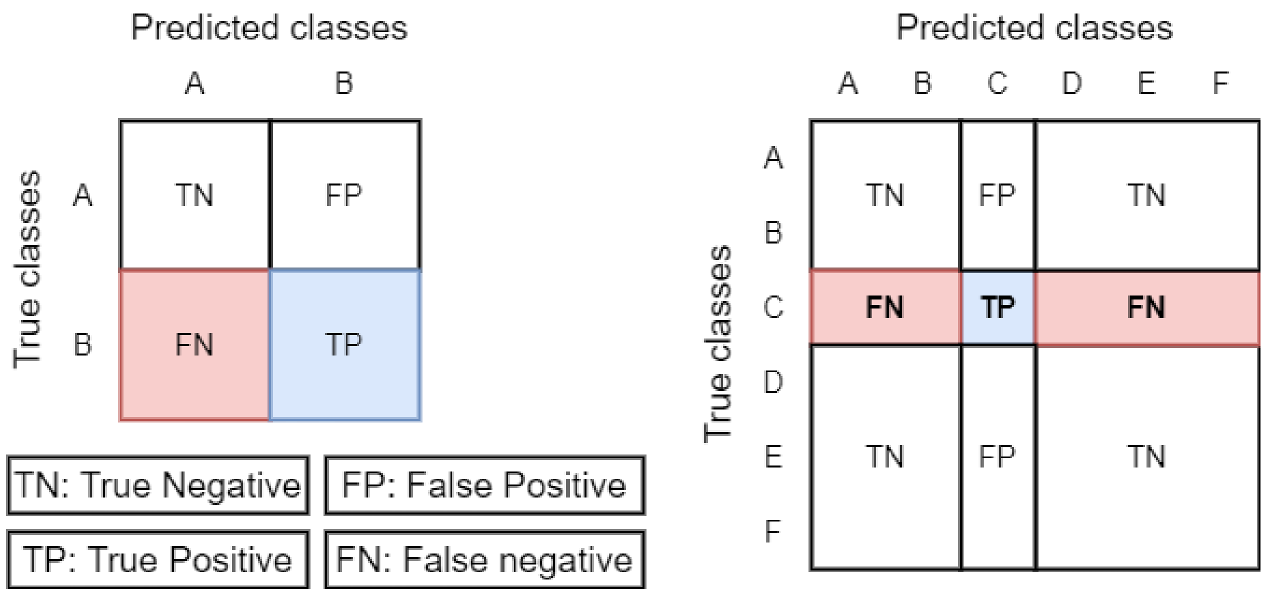


Figure 2.1: Example on how a confusion matrix works. On the left, a binary-classifier confusion matrix is shown, on the right, a multi-class classifier confusion matrix is shown.

matrix. With a confusion matrix we can visually observe the 4 possible outcomes when classifying a dataset, True-Positive (TP), False-Positive (FP), True-Negative (TN), False-Negative (FN). For example, if we have two classes A and B, where we define A as non-B. Predicting B for a B object is a TP (good prediction), predicting B for a non-B is a FN (bad prediction), predicting non-B for a non-B is a FP (good prediction), and predicting non-B for a B is a FN (bad prediction). Figure 2.1 illustrates this Confusion matrix and shows how it can be extended for multi-class classifiers. The key fact is that the diagonal of the confusion matrix will always show the good predictions and the non-diagonal ones are the bad predictions. These confusion matrices can be normalized by row in order to get a predicted probability per class.

Another useful way of describing results on data mining application is the Five-Number Summary, used when experiments involve repetitions and those repetitions show certain distributions. Here we compute the quartiles Q_1, Q_2, Q_3, and the interquartil range $IQR = Q_3 - Q_1$, which gives account of data dispersion. Then, the Five-Number Summary consist of: minimum value of the results, Q_1, median (Q_2), Q_3, and maximum value of the results. For outliers identification, the IQR is used, where value above $Q_3+1.5IQR$ or below $Q_1-1.5IQR$ are considered outliers and the respective minimum or maximum values are replaced, transforming the summary $(min, Q_1, Q_2, Q_3, max)$ into $(Q_1 - 1.5IQR, Q_1, Q_2, Q_3, Q_3 + 1.5IQR)$. Additionally, The Five-Number Summary can be visualized using box-plots (See Figure 2.2).

2.2. Similarity measures

A representation method by itself cannot solve any problem of interest since on its basis it is only a transformation from raw data space to a feature data space. In general, a similarity or dissimilarity measure is applied to compare Time Series on the feature data space and solve

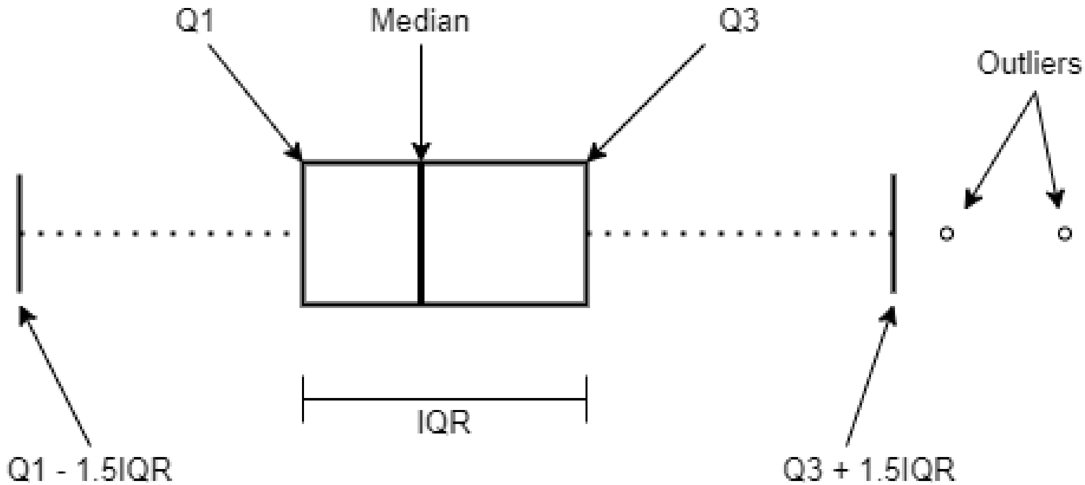


Figure 2.2: Example on how boxplots are constructed. Where Q1, median (Q2) and Q3 are the Quartiles, and IQR is the Inter Quartile Range. The limits Q1 - 1.5IQR and Q2 + 1.5IQR defines outliers (lower than/higher than respectively). These limits can be adjusted depending on the application, the expected distribution and the number of outliers desired.

data mining tasks based on similarity search. Some of the most used similarity/dissimilarity measures in literature are described next.

2.2.1. Minkowski distance

The Minkowski distance is a similarity measure between two points in vector space. Different normalization are identified in Minkowski space, the most famous being the L2-norm, also known as the *Euclidean distance* (Figure 2.3 at left):

$$\text{L1-Norm}(\boldsymbol{P}, \boldsymbol{Q}) = \|\boldsymbol{P} - \boldsymbol{Q}\|_2 = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{2.1}$$

Another example is the L1-norm, also known as the *Manhattan distance*:

$$\text{L2-Norm}(\boldsymbol{P}, \boldsymbol{Q}) = \|\boldsymbol{P} - \boldsymbol{Q}\|_1 = \sum_{i=1}^{n}|p_i - q_i| \tag{2.2}$$

where both vectors (Time Series) $\boldsymbol{P}$ and $\boldsymbol{Q}$ have the same number of samples n. These distance measures are used in Time Series due to their linear complexity. However, they are very simple and can fail in more complex cases scenarios [22].

2.2.2. Cosine Similarity

The Cosine similarity is a measure of similarity between two non-zero vectors. This similarity is based on the angle generated between the two vectors and is defined as:

$$CS(\boldsymbol{P},\boldsymbol{Q}) = cos(\theta) = \frac{\boldsymbol{P}\cdot\boldsymbol{Q}}{\|\boldsymbol{P}\|\|\boldsymbol{Q}\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} p_i^2}\sqrt{\sum_{i=1}^{n} q_i^2}} \tag{2.3}$$

where $\boldsymbol{P}$ and $\boldsymbol{Q}$ are vectors (Time Series) of the same length n. If the vectors are normalized, in which case $\|\boldsymbol{P}\| = \|\boldsymbol{Q}\| = 1$, this similarity measure can be seen as the euclidean distance if we apply the conversion:

$$\begin{aligned} L_2 - Norm(\boldsymbol{P},\boldsymbol{Q})^2 &= \|\boldsymbol{P}-\boldsymbol{Q}\|_2^2 \\ &= \|\boldsymbol{P}\|_2^2 - 2\boldsymbol{P}\cdot\boldsymbol{Q} + \|\boldsymbol{Q}\|_2^2 \\ &= 2 - 2CS(\boldsymbol{P},\boldsymbol{Q}) \end{aligned} \tag{2.4}$$

2.2.3. Dynamic Time Warping (DTW) distance

The Dynamic Time Warping (DTW) distance is a technique that finds the best path of alignments between two Time Series (Figure 2.3, right). The idea is to compute a cumulative cost matrix C in the following way [98]:

$$\begin{aligned} &C[0,0] = 0 \qquad C[i,0] = \infty \qquad C[0,j] = \infty \\ &C[i,j] = cost(p_i, q_j) + min \begin{cases} C[i-1,j] \\ C[i,j-1] \\ C[i-1,j-1] \end{cases} \\ &\Rightarrow DTW(\boldsymbol{P},\boldsymbol{Q}) = C[n^P, n^Q] \end{aligned} \tag{2.5}$$

Where n^P and n^Q are the last index of Time Series $\boldsymbol{P}$ and $\boldsymbol{Q}$ respectively. The cost function $cost(p_i, q_j)$ can vary between applications, but it is mostly based on the Minkowski distances. An example of a cost function is:

$$cost(p_i, q_i) = \begin{cases} (p_i - q_i)^2 & \text{based on L2-Norm} \\ |p_i - q_i| & \text{based on L1-Norm} \end{cases} \tag{2.6}$$

DTW distance is widely used on different application domains since it works fine on Time Series with more complex variations such as time shifting or longitudinal scaling. However, since the original algorithm of DTW distance has a computational complexity of $O(n^P \cdot n^Q)$, it is not well suited for large datasets. To speed up the distance, various modifications to the algorithm have been proposed. For example, the constrained Dynamic Time Warping (cDTW) distance is an application of DTW to a reduced range around the diagonal of the cost matrix reducing the number of computations [47, 57]. There are other optimizations of DTW distance such as lower bounding distance LB_Keogh [54], DTW distance in multi-level resolutions[96] or techniques for early abandoning the DTW distance algorithm [93].

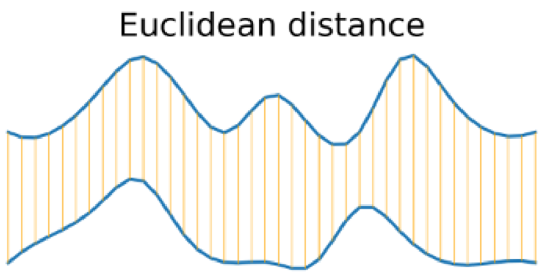


Figure 2.3: Example of two distance measures for the Time Series. Euclidean distance on the left, where the elements are compared one-by-one, represented by the vertical yellow lines. Dynamic Time Warping (DTW) on the right, where the best alignment is found, which can be one-to-one or one-to-many matching, this is represented by many different yellow lines going from one single point on one of the curves to many points on the other curve. As long as the matches never go back in the sequences, one point can have as many matches as required to find the best alignment.

2.2.4. Edit Distance

Edit-based distances are designed to measure the similarity of two symbol sequences counting the minimum number of operations (insertion, deletion, substitution) required to transform a sequence into the other [98]. Edit Distance on Real sequences (EDR) [23] was the first method to adapt edit-based distance for time series and is defined as:

$$EDR(\boldsymbol{P},\boldsymbol{Q}) = \begin{cases} n & \text{if } m = 0 \\ m & \text{if } n = 0 \\ min \begin{cases} EDR(Rest(\boldsymbol{P}), Rest(\boldsymbol{Q})) + subcost \\ EDR(Rest(\boldsymbol{P}), \boldsymbol{Q}) + 1 \\ EDR(\boldsymbol{P}, Rest(\boldsymbol{Q})) + 1 \end{cases} & \text{otherwise} \end{cases} \tag{2.7}$$

where $Rest(\boldsymbol{P})$ is the sub-sequence of $\boldsymbol{P}$ without the first element, the same for $Rest(\boldsymbol{Q})$. $subcost$ is a value defined by:

$$subcost = \begin{cases} 0 & \text{if } match(p_1, q_1) \\ 1 & \text{otherwise} \end{cases} \tag{2.8}$$

where $match(p_1, q_1)$ is True if and only if $|p_{1,x} - q_{1,x}| \leq \epsilon$ and $|p_{1,y} - q_{1,y}| \leq \epsilon$ with ϵ the matching threshold, $p_1 = (p_{1,x}, p_{1,y})$ and $q_1 = q_{1,x}, q_{1,y}$. This definition assumes that the cost of a replace, insert or delete operation is always 1. Another adaptation of edit distance for time series is Edit distance with Real Penalties (ERP) [22]. The main difference between EDP and ERP is that the last one has the properties of a metric measure, including triangle inequality property, which is used to define a lower bounding function. Another measure based on edit distance is Longest Common Sub-Sequence (LCSS) [115], which finds the largest number of

Table 2.1: Summary of studied (dis)similarity measures, specifying the paper that defined the measure, the theoretical time complexity, if it is considered a metric or not, if it has a lower-bounding version ($Dist_{LB}$) for fast computing and for which data type it was designed.

$\boldsymbol{Dist}$	**Complexity**	**Metric**	$\boldsymbol{Dist_{LB}}$	**Data type**
L2/L1-Norm [94]	$O(n)$	✓	[94]	Regular
CS [66]	$O(n)$	-	-	Regular
DTW [64]	$O(n^2)$	-	[57]	Regular
EDR [23]	$O(n^2)$	-	[23]	Regular
ERP [124]	$O(n^2)$	✓	[22]	Regular
LCSS [115]	$O(n^2)$	-	-	Regular
ACSS [117]	$O(n^2)$	-	-	Regular
TWED [74, 111]	$O(n^2)$	✓	[78]	Irregular

matching values between two sequences. This measure was later extended to All Common Sub-Sequences (ACSS) [117].

Time Warp Edit Distance (TWED) [78] is a different approach to adapt edit distance for the Time Series, in this case the operations of insert, delete and match are replaced by the operations of match, delete-x, delete-y. The idea of TWED is to provide an elastic metric for Time Series matching by taking the time difference into account when penalizing edit operations. Table 2.1 summarizes comparisons between all of these similarity measures.

2.2.5. Similarity measures for irregular-multivariate Time Series

Distance measures for Univariate Time Series (UTS) can be adapted to work in a Multivariate case (MTS). Many authors have proposed adapted methods that works on MTS [7, 14, 23, 36, 64, 91, 98, 115, 116, 118, 122] with regular sampling, for example, Correlation Based Dynamic Time Warping (CBDTW) [11]. Other works propose distance measures especially designed for multivariate application, for example: correlation measure for streaming Time Series [99], triangle distance to measure the triangle cosine between two Time Series [126] and Bounded Coordinate System (BCS) [121].

For irregularly sampled Time Series representations, measures like the DTW distance, the Time Warp Edit Distance (TWED) or the Fréchet distance [24, 32, 34] can be used on the similarity search problem. However, how to use distance measures for irregularly sampled MTS is yet an open problem.

2.3. Efficient similarity search

A Time Series similarity search can be described as a query search of a query $\boldsymbol{Q}$ in a database $\boldsymbol{B}$. If the database $\boldsymbol{B}$ is so large that does not fit in main memory (RAM), a query search can result in a sequential scan of the database, making several reads to secondary

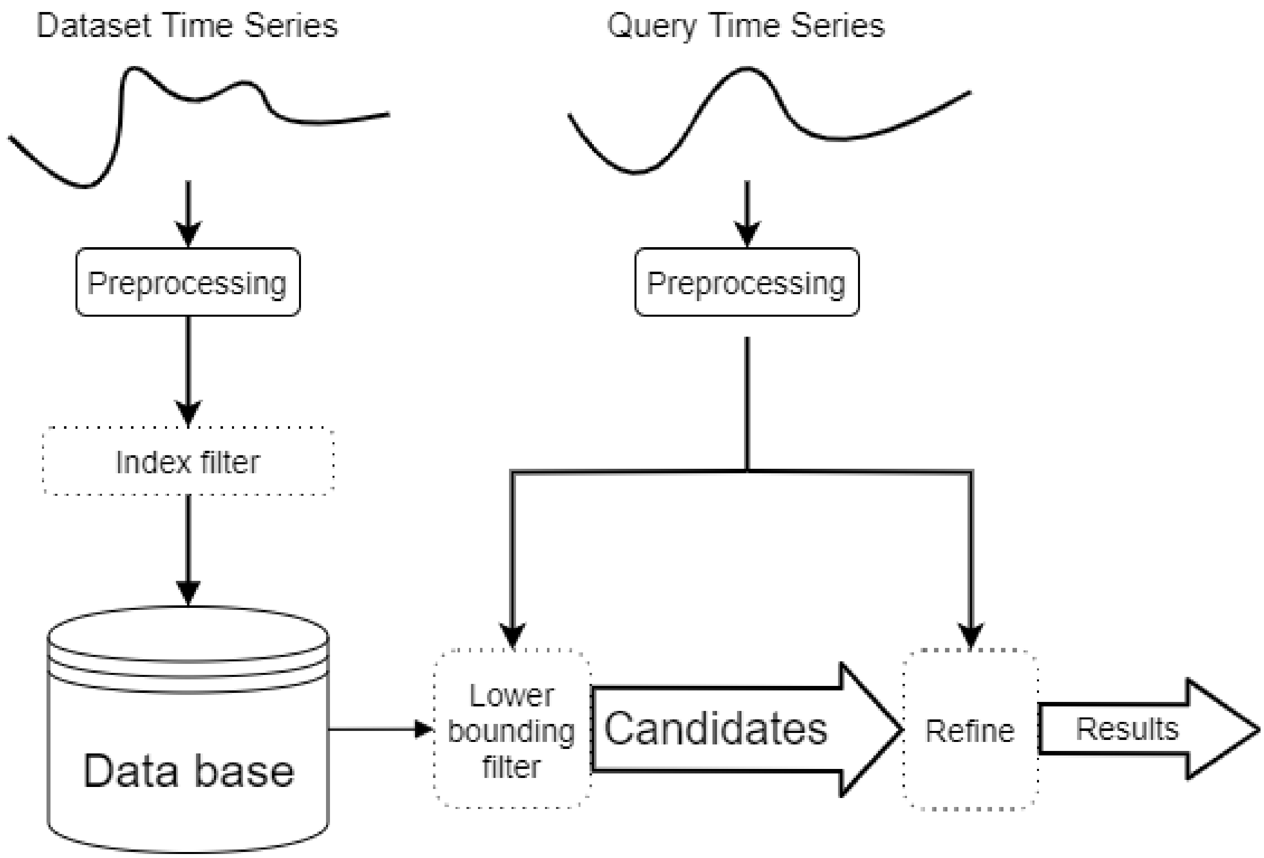


Figure 2.4: Diagram of the general multistep query processing architecture. The Time Series in the dataset are preprocessed and indexed into a database, which uses a filter-and-refine techinque to solve the query.

memory (disk), which is time-consuming. Filter-and-refine methods have been proposed to reduce computational time of similarity search in large datasets [6, 38, 115, 125]. The basic of these methods is to apply a multistep query process (See Figure 2.4), where the first step is to filter the database to get a reduced set of candidates, and the second step is to refine the search applying a similarity/dissimilarity measure. Indexing strategies and lower bounding distances are two filter methods usually used.

2.3.1. lower bounding distance

A Lower Bounding Distance ($Dist_{LB}$) is an adaptation of a true distance ($Dist_{true}$) with the characteristic of faster computation and lower value compared to true distance ($Dist_{LB}(\boldsymbol{P}, \boldsymbol{Q}) \leq Dist_{true}(\boldsymbol{P}, \boldsymbol{Q})$) without getting any false dismissal (discard Time Series that are actually similar). Some distances that were adapted to a lower bounding function are: Dynamic Time Warping [57], Edit distance [61], Edit Distance on Real sequences (EDR) [23], Edit distance with Real Penalties (ERP) [22], Time Warp Edit Distance (TWED) [78]. Furthermore, any metric distance can be adapted to lower bounding distance by using the triangle inequality property.

2.3.2. indexing strategies

Depending on the dimensionality of the input data, different types of indexes can be applied. In general, this can be divided into two groups, an index for 1-dimensional data and an index for high-dimensional data. Here we will make a quick review of some indexing structure.

2.3.2.1. One-dimensional data

When a Time Series is represented by 1-dimensional data like text symbols or bit sequence, an index structure such as B-tree can be applied. Signature Tree and Paged Trie Structure [4] are text indexing methods designed for symbolic representation of Time Series. The first one provides results with very low false rate, and the second one is especially designed for main memory applications. Symbolic Aggregate approXimation (SAX) [70, 75, 103] is a representation method that transforms raw Time Series into text strings. Several indexes where applied over SAX strings such as hashing [69], prefix tree [56], suffix tree [70] and iSAX multi-resolution indexing [103].

2.3.2.2. High-dimensional data

For high-dimensional data, Spatial Access Methods (SAMs) such as R-Tree, R*-Tree, TS-Tree [6] and M-Tree [26] can be applied by mapping the Time Series representation to a Minimum Bounding Rectangle (MBRs). However, in high dimensional space MBRs of sub-trees overlap to a high degree, producing more access during the query process. This is called the *curse of dimensionality* [114] and affects almost every SAM index.

To address this problem, several approaches have been proposed. M-Tree was adapted for a low-dimensional representation using Wavelets Transform. L-index [85] was a special index designed for a representation method based on Piecewise Approximation approach. For LCSS similarity measure, a 2-dimensional index structure was proposed [115] based on the construction of MBRs and Minimum Bounding Envelope (MBE) as an additional pruning. Time Warping in Indexed Sequential sTructure (TWIST) [89] is an index structure proposed for Time Series that works with sequential structures and index structures using MBEs. A structure that joins properties from R*-Tree and B-Tree is TS-Tree (Time Series Tree) [6]. B-tree properties help TS-Tree to avoid overlapping in sub-trees, producing better performance for higher dimensional data.

2.3.2.3. Irregular sampled and multivariate data

For irregular Time Series, almost every SAM structure can be applied using the correct similarity measure and representation method. For example, TWED, DTW or Fréchet Distance can be applied over unevenly sampled Time Series without major variations in the algorithm [74, 111].

For Multivariate Time Series, if the representation method produces a global feature

Table 2.2: Distribution of studied work on the four different groups of Time Series data defined by sampling (regular or irregular) and number of variables (univariate or multivariate).

	Regular	**Irregular**
Univariate	[2, 9, 12, 13, 18, 20, 27, 46, 55, 59, 66, 69, 71, 73, 75, 79, 80, 85, 90, 97, 102, 103, 119, 127]	[21, 74, 88, 111]
Multivariate	[15, 36, 52, 64, 98, 120, 123]	[16, 42, 43, 87, 106]

vector or if the Time Series is projected to univariate case, the indexation can be directly made with SAM structure [4, 83] such as R*-Tree variant for Time Series [14, 57]. If the Multivariate Time Series is evenly sampled and represented as a matrix, a metric access method can be used for indexing such as M-Tree [25]. However, indexing irregular MTS has not direct indexing structure yet, which means that the most viable possibility is to design a creative representation method or similarity measure.

2.4. Time Series Representations

We group existing methods into four major categories depending on their target Time Series data. Methods that work with Regular Sampling Time Series on one variable (RU data) are reviewed in Section 2.4.1, for the ones working with Regular Sampling Time Series and two or more variables (RM data) are reviewed in Section 2.4.2. Section 2.4.3 and 2.4.4 reviews the studies on methods that work with Irregular Sampling Time Series of one variable (IU data) and two or more variables (IM data) respectively. Table 2.2 shows how the studied works distribute over the different Time Series data types.

2.4.1. Regular-Univariate data approaches

Methods that work with regular sampled Time Series of one variable (RU data) are studied here. The Time Series to compare are usually sampled at the same sampling rate but with variable length (number of measures). Since most of the literature work can be found in this category, we group the studied works in three sub-groups: Feature extraction, Piecewise approximation and Symbolic representation.

2.4.1.1. Feature extraction

Fourier Transform was one of the first methods applied to represent Time Series in a different feature space. Agrawal et. al. [2] proposed a representation method using Discrete Fourier Transform (DFT) and an indexing structure for Fourier features called *F-index*. Inspired by the same approach, Chan [20] proposed a Wavelet representation using Discrete Wavelet Transform (DWT). In particular, he uses the Haar Wavelet, which is a sequence of square-shaped functions used to build a Wavelet Familiy (basis). As an indexing structure, Haar Wavelet has a better pruning capability compared to DFT. Using a R-tree structure on the first few coefficients of Haar wavelet is enough to outperform the F-index structure in terms of pruning power, number of pages access and complexity.

2.4.1.2. Piecewise approximation

Shatkay et. al. [102] were one of the first to suggest piecewise segmentation for representing Time Series. They propose a method that breaks the Time Series into regions that are approximated by a function (e.g. line). The key idea is to compress the information of each region by a well-defined, continuous, differentiable function producing a representation method that generates a lower dimensional Time Series. The break algorithm to generate the regions is based on Bézier curve fitting, with a recursive implementation that starts by fitting a curve to the whole sequence, finding the point in the sequence with a maximum deviation from the curve and repeat for the two new generated sequences. Their experiments are based on query search by number of peaks using electrocardiograms (ECG) data. For an ECG they look for all 24-hour sequences that presented exactly two peaks.

Continuing with the concept of piecewise segmentation, several different variations can be found in literature. Keogh et. al. [58, 59] proposed Piecewise Linear Approximation (PLA). This algorithm initializes with the smallest segments and starts merging neighboring segments that lead to the least increase in square error, until it reaches K segments (defined by the user). Piecewise Aggregate Approximation (PAA) [55] is a work where the Time Series is divided into equal length segment and each segment is represented by the mean value, this approach was extended in Piecewise Linear Aggregate Approximation (PLAA) [46] by combining mean-value and slope-value to produce a more meaningful symbolic representation. Other variations of PLA/PAA include monotonically increasing or decreasing segments [90], adapting break points according to slope values [127] or local-trend values [27], using splines to fit on each segment [12, 85], representing each segment by different statistical quantities [18], or using probabilistic models for each segment [59].

Sanchez and Bustos [97] proposed Multi-resolution Trend-Value Approximation (MTVA), a method based on Trend-Value Approximation (TVA), which was originally designed for regular-multivariate data. In MTVA, a bottom-up construction algorithm is proposed where the representation starts at the lower level of resolution (1 large segment) and iteratively increase the level of resolution until a threshold limit resolution level is reached. All levels of resolution are concatenated into one single feature vector. Each segment on this representation method is represented by the mean value and the slope of the best linear fit. Since the representation takes into consideration two variables (mean and trend), the resulting representation is a multivariate sequence, which needs a special similarity measure. In this work, they define the special cost function:

$$cost(p_i, q_i) = |v_i^p - v_j^q|^2 + |s_i^p - s_j^q|^2 \tag{2.9}$$

Where both value-domain and slope-domain need to be normalized and the distance measure is defined as:

$$MDist(P, Q) = \sum_{l=1}^{L} \sum_{i=2^{(l-1)}}^{2^l-1} cost(p_i, q_i) \tag{2.10}$$

Furthermore, they include a symbolic representation based on SAX and an indexing

Table 2.3: Overview of representation methods for Regular-Univariate Time Series. Sub-groups are: Feature Extraction (FE), Piecewise Segmentation (PS), Symbolic Regresentation (SR) and Bag-of-Words (BOW).

Article	**Sub-group**	**Build Time**	**Distance measure**	**Characteristics**
Agrawal et. al. [2]	FE	$O(nlog(n))$	L2-Norm	Discrete Fourier Transform.
Chan et. al. [20]	FE	$O(n)$	L_2-norm	Discrete Haar Transform.
Shatkay et. al. [102]	PS	$O(n)$	L_2-norm	Curve fitting (linear, Bezier).
Keogh and Pazzini [58]	PS	$O(n)$	Custom	Linear interpolation on segments and weighting scheme.
Keogh and Smyth. [59]	PS	$O(n)$	Probabilistic similarity	Linear interpolation on segments using a prototype template for probabilistic similarity.
Keogh et. al. [55]	PS	$O(n)$	L2-Norm	Compute mean-value to represent each segment.
Hung et. al. [46]	PS	$O(n)$	L2-Norm	Mean-value and slope on each segment.
Park et. al. [90]	PS	$O(n)$	DTW	Monotonically increasing or decreasing segments.
Zhou et. al. [127]	PS	$O(n)$	L_2-norm	Segments based on slope change.
Dan et. al. [27]	PS	$O(n)$	DTW	Segments based on local-trend and global-trend.
Morinaka et. al. [85]	PS	$O(n)$	L_1-norm	Approximate segments with lines.
Bar et. al. [12]	PS	$O(n)$	L_2-norm	Spline Curves for each segment.
Sanchez et. al. [97]	PS	$O(n)$	DTW	Mean-value and slope for each segment.
Cai et. al. [18]	PS	$O(n)$	DTW	Computes several different statistical values for each segment.
Lin, et. al. [69]	SR	$O(n)$	MINDIST	Defines special distance and introduces SAX.
Shieh and Keogh [103]	SR	$O(n)$	MINDIST	SAX but using bits instead of symbols allows multi-levels of discretization and fast indexation.
Malinowski et. al. [75]	SR	$O(n)$	MINDIST	Combines mean-value and slope-value with SAX.
Lkhagva et. al. [73]	SR	$O(n)$	L2-Norm	Extends SAX to work with 3 values per segment.
Lin, Khade and Li [71]	BoW	$O(n)$	L_2-norm	Uses SAX on sub-sequences to generate a document of words.
Li and Lin [66]	BoW	$O(n)$	CS	Feature selection using ANOVA.
Megalooikonomou et. al. [79]	BoW	$O(n^2)$[1]	Hist. Model	Clustering algorithm to find its own vocabulary (codebook).

Continued on next page

[1] Time build is dominated by clustering algorithm. Taking k-means as reference.

Table 2.3 (*continued*)

Article	**Sub-group**	**Build Time**	**Distance measure**	**Characteristics**
Baydogan et. al. [13]	BoW	$O(nlog(n))$	DTW	Combines local features with global features.
Wang, Liu, et. al. [119]	BoW	$O(n^2)$[1]	Several	Generates discretization on Wavelet coefficients.
Bailly et. al. [9]	BoW	$O(n^2)$	Several	Extract features using SIFT-framework.

Table 2.4: Overview of representation methods for Regular-Multivariate Time Series, identifying the original Article that proposed the method, the time required to build the representation, the distance measure used (if any), and some relevant characteristics.

Article	**Build Time**	**Distance measure**	**Characteristics**
Wang et. al. [120]	$O(n^3)$	Custom	Address correlation using PCA and design special multivariate similarity measure based on BORDA voting.
Yang and Shahabi [123]	$O(n^3)$	PCA-based	Propose a PCA-based similarity measure for multivariate data.
Kane and Shiri [52]	$O(n^3)$	Pearson correlation	Applies PCA to each Time Series and perform similarity query using a correlation function.
H. Li [64]	$O(n)$	DTW	Combines multivariate sequence using element-wise mean-value.
H. Sanchez [98]	$O(n)$	Correlated L_2-norm	Applies a representation method on each variable and then uses a correlation similarity measure.
Bianchi et. al. [15]	Heuristic	L_2-norm	Applies Auto-encoders to learn features.
Esmael et. al. [36]	$O(n)$	None[2]	Combines mean-value and trend-value to generate a matrix representation.

[2] Uses a classification algorithm that doesn't need any similarity measure.

structure for fast similarity search. Their experiments are performed on anomaly detection tasks using the approach of discord discovery.

2.4.1.3. Symbolic representation and Bag-of-Words

Lin et. al. [69, 70] introduced a discretization technique for Time Series based on PAA and symbolic representation. This representation method, called Symbolic Aggregate approXimation (SAX) takes a normalized Time Series, computes its PAA representation, and each segment value is discretized and represented by a symbol. The breakpoints for the discretization are computed by dividing the normal distribution curve in k segment of equal area. SAX has been used in many application domains and extended by several different works. Indexable SAX (iSAX) [103] , 1d-SAX [75] and Extended SAX [73] are examples of improvements made to SAX method.

Although a Time Series can be transformed to a word by SAX representation, this is not the only possibility for symbolic approaches. Lin et. al. [71] proposed a representation method using concepts of Information Retireval Theory. In their work, a Time Series is transformed to a document of words by extracting all sub-sequences of the Time Series and represent each one of them as a word using SAX representation. Then, the sequence of words can be represented as a Bag-of-Word (BoW) vector, called Bag-of-Patterns, where the words in the document are counted and represented as a histogram. Furthermore, as we can see in Li and Xin [66], this representation method will generate a sparse high-dimensional vector that can easily be reduced to only the relevant features for classification tasks. Using Analysis of Variance (ANOVA) [108], the vectors are reduced to the top k-features that best represent the dataset.

In order to get a BoW representation, we don't necessarily need to generate a word transformation. Megalooikonomou et. al. [79] proposed a representation method based on key sequences, clustering algorithm and Histogram representation. Here, the Time Series are segmented into equal-length sub-segments, and each sub-segment is represented by its closest sub-sequence from a codebook. The k sub-sequences in the codebook are obtained by a clustering algorithm, and the distance between sub-sequences of the codebook are precomputed. Later, the same author extended this algorithm to the Multiresolution Vector Quantized (MVQ) approximation [80].

Baydogan et. al. [13] proposed a combination of BoW representation and global features to represent each Time Series for classification tasks. Wang et. al. [119] applies Discrete Wavelet Transform (DWT) to obtain local features that are used to generate the codewords and the BoW vector. Bailly et. al. [9] used the SIFT framework (Scale-Invariant Feature Transform) to extract key-points from each Time Series and use them for the BoW vector.

Table 2.3 shows an overview of all the works studied here, summarizing the major characteristics of each method. We see that most of the representation methods are very fast, with a linear build time.

2.4.2. Regular-Multivariate data approaches

Using multivariate data involves correlation between variables that may contain relevant information for the representation method. There are different ways to handle correlation between variables, some more efficient than others, but they all depend on the target application and the field of study. Because of this, a method that works well on multivariate data on a specific field may not work well on a different field since the correlation behaves different. The works studied here are summarized in Table 2.4.

Using Principal Component Analysis (PCA) is the straightforward approach to transform a correlated set of data to an uncorrelated and lower dimensional space. Wang et. al. [120] applies this approach. On each uncorrelated variable, a separated similarity search algorithm is used to match similar sequences. Then, top k matching sequences are aligned and most similar multivariate candidates are chosen based on weighted BORDA voting algorithm [65]. This method was tested on different application fields of regular-multivariate series such as EEG, Japanese vowel and robot execution failure.

Yang and Shahabi [123] proposed a PCA-based similarity measure called *eros* on which PCA is computed for multivariate sequences and a similarity measure is generated from eigenvectors and eigenvalues. Kane and Shiri [52] proposed an algorithm to transform regular-multivariate sequences to regular-univariate sequences by linear combination where the weights are computed using PCA. This approach uses PCA as unsupervised learning and extracts the weighted scores from the implicitly computed Singular Value Decomposition (SVD) matrices.

Since PCA has high computational complexity, some alternatives are presented in the literature for applications that require faster algorithms. Trend-Value Approximation (TVA) [36] is an alternative method based on piecewise segmentation but applied to multivariate data. The algorithm generates a representation for each variable and represents the multivariate sequences as a matrix of features, where the features are two values per each segment, the mean value and the slope value. They train a memory-based classifier, which takes as input the computed features for each multivariate segment. Li [64] proposed a PAA-based method where a new univariate sequence is generated by computing the element-wise mean-value of the multivariate sequence.

Sanchez [98] proposed a different approach to similarity search problem on multivariate Time Series, especially designed for anomaly detection using discord discovery technique. He designed a univariate representation method called Multi-resolution Trend-Value Approximation (MTVA), which is applied to each variable of the multivariate sequence. Then, two multivariate Time Series P and Q are compared through their univariate MTVA representations using a special adaptation of euclidean distance for multivariate sequence defined as:

$$ED(P, Q) = \sqrt{\sum_{i=1}^{d} \sum_{j=1}^{n} (p_{i,j} - q_{i,j})^2} \tag{2.11}$$

Where n is the number of features on each variable, d is the number of variables compared and $p_{i,j}$, $q_{i,j}$ are the MTVA value corresponding to the j-th feature of the i-th variable of P and Q, respectively.

Bianchi et. al [15] proposed an autoencoders-based approach for RU data where the multivariate Time Series is flattened into a uni-dimensional vector and fed to an autoencoder, using the latent space vector as the representation features. The authors also consider an algorithm to work with RM data with missing values, which is similar to IM data. Their experiments are performed on blood measurements and different approaches to replace missing data are compared.

2.4.3. Irregular-Univariate data approaches

In the case of univariate Time Series with irregular time intervals, a representation method has to solve some extra challenges related to the data format. In particular, for most of the data mining algorithm we cannot use directly the raw irregular Time Series since the algorithm expects a regular sequence of features. Usually, representation methods designed for irregular data transform the raw Time Series to a regular feature space that can be used easily on different data mining tasks. Here we study methods based on clustering and Machine Learning that produce a representation vector of Time Series. Table 2.5 shows an overview of all studied works.

Mackenzie et. al. [74] proposed a cluster-based approach for representing irregular Time Series. They apply a clustering algorithm called *affinity propagation* [40], which aims to find representative exemplars from its input data by transmitting real-value messages between data points. The input to the algorithm is a distance matrix, which in this case is computed using Time Warp Edit Distance (TWED). After training and generating k clusters with affinity propagation, each irregular Time Series sequence is transformed into a k-dimensional vector, where the value of the i-th element will represent the similarity of the sequence with the i-th cluster centroid. The method is tested on astronomical Time Series using Support Vector Machine (SVM) classifier [17].

Valenzuela and Pichara [111] proposed an extension of Mackenzie et. al. [74] by developing a tree-based representation method. In this approach, the clustering algorithm is built on a tree-structure, producing a hierarchical tree-structure of clusters and sub-clusters. On this tree-structure, each node represents a cluster, and they will be divided into sub-clusters until a threshold condition is met. To build and populate the tree-structure, random sub-sequences are extracted from the irregular Time Series dataset and the branching factor of the tree, which is the number of clusters to build on each level of the tree, must be defined by the user. To build the representation vector of each Time Series, a sliding window technique is applied to extract sub-sequences of the Time Series and pass them to the tree-structure, counting the number of times each node is visited. A node is visited when a sub-sequence is closer to the centroid of the respective cluster. Finally, the representation vector of each Time Series will be a n-dimensional vector, with n the number of nodes, where the i-th value is the number of times the node i-th was visited by the sub-sequences of the respective Time Series.

From the machine-learning approach, there are a few algorithms based on recurrent networks that aim to learn features from a dataset of irregular Time Series. Naul et. al. [88]

Table 2.5: Overview of representation methods for Irregular-Univariate Time Series, identifying the original Article that proposed the method, the time required to build the representation, the distance measure used (if any), and some relevant characteristics.

Article	**Build Time**	**Distance measure**	**Characteristics**
Mackenzie, Pichara And Protopapas [74]	$O(n^3)$	TWED	Generates a codebook using a k-medoids clustering algorithm with TWED for irregular time serie.
Valenzuela And Pichara [111]	$O(n^3)$	TWED	Generate a more detailed codebook with hierarchical clustering on a tree-structure.
Naul, Bloom, et. al. [88]	Heuristic	None [3]	Applies Recurrent Neural Network to learn features and directly classify Time Series.
Charnock and Moss [21]	Heuristic	None [3]	Applies Deep Recurrent Neural Networks to learn features.

[3] Method uses Machine learning classification algorithms without the need for similarity measures.

proposed a Recurrent Neural Network (RNN) feature extraction architecture in which the network takes as input the measurement value of the raw Time Series and the difference between sampling times, and learns to generate a feature vector of fixed length. This feature extraction method was tested on different datasets of astronomical Time Series using a Random Forest classifier. Charnock and Moss [21] proposed the application of deep recurrent neural networks to learn relevant features of Time Series and use them on classification tasks. They design their method for a specific sub-group of Astronomical Time Series called supernovae, solving classification of different kinds of supernovae events.

2.4.4. Irregular-Multivariate data approaches

So far we have studied many different Time Series representation methods designed for different Time Series data across a variety of disciplines. However, there is an additional variation of Time Series characterized by multivariate measurement at irregular, non-simultaneous, time samplings. This means that two or more variables may be observed at different time instants without the need for simultaneous measurement. This variation of the Time Series is less common and mostly limited to Astronomy, where the telescopes may observe different variables of the same object at different times. Furthermore, since the irregular-multivariate case is the most general, we can expect a solution that works on this variation to work on any other variation of Time Series data.

Although we can find other fields of study with special cases of IM data (RM data with missing values), we will focus our study on works related to Astronomy field. Luckily, Astronomy usually presents challenges to the community to help the development and impulse further research on this field. One of those challenges is PLaSTiCC.

The Photometric LSST Astronomical Time-Series Classification Challenge (PLAsTiCC) was a challenge presented in 2018-2019 where the participants have to classify a simulated dataset of 14 different astronomical objects (classes) with irregular-multivariate Time Series (usually called in astronomy Multi-band Time Series). The objective of the challenge was to classify, with the highest precision as possible, a very large test set using a very small train set, which was not representative of the test set. Many different solutions were proposed to this challenge, most of them using feature extraction methods to transform the raw irregular-multivariate Time Series into a regular feature vector.

One of the top results during the challenge was proposed by Gabruseva et. al. [42] based on gradient boosting of decision trees, feature extraction and selection, and data augmentation. Their solutions include the computation of a variety of features of different kinds. Statistical quantities such as median, standard deviation or skewness, peak analysis, parametric curve fitting and magnitude (derived from the observed flux) are some of the features computed. Then, they rank the features importance and choose the top 100 features to represent each Time Series.

Boone [16] proposed the algorithm that won the challenge with the best classification performance. Its algorithm, named *Avocado*, apply a feature extraction based on Gaussian Process (GP) regression. The key to this approach is to apply GP regression on time-wavelength space (where different variables measure different wavelengths), where the raw astronomical

Table 2.6: Overview of representation methods for Irregular-Multivariate Time Series, identifying the original article where the method was proposed, the time required to build the representation, the distance measure used (if any), and some relevant characteristics.

Article	**Build Time**	**Distance measure**	**Characteristics**
Gabruseva, Zlobin and Wang [42]	Heuristic	None [4]	Computes many different features using Pearson correlation function to compare and remove redundant features.
K. Boone [16]	$O(n^3)$	None [4]	Perform multivariate GP-regression to generate a model that takes into consideration correlation.
Soraisam, Saha, et. al. [106]	$O(n)$	KDE-based	Using as features value and time-difference, probabilistic models are used for each variable and cross-variables.
Muthukrishna, Narayan, et. al. [87]	MCMC [5]	None [4]	Model the Time Series and interpolate values using linear interpolation.
Gomez, et. al. [43]	MCMC [5]	None [4]	Combines a model-based representation with contextual data.

[4] Method uses Machine learning classification algorithms without the need for similarity measures.
[5] Build time is dominated by time complexity of Markov Chain Monte Carlo (MCMC) algorithm.

Time Series lives. Using the two-dimensional GP regression, the correlation between variables is captured in the fitted model from which several different features are computed. Thanks to the effect of redshift on wavelength (true wavelength is affected by redshift producing a different observed wavelength), the GP regression model can also be used to augment the dataset by sampling Time Series at different wavelengths and redshifts. This advantage on data augmentation was key for this solution to win the challenge.

After the challenge was finished, authors kept using the PLaSTiCC set (unblinded, which was released after the challenge was closed) to evaluate their algorithms. Soraisam et. al. [106] proposed a feature extraction method based on a probabilistic approach. They take as input the differences in time and value of each Time Series on each variable measured. Then, two quantities are derived from these differences, a probabilistic density function for each variable on the Time Series and a likelihood score between a train and test set. The classification experiments are performed using naive Bayes on different datasets, including PLaSTiCC.

Muthukrishna et. al. [87] developed two different approaches to classify IM data. One is based on deep learning where the Time Series are preprocessed before start feeding the neural network. First, the Time Series are corrected and normalized using different astronomical concepts. Then, using this Time Series, a model of the peaks of each event is fitted and the Time Series are re-sampled. This first method designed was called RAPID and it is a model-based representation of astronomical Time Series. To compare the performance of RAPID, the authors adapted a traditional feature-based approach computing different statistical quantities to use as features and feed to a random forest classifier. They discovered that both approaches work similarly well with the difference that RAPID is a more flexible method allowing different time-length events.

FLEET is another model-based feature extraction method proposed by Gomez et. al. [43]. FLEET is a method that combines contextual data with feature extraction based on parametric model-fit. Their method is tested on different astronomical datasets using random forest to classify a specific type of astronomical event called Super Luminous Super Novae (SLSN).

An overview of the studied representation methods for IM data is presented in Table 2.6.

2.5. Discussion

2.5.1. Data mining application

During this study we found that by itself, a representation method of Time Series cannot solve any data mining problem. Most of the work in the literature applies data mining algorithms based on similarity/dissimilarity measures for which many different measures have been proposed. Classical distances, like Minkowski Distance and Cosine Similarity, have proved to be the faster and simplest options for comparing two Time Series Vectors. For raw Time Series or more complex representation vectors (i.e with time shifting or longitudinal scaling), Dynamic Time Warping (DTW) distance is the most suited measure, being widely used across different application domains. From the very few works presented on irregularly sampled Time Series, it seems that Time Warp Edit Distance (TWED) is the best option for such application domain as it takes into consideration the time intervals between measurements.

Additionally, we found several works on time series presenting different pruning techniques for speeding up the query process based on similarity search. We identified two families of approaches: those based on Lower-Bounding (LB) function and those based on indexing structures. For LB function, most of the similarity measures introduced in the literature have some LB function formulation, except for LCSS/ACSS.. For indexing structures, we identify two categories related to the kind of data we would like to index. For 1-dimensional data (e.g. SAX), indexing structures like B-Tree or suffix-tree are preferred. In the case of high-dimensional data (e.g., PAA), Spatial Access Methods (SAMs) are mostly used, for example, R-Tree. A special structure that combines properties from B-Trees and SAM is TS-Tree, which was especially designed to handle high-dimensional Time Series.

For representation methods, among the four groups identified in this study, the one with the most variety of works is RU data. Some methods such as PAA [55] and SAX [70] divides the Time Series into equal-length segments, and for each segment they compute one value, usually the mean-value. Few cases includes the computation of two values (e.g. the pair mean-value/trend-value used in MTVA [97]) or more (e.g. the statistical quantities computed in PSA [18]).

Most of the representation methods designed for RM data are extensions from methods designed for RU data (except for MTVA, which was designed after its RM data version). PCA seems to be the best option for RM data since it handles the correlation between variables properly (Wang et. al. [120], and Yang and Shahabi [123]). However, it has a very high computational complexity and is not well-suited for large datasets. Despite some loss of information about the correlation, some alternative and faster methods compared to PCA have been proposed. Li [64] proposed to compute the element-wise mean value between variables in order to transform the RM data into RU data. Although this approach was naive and simple, it has the same basis of AVOCADO [16] where they use Gaussian Process (GP) regression to fit a model on IM data. The MTVA algorithm was also extended for RM data [98] by designing a multivariate euclidean distance, which addresses the correlation between variables but at a very high computational cost.

For IU data, the few works studied in this chapter presented representation methods that transform the IU raw data to a RU feature vector. Valenzuela et. al. [111] and Mackenzie et. al. [74] proposed a representation method based on clustering sub-sequences of the Time Series, using Bag-of-Word to represent the number of times each cluster is used to represent each Time Series. In order to apply the clustering algorithm directly to IU data, they used Time Warp Edit Distance (TWED), which, as we see from Table 2.1, is the only distance measure (studied here) designed for IU data. On the other hand, Naul et. al. [88] and Charnock et. al. [21] proposed a feature extraction method based on the application of neural networks. In the first, the features are directly learned during the execution of the neural network that solves the classification task, and in the second, a special neural network is designed to learn features from the IU data.

In the case of IM data, which is the most complex and general data type we have studied here, we found that most of the current work is related to Astronomy, including some works on Medicine using RM data with missing values [15]. Among all the methods studied for IM data, AVOCADO [16] is the only one that addresses the correlation between variables using two-dimensional Gaussian-Process regression. Sorsaim et. al. [106] is the only one proposing a linear time representation method, with a similarity measure that takes into account the correlation between variables. Although all of the studied works here presented experiments for Classification, only this last one used similarity-based methods, where the rest used Machine Learning algorithm to learn to compare and classify the features.

2.5.2. Research directions

From the study presented, we can outline various different research directions that could be explored. First, from Table 2.2 we noted that there are very few works related to irregular sampled Time Series, being a potential direction of research, mainly for Astronomy where most of the data used is irregular sampled. Here, the principal challenges to solve are lower computational complexity, correlation between variables and large dataset sizes. Additionally, many works reviewed here have some potential to be extended for more general applications. For example, Bag-of-Pattern Feature (BOPF) [66] can be extended for IU data and IM data, the same goes for MVQ [80]. As another example, Variablity Tree [111] is a very solid work that could be tested on different fields of study, applied to RU data and extended to RM/IM data. Another possible research direction is to design a similarity measure that can be used on IM data, which would help to develop new data mining algorithms for IM data without the need of applying feature extraction.

Chapter 3

Information Retrieval theory for Time Series

This chapter introduces the use of Information Retrieval theory for Time Series. We describe shortly the base of Information Retrieval theory and how it can be used for representing Time Series. In Section 3.1 we describe the characteristics of a Time Series represented as a document of words. Next, we introduce the theory of Vector Space Models (Section 3.2) where representation models based on counting words are described. In particular, we describe the Bag-of-Words (Section 3.2.1) and Term-Frequency Inverse-Document-Frequency (Section 3.2.2) vector models, including variants and normalization. To address the high dimensionality of the resulting representation by using Vector Space Model, two techniques to generate compact vectors are described in Section 3.3, Latent Semantic Analysis, which is a transformer of dimensional space, and Analysis of variance, which is a feature selection method, with an extension for multivariate application. Finally, in Section 3.4 we present a brief summary about the described theory and how it is used for Time Series.

3.1. Time Series Transformation

Before we start using Information Retrieval theory on Time Series data, we have to transform those Time Series into documents of words. Our goal is to extract all the relevant information of a Time Series and generate a representation vector of symbols called words.

3.1.1. Document representation

A Time Series is composed of many patterns, some of them overlapped depending on the complexity of the Time Series. A naive approach to extract those patterns, or at least, a part of them, is to slide a window across the Time Series data, extracting the corresponding sub-sequences (Figure 3.1 left side). These windows can be of variable or fixed width and the offset applied to the window while slides over the data can be constant or variable. Furthermore, the sub-sequence extraction can be done using many different windows at once. Here we assume that the sub-sequences are already extracted without knowing which method was used to extract them, which is addressed in Chapter 4.

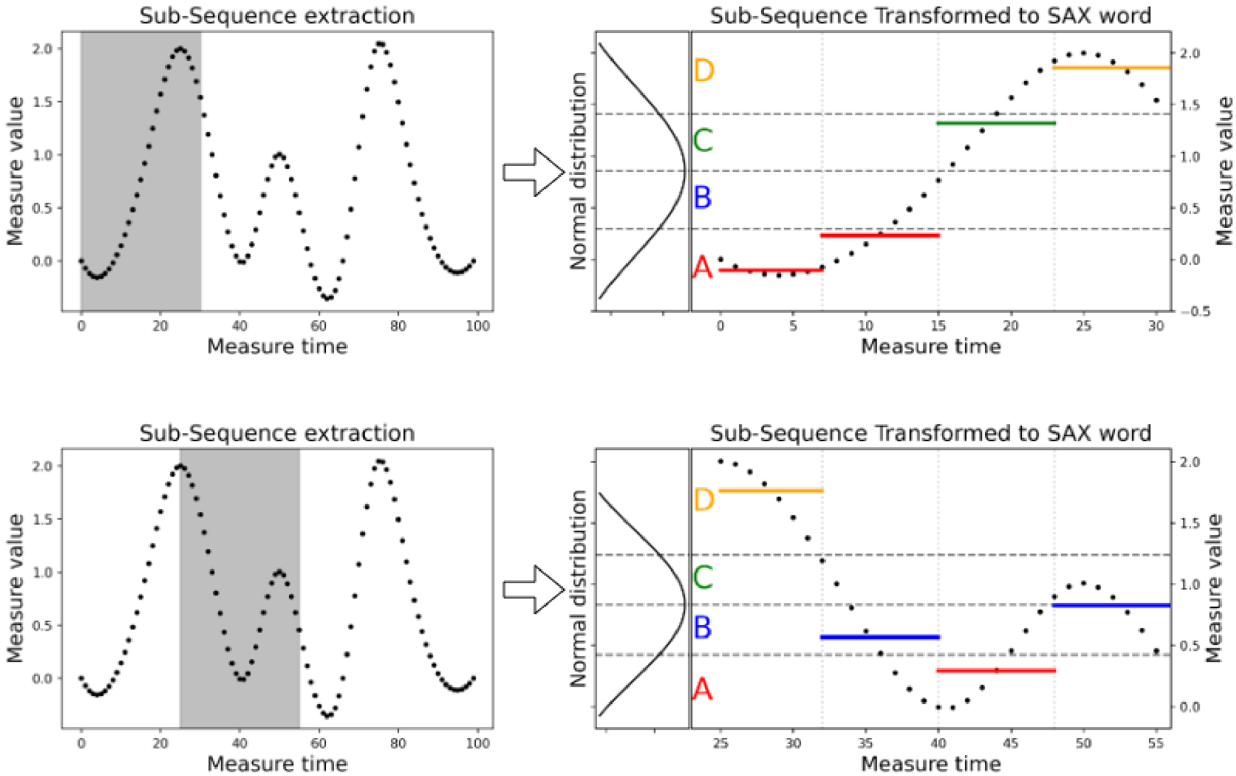


Figure 3.1: Two examples of transforming a sub-sequence of a regular UTS to a SAX word using sliding window. Extracted sub-sequences are of length 30 (measure time), the generated words have 4 characters each and the SAX alphabet is $\{A, B, C, D\}$. Resulting words are $AACD$ (top) and $DBAB$ (bottom).

For each sub-sequence (pattern) a discretization process has to be applied in order to represent that pattern as a word. The most used discretization method in literature is Symbolic Aggregate Aproximation (SAX) [70]. In SAX, the sequence is divided into w equal-width segments for which a representative quantity is computed, reducing the data from n measures in the sequence to w values. When the representative quantity is the mean value of the segment, this process is called Piecewise Aggregate Approximation (PAA) [55]. Having the representative sequence of w values, k break-points are defined, and the representative values of each segment is transformed into representative symbols for each break-point interval to build a word. An example of this process can be seen in Figure 3.1.

When all possible sub-sequences of a Time Series are transformed to symbolic representation, they are grouped together in a *document*. If q sub-sequences are extracted from a regular UTS of length n, where $q < n$, then the corresponding document has q words. In general, the SAX method is defined to generate words of fixed length w based on a fixed alphabet Σ of size $|\Sigma| = \alpha$, this means that we always know the possible words that can be produced by the SAX transformation. The collection of possible words or patterns is called *vocabulary*, whose size $|vocabulary|$ is defined as:

$$|\text{vocabulary}| = \alpha^w \tag{3.1}$$

When all the Time Series of a dataset are transformed to documents, they are grouped together in a *corpus*. In summary, we transform a dataset of m Time Series, each with n_i measures, $i \in \{1, 2, ..., m\}$, to a corpus of m documents, each with q_i words, $i \in \{1, 2, ..., m\}$, of length w living in an alphabet of size $|\Sigma| = \alpha$ with a total vocabulary size of α^w.

3.1.2. Numerosity reduction

When extracting all sub-sequences from a Time Series, two or more consecutive sub-sequences can be very similar depending on the sampling rate of the Time Series. Furthermore, once discretized and transformed to a SAX word, they could be mapped to the exact same word. This will derive in repetitive data, which does not give any relevant information about the Time Series.

Numerosity reduction is a technique used in previous works [66, 71] to reduce this redundant information produced by consecutive sub-sequences. The main idea is to count only the first occurrence of a word and ignore the rest until the occurrence of a different word. For example, for the document:

$$\{ABB, ACA, ACA, ACA, BBB, BCA, BAC, BAC, BAC, BAC, BBB, ...\} \tag{3.2}$$

Numerosity Reduction results in the following:

$$\{(ABB)_1, (ACA)_2, (BBB)_5, (BCA)_6, (BAC)_7, (BBB)_{11}...\} \tag{3.3}$$

Where the subscript denotes the first occurrence position in the original document.

This technique modifies the document by reducing the impact of those common patterns, especially useful when Time Series have smooth variations. Figure 3.2 exemplifies a case of application. For a simple event of a peak with linear behavior, three different Time Series are generated (Figure 3.2 left column), each one measuring the peak of the event at different time instants. The figure shows the effect of Numerosity Reduction (NR) by generating two histograms of counting words for each Time Series, one using the full document of words and the other using the document of words after NR. It is clear that after using NR the histogram are exactly the same for all Time Series, which correlates with the fact that they all measured the same event. Although this is a potentially good approach to reduce bias on documents of words, its performance may vary depending on the application domain.

3.2. Vector space model

We have achieved a representation of Time Series as a collection of patterns in the form of a document of words. However, as it is now, we cannot apply this representation to any data mining technique, so we need to apply an extra transformation, in this case, to a vector space.

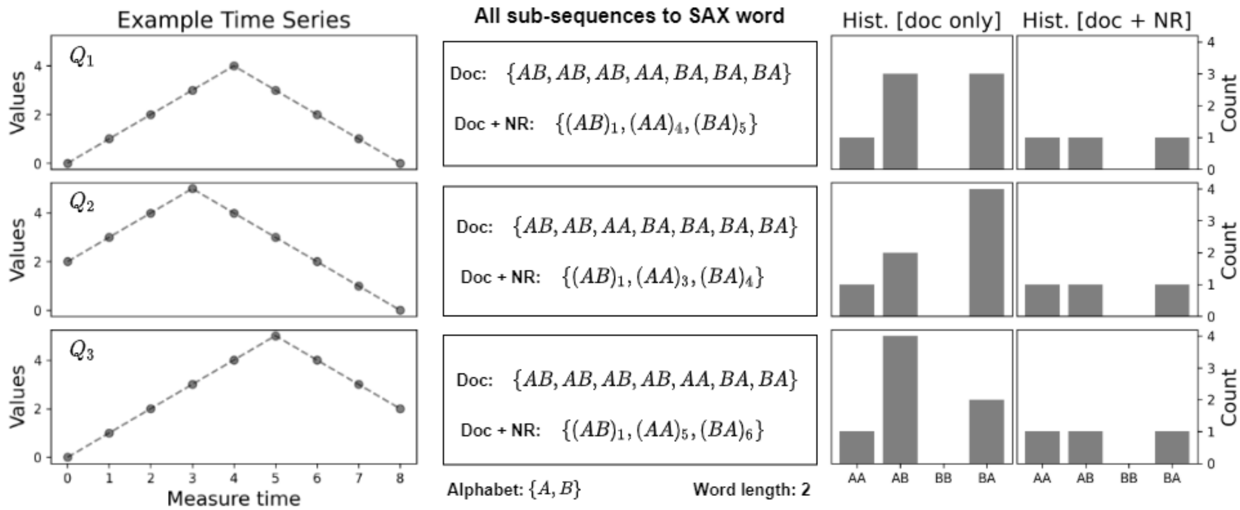


Figure 3.2: Effects example of Numerosity Reduction (NR). For the same class event (simple peak), three Time Series are generated (Q_1, Q_2, Q_3), measuring the peak of the event at different time instants (left column). Their documents are constructed and the NR is applied (middle column). Finally, the documents are shown in the form of Histogram considering the vocabulary $\{AA, AB, BB, BA\}$.

The representation of a set of documents as vectors in a common vector space is known as the *vector space model* [77]. Applying this model is key for most of the data mining algorithms. However, before we ensure common vector space for a corpus, we have to transform that corpus of words into a set of document vectors. For that, we describe two well known document vectors: Bag-of-Words and TF-IDF, including TF-IDF variants for special applications.

3.2.1. Bag-of-Words model

An intuitive approach to represent a document of words as a vector is based on histograms (as in Figure 3.2). Since our document has a finite number of possible words, we can just count the number of times each word occurs in the document and represent that information as an Histogram. This intuition is applied in Bag-of-Words (BoW) with the addition of including those words that are known to be present in the vocabulary even if they are not present in the document. The BoW vector is written as:

$$BoW(d) = \{tf(w_1, d), tf(w_2, d), ..., tf(w_n, d)\} \tag{3.4}$$

Where $tf(\omega_i, d)$ is the *Term-Frequency (TF)* of the word ω_i in the document d and is defined as follows:

Definition 3.1 *(**Term Frequency (TF)**) For a document* $\boldsymbol{d}$ *and a vocabulary* $\boldsymbol{V}$*, the term frequency of a word* $\boldsymbol{\omega} \in \boldsymbol{V}$ *is the number of times that word occurs in the document:*

$$tf(\omega, d) = |\{\omega' \in d : \omega' = \omega\}| \tag{3.5}$$

However, there is a main question that this approach cannot answer: are all words in a document equally important? [77]. In short, the answer is no. For a Time Series application of Information Retrieval all the Time Series are going to have several different patterns, but some of them may be very rare and others may be very common depending on the taxonomy nature of the used dataset. The common patterns are usually not relevant for classification or other data mining tasks since they do not help differentiate between vectors. Furthermore, they can produce a bias in the feature vector that affects negatively the representation method (something that Numerosity Reduction tries to solve, but not totally).

3.2.2. TF-IDF model

Extending the idea of Term Frequency (TF) used in the Bag-of-Words (BoW) model, we add a penalization factor depending on the frequency of a word on the whole document, this penalization term is called *Inverse Document Frequency* (IDF) and penalizes the common words, having high IDF values for rare words and low IDF values for common words. A formal definition is presented below:

Definition 3.2 ***(Inverse Document Frequency (IDF))****For a set of documents* $\boldsymbol{D}$ *and a vocabulary* $\boldsymbol{V}$*, the inverse document frequency of a word* $\boldsymbol{\omega} \in \boldsymbol{V}$ *is defined as:*

$$idf(\omega) = log\left(\frac{|D|}{|\{d \in D : \omega \in d\}|}\right) \tag{3.6}$$

The TF-IDF model is the product of TF and IDF values for each word on each document. Then, a corpus (dataset) is transformed into a *TF-IDF matrix*, where each row contains the TF-IDF value of each document. The dimensions of the matrix depends on the corpus size and the vocabulary size, which can be arbitrarily big depending on the alphabet size and the word length used during the SAX transformation of each sub-sequence. For clarity, the TF-IDF matrix is defined next:

Definition 3.3 ***(TF-IDF matrix)****For a set of documents* $\boldsymbol{D}$ *and a vocabulary* $\boldsymbol{V}$*, the TF-IDF matrix is defined as:*

$$tf\text{-}idf_D = \begin{pmatrix} tf(\omega_1, d_1)\, idf(\omega_1) & tf(\omega_2, d_1)\, idf(\omega_2) & ... & tf(\omega_{|V|}, d_1)\, idf(\omega_{|V|}) \\ tf(\omega_1, d_2)\, idf(\omega_1) & tf(\omega_2, d_2)\, idf(\omega_2) & ... & tf(\omega_{|V|}, d_2)\, idf(\omega_{|V|}) \\ ... & ... & ... & ... \\ tf(\omega_1, d_{|D|})\, idf(\omega_1) & tf(\omega_2, d_{|D|})\, idf(\omega_2) & ... & tf(\omega_{|V|}, d_{|D|})\, idf(\omega_{|V|}) \end{pmatrix} \tag{3.7}$$

TF-IDF vectors are likely to be more robust than Bag-of-Words in the sense that they penalize common patterns focusing more on rare patterns as features. TF-IDF are the prefered vector model in many Information Retrieval applications, however, it has some weak points that are addressed by variants of TF-IDF like sublinear TF or class based TF-IDF.

3.2.3. Sub-linear TF weights

The simple TF of counting the number of occurrences of a word has a problem with complex structure documents. The linear nature of TF says that if a word occurs twenty times in a document, it means it carries twenty times the significance of a single occurrence [77]. With this, the significance of a word is linearly related to the number of occurrences of that word in the document, which results in common words TF dominating rare words TF. To reduce the impact of words with more occurrences, sub-linear TF is used.

With a sub-linear TF, the significance of a frequent word is not much larger than the significance of a rare word, but it is still more significant. A common sub-linear TF is to use the logarithm of the term frequency, setting to 0 when the logarithm cannot be computed:

$$\mathrm{tf}_{log}(\omega, d) = \begin{cases} 1 + log(\mathrm{tf}(\omega, d)) & \text{if } \mathrm{tf}(\omega, d) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

In consequence, the TF-IDF matrix elements of eq. (3.7) are redefined as:

$$\text{TF-IDF}_{i,j} = \mathrm{tf}_{log}(\omega_i, d_j)\mathrm{idf}(\omega_i) \tag{3.9}$$

3.2.4. Cosine normalization

For some cases where the documents in a corpus contains a variable number of words, the resulting TF weights may vary enormously. For example, for a document d_1 where $||d_1||$ is small and another document d_2 where $||d_2||$ is larger, the chances of match is higher for document d_2 since it has more data to match, producing a bias that must be handled properly.

With a normalization of any kind, the relevance of larger documents does not exceed the relevance of short documents by much, resulting in a more balanced comparison and not biased by the length of the document. A useful normalization in this sense is cosine normalization, where each element in the vector is divided by its module:

$$\text{nTF-IDF}_{i,j} = \frac{\text{TF-IDF}_{i,j}}{||d||} \tag{3.10}$$

where $||d||$ is the total number of words in the document d. To avoid the normalization problem, cosine similarity is usually applied as the distance measurement, which by definition normalizes the data.

3.2.5. Class based TF-IDF

In some cases the produced documents are too short for any real data mining tasks, resulting in low-populated document vectors with non-representative values. To solve this problem, the documents can be merged by class label, resulting in more large and consistent documents. Then, the same procedure is followed to compute TF, TF_{log}, IDF, TF-IDF and/or nTF-IDF, with the only difference that now the documents represent classes and the number of documents is smaller.

3.3. Compact vectors

Since the document vector may be really large and sparse depending on the size of the alphabet and the length of the word constructed, the dimension of the vector needs to be reduced to a more manageable size. We define this as compact representation:

Definition 3.4 *(**Compact Representation**) For a raw Time Series T with spatial complexity $SC(T)$, a representation R with spatial complexity $SC(R)$, is compact if and only if:*

$$SC(T) < SC(R) \tag{3.11}$$

Where R is a feature vector that represents the original Time Series T through a transformation algorithm.

The more compact a representation method is, the better. However, we have to consider that a reduction in spatial complexity will likely come with a reduction of performance in classification or similarity search tasks as well. To minimize the impact on the representation method, we have to design a compact method that is able to reduce the dimension while preserving as most information as possible.

Almost every variant of Vector Space Model (VSM) can easily be transformed to a compact representation, but the usage theory depends on the variant of TF-IDF used and the target application. For example, class-based TF-IDF may not need a compact representation since they have been already merged into class vectors. On the other hand, if the original dataset has Time Series with variable spatial complexity we need to define a criteria for the compact representation. In this case we use the mean spatial complexity of the dataset.

We will describe two known approaches to generate compact representation: feature selection through analysis of variance and dimensionality reduction through singular value decomposition.

3.3.1. Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA) was first proposed by Deerwest et. al. [28] as an Information Retrieval technique for improving indexing and search query perfomance. The latter was addapted for several other fields including representing the meaning of words by humans or category judgments [62]. The base idea of LSA from the perspective of human words is that the meaning of a paragraph or document is related to patterns of presence or absence of individual words, whereas a collection of documents (corpus) is modeled as a system of

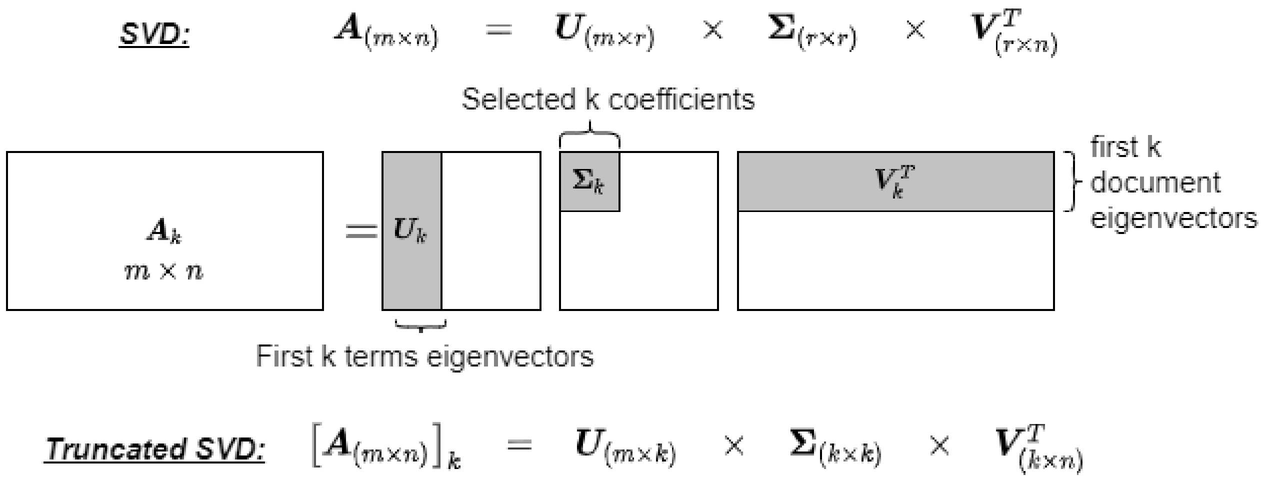


Figure 3.3: Diagram of dimensionality reduction in LSA based on Truncated SVD.

simultaneous equations that can determine the similarity of meaning of words and document to each other [37].

LSA is usually applied to text quantification vectors or Vector Space Models (VSM) such as Bag-of-Words or TF-IDF. The mathematics behind LSA is based on Singular Value Decomposition (SVD) and it is very similar to Principal Component Analysis (PCA). Applying LSa over VSM means we have as input a matrix $\boldsymbol{A}$ of m terms and n documents. Subsequently, $\boldsymbol{A}$ is subjected to SVD:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{3.12}$$

Where $\boldsymbol{U}$ are the term eigenvectors, $\boldsymbol{V}^T$ are the document eigenvectors transposed, and $\boldsymbol{\Sigma}$ is the diagonal matrix of singular values, which can be identified as the square roots of common eigenvalues between terms and documents, being $\boldsymbol{U\Sigma}$ the term loadings on the common principal components of terms and documents, and $\boldsymbol{V\Sigma}$ the respective document loadings.

Two important characteristics have to be noticed here. First, the generated matrix $\boldsymbol{A}$ is highly sparse since most of the documents have a very small subset of words from the whole possible vocabulary. Second, the vocabulary size of possible words can be extremely large, producing a matrix $\boldsymbol{A}$ with a large number of features. Both factors influence in that the computed matrix during SVD is very expensive in time and space. In particular, the matrices $\boldsymbol{U}$ and $\boldsymbol{V}^T$ are an orthonormal dense matrix with a much larger storage consumption compared to $\boldsymbol{A}$.

Over the years several different techniques have been proposed to reduce the dimensionality of the computed matrices [37], being one of the most used Truncated SVD. This methods works by reducing the length of the eigenvectors used for computing the diagonal matrix.

The idea is to define $k < min(m, n)$ as the number of desired components on the SVD. Then, $\boldsymbol{U}_k$ is the matrix with the first k columns of $\boldsymbol{U}$, $\boldsymbol{V}_k^T$ is the matrix with the first k rows of $\boldsymbol{V}^T$, and $\boldsymbol{\Sigma}_k$ is the diagonal matrix with the k largest singular values (See Figure 3.3), defining the Truncated SVD as:

$$\boldsymbol{A}_k = \boldsymbol{U}_k \boldsymbol{\Sigma}_k \boldsymbol{V}_k^T \tag{3.13}$$

LSA with dimensionality reduction based on Truncated SVD generates a lower dimensional representation matrix $\hat{\boldsymbol{A}}$ that can be used on a variety of application domains such as query search, clustering or classification. For complexity analysis, in the case of large number of documents compared to number of features $n > m$, the time complexity will be dominated by the decomposition of the matrix, which is $O(n^3)$ [67], otherwise, the time complexity will be $O(m^2 n)$

3.3.2. One-way Analysis of Variance (ANOVA)

Any method that allows us to rank features and drop the least relevant ones can be used as a feature selection method. For instance, we can take the IDF term and preserve only the top 80 % features with the highest IDF values. Several different methods have been proposed in literature as feature selections [30, 66, 72, 101]. Here we will describe one-way Analysis of Variance (ANOVA) since it has potential for multivariate approaches.

ANOVA is a statistical technique used to analyze variation in a response variable measured under conditions defined by discrete factors [63]. In the literature discrete factors are usually described as fixed factors with specific levels of interest, where each level represents a distinct population with a unique response mean. In this context, the response variable is going to be a vector with a set of scalar measures for a specific population on a specific discrete factor. The discrete factors are just features sampled or observed from an underlying continuous phenomena, which is assumed to be normal distributed. The population is the statistical normal distribution that can describe a set of response variables by a unique mean (i.e. response mean) and variance value. To our application on features extracted from astronomical Time Series, ANOVA is a method that works on each feature, evaluating how these features relate within and between populations in terms of variance, assuming normal distribution. With the correct test statistic and ranking, ANOVA can be used to select features with higher relevance in terms of variance.

Using our application case for the formula derivation, we have a dataset with c classes and m Time Series transformed into a corpus matrix with w features using any Vector Space Model. Here we have c different populations (treatment) and up to m response values (observations) for each feature, which are considered as random variables. ANOVA is applied to each feature independently since our goal is to perform a feature selection. However, before we can use ANOVA, some assumptions have to be made [63]:

Independence : The observations are independent within each others. For our case, this means no correlation between values of a feature.

Additivity : The response data can be represented using a statistical model with additive components. The model used for ANOVA is:

$$Y_{ij} = \boldsymbol{\mu} + \boldsymbol{\tau_i} + \boldsymbol{\epsilon_{ij}} \tag{3.14}$$

Where Y_{ij} is the j-th observations on the i-th group, $\boldsymbol{\mu}$ is the grand mean, $\boldsymbol{\tau_i}$ is the treatment effect for group i, and $\boldsymbol{\epsilon_{ij}}$ is a random error. Here we can consider $\mu_i = \boldsymbol{\mu} + \boldsymbol{\tau_i}$ with μ_i the group mean.

Normality : assume a normal distribution for the random errors.

Homogeneous Variance : Assume identical variance for all groups, represented by σ^2.

Having defined the initial assumptions, we write the formulation of group mean and grand mean. If each group (class) has n_i, $i = \{1, 2, ..., c\}$ observations Y_{ij}, $j = \{1, 2, .., n_i\}$, such that $n_1 + n_2 + ... = m$ (i.e. all Time Series are labeled), the group total sum and group mean are written as:

$$y_{i.} = \sum_{j=1}^{n_i} Y_{ij}, \qquad \overline{y}_{i.} = \frac{y_{i.}}{n_i} \quad i = 1, 2, ..., c \tag{3.15}$$

In the same way, the grand total sum and grand mean are:

$$y_{..} = \sum_{i=1}^{c} \sum_{j=1}^{n_i} Y_{ij}, \qquad \overline{y}_{..} = \frac{y_{..}}{m} \quad i = 1, 2, ..., c \tag{3.16}$$

Then, the ANOVA technique consists of a test hypothesis on the equality of group means. We define a null hypothesis where all group means are the same with an alternative hypothesis of at least one group mean with different value:

$$\begin{aligned} H_0 &: \ \overline{y}_{1.} = \overline{y}_{2.} = ... = \overline{y}_{c.} \\ H_1 &: \ \overline{y}_{i.} \neq \overline{y}_{k.} \text{ for at least one } i \neq k \end{aligned} \tag{3.17}$$

If the null hypothesis is true, there is no statistically significantly difference between groups. Otherwise, we accept the alternative hypothesis, i.e., there is a difference between at least two pairs of group means.

The ANOVA test is based on the total variability in the data, partitioned into two terms using the sum of squares identity [84]:

$$\begin{gathered} \sum_{i=1}^{c} \sum_{j=1}^{n_i} (Y_{ij} - \overline{y}_{..})^2 = \sum_{i=1}^{c} n_i (\overline{y}_{i.} - \overline{y}_{..})^2 + \sum_{i=1}^{c} \sum_{j=1}^{n_i} (Y_{ij} - \overline{y}_{i.})^2 \\ SS_{total} = SS_{treat} + SS_{error} \end{gathered} \tag{3.18}$$

The two terms on the right are identified as the treatment sum of squares (SS_{treat}), and the error sum of squares (SS_{error}), respectively. The degrees of freedom are also affected in

Equation 3.18. SS_{total} has $m-1$ degrees of freedom given by the total number of observations. There are c levels (classes), which means SS_{treat} has $c-1$ degrees of freedom. This gives as result $m-c$ degrees of freedom from SS_{error} in order to preserve the equality:

$$m - 1 = (c - 1) + (m - c) \tag{3.19}$$

To have statistically comparable values based on SS_{treat} and SS_{error} we need to normalize those quantities by their respective degrees of freedom, computing their mean values:

$$MS_{treat} = \frac{SS_{treat}}{c-1} \qquad MS_{error} = \frac{SS_{error}}{m-c} \tag{3.20}$$

Where MS_{treat} is called the mean square of treatments and MS_{error} the mean square of errors. MS_{error} is an estimator of the variance error σ^2 [84] and in the case of all treatments with the same group mean (i.e. null hypothesis satisfied) then MS_{treat} is also an estimator of σ^2. In this case, the ratio of the two variances:

$$F = \frac{MS_{treat}}{MS_{error}} \tag{3.21}$$

follows the Fisher statistical distribution (F-distribution) and the ratio is called the *F-value*, with $(c-1)$ and $(m-c)$ degrees of freedom. Larger F-value provides evidence against the null hypothesis. Then, features can be ranked by their F-value, selecting only the top-k features. A more detailed deduction for ANOVA F-value is presented in Larson [63], and Montgomery and Runger [84].

For a time complexity analysis, From Equation 3.18 we see that:

$$SS_{total} = \sum_{i=1}^{c}\sum_{j=1}^{n_i}(Y_{ij} - \frac{1}{m}\sum_{l=1}^{c}\sum_{k=1}^{n_l} Y_{lk})^2 \tag{3.22}$$

If $n_i = n/c$ for all i, and the same for n_l, then, the computational complexity is $O(c \cdot n/c \cdot c \cdot n/c) = O(n^2)$, for n documents. Considering that ANOVA is computed for each feature, the final time complexity is $O(wn^2)$ for w features.

3.3.3. One-way Multivariate ANOVA (MANOVA)

The ANOVA test statistic can be extended for multivariate observations by extending some scalar quantities to vectors. Here, the observations, the group mean and the grand mean are going to be vectors:

$$\boldsymbol{Y}_{ij} = \begin{pmatrix} Y_{ij1} \\ Y_{ij2} \\ \dots \\ Y_{ijw} \end{pmatrix} \quad \overline{\boldsymbol{y}}_{i.} = \begin{pmatrix} \overline{y}_{i.1} \\ \overline{y}_{i.2} \\ \dots \\ \overline{y}_{i.w} \end{pmatrix} \quad \overline{\boldsymbol{y}}_{..} = \begin{pmatrix} \overline{y}_{..1} \\ \overline{y}_{..2} \\ \dots \\ \overline{y}_{..w} \end{pmatrix} \tag{3.23}$$

Where

$$\overline{y}_{i.k} = \frac{1}{n_i}\sum_{j=2}^{n_i} Y_{ijk}, \quad \overline{y}_{..k} = \frac{1}{m}\sum_{i=1}^{c}\sum_{j=1}^{n_i} Y_{ijk} \tag{3.24}$$

From this, the SS_{treat} and SS_{error} are going to be matrices called $\boldsymbol{SS_T}$ and $\boldsymbol{SS_E}$ respectively. For each pair (p, q) we have:

$$\boldsymbol{SS_T}(p, q) = \sum_{i=1}^{c} n_i(\overline{y}_{i.p} - \overline{y}_{..p})(\overline{y}_{i.q} - \overline{y}_{..q}) \tag{3.25}$$

$$\boldsymbol{SS_E}(p, q) = \sum_{i=1}^{c}\sum_{j=1}^{n_i}(Y_{ijp} - \overline{y}_{i.p})(Y_{ijq} - \overline{y}_{i.q}) \tag{3.26}$$

Where the cross product was included into the original definitions of SS_{treat} and SS_{error}. An advantage of this cross product is the inclusion of correlation between multivariate observations.

For the hypothesis, they are almost the same, but now the group means are vectors and the alternative hypothesis has an extra condition:

$$\begin{aligned} H_0 :&\quad = \boldsymbol{\overline{y}_{1.}} = \boldsymbol{\overline{y}_{2.}} = \boldsymbol{\ldots} = \boldsymbol{\overline{y}_{c.}} \\ H_1 :&\quad \overline{y}_{i.l} \neq \overline{y}_{k.l} \text{ for at least one } i \neq k \text{ and at least one variable } l \end{aligned} \tag{3.27}$$

Since the hypothesis to be tested has vectors instead of scalars, alternative test statistics based on the eigenvalues of the matrices $\boldsymbol{SS_T}$ and $\boldsymbol{SS_E}$ have been proposed [35, 53, 86, 110]. In particular, we describe the Wilks's Lambda test without entering in detailed formulation (for further details see Kent, Mardia and Bibby [53]). The Wilks's Lamba function for MANOVA is the ratio between the determinant of the matrix mean square error $\boldsymbol{MS_E}$ and the determinant of the total mean square $\boldsymbol{MS_T} = \boldsymbol{MS_E} + \boldsymbol{MS_T}$:

$$\lambda = \frac{|\boldsymbol{SS_E}/(m-c)|}{|\boldsymbol{SS_E}/(m-c) + \boldsymbol{SS_T}/(c-1)|} = \frac{|\boldsymbol{MS_E}|}{|\boldsymbol{MS_T} + \boldsymbol{MS_E}|} = \frac{|\boldsymbol{MS_E}|}{|\boldsymbol{MS_T}|} \tag{3.28}$$

This quantity works inversely compared to F-distribution, in the sense that relevant features have a λ value closer to 0. Repeating the same procedure as that applied in ANOVA, λ is computed for each feature and the top-k features are selected by ranking their λ, with the only difference that now the best features are the ones with the lowest scores.

In the case of MANOVA, the time complexity includes the multivariate computation, an aggregated computational cost compared to ANOVA. For each term in the matrices $\boldsymbol{SS_T}$ and $\boldsymbol{SS_E}$ the computational complexity is the same as ANOVA ($O(n^2)$) where each matrix has b^2 terms, with b the number of variables, this results in a time complexity within $O(wb^2n^2)$, which reduces to $O(wn^2)$ when b is constant and small.

3.4. Summary

We described the usage of the Information Retrieval theory for Time Series representation. We presented some relevant concepts and definitions that are necessary to understand the basics of Information Retrieval theory and how it is applied to the Time Series dataset. We show that the key factors to use Information Retrieval theory for the Time Series dataset is how we transform each Time Series into a document of patterns/words and a feature vector using Vector Space Model (VSM).

We describe the use of the Numerosity Reduction technique and the SAX method to transform a Time Series into a document of patterns/words. Next, we introduce the concept of VSN and Bag-of-Words, which is a simple vector model used to represent a document of words in a more manageable form.

Then we go deeper into the VSM by introducing the usage of TF-IDF vectors, which are a more polished vector model compared to the simple Bag-of-Words. Furthermore, we presented two variants for TF-IDF called log-TF-IDF and class-TF-IDF. The first one uses sublinear Term Frequency to penalize those patterns, which repeats many times in a document. The other variants, called class-TF-IDF, is used for a small dataset where the generated documents are too short to be used to compute any representative vector space model. To solve this, all the documents of a common class are merged into only one single document, reducing a dataset of m Time Series and c classes into c large documents from which vector space models are computed.

Finally, we introduce the concept of compact vector and describe two techniques that can help to achieve compact vectors: Latent Semantic Analysis (LSA) for dimensionality reduction and Analysis of Variance (ANOVA) for feature selection. Furthermore, we derive the multivariate version of ANOVA, called Multivariate ANOVA (MANOVA), which addresses the correlation between variables when selecting the best features.

Chapter 4

A new representation method for astronomical Time Series

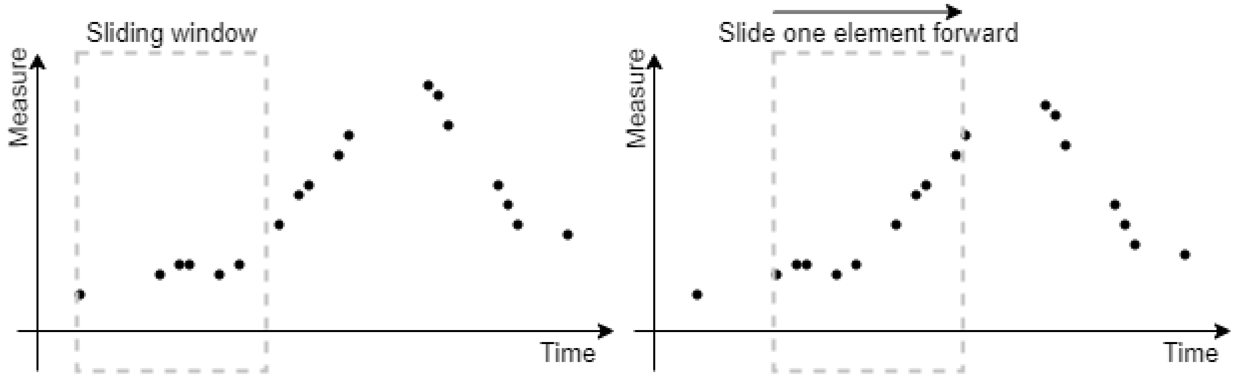
In this chapter, we describe our proposed method for representing astronomical Time Series (irregular Multivariate Time Series), based on Information Retrieval theory (see Chapter 3) and Bag-of-Pattern Feature method [66]. In Section 4.1 we define the procedure adopted to extract sub-sequences (patterns) from irregular Univariate Time Series and transform those patterns to words using a special character approach for empty segments. Next, we describe our adapted version of Irregular Bag-of-Pattern Feature method (Section 4.2), combining the classic method with the procedures defined in Section 4.1. Section 4.3 generalizes the method for irregular Multivariate Time Series, presenting the application of feature reduction techniques such as Latent Semantic Analysis (LSA) or Multivariate Analysis of Variance (MANOVA). Two additional extensions are proposed to improve the base algorithm of the Irregular Bag-of-Pattern Feature method, one focusing on the use of multiple statistical quantities (Section 4.4.1) and another focusing on the use of multiple levels of resolution (Section 4.4.2). Finally, we conclude this chapter in Section 4.5 with a theoretical analysis of performance and computational complexity of our method.

4.1. Irregular time Series

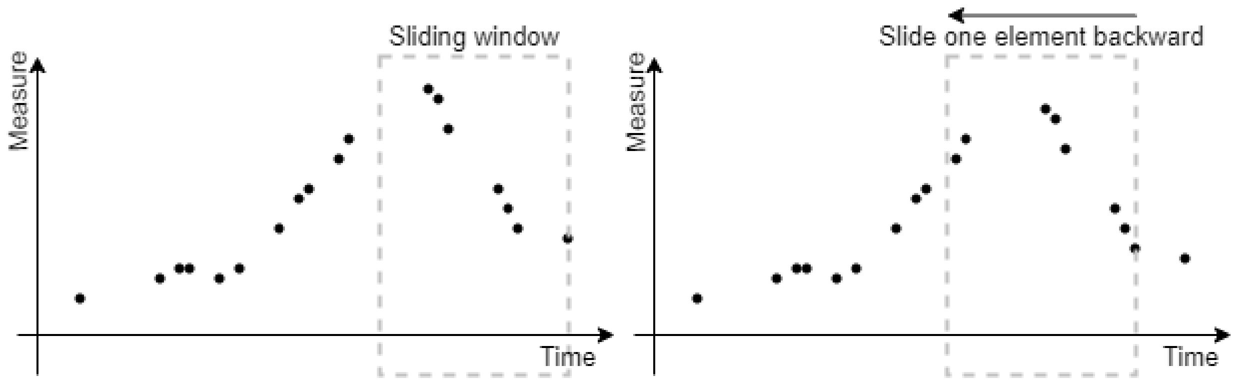
In order to apply a representation method based on Bag-of-Pattern Feature representation for irregular Time Series, we need to address the irregularity in the sampling intervals. Here two major factors are identified, the technique used to extract patterns from Time Series and the further segmentation of each pattern to generate a symbolic representation.

4.1.1. Pattern extraction

The first step on Bag-of-Pattern approaches is to extract the respective patterns to be used for the feature representation. The classical approach applies a Sliding Window technique on the number of measures, which we refer to as *Sample-based Sliding Window*. More generalized approaches use Sliding Windows on the time of measures, which we refer to as *Time-based Sliding Window*.



(a) Forward Sliding Window.



(b) Backward Sliding Window.

Figure 4.1: Two-ways Sliding Window example. A fixed window width slides forward across a Time Series, reaches the end, and then slides backward.

On Sample-based Sliding Window, a window of W measures is defined and a fixed step of Δn measures is used to slide the window across the Time Series. With this, the window starts at the beginning (first measure on the Time Series), extract all the measures that are contained in the window (the pattern) and advance Δn measures to repeat the process.

For Time-based Sliding Window, the window is defined based on a time interval T and the step is now a Δt time. Here, the window starts at the time of the first measure (time zero) and advances across the Time Series in Δt steps.

A combination of both approaches is to define the window based on a fixed time interval T such that on every step it has a different sample aligned at the beginning of the window, i.e., if the window starts at the first measure (time zero), then on the next step it will have the second measure aligned with the start of the window, next the third measure, and so goes on. This approach extracts all possible sub sequences from a Time Series regardless of its sampling intervals.

The challenge for irregular Time Series is that some extracted patterns may have empty tails due to high irregularities in the time intervals. Furthermore, to generalize the procedure, we can combine generation of *empty-tail patterns* with *empty-head patterns*, since the pattern extraction process should not be affected by the time direction of the sliding window. The direct approach to address this is to apply a **two-way Sliding Window**, which slides the window forward and backwards, aligning each measure once with the start of the window and once with the end of the window. Figure 4.1 shows this two-way Sliding Window approach.

4.1.2. Pattern segmentation

After the patterns are extracted, two main approaches can be used for the representation method. One is based on codebooks, where a custom set of key-patterns is generated using a clustering strategy such as K-means or K-medoids [111]. Then, each pattern is represented by its closest key-pattern, and the Bag-of-Pattern matrix is constructed. The second approach is based on discretization, where each pattern is discretized to a symbolic representation (word) and a Bag-of-Pattern matrix is constructed based on a vocabulary of possible words. The first approach was already explored for the irregular Time Series by Valenzuela and Pichara [111] and Mackienzie et. al. [74]. Here we focus our work on the second approach, since it has more potential for linear time algorithms. Using a codebook implies using a similarity measure between the raw Time Series to find the best match between the sequences. Since our objective Time Series are irregular and multivariate, the similarity measure would most likely require a complex and computational expensive function.

To discretize a pattern to a collection of symbols, we have to divide each pattern into k segments and discretize each segment to a symbol. The segmentation criteria can be based on a number of samples or time intervals. Here, for our application domain we decide to segment each pattern by using the second case since it allows us to keep the time-significance of each segment. The number k is an argument introduced by a user or selected through grid-search with cross-validation.

As a result of this approach, some patterns are produced with *empty segments*. For example, as in figure 4.2, a pattern extracted from an irregular Time Series can have large gaps without any data, which leads after the segmentation to segments without data. The challenge of representing these empty segments is addressed by the discretization step described below.

4.2. Irregular Bag-of-Pattern Feature approach

Once we have defined the fixed methods of pattern extraction and pattern segmentation for irregular Time Series, we can apply the same approach as Li et. al. with Bag-of-Pattern Feature (BOPF) [66] for the pattern discretization. The first step here is to compute the mean-value for each segment as the representative quantity, excepting the empty segments for which no value is computed.

For discretization, the whole range of possible values is divided into α intervals defined by a distribution function. In the case of Symbolic Aggregate Approximation (SAX) [70], they use a normal distribution fitter for the whole pattern, which is divided into α intervals of the same area under the curve. An alternative option is to use a uniform distribution for a simpler discretization. The selected option will depend on the statistical quantity used. For example, for mean-value we can directly use a normal distribution. However, in the case

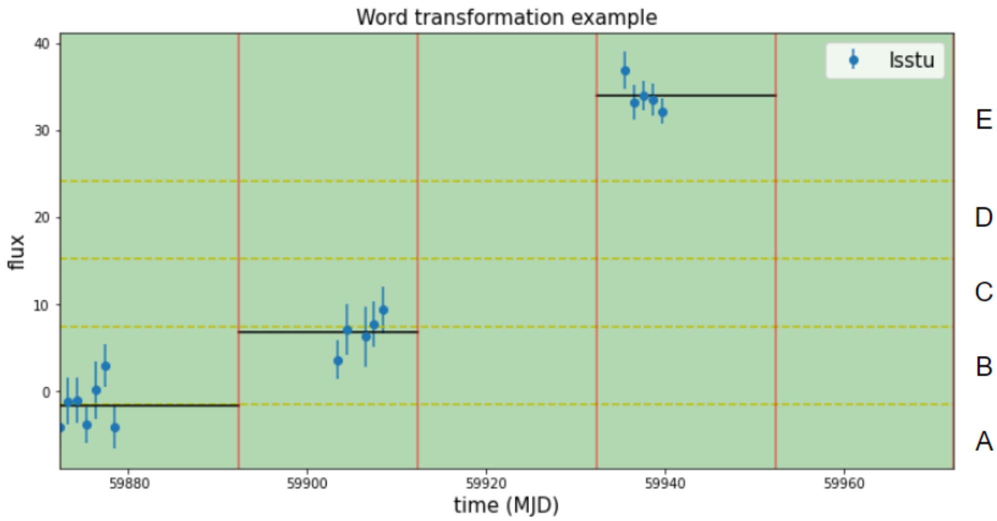


Figure 4.2: Pattern segmentation example, where a sub-sequence was divided into 5 segments and for each one of them a mean value is computed.

of slope-value, it is better to use a uniform distribution, dividing the whole possible range $[-\pi/2, \pi/2]$ into equal arcs.

Once the α intervals are defined, a symbol is assigned to each one and the patterns are discretized to a word. If the number of segments k is large enough, it will produce empty segments just as it was described before. Here we propose a naive approach based on the assumption that the absence of information (empty segment) is also relevant information, which is represented by a special character. For the example shown in Figure 4.2 the resulting word is $AB\#E\#$ where $\#$ is the special character. However, this approach is not useful on all possible values of k and its application variates depending on the value of k. For $k \leq 2$ we will directly ignore sub-sequences with empty segments since using the special character approach will produce meaningless words. In the case of $k > 2$, we will allow empty segments as long as they are less than half of the word and non-consecutive. For example, if $k = 5$, we will allow at most 2 non-consecutive empty segments, otherwise, we drop the sub-sequence. this is the more simpler choice to ensure integrity of symbolic representation, however, the threshold could be defined as an hyper-parameter and adjusted depending on the specific application.

Having defined our approach for discretizing a pattern, the following steps of Bag-of-Pattern Feature (BOPF) [66] can be applied in a straightforward way, after some modifications. Code 4.1 shows the general algorithm followed by our proposed method. Here it starts by defining a slider object, which has a window that will slide following the two-ways Sliding Window technique across the Time Series, extracting the respective sub-sequences. Then, each sub-sequence is segmented into ω segments, computing for each one of them the respective statistical quantity value and discretize using the Bag-of-Pattern approach. The discretization has α levels with the special character positioned at the discretization level $\alpha + 1$. The words are represented numerically with the corresponding position on the vocabulary size, which has $(\alpha + 1)^{\omega}$ different words, producing a list of numbers as the final document.

Code 4.1: I-BOPF algorithm for generating the document of words.

```
Inputs:
- Irregular Univariate Time Series 'ts'
- Statistical quantity 's'
- Window Width 'T'
- Word length 'ω'
- alphabet size 'α'
output: document of word ids 'd'

slider = TwoWaysSlidingWindow(ts, T)
d = ∅
while slider.not_finished() do
    sub_sequence = slider.get_sequence()
    symbols_set = ∅
    i = 0
    for segment ∈ segmentate_sequence(sub_sequence, ω) do
        if is_empty(segment) then
            val = α + 1
        else
            val = compute_quantity(s, segment)
        end if
        word_code = power(α + 1, i) * val
        symbols_set.add(word_code)
        i += 1
    end for
    word_id = symbols_set.sum()
    d.add(word_id)
    slider.advance_window()
end while
```

4.3. Generalized approach for Multivariate cases

For any irregular Multivariate Time Series, different types of correlations can exist between the measured variables. These correlations could be simple, like dependencies at the same time instant or more complex, like dependencies at large-scale or at different time instants. Most literature methods address the correlation between variables on the irregular Multivariate Time Series by re-sampling the Time Series to regular intervals. One of the most robust methods to perform this re-sampling is Gaussian Regression, which allows to modelate the irregular Multivariate Time Series in 2-dimensional space, addressing correlation between variables, even at different time instants [16]. However, these types of re-sampling techniques are extremely expensive and not suitable for a fast computing method. An alternative to re-sampling is to explore the application of Neural Network architecture to astronomical Time Series such as deep Recurrent Neural Networks using Phased Long Short-Term Memory units [31]. Although they can be more optimized than costly re-sampling methods, they are still computational expensive due to the large number of matrices involved in the many layers of the deep architecture.

A middle ground alternative is to only address correlation of variables at relatively close time instants, which will capture short-scale correlation but it will miss the large-scale correlation. This could be easily included in our proposed method by applying a synchronous Sliding Window on the irregular Multivariate Time Series, extracting features on each variable for the same time window. In the case of an irregular Multivariate Time Series, which has large time intervals without measure on any variable, an empty word is generated, using the special character strategy (See Section 4.1.2). Although this alternative addresses some part of the correlation, it produces a large number of empty-segments for large words, which leads to a large presence of words with empty-segments deriving in less significant representations. This can be seen from the fact that empty-segments represent the absence of information, which means that a high presence of words with empty-segments is a low level of valid information.

A simpler and naive approach is to consider the irregular Multivariate Time Series as many separate irregular Univariate Time Series and apply the Irregular Bag-of-Pattern Feature method to each one of them. This approach is simpler than previously described options and reduces the usage of special character strategy while losing most of the information of the correlation in raw-data. In the case of a Time Series without measures on one of the variables, the resulting feature vector for that irregular Univariate Time Series will be empty (only zeros).

We chose to use the latter method and left the handling of correlation to the compact technique. Two options are presented in Chapter 3 for producing a reduced and compact feature representation of the astronomical Time Series, MANOVA and LSA. MANOVA handles the correlation between multivariate-equivalent patterns. This is achieved after transforming each variable of the astronomical Time Series, producing its respective bag-of-Words (BOW) vectors and evaluating the relevance and correlation of each BOW symbol through all the variables. In the case of LSA, the method tries to characterize the latent relationships within a collection of documents (matrix of features). Here the strong assumption is that words close in meaning will occur in similar pieces of text [51]. To our representation method, this translates to similar patterns occurring in similar sequences of patterns or similar documents. If we include the multivariate patterns, similar patterns in one variable can be related to sequences of patterns involving multiple variables. Although this approach is not strictly the same as a correlation between variables, it has the potential to capture a large portion of this correlation.

4.3.1. Compact representation using MANOVA

For Multivariate Analysis of Variance (MANOVA), the features are ranked by their statistically significant score based on a hypothesis test on the variance between groups. Here we apply this feature selection on the BoW matrix, reducing the number of features and then apply TF-IDF Vector Space Model (VSM) for weighting the features.

Here MANOVA handles the correlation of the dataset in text-form, which is different from the correlation in raw-form. Since our representation method separates the variables in

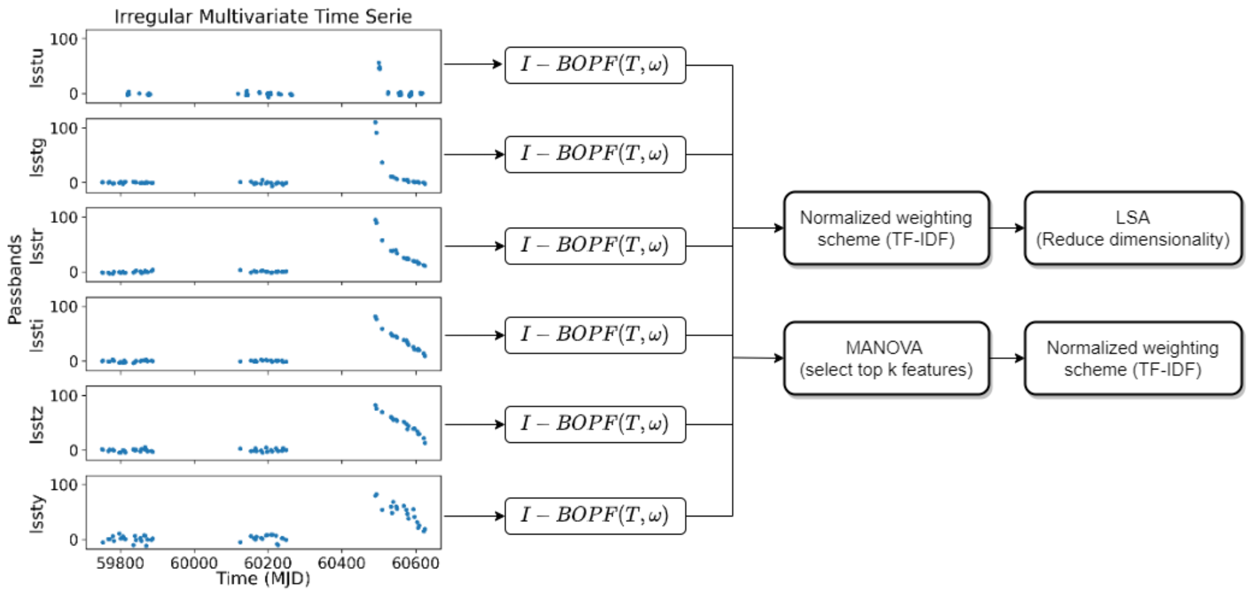


Figure 4.3: Generalized Irregular Bag-of-Pattern Feature diagram.

Univariate Time Series, we have lost the correlation information from raw-data and cannot be recovered. Instead, MANOVA handles the correlation between similar patterns represented by the same textual words and selects those more relevant by computing their cross-product impact to the variance between classes.

4.3.2. Compact representation using LSA

For Latent Semantic Analysis (LSA), the multivariate features are flattened and considered as separate features with some degree of correlation between them. LSA acts by decomposing the feature matrix, which was already flattened from 3D to 2D, into eigenvectors and coefficients, using them to represent the features in a dimensionaly reduced space. Here, LSA addresses the correlation between words and documents (rows and columns) during the Singular Value Decomposition (SVD). The final dimensional space includes the information of the top features in terms of variance. LSA quantifies the lost information due to dimensionality reduction by the loss of variance from those eigenvectros and coefficients not used on the transformation.

LSA is applied to the TF-IDF matrix normalized by cosine normalization, and the output can be directly introduced to any data mining algorithm. LSA directly handles the sparsity level of the TF-IDF matrix giving as output a dense matrix. An extra filter step is introduced in this compact technique, which discard features with zero variance across the dataset since it is expected that those features only produce bias to the representation method. This slightly speeds up the LSA method by reducing the number of features to be compacted.

A diagram showing how our Generalized Irregular Bag-of-Pattern Feature method works is presented in Figure 4.3, including both options for compacting the features. On the dia-

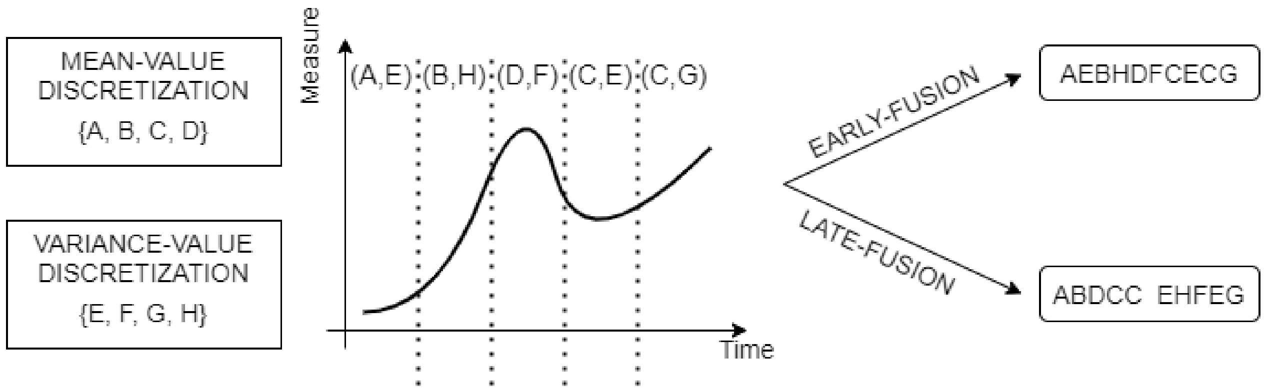


Figure 4.4: Example of early and late fusion schemes for statistical quantities. For the same sequence, divided into 5 segments, early-fusion and late-fusion are applied separately, generating two different word representations.

gram, we first split the astronomical Multi-Band Time Series into irregular UTS according to the measured variables. For each UTS we apply the proposed Irregular Bag-of-Pattern Feature method, producing its respective Bag-of-Word vector representation. For the example, an astronomical Time Series with 6 bands (variables) is used, which produces 6 different representation vectors. These vectors are then processed using either LSA or MANOVA techniques. For LSA, we concatenate the 6 representation vectors, apply a normalization weighting scheme and reduce its dimension through LSA. For MANOVA, we first apply the feature selection technique on the multivariate representation vectors, where each feature has 6 dimensions, then, the selected multivariate-features are flattened into one single 1-dimensional vector and a normalization weighted scheme is used. For our application we decided to use Term-Frequency Inverse-Document-Frequency (TF-IDF) as the normalization weighted scheme, which is described in Chapter 3.

4.4. Extensions on Irregular Bag-of-Pattern Feature

Although we presented an adaptation of BOPF method for irregular Time Series and generalized to irregular Multivariate Time Series by proposing the Generalized Irregular Bag-of-Pattern Feature method, it is still a very limited approach that may miss some relevant information on more complex applications. Two different extensions are introduced for a more robust representation method, which first needs to be validated in order to prove that they are actually improving the performance of the method.

4.4.1. Including Multi-Quantity representation

Our first proposed extension is based on adding more representative quantities to each pattern segment. Existing work have proposed to use mean-value combined with trend-value [75] or mean-value combined with min-value and max-value [73]. We propose a method to combine an arbitrary number of quantities using *early fusion* or *late fusion* schemes.

Code 4.2: I-BOPF algorithm for generating the document of words using early-fusion Multi-Quantity extension.

```
Inputs:
- Irregular Univariate Time Series 'ts'
- Statistical quantities 'S'
- Window Width 'T'
- Word length 'ω'
- alphabet size 'α'
output: document of word ids 'd'

slider = TwoWaysSlidingWindow(ts, T)
combined_alpha = power(α, |S|) + 1 \\ new line

d = ∅
while slider .not_finished() do
    sub_sequence = slider.get_sequence()
    symbols_set = ∅
    i = 0
    for segment ∈ segmentate_sequence(sub_sequence, ω) do
        if is_empty(segment) then
            val = combined_alpha \\ modified
        else
            val = 0
            weight = 1 \\ new line
            for s ∈ S do \\ new line
                quantity_result = compute_quantity(s, segment)
                val += weight * quantity_result \\ new line
                weight *= α \\ new line
        end if

        word_code = power(combined_alpha, i) * val \\ modified
        symbols_set.add(word_code)
        i += 1
    end for
    word_id = symbols_set.sum()
    d.add(word_id)
    slider .advance_window()
end while
```

For $Q = \{q_1, q_2, ...\}$ statistical quantities, early fusion combines the computed values for each segment and produce a higher discretization level. For example, if we define a discretization level of α intervals (symbols) and use the Q statistical quantities through early fusion scheme, the final discretization level for each segment will raise to:

$$\alpha_Q = \alpha_{q_1} \cdot \alpha_{q_2} \cdot ... \cdot \alpha_{q_{|Q|}} \tag{4.1}$$

However, we set the same discretization level α for all statistical quantities $q_i \in Q$, which reduces eq. 4.1 to:

$$\alpha_Q = \alpha^{|Q|} \tag{4.2}$$

where $|Q|$ is the number of quantities used. This early fusion scheme increases the complexity of the discretization but also adds more precision and information to the representation method.

For the late fusion scheme, we consider the Q statistical quantities as separate quantities, generate the respective symbolic word for each pattern and each statistical quantity, and add all the resulting words to the final document. For example, if we use two statistical quantities combined by late fusion, then for each pattern, two words are generated, one for each statistical quantity. Figure 4.4 illustrates the early fusion and late fusion schemes for Multi-Quantity approach. This diagram presents an example pattern, which is transformed into symbolic words using mean-value and variance-value. If we only use mean-value, the resulting word is $ABDCC$, and if we only use variance-value, the representation is $EHFEG$. Under a late-fusion scheme, we would just concatenate these two symbolic words into a phrase $ABDCC\ \ EHFEG$. However, if we use the early-fusion scheme, the symbolic words are merged symbol by symbol, producing one combined symbolic words of twice the length of the original, in this case, $AEBHDFCECG$. Code 4.2 shows how the original code presented in Code 4.1 is modified to work with early-fusion scheme on Multi-Quantity.

The quantities proposed to use are Mean-value, Trend-value, Variance-value, Min-value, Max-value and MinMax-value. The formulation and discretization strategy used for each quantity are presented below.

Mean-value Computes the statistical mean value within a pattern segment by

$$\mu = \frac{1}{n}\sum_{i=1}^{n} v_i \tag{4.3}$$

Where n is the total number of measures v_i in the segment. The discretization is based on fitting normal distribution to the whole pattern and divide the distribution curve into α intervals of the same area.

Trend-value Computes the slope of a segment by linear regression.

$$m = \frac{\sum_{i=1}^{n}(t_i - \bar{t})(v_i - \mu)}{\sum i = 1^n (t_i - \bar{t})^2} \tag{4.4}$$

Where t_i is the time instant of the measure v_i and $\bar{t}$ is the mean time of the segment. The slope is then transformed to angle by $m_{angle} = tan^{-1}(m)$ and discretized in the range of $[-\pi/2, \pi/2]$ using uniform distribution.

Variance-value Computes the statistical sample variance within a pattern segment by

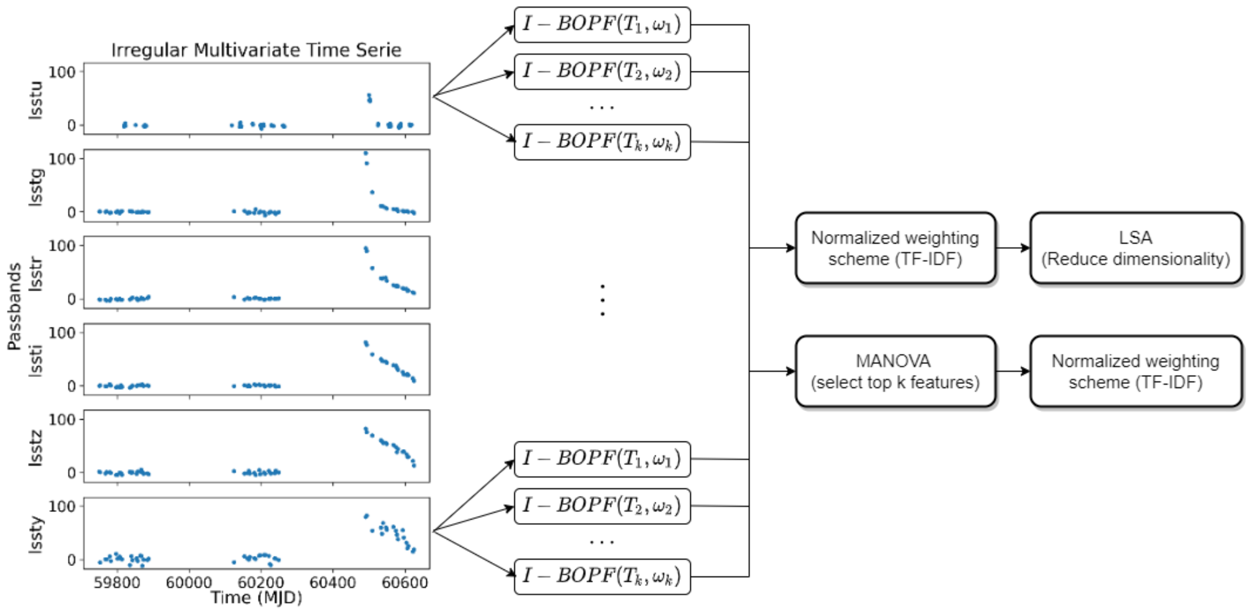


Figure 4.5: Generalized Irregular Bag-of-Pattern Features diagram with Multi-Resolution extension.

$$Var[\text{segment}] = \sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(v_i - \mu)^2 \tag{4.5}$$

The discretization is defined by the range of $[0, max(\text{pattern}) - min(\text{pattern})]$ using a uniform distribution.

Min-value Takes the minimum value within a pattern segment $min(\text{segment})$ and discretize in the same way as Mean-value does.

Max-value Takes the maximum value within a pattern segment $max(\text{segment})$ and discretize in the same way as Mean-value does.

MinMax-value Takes the difference between the maximum value and the minimum value within a pattern segment $max(\text{segment}) - min(\text{segment})$ and discretize in the same way as Variance-value.

All of these quantities have linear computational complexity, which is key for a fast method. However, we cannot expect to use all of these five quantities to represent a Time Series, mainly because they will add a large number of features to the representation method resulting in an extremely large BoW matrix. To reduce this, the user can define the desired quantities to use or apply a grid-search algorithm to find an optimal combination of quantities.

4.4.2. Including Multi-Resolution representation

We also propose to extend the Bag-of-Pattern method for a Multi-Resolution scheme of multiple pattern extractions and pattern segmentation. For pattern extraction, the window width T of the two-ways Sliding Window technique is a fixed value. The same goes for the number of segments ω of the pattern segmentation process. Here we propose to use a series of combinations (T_i, ω_i) for different *levels of resolution*.

For example, a direct approach is to change T over different values and fix ω, producing a Multi-Resolution representation defined by $\{(T_1, \omega), (T_2, \omega), ..., (T_{mr}, \omega)\}$ for mr levels of resolution. The values of T_i are defined by an arbitrary sequence. Another option is to fix T and vary ω producing a Multi-Resolution representation defined by $\{(T, \omega_1), (T, \omega_2), ..., (T, \omega_{mr})\}$ for mr levels of resolution and ω_i defined by an arbitrary sequence of natural numbers. Furthermore, a user can choose to vary both T and k simultaneously, producing a representation defined by $\{(T_1, \omega_1), (T_2, \omega_2), ..., (T_{mr}, \omega_{mr})\}$. The levels of resolutions can be defined by the user or through an incremental grid-search algorithm.

For the representation method, the levels of resolution are combined by late fusion scheme where the final feature matrix is a concatenation of feature matrices for each level of resolution. Depending on how many levels of resolution we use, and the values of ω_i, the resulting number of features for each Time Series can be extremely large. However, the sparsity level is high as well, allowing us to reduce the memory usage drastically. Figure 4.5 shows an illustration example of our Multi-Resolution approach.

The final pipeline is presented in Code 4.3, where each run of I-BOPF(T, w, s) has a computational complexity of $O(n)$ for a Time Series of n measures. VectorSpaceModel() is a function that generates the respective space model according to the Information Retrieval theory described in chapter 3. LSA and MANOVA both require the target number of features for the features reduction, with MANOVA also requiring the labels of each Time Series on the dataset in order to fit the model. The Vector Space Model labeled as *bow-3d* is just a method that generates a 3-dimensional Bag-of-Word matrix where each feature has multiples values, one for each variable.

4.5. Discussion

We have proposed a new representation method oriented for irregular Multivariate Time Series data with extensions on Multi-Resolution and Multi-Quantity approaches. Our method combines the classic Bag-of-Pattern Feature (BOPF) method [66] with an adaptation for irregular Time Series and Multivariate measures, and the option to apply Latent Semantic Analysis (LSA) or Multivariate Analysis of Variance (MANOVA) for reducing the number of features.

Code 4.3: Generalized I-BOPF, with Multi-Quantity and Multi-Resolution.

```
Inputs:
- Dataset D
-  Statistical  quantities  'Q'
- Levels of resolution  'R'
- alphabet size  'α'
- Compact method 'C' (options: LSA, MANOVA)
- Number of components 'N'
output: feature matrix 'M'

corpus = ∅
for ts ∈ D do:
    document = ∅
    for (T, w) ∈ R do
        for q ∈ Q do
            words_feature = I-BOPF(T, w, s, α)
            document.add(words_feature)
        end for
    end for
    corpus.add(document)
end for

if C == LSA then
    tf_idf_corpus = VectorSpaceModel(corpus, 'tf_idf')
    M = LSA(n_components=N).fit_transform(tf_idf_corpus)
else
    bow_3d_corpus = VectorSpaceModel(corpus, 'bow-3d')
    bow_reduced_corpus = MANOVA(k=N).fit_transform(bow_corpus, D.labels)
    M = VectorSpaceModel(bow_reduced_corpus, 'tf_idf')
end if
```

4.5.1. Proposed methods

Our work includes the suggestion of an extended Sliding Window technique called two-ways Sliding Window, which extracts all sub-sequences (patterns) from the Time Series by sliding a window forward and backward. We also propose a symbolic representation method for sequences with missing values by adding a special character to represent the absence of information. This produces a representation method where the measuring instants have an impact on the representation method.

The generalization proposed for irregular Multivariate Time Series consist of a naive approach of using the multivariate values by separate, applying Irregular Bag-of-Pattern Feature to each irregular Univariate Time Series and combining all of those features into one single representation, letting the compact method (LSA or MANOVA) decide whether there is any correlation between the respective features. Although this approach is naive, it is as faster as we can achieve for our method on multivariate data while somehow still addressing a correlation between variables.

The extensions proposed to our Irregular Bag-of-Pattern Feature method are two. The first is related to statistical quantities where we allow the algorithm to work with two or more quantities for symbolic representation. For example, classic BOPF only uses Mean-value as discretization quantity, in our method we can use Mean-value, Trend-value, Variance-value, Min-value, Max-value, MinMax-value or a combination of them adding more precision to the representation method. The second extension is for the resolution level defined by the window width T of the Sliding Window process and the segmentation size ω of the pattern segmentation step. We propose to use an arbitrary number of resolutions (T_i, ω_i) with values defined by the user or optimized by a grid-search algorithm.

4.5.2. Computational complexity

The computational complexity of our methods has the advantage of being linear $O(n)$. The training phase may be more expensive due to the application of MANOVA or LSA techniques, but after the computation of those methods, the time required to transform a Time Series to a feature vector is always $O(n)$.

The computational complexity of MANOVA is $O(wb^2n^2)$ for w features, b bands and n Time Series, being w a large constant and b a small number. On the other hand, LSA has a computational complexity of $O(n^3b^3)$ when n is larger than w, otherwise it is $O(nbw^2)$. For a large train-set ($n > w$), we will have that MANOVA should be faster than LSA. For small datasets ($n < w$), we should have LSA faster than MANOVA.

In the case of our proposed grid-search algorithm for Multi-Quantity and Multi-Resolution, we do a series of repetitions for each combination of parameters, where each repetition involves the re-computation of the whole algorithm. This is the most expensive step from our representation method since it performs cross-validation on each iteration of the grid-search. Luckily, we only need to run the grid-search algorithm once.

4.5.3. Data Mining application

Our proposed method takes as input the raw Time Series data, applies a series of transformations and generates a feature vector for each Time Series. These vectors are of the same length, and they live in a regular feature space where element-wise comparison can be applied (for example, L2-norm or Cosine Similarity). This means that our representation method can be applied on almost any data mining algorithm without major issues.

In particular, we propose our method to solve classification tasks and similarity search queries. Our method is designed to work by itself or in combination with other features (from metadata or quantities computed by the user).

Chapter 5

Experimental Evaluation

In this chapter, we describe our experiments and results related to our proposed representation method (see Chapter 4). In Section 5.1 we describe the implementation of our algorithm, which is an open source code. Then, we present the dataset used for the experimental evaluation (Section 5.2) and the literature methods used for comparison (Section 5.3). On the parameter exploration, we evaluate the performance of including Multi-Quantity (Section 5.4.1) and Multi-Resolution (Section 5.4.2) extensions to our proposed method, including a summary with the hyper-parameter selected to run our proposed representation method algorithm. The final experiments are on classification tasks (Section **??**) and computing times (Section 5.7) where our proposed method shows to be faster but AVOCADO shows to be better at classifying the dataset.

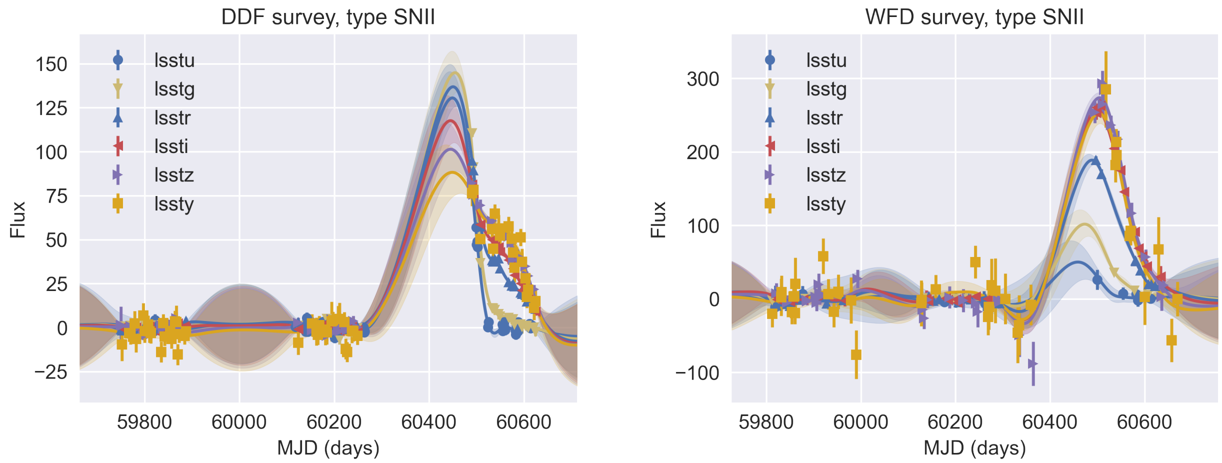
5.1. Implementation

We implement our method using Python language under open source license MIT. The principal libraries used are **Scikit-learn** for machine learning methods and **Numpy** for fast numerical computation. The whole code is available on github for free [1].

The implementation includes a feature extraction module designed for Multivariate Time Series where patterns are extracted following the method described in Chapter 4. This feature extraction module is programmed using **Numpy** and optimized using **Numba**. The rest of the code consist on adaptations of different **Scikit-learn** modules (e.g. *KNeighborsClassifier*, *TruncatedSVD* or *TfidfTransformer*) for our usage case. Although **Scikit-learn** is a very complete library, which has almost every machine learning method we use, it does not have implemented Multivariate Analysis of Variance (See Section 3.3.3) for which we have used the library **Statsmodels**.

The public code also includes the scripts used for experimental evaluation and visualization, for which results are shown in the following sections.

[1] https://github.com/frmunozz/irregular-bag-of-pattern



(a) well-sampled, high Signal-to-Noise ratio light curve.

(b) poorly-sampled, low Signal-to-Noise ratio light curve.

Figure 5.1: PLaSTiCC Light curve examples with GP-fit model, where the mean GP flux prediction is shown as a solid line surrounded by a shaded contour indicating the 1σ deviation [16].

5.2. Dataset

Our experimental evaluation focuses on PLaSTiCC (Photometric LSST Astronomical Time-series Classification Challenge) dataset. This dataset contains synthetic irregular Multivariate Time Series and, as the name indicates, was generated for a classification challenge. The data simulates observations from the Large Synoptic Survey Telescope (LSST) on two different kind of suverys, the Deep Driling Fields (DDF) that captures less objects (i.e. a small portion of the sky) more often, and the Wide-Fast-Deep (WDF) that captures more objects (i.e. a larger portion of the sky) less frequently. The DDF survey is oriented to observe a small number of fainted objects with precise light curves and the WDF is oriented to discover more new objects with noisy light curves (see Figure 5.1). The challenge consist on classifying an extremely large test set (more than 3 million objects) using a very small train set (only 7846 objects), which is highly unbalanced (see Figure 5.3) and not representative, containing only 14 classes out of the 19 used for simulating the observations [109]. Table 5.1 shows a brief summary of the object types distribution on PLaSTiCC.

As it was shown in Figure 5.1, the objects included in PLasTiCC dataset have irregular Multivariate Time Series curves, with extreme variations in the number of observations per Time Series (see Fig 5.2), mainly due to the DDF/WDF surveys. The variables measured in this Time Series are electromagnetic wavelength passbands identified with the labels u, g, r, i, z, y. The u-band measures the range of 300-400 nanometers, the g-band measures the 400-600 nanometers, the r-band measures the 500-700 nanometers, the i-band measures the 650-850 nanometers, the z-band measures the 850-950 nanometers, and the y-band measures the 950-1050 nanometers.

To handle the unbalanced train set, a data augmentation is performed following the pro-

Table 5.1: Summary of the object types included in the PLaSTiCC dataset. The table includes the random ID number, the full and short name of each object type, the number of objects on train set N_{train}, test set N_{test}, and the ratio between both sets N_{train}/N_{test}. A more complete summary can be found on the Unblinded release of PLaSTiCC [109]

ID	Object type name	N_{train}	N_{test}	N_{train}/N_{test}
90	Type Ia Supernova (SNIa)	2,313	1,659,831	0.0014
67	Peculiar type Ia-91bg Supernova (SNIa-91bg)	208	40,193	0.0052
52	Peculiar type Ia Supernova (SNIax)	183	63,664	0.0029
42	Type II Supernova (SNII)	1,193	1,000,150	0.0012
62	Type Ibc Supernova (SNIbc)	484	175,094	0.0028
95	Superluminous Supernova (SLSN)	175	35,782	0.0049
15	Tidal Disruption Event (TDE)	495	13,555	0.0365
64	Kilonova (KN)	100	131	0.7634
88	Active Galactic Nuclei (AGN)	370	101,424	0.0036
92	RR Lyrae (RRL)	239	197,155	0.0012
65	M-dwarf stellar flare (M-dwarf)	981	93,494	0.0105
16	Eclipsing Binary stars (EB)	924	96,572	0.0096
53	Pulsating Variable Star type Mira (Mira)	30	1,453	0.0206
6	Microlens from single lens (Single μ-lens)	151	1,303	0.1159
991	Microlens from binary lens	0	533	0.0000
992	Intermediate Luminuos Pptical Transient	0	1,702	0.0000
993	Calcium Rich Transient	0	9,680	0.0000
994	Pair Instability Supernova	0	1,172	0.0000
	TOTAL: sum of all types	7,846	3,492,888	0.0022

cedure designed in AVOCADO [16]. This method consists in performing a Gaussian Process regression in two-dimensions (time and wavelength) and creating a new Time Series modifying both values, time and wavelength. Given the data augmentation method, the procedure we follow is:

1. Generate up to 100 augmented Time Series from each Time Series in the train set. Well sampled Time Series can easily be augmented 100 times using Random number Generation (RNG). However, when the Time Series are poorly samples, such as the ones in the WDF survey, most of the RNG augmentations are going to produce augmented Time Series that are only noise and thus discarded. This produce an imbalance in the data augmentation that it is solved in the next step.

2. Set a target train set size and maximum number of augmented Time Series per class. This is included to reduce those classes with an excessive number of augmented Time Series due to well sampled sequences, for example, those Time Series in the DDF-Survey. The method ensures to keep the original Time Series and only drop augmented sequences.

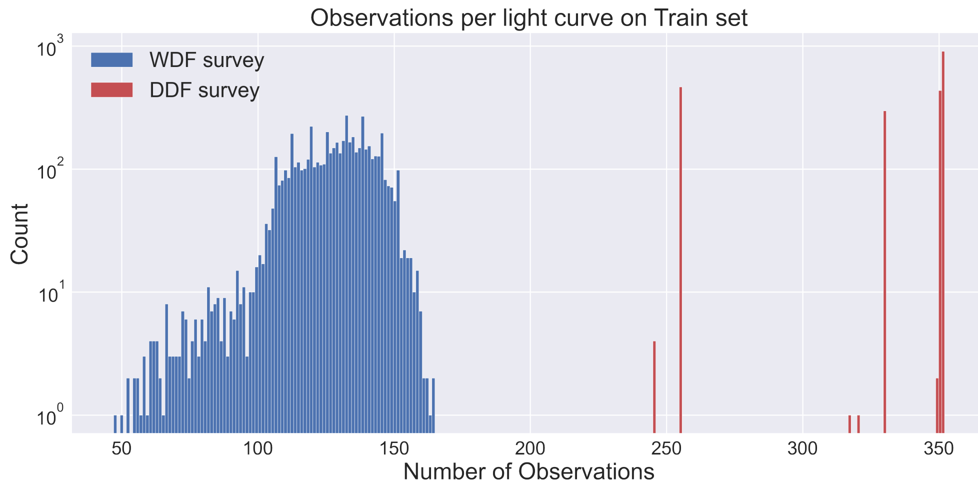


Figure 5.2: Number of observations per Time Series on train set grouped by DDF/WDF surveys.

For our experimental evaluation we choose to use up to 6000 augmented Time Series per class, this means, for example, that if the class on the original train set has 600 Time Series, then each Time Series needs to produce 10 new augmented Time Series, resulting in 6600 Time Series in the augmented train set. Sadly, not all Time Series can produce the same number of augmented Time Series since the augmentation method uses random parameters, which in some cases can produce Time Series that are only noise, which are discarded. Each augmented Time Series simulates detections through a Random Number Generation probability on the Signal-to-Noise ratio (SNR). The SNR is defined as

$$\mathrm{SNR}_i = \frac{y_i}{err(y_i)} \tag{5.1}$$

where y_i is the simulated flux and $err(y_i)$ is the simulated flux error for the i-th measure, both produced through Gaussian Process regression. Using the error function

$$erf(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt \tag{5.2}$$

A simulated detection is valid if and only if [16]

$$\mathrm{RNG}_i < \frac{erf((\mathrm{SNR}_i - 5.5)/2) + 1}{2} \tag{5.3}$$

Where RNG_i is the Random Probability produced for the given detection. Finally, an augmented Time Series is considered only noise if it has 2 or less valid detections. This results in some classes not having more than 6000 Time Series after the augmentation. Figure 5.3 shows the histogram of this augmented train set. One important remark about the original dataset is that there are 4 classes that are not present in the train set (See Table 5.1). They were ignored in our experiments and dropped from the test set for the sake of simplicity.

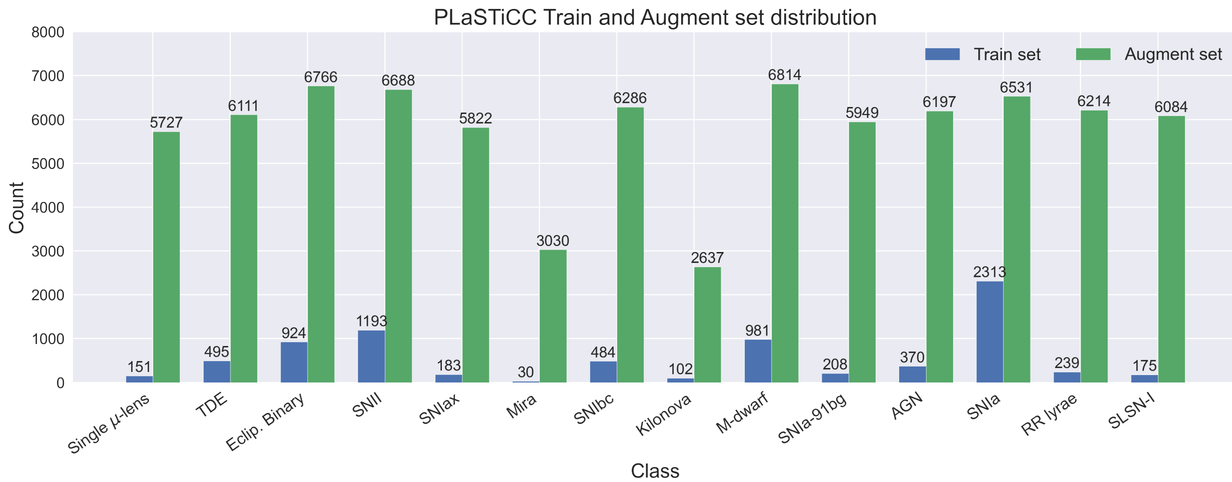


Figure 5.3: Class distribution on the PLaSTiCC train set and augmented train set.

5.3. Literature methods for comparison

For comparison we use the winning method of PLaSTiCC, proposed by Kyle Boone and called **AVOCADO** [16]. This solution proposes to modelate each irregular Multivariate Time Series as a 2D Gaussian Process (2D-GP) on time-wavelength domain. The 2D-GP model is used to fit the light curve on a target wavelength passband, allowing to re-sample the observations and compute different features used for classification such as Peak magnitude of the flux, fraction of observations that have an absolute signal-to-noise less than 3, the total absolute signal-to-noise ratio or the positive flux ratio, among many others [16], using in total 41 features, where some of them can be undefined if the respective Time Series does not have enough data. The main advantage of 2D-GP is that it handles the correlation between passbands (variables) by generating a multivariate model. Furthermore, the same 2D-GP fit is used for data-augmentation, which is explained in the AVOCADO's article [16].

AVOCADO performed very well using a Light Gradient Boosting Machine (LightGBM) classifier with the features extracted from the Time Series. On this classifier, 27 features were provided from the AVOCADO method and 2 additional features were added from the metadata table, which results in a balanced accuracy score of 90 % [16]. For comparison, we evaluate our proposed method on the same classifier using our computed features and the same 2 metadata features. The only difference is the augmented dataset, which for our case is smaller and more balanced than the one used in previous works for AVOCADO.

5.4. Parameter explorations

Our proposed method requires many different hyper-parameters, most of them defined by the user and other computed internally by the algorithm. From literature we already know that the idea of Bag-of-Pattern Feature can work on Time Series, and from that we can expect that our adaptation to irregular Multivariate Time Series can work too. However, we

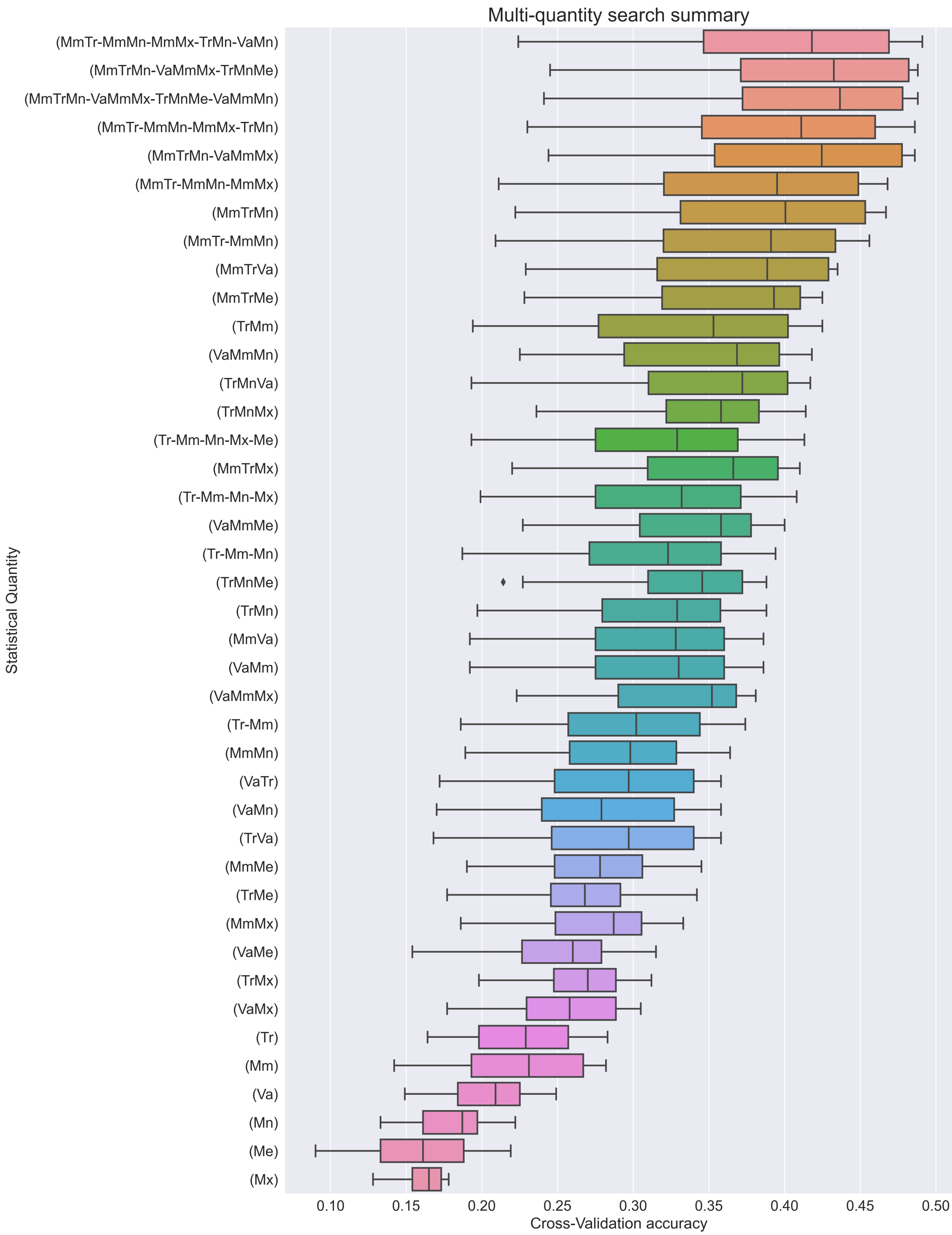


Figure 5.4: Results of Multi-Quantity evaluations for the statistical quantities Mean (Me), Min (Mn), Max (Mx), Min-Max (mm), Trend (Tr) and Variance (Va). The notation follows the pattern s_1-s_2-... where each s_i is a statistical quantity, single or composed through early-fusion, and all s_i are combined through late-fusion. For an explanation on what these boxplot are, see Section 2.1.3.

do not know if our proposed extensions for this method, called Multi-Quantity and Multi-Resolution, can effectively improve the performance of the representation method. In order to validate this assumption, we perform many different experiments using k-fold cross-validation technique on the original train set (not augmented). In particular, two explorations are made, first for the Multi-Quantity and second for the Multi-Resolution, using on the last one the results obtained from the first.

The values for all of the other hyper-parameters are based either on theory or previous literature results:

- Alphabet size (α): we choose $\alpha = 4$ since it is the most used value on literature, where they have shown that for $\alpha < 4$ the performance is worst and for $\alpha > 4$ the increase in performance does not compensate the increase in the vocabulary size [66, 71].
- Word length (ω): can vary between some margins given by the vocabulary size, which is set to be not higher than 4^6 features. In practice, this means that $\omega \in (1, 6)$.
- Window Width (T): can vary between an interval given by the data itself. If the dataset has a mean time duration of Time Series of μ and a variance of σ, then $T < \mu + \sigma$. Furthermore, for PLaSTiCC a lower limit is also present, where for $T < 20$ the method is not able to capture enough information to produce a meaningful representation. The discrete values are produced through a logarithmic scale following the equation

$$log_{10}(T_i) = \frac{(i-1)}{m_i}\left[log_{10}(\mu + \sigma) - log_{10}(20)\right] + log_{10}(20), i \in \{1, 2, 3, ..., m_i\}, \tag{5.4}$$

  Which produces closer short windows and distant long windows.
- Compact technique (C): either LSA or MANOVA could be used. For parameter exploration we choose to use LSA, but all experiments can be replicated using MANOVA as well.
- Target number of features (N): the target value is set to be the highest possible value as long as the representation method can still be considered a compact representation. Although we know that on the multi-variate measures each observation on the Time Series involves (time, measure, variable), we will only consider (time, measure) as the used data. Following this, for PLaSTiCC we have 181 observations per Time Series in average, which mean 362 values in average, resulting in $N = 361$ values for the compact representation method. LAter on this chapter we will explore representations with different number of features to see how it changes when N variates.

5.4.1. Multi-Quantity search

In the literature we found some work that proposed more than one statistical quantity for representing Time Series on the Bag-of-Pattern Feature approach with promising results [18].

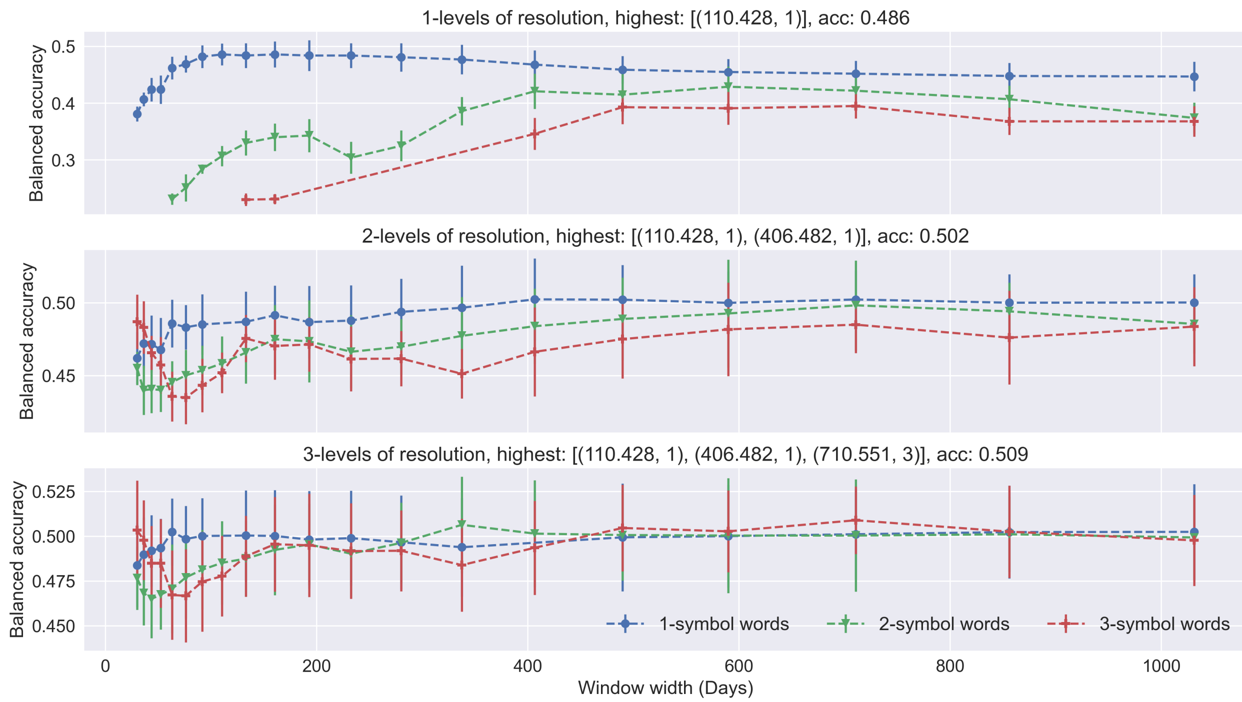


Figure 5.5: Results of Multi-Resolution evaluations for an incremental number of levels of resolution. On each level, different values for word length ω and window width T were tested and the best pair on each level is selected.

However, they only provide a theoretical analysis as a basis for choosing the statistical quantities, and they only combine them using a late-fusion scheme.

As we described in Chapter 4, there are two ways of combining statistical quantities, early-fusion and late-fusion, with an additional possibility of combining both schemes, applying first many different early-fusions, and combining all of them with a late-fusion scheme. Here we try to evaluate which one is the best combination of statistical quantities using a k-fold cross-validation score. For the experimentation we set a maximum number of early-fusion to be 3 and a maximum number of late-fusion to be 6, having as absolute upper limit a vocabulary size of 4^8. For each combination of statistical quantities, we get the k-fold cross-validation score for different combinations of window width T and word length ω, this is made in order to produce more robust results testing the performance of the combination of statistical quantities on different parameter configurations.

The exploration start considering six different statistical quantities: Mean (Me), Min (Mn), Max (Mx), Min-Max (Mm), Trend (Tr) and Variance (Va), and the first evaluation is made on single-quantities, without early-fusion and with late-fusion. Then we proceed to evaluate double-quantities, with an early-fusion of 2 quantities and a late-fusion of at most 5, where the top-3 quantities from the previous single-quantity step are used as basis for the early fusion, reducing the spectrum of possibilities. The final step is for triple-quantities, with an early-fusion of 3 statistical quantities and a late-fusion of at most 4, where again the top-3 quantities from the previous double-quantity step are used as basis for the early-fusion. In total, 41 different combinations of statistical quantities were tested, from which the

Table 5.2: Parameters required by our proposed method, describing each parameter name, symbol, criteria applied for its use and used value.

Parameter	Symbol	Criteria	Value
Alphabet size	α	Optimized	4
Statistical quantities	S	Optimized	(MmTr-MmMn-MmM-TrMn)
Levels of resolution	R	Optimized	$[(110.428, 1), (406.482, 1)]$
Representation size	N	User-defined	< 362
Compact repr. method	C	To be chosen	LSA/MANOVA/UMAP

selected Multi-Quantity is (MmTr-MmMn-MmMx-TrMn) since it provides the highest score while having a relatively small vocabulary size. A Summary of all the experiments can be seen on Figure 5.4, where the distribution correspond to evaluations performed on different combinations of (T, ω).

5.4.2. Optimal Levels of resolution

Having decided the best combination of statistical quantities S, we find the best combination of levels of resolutions R. For this, we perform an incremental grid search, where on each step we use as a base the best combination of the previous step. The search starts looking the best pair (T_1, ω_1), once it is selected, it search again for a second pair (T_2, ω_2) such that both pairs combined gives better accuracy than before. The process repeats until 4 pairs are selected.

In this case we cannot repeat the procedure used before of testing on different pairs (T, ω) since here we are looking for an optimal set of pairs (T, ω). Instead, we perform an incremental grid-search on the evaluated pairs (T, ω), where we first find one optimal pair, then search for a second optimal pair that combined with the previus pair increase the classification score, and repeat the procedure until no new pair can increase the score. The incremental steps can be seen on Figure 5.5. Here we directly see that the improvement made by the Multi-Resolution is not that relevant compared to the improvement made by the Multi-Quantity. This can be noticed by the relatively high standard deviation obtained from the k-fold cross-validation, which on 3-levels of resolution made it very difficult to differentiate any improvement. Hence, we found that 2-levels of resolution is enough to increase the performance of our method to its best results.

A summary of the optimal parameters selected for further experimental evaluations can be seen on Table 5.2, which translates into 816 sparse features per Time Series using our proposed method. The compact method is yet to be defined in the next sections, but we already know that for the representation to be compact there must be at most 361 dense features.

5.5. Classification Comparison with the State-of-the-art

The classification comparison experiments are divided in three steps. First, we perform a comparison among the three compact techniques described in Chapter 3, including combinations of our proposed method with the state-of-the-art method. Here we find which one of the three compact techniques is going to be used for further evaluations by performing classifications on 10 % of the full test-set. Then, we perform a full-classification experiment where we use the full test set to evaluate performance of the selected configuration of our proposed method and the state-of-the-art method. Finally, we include additional experiments where we evaluate classification on subsets dataset using certain classes.

5.5.1. Optimal compact technique

Once the parameters for the textual sparse representation are selected, we have to find the compact method with the best feature space for classification. On Chapter 3 we have described two potential compact methods; LSA and MANOVA. Additionally, we evaluate Uniform Manifold Approximation and Projection (UMAP), a dimension reduction method mostly used for visualization that can be applied to transform new data [5, 33, 95]. Since UMAP is a non-linear method that mostly focuses on preserving local structure (manifold approach), we use it as a comparison with LSA and MANOVA, which are linear-methods that preserve global structure (variance approach). Three variations for UMAP are included, UMAP using Cosine distance, UMAP using Euclidean distance and UMAP using a semi-supervised approach, where it uses the labels during the training set to guide the learning of the representation space. Since the UMAP method involves computing K-Nearest Neighbor, we set the number of neighbors to 100 so it can capture more global structure, and the minimum distance to 0.0 to reduce the dispersion on the inferred space.

In addition to the compact method variants on our proposed method, we evaluate the possibility of combining the features produced by our method with the features produced by AVOCADO, which evaluates if the information captured by both methods can be complemented or not.

Figure 5.6 shows the classification results of our proposed method with all the previously described compact method variants and the combinations of AVOCADO features with our proposed method on certain configurations. For the classification, LightGBM classifier was applied, using the parameter configuration established by AVOCADO [16]. Since we tested a large number of configurations, we have used only 10 % of the test set. IBOPF-(LSA), IBOPF-(MANOVA), IBOPF-(UMAP Euclidean), IBOPF-(UMAP Cosine) and IBOPF-(sup-UMAP) shows the classification accuracy using our proposed method with a compact technique at different number of components. IBOPF-sparse|AVOCADO-(LSA), IBOPF-sparse|AVOCADO-(UMAP) and IBOPF-LSA|AVOCADO shows the classification accuracy using a combination of our proposed method with AVOCADO, the first combines the sparse IBOPF representation with AVOCADO and then applies LSA to reduce dimension, the second is similar to the previous one, but applies UMAP Cosine instead of LSA, the last one combines IBOPF

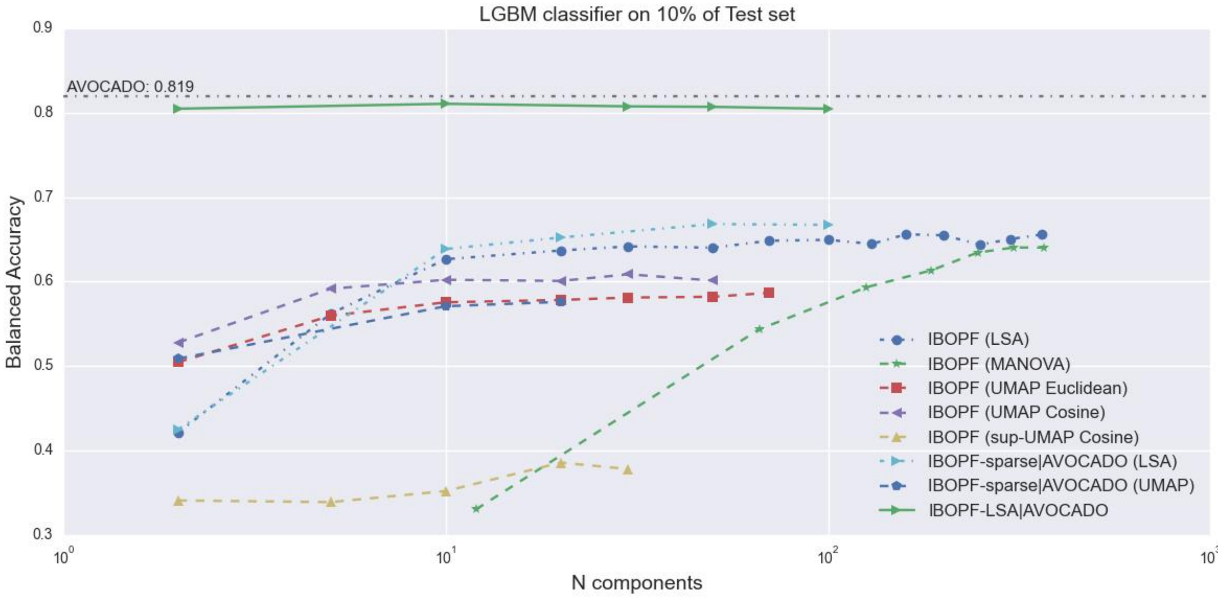


Figure 5.6: Comparison of different combinations of IBOPF with compact methods. The metric used is balanced accuracy on classification tasks considering a random 10 % of the full test-set. The figure includes configurations that combines the proposed method with state-of-the-art method, these are named *IBOPF-sparse|AVOCADO (LSA)*, *IBOPF-sparse|AVOCADO (UMAP)*, and *IBOPF-LSA|AVOCADO*. The dashed horizontal lines represent the balanced accuracy achieved by AVOCADO alone.

compacted through LSA with AVOCADO.

The results show that combining the features of our proposed method with the features of AVOCADO is not a viable option on classification tasks, since it does not improve the accuracy compared with the state-of-the-art method alone. For the compact methods, we see that the supervised UMAP is not able to produce a robust representation and fails drastically on classifying the test set. For MANOVA we see that it has the lowest accuracy for low dimensions and performs average on high dimension. Finally, LSA shows the best results for 10 dimensions or higher. Furthermore, since LSA is orders of magnitude faster than UMAP, we directly choose to use LSA as the main compact technique for our proposed method.

On the number of components, we see that LSA improves significantly up to 10 components, and from there the increase in accuracy is very slow up to 100 components, from where it remains almost constant until reaching 361 components, which is the maximum number of components to keep the concept of compact. This means that although we could use 361 components to minimize the loss of information, with 100 components is enough to generate an optimal representation for solving classification tasks. Furthermore, with 10 components we could produce a good classifier considering the trade-off between spatial complexity and accuracy. For less than 10 dimensions, UMAP shows the highest results, which is expected since it is a method designed for visualization on low dimensions.

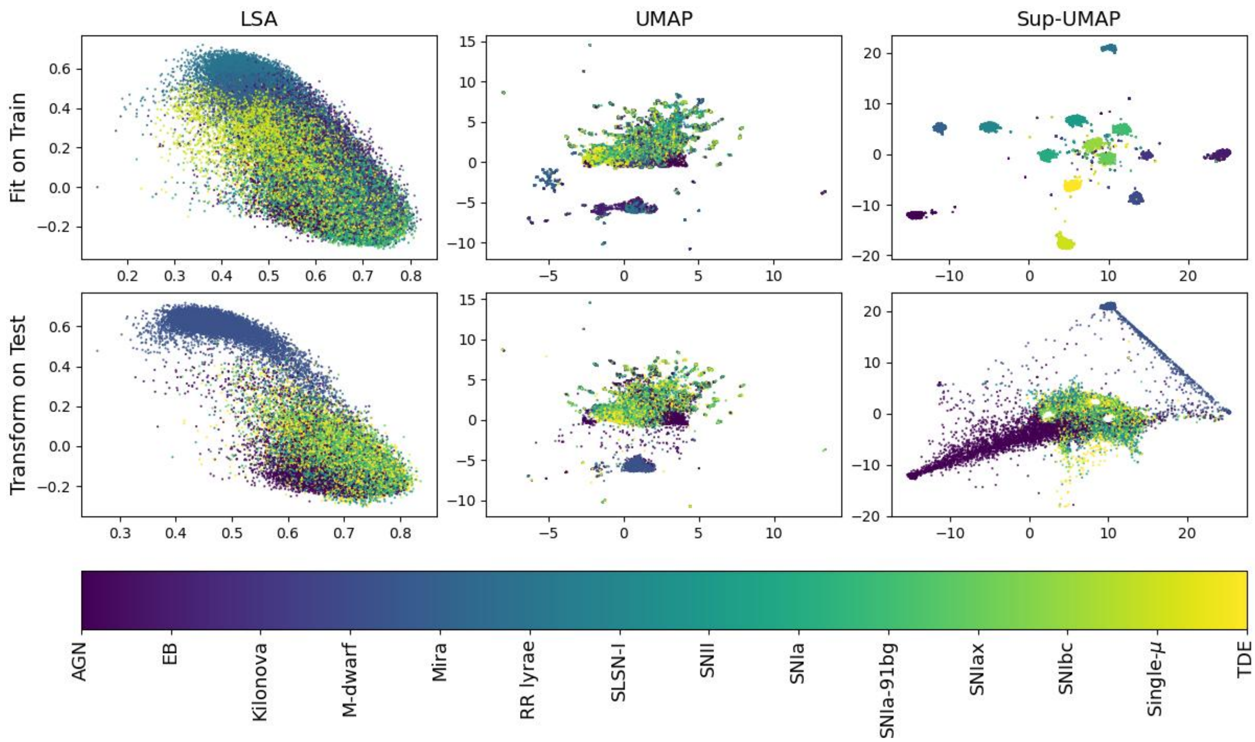


Figure 5.7: 2-dimensions visualization of compact methods LSA, UMAP with Cosine distance and Supervised UMAP with Cosine distance, on train set (fit) and test set (transform). The transform is evaluated on a small subsample of the test set. Colors represents the classes, where it is expected that similar color groups closer and separated from the rest.

Figure 5.7 shows a two-dimensions visualization of IBOPF-(LSA), IBOPF-(UMAP Cosine) and IBOPF-(sup-UMAP Cosine). This visualization helps to clarify the extremely bad performance of Supervised UMAP on classification tasks. Here we see that although Supervised UMAP learns an excellent compact space for the sparse features on the training set, showing very dense visual clusters, it is not able to reproduce this learned space on the test set, producing a similar effect to overfitting. This is related to the issue of the training set not being representative of the test set since it is small and imbalanced (even after data augmentation). On the figure, we see that the features on the test set are very dispersed in the 2-dimensional feature space, and the original visual clusters are almost lost. On IBOPF-LSA, although we see that points are overlapped, we can still identify some visual clusters. On IBOPF-UMAP we see that classes of super novae (SNIa, SNII, SNIax, etc) are overlapped, and classes of variable stars (RR-Lyra, EB, MIRA) are clearly separated. However, on the test set, UMAP shows some degree of dispersion on these classes as well.

5.5.2. Classification on full test set

To perform a full comparison with the state-of-the-art method, we evaluate IBOPF-LSA and AVOCADO on the full test set using LightGBM classifier. This evaluation also includes the case of combining AVOCADO with IBOPF-LSA for completitud. Table 5.3 shows the Balanced Accuracy, Presicion, Recall, Adjusted Mutual Information, Adjusted Rand and

Table 5.3: LightGBM classifier results on the full test-set, comparing the proposed method with the state-of-the-art method. The methods evaluated are AVOCADO as the state-of-the-art method, the proposed IBOPF-LSA method, and the combination of AVOCADO features with IBOPF-LSA features, named COMBINED. Flat-weighted metric is a classification metric using in the original AVOCADO's paper [16], which is computed from the multi-logloss metric on the training set.

LightGBM Classifier (Features with metadata)			
Metric	AVOCADO	IBOPF-LSA	COMBINED
Balanced accuracy	**0.82**	0.65	0.81
Precision weighted	0.85	0.73	0.84
Recall weighted	0.75	0.57	0.75
Adjusted Mutual Information score	0.55	0.38	0.55
Adjusted Rand score	0.57	0.35	0.57
Flat-weighted metric	0.595	0.976	0.598
AUC - 90: Type Ia SN	0.945	0.873	0.941
AUC - 67: Peculiar Type Ia SN - 91bg-like	0.958	0.864	0.956
AUC - 52: Peculiar Type Ia SN - SNIax	0.837	0.690	0.815
AUC - 42: Type II SN	0.923	0.780	0.922
AUC - 62: Type Ibc SN	0.915	0.795	0.910
AUC - 95: Superluminous SN	0.990	0.975	0.990
AUC - 15: Tidal disruption event	0.990	0.868	0.990
AUC - 64: Kilonova	0.997	0.984	0.997
AUC - 88: Active galactic nuclei	0.997	0.966	0.997
AUC - 92: RR Lyrae	**0.999**	**0.999**	**0.999**
AUC - 65: M-dwarf stellar flare	**0.999**	**0.999**	**0.999**
AUC - 16: Eclipsing binary stars	**0.999**	0.997	**0.999**
AUC - 53: Mira variables	**0.999**	**0.999**	**0.999**
AUC - 6: Microlens from single lens	**0.999**	0.995	**0.999**

Flat-weighted metric (from AVOCADO's paper) results on these classifications, including the AUC scores for each individual class from LightGBM classifier probabilities. The input used for these classifications consist of a combination of computed features and metadata features that potentially add relevant information to differentiate classes [16], and the LightGBM classifier will decide which features (or metadata features) are relevant for classification.

The overall results show that our proposed method does not improve the accuracy compared to AVOCADO, however, on some classes it shows very similar results. This can be identified on the AUC values, where for RR Lyrae, M-dwarf stellar flare and Mira variables it has the same AUC score up to the third decimal. Furthermore, for Eclipsing Binary stars and Microlens from single-lens, it shows almost similar AUC scores. The confusion matrix for our proposed method on the full test set is shown on Figure 5.9a where we see that the highest confusion classes are SNII, SNIax, SNIbc and SNIa-91gb, and the more accurate classified classes are Mira, RR Lyrae, Eclip. Binary and M-dwarf. The rest of the confusion matrices, including ROC-AUC curves plots, are included in Appendix **??**.

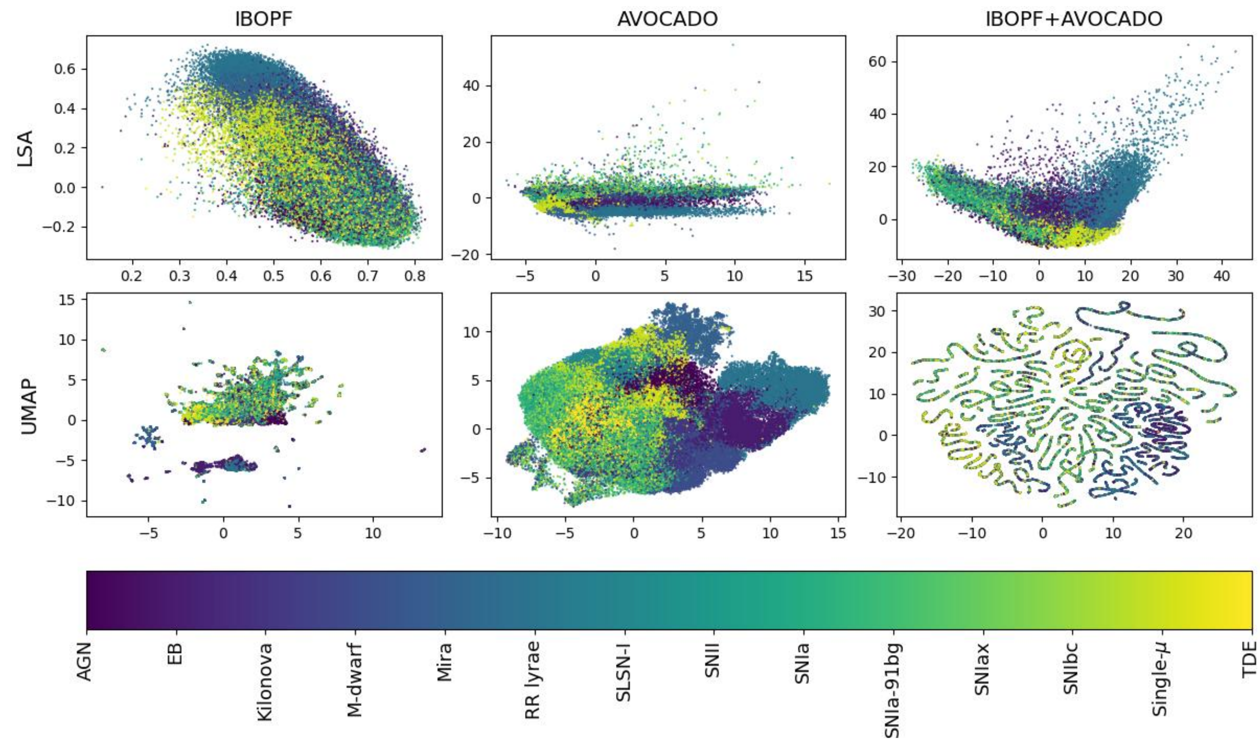
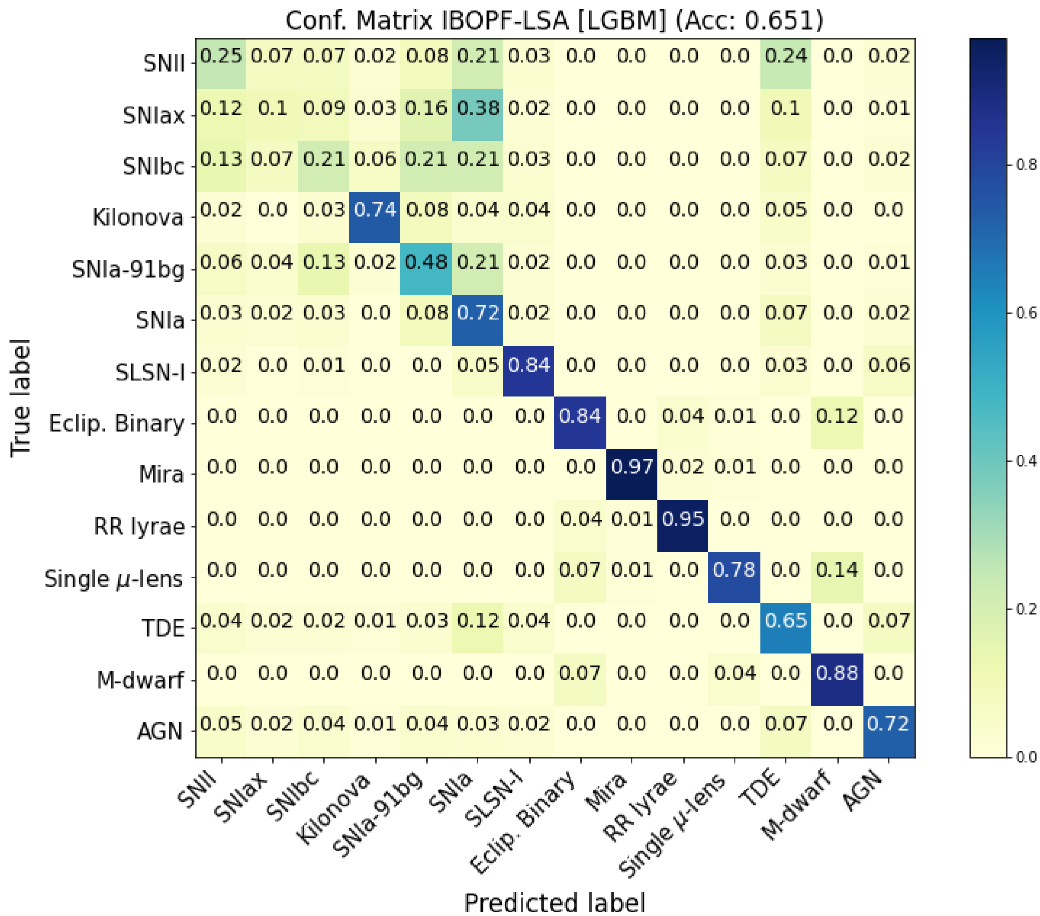


Figure 5.8: 2-dimensions visualization of AVOCADO, IBOPF-LSA and their combination, using LSA and UMAP as the dimension reduction technique. The visualization is produced from features extracted from the train set. Colors represents the classes, where it is expected that similar color groups closer and separated from the rest.

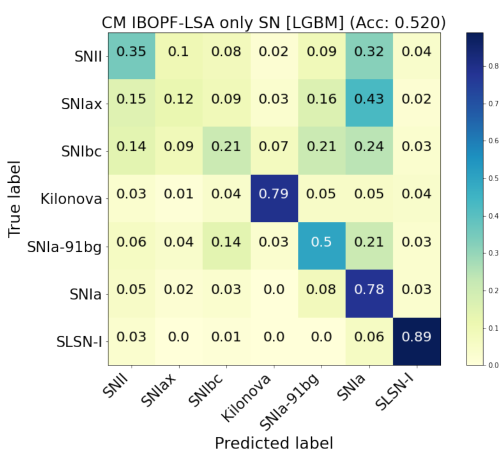
On the combination of both methods, the results shown on Table 5.3 indicates that our proposed method does not add any valuable information to AVOCADO features. Even more, it shows worse performance by some metrics, such as accuracy, precision and AUC score on certain classes. Figure 5.8 shows a 2-dimension visualization of the inferred spaced on the training set produced by AVOCADO, IBOPF and their combination, using LSA and UMAP as the dimension reduction technique. On the figure, it is easy to identify some clusters on AVOCADO using the UMAP method. On IBOPF we see that visual clusters are more separated and less dense compared to AVOCADO. However, both methods show clear confusion on objects of supernovae-type. On the combination of IBOPF-LSA and AVOCADO we see that on UMAP it shows a complex, spaghetti-like structure where it is very difficult to visually identify clusters. However, they seem to be very well-separated, except for supernovae-type classes.

5.5.3. Extra case study on classification

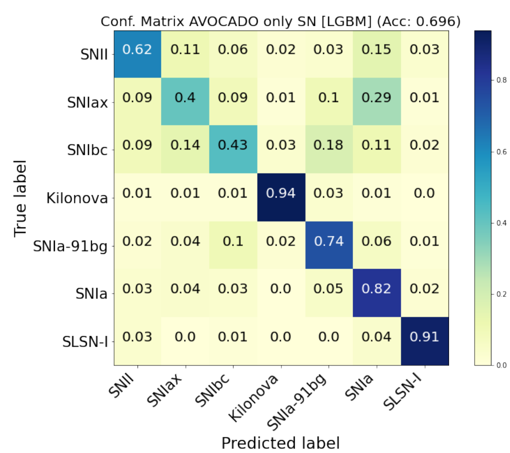
Results from the previous section have shown that our proposed method is not able to produce comparable accuracy to AVOCADO when classifying all of the 14 classes. However, since most of its confusion comes from supernovae-type classes, it is of interest to produce a split on datasets that separates supernovae-type classes from the dataset. To achieve this, we produce a subset from the original dataset (augmented train and test set) without supernovae-



(a) Confusion Matrix IBOPF, full-dataset.



(b) Confusion Matrix IBOPF, subset supernovae-type only.



(c) Confusion Matrix AVOCADO, subset Supernovae-type only.

Figure 5.9: Confusion Matrices of classification results using LightGBM Classifier on PLaSTiCC dataset. (a) The confusion matrix of our proposed method for the full test-set. (b) The confusion matrix of our proposed method for the subset with only Supernovae-type classes. (c) The confusion matrix of AVOCADO for the subset with only supernovae-type classes. The rest of confusion matrices, including the ROC-AUC curves, can be found on Appendix **??**

Table 5.4: Extra LightGBM classification experiments on subsets from the original PLaSTiCC dataset. The first subset consists of dropping the supernovae-type classes and the second subset consist of using only the supernovae-type classes. The same metrics are evaluated for both subsets. Flat-weighted metric is a classification metric used in the original AVOCADO's paper [16], which is computed from the multi-logloss metric on the training set.

Dataset without supernovae-type classes			
metric	AVOCADO	IBOPF-LSA	COMBINED
Balanced accuracy	**0.95**	0.87	**0.95**
Precision weighted	0.97	0.91	0.97
Recall weighted	0.96	0.88	0.96
Adjusted Mutual Information score	0.90	0.77	0.90
Adjusted Rand score	0.92	0.80	0.93
Flat-weighted metric	0.146	0.330	**0.134**
ROC-AUC weighted score	0.998	0.988	0.998
Dataset with only supernovae-type classes			
Balanced accuracy	**0.70**	0.52	0.69
Precision weighted	0.82	0.70	0.81
Recall weighted	0.72	0.59	**0.73**
Adjusted Mutual Information score	0.33	0.15	0.32
Adjusted Rand score	**0.48**	0.27	**0.48**
Flat-weighted metric	0.961	0.126	0.975
ROC-AUC weighted score	0.917	0.820	0.914

type classes, reducing from 14 to 7 classes, and a subset with only supernovae-type classes, with also 7 classes. The classification results on these subsets using the LightGBM classifier are shown on Table 5.4, where we have replaced the AUC score per class by an AUC macro-weighted.

The results show that even without supernovae-type classes, AVOCADO is still superior to our proposed method. Furthermore, we see that although the separation produces a slightly better classification accuracy per class for our proposed method, it is not enough to open the possibility of a hierarchical classification. We can see the Confusion on the subset with only Supernovae-type classes for our proposed method and state of the art method on Figures 5.9b and 5.9c, respectively, where it is clear that our proposed method has more confusion than avocado, specially on classes SNIax and SNIbc. When comparing these confusion matrices with the results on full dataset (Figure 5.9a) we see that the accuracy improvement most likely comes from the reduction of confusion between Tidal Disruption Event (TDE) or Active Galactic Nuclei (AGN) with Supernovae-type classes.

Table 5.5: Similarity search results using K-Nearest Neighbor approach with Euclidean distance. The input used for both methods is computed features plus metadata, as specified by AVOCADO [16]. The metric used to evaluate similarity search is the mean Average Precision at k (mAP@k) for $k = \{1, 5, 10, 20\}$. The table includes a macro average of mAP@k and separated mAP@k for each class on the dataset.

Class	IBOPF-LSA				AVOCADO			
	k=1	k=5	k=10	k=20	k=1	k=5	k=10	k=20
mAP - macro average	0.31	0.34	0.34	0.32	0.67	**0.69**	0.67	0.64
mAP - 90: Type Ia SN	0.24	0.30	0.30	0.27	0.52	0.54	0.51	0.45
mAP - 67: Type Ia SN - 91bg-like	0.18	0.24	0.25	0.23	0.51	0.54	0.51	0.46
mAP - 52: Type Ia SN - SNIax	0.15	0.21	0.22	0.21	0.31	0.37	0.35	0.32
mAP - 42: Type II SN	0.15	0.21	0.22	0.21	0.38	0.43	0.41	0.36
mAP - 62: Type Ibc SN	0.14	0.19	0.20	0.19	0.38	0.43	0.41	0.37
mAP - 95: Superluminous SN	0.18	0.21	0.20	0.19	0.81	0.81	0.80	0.77
mAP - 15: Tidal disruption event	0.13	0.16	0.17	0.16	0.79	0.78	0.76	0.73
mAP - 64: Kilonova	0.11	0.17	0.18	0.17	0.64	0.64	0.63	0.60
mAP - 88: Active galactic nuclei	0.23	0.26	0.25	0.23	0.77	0.77	0.75	0.73
mAP - 92: RR Lyrae	**0.97**	**0.97**	**0.97**	**0.96**	0.95	0.95	0.94	0.94
mAP - 65: M-dwarf stellar flare	0.46	0.50	0.47	0.41	0.85	0.84	0.82	0.80
mAP - 16: Eclipsing binary stars	0.47	0.49	0.46	0.41	0.94	0.94	0.93	0.92
mAP - 53: Mira variables	0.56	0.57	0.54	0.50	0.94	0.94	0.94	0.93
mAP - 6: Microlens from single lens	0.32	0.33	0.32	0.30	0.66	0.67	0.65	0.63

5.6. Similarity search experiments

Similarity search experiments are performed based on a distance metric and K-NN algorithm. The evaluations are made on the full test set using Euclidean distance as the similarity measure for finding the neighbors. The methods compared are IBOPF-LSA, the best configuration for our proposed method, and AVOCADO, the state-of-the-art method. The metric evaluated is mean Average Precision at different k values (mAP@k). Table 5.5 shows the mAP@k for $k = \{1, 5, 10, 20\}$ describing the metric for each class by separate and for the overall dataset under a macro average scheme.

In general, it is shown that AVOCADO has significant better performances on similarity search under the mAP metric. For our proposed method, the only class that was correctly searched was RR Lyrae with 97 % mAP, 2 % more than AVOCADO. This class shows to be almost perfectly searched on the first neighbor (mAP@1) and does not improve performance for higher k values. For AVOCADO the highest mAP was at k=5, reducing its performance afterwards.

5.7. Computing time experiments

Although having a method with high performance on classification is a really important part of this work, it is not the only relevant one. We also want to produce a representation method that is fast to compute compared to the literature method. In particular, our proposed

Table 5.6: Computing times per representation method. For AVOCADO only 10 % of the whole test set was measured for computing times, which means that the average value could fluctuate. third column indicates the average time that takes to transform 1 time series and fourth column indicates the estimated time that could take to transform the whole test set, with 3.479.801 Time Series.

Method	Threads	Time Series time (ms)	Dataset time (hours)
AVOCADO	6	256 ± 66	247.5 ± 63.8
Extended I-BOPF	1	24 ± 5	3.9 ± 0.8

method has a theoretical asymptotic upper bound of $O(n)$ since all the computing algorithms that involve the method are of linear nature. However, the method can have a significant bias due to the complex nature of the algorithm or un-optimized code, which can impact negatively on the computing times.

To evaluate the effective computing times, we measure the time it takes to transform each Time Series in the test set using AVOCADO and our proposed method, and estimates the total time that would take to transform the whole dataset. This is summarized on Table 5.6. Both experiments were performed on the same machine with an i5-9400f CPU (6 cores/6 threads), where AVOCADO transform 1 Time Series in 256 milliseconds using 6 parallel process and our proposed method IBOPF-LSA transform 1 Time Series in 24 milliseconds using 1 process, which means it can transform 6 Time Series in 24 milliseconds, with each Time Series running in a different process. This leads to our proposed method being 2 orders of magnitude faster than AVOCADO.

Following the results, we get that to transform the 3,479.801 Time Series included in the test set (excluding classes not present in the train set), AVOCADO takes around 250 hours (approximately 10 days), and our proposed method takes around 4 hours. However, due to the sequential read bias and pre-processing required for our proposed method, the effective computing time results in around 7 hours.

Chapter 6

Conclusion

In this chapter, we present our conclusions on the experimental evaluations presented in Chapter 5 and the proposed method presented in Chapter 4. In Section 6.1 we summarize the experimental results. Then, a complete discussion of the current work is performed (Section 6.2), including an analysis of the proposed method and the experimental results. Finally, we describe some possible future research directions (Section 6.3) related to our proposed method and the application of Information Retrieval theory on Time Series.

6.1. Contributions

Our contributions focus on a new proposed method for which different experimental evaluations were performed using the PLaSTiCC dataset. This dataset contains approximately 3.5 million Time Series on the test set and 7846 Time Series on the train set (augmented to 80856 Time Series). The first part of the evaluations focuses on validating the proposed method, in particular, the proposed generalizations of Multi-Quantity and Multi-Resolution. For Multi-Quantity, 41 different combinations of statistical quantities where evaluated, from which the selected statistical quantities were a late-fusion combination of the early-fusion pairs (Min-Max, Trend), (Min-Max, Min), (Min-Max, Max), (Trend, Min-Max). These combinations show to be better to represent the dataset than using only one statistical quantity, which validates the inclusion of this generalization in the algorithm.

For Multi-Resolution, we evaluate through an incremental grid-search if adding more resolution levels does effectively improve the performance. The experiments were performed for different resolutions levels, defined within specific ranges of window width and word length, and iterating using an incremental grid-search technique. The algorithm was designed to work until adding more resolution levels improved the accuracy by less than 1 %. The results show that 2-levels of resolutions was optimal, using the pairs (110 days window, 1-length word) and (406 days window , 1-length word), which give a relatively small improvement in the accuracy compared to using only one resolution level.

Having evaluated and selected the parameters for the Multi-Quantity and Multi-Resolution generalizations, we proceeded to perform the experiments on classification and computing time. For these experiments, we extracted the maximum number of features using our method such that the method can still be recognized as a compact representation. This means that, for the PLaSTiCC dataset, we extracted 361 compact features from the 816 sparse features produced by the algorithm, reducing the features dimensionality by approximately 44 %. For the literature method, we used AVOCADO, which is one of the best methods for classifying PLaSTiCC dataset. This method applies Gaussian Process to re-sample the irregular Multivariate Time Series to a regular-space and then computes 41 relevant features.

On classification tasks, we first evaluate which compact technique gives the better classification performance and how it changes depending on the number of components (i.e., dimensionality) used. Once this configuration is selected, a full classification comparison is performed between our proposed method and AVOCADO method, using the LightGBM classifier. The results show higher classification on AVOCADO when compared to our proposed method, with higher confusion on Supernovae-type Time Series on our method. Further evaluations separating the Supernovae-type classes from the dataset shows that our proposed method on the selected configuration effectively fails to separate different supernovae-type classes, since they have very similar patterns.

On the similarity search experiments, AVOCADO is again the highest performance methods, showing mAP scores of 69 %. On the other hand, our proposed method fails to retrieve good similarity matches for almost every class, expect for RR Lyrae, where it shows almost 100 % performance. These results show that our proposed method, on the selected configuration, is not appropriate for performing similarity search queries in the full PLaSTiCC dataset.

For the computing times evaluations, using the same machine with an i5-9400f CPU (6 cores/ 6 threads), AVOCADO took around 250 milliseconds to extract the features from a Time Series using the 6 cores through multi-processing. On the other hand, our proposed method took only 25 milliseconds in average to extract the features from a Time Series using only 1 core, which means 6 Time Series are transformed to their respective feature vectors in 25 milliseconds using 6 parallel processes, running each Time Series on a different process.

6.2. Discussion

We have proposed, and evaluated, a new representation method for irregular Multivariate Time Series using Information Retrieval theory. The proposed method includes extensions on the use of multiple statistical quantities (Multi-Quantity) and multiple levels of resolution (Multi-Resolution). The evaluations were performed on classification performance and computing time.

6.2.1. Proposed approach

This thesis presents basic foundations for the inclusion of Information Retrieval theory on irregular Time Series, putting together all the involved concepts and formulas to produce a compact feature vector using a modified Bag-of-Pattern approach. The modifications on the approach are made in order to support irregular sampling, solving the challenges of empty segments during the discretization and variations in sub-sequence lengths. The first is handled by interpolation or adding a special character to represent this absence of information, the second is handled by proposing a two-ways sliding window with a fixed window width but variable window steps based on data time instants.

Our proposed method consists in this adapted Bag-of-Pattern approach, generalized for irregular Multivariate Time Series, where the Bag-of-Pattern method is applied to each variable in the Time Series by separating and then merging them together using Latent Semantic Analysis (LSA) or Multivariate Analysis of Variance (MANOVA) to address the possible correlations between representations. Nonetheless, this proposed method suffers from loss of information on certain correlation between variables, for example, large-scale correlation or correlation between variables at different time intervals are lost when we apply our proposed method. This is because MANOVA and LSA only address correlation between patterns, drooping the temporal order of sequences, and thus any large-scale correlation between patterns is most likely lost. LSA could capture some of these large-scale correlations if they are found to be happening on similar sequences of patterns. This loss in correlation information has most likely a negative impact in the final representation method, being a possible explanation to the low classification metrics obtained on our proposed method when compared to state-of-the-art method (Figure 5.3). However, to prove this negative impact on classification, we need to validate on a dataset with well-known correlation, which is an experiment we could not include in our current work and thus is proposed as future work.

Together with the base method for irregular Multivariate Time Series, we have proposed two extensions designed to improve the performance of the representation method. The first extension was designed to include multiple statistical quantities, which allow to extract more detailed information from the Time Series. The second extension focuses on the use of multiples levels of resolution, which aims to combine different resolution levels to represent different kinds of features for the Time Series. For example, for short-time patterns, a small window width is better, but for large-time patterns, a large window width is better, which means that if we combine a small window width with a large window width, it will improve the performance.

6.2.2. Representation method performance

On the experiments, we have shown the effectiveness of our method applied to PLaSTiCC dataset, validating the performance of our proposed representation method for irregular Multivariate Time Series, including the extensions with Multi-Quantity and Multi-Resolution. In particular, we have shown that it is better to use a complex combination of statistical quantities and simple discretization rather than complex discretization with a simple combination of statistical quantities. This directly means that Multi-Quantity has more impact on the representation method than Multi-Resolution.

This is an interesting result considering that we are only using linear time complexity quantities, which rise the question of: What could have been the performance of our proposed method if we include more complex statistical quantities?. Possibilities on this topic are discussed in Section 6.3 as Future Work. Another advantage of using Multi-Quantity with a simple discretization of words with no more than 3 symbols (segments) is that allows us to bypass the problem of empty segments. This happens because for simple discretization levels and a large enough window width, the cases where a word has a missing symbol (empty segment) are so rare that they can be ignored without losing relevant information.

Finally, since our evaluation relies more on the use of Multi-Quantity rather than Multi-Resolution, the experiments show that the latter does not improve the performance of the method by a significant margin. However, using simple 2-levels of resolution in combination with a complex Multi-Quantity is enough to get the optimal configuration of parameters for PLaSTiCC dataset, which shows that both extensions together are useful to improve our proposed method.

We have also shown that the applied technique for producing a compact representation is capable of conserving enough relevant information even when reducing its dimensionality. This is seen in Figure 5.6 where IBOPF-LSA shows no major change in classification accuracy for 100 dimensions or more.

6.2.3. Classification comparissons

For classification evaluations, we first perform a comparison on all the candidate configurations on our proposed method in terms of compact techniques. This experiment, performed on 10 % of the test set using LightGBM classifier, evaluates many different possibilities of application of our method. These possibilities are combinations between the features produced by our proposed method, the features produced by AVOCADO, and the compact methods LSA, MANOVA and UMAP, including the case of sparse representation (i.e., without using any compact method). These evaluations are shown in Figure 5.6, where the selected configuration is the combination of our proposed method with Latent Semantic Analysis (LSA), a linear method that reduces dimensions using the Singular Value Decomposition method, which preserves global structure. To further understand how the preservation of global structures impact on these results, we have included IBOPF-UMAP, which uses Uniform Manifold Approximation and Projection (UMAP), a non-linear method that reduces dimensions using the Maniforld approach, which preserves local structures. By comparing IBOPF-LSA with IBOPF-UMAP, we can see that our proposed method produces features more suited for the global structure approach, producing bad results when using methods like UMAP for dimension reduction.

To analyze this, we consider the case of Supernovae-type classes where their Time Series are very hard to differentiate due to very similar and complex taxonomy. Our proposed method extracts general features that do not differentiate very well between these slightly different Supernovae-type classes. Although we do not have a way to prove that these differences between Supernovae-type classes are of local-structure nature, this intuition leads us to a most likely bad optimization on the Multi-resolution extension due to the use of very large time windows with a compact method that preserves global structures (IBOPf-LSA). In fact, almost every Supernovae-type event have a very short time duration on its light curve peak (no more than a few weeks), and thus, using time windows of 100 and 400 days is too large for extracting pattern features that can tell the slight differences between these Supernovae-type classes. This could be easily validated and fixed if we evaluate Multi-resolution extension using a different compact technique and on shorter time windows, comparing its resulting accuracy. However, due to time limit, we propose this extra validation as future work only (See Section 6.3).

On the case of Supervised UMAP (sup-UMAP), Figure 5.6 and 5.7 show that this supervised approach, which uses the train labels to learn a better dimension reduction, has a very clear over-fit on the model inferred space, differentiating extremely well on train set classes but failing drastically on test set classes. This is a clear illustration of how PLaSTiCC dataset has a non-representative train set, which renders extremely difficult the challenge of using a method that can learn to differentiate Time Series on test set using the small, imbalanced and non-representative train set extremely difficult. We know that these experimental evaluations on UMAP are very superficial, and this method has much more potential to explore, but we have limited our experimentation to its more common and simpler algorithms.

Once we have selected the compact technique, we performed evaluations and comparisons on classification between our proposed method and AVOCADO, one of the best known methods for representing and classifying PLaSTiCC dataset.

For the Light Gradient Boosting Machine (LightGBM) Classifier, AVOCADO shows excellent classification results, with an accuracy of 82 %, very similar results to the ones exposed in the literature [16]. On the other hand, our proposed method shows relatively good results, with an accuracy of 65 %. On these classification experiments, a combination of computed features and metadata was used, just like AVOCADO proposed, indicating that our proposed method shows high confusion when classifying Supernova-type Time Series, not being able to efficiently differentiate between all different sub-types like type II Supernova, type Ia Supernova, Kilonova, etc. Table 5.3 shows the summary of classification experiments with different metrics. Here we see that AVOCADO has higher metrics values such as precision and recall and accuracy. Additionally, it has a lower flat-weighted metric, which is a metric that combines multi-logloss score from cross-validation training on a train set, this means that the features from AVOCADO are more easy to learn for the LightGBM Classifier. On the AUC scores we see that for Supernovae-type classes it has lower scores, which directly indicates higher confusion (more False-Positive or False-negatives). The confusion matrices exposed in Figure 5.9 directly show this confusion when compared to AVOCADO, Here we see for example that Superonvae type-II, type-Iax and Type-Ibc are confused with Type-Ia. As we discussed before, this is mostly because Supernovae-type Time Series are very similar, with similar patterns hard to differentiate when using discretization, even more when the Multi-resolution extension is using large time windows. We could try to find a custom and complex statistical quantity that extracts more relevant information for Supernovae-type classification, which is proposed as future work (See Section 6.3).

From the two clustering metrics; Adjusted Mutual Information score and Adjusted Rand score, we see that if we consider our classification as a supervised clustering, then neither method performs well on these metrics. For an optimal value of 1, AVOCADO achieved the highest scores of 0.55 and 0.57, respectively. Since we do not employ a clustering method, we cannot extract much information from these metrics. However, these results suggest that neither IBOPF-LSA nor AVOCADO could work directly on clustering tasks and would require further adaptations.

A promising result obtained from classification on LighGBM is that our proposed method shows relatively good accuracy on classifying periodic Time Series while it remains with a theoretical linear time complexity. This is more clearer on the extra experiment performed on a subset of the dataset without Supernovae-type classes (see Table 5.4) where our proposed method reaches 87 % accuracy and AVOCADO reaches 95 % accuracy, only 8 % more accuracy, which is a much better result compared to the difference of 17 % on the full dataset, even more if we consider the computing times (discussed on Section 6.2.5). On the Supernovae-classes subset no major difference are seen since our proposed method remains with the same confusion, improving very little the accuracy thanks to the absence of Tidal Disruption Events (TDE) and Active Galacitc Nuclei (AGN) classes, which are highly confused with Supernovae-type classes (See Figure 5.9.(a)).

6.2.4. Similarity search

On the Similarity search, the experimental evaluation reflects a wrong optimization of our method. Since during the parameter exploration we used classification accuracy as a score for deciding on the parameter values, the used configuration is optimal only for classification and not for similarity search. This is an interesting phenomena, since the classifier used on the parameter exploration is a similarity-based classifier. We would have expected to produce much better similarity search results. One possible explanation is that the parameter exploration was performed only on the train set, which even after data augmentation is not representative enough of the test set. This means that simple similarity-based algorithms will perform very differently on a test set and a train set. In order to validate these results we would need to re-calculate the parameter configuration using a similarity search score instead of a classification accuracy score on the k-fold cross-validation, or use part of the test-set during the k-fold cross validation. Due to limited time and since k-fold cross validation is a very slow process, we could not perform this extra validation on the current work and it is proposed as a future work.

In particular, the results shown on Table 5.5 indicates that AVOCADO's features are robust enough to produce relatively good similarity search results, obtaining an mAP@10 of 0.69, a much higher score compared to the mAP@10 of 0.34 obtained by our proposed method. There is one particular class, the RR Lyrae, where our proposed method shows superior query search results with an mAP@10 of 0.97, compared to mAP@10 of 0.95 obtained by AVOCADO on this specific class. The fact that it gives so good results on only 1 class and extremely bad results on the other classes is another indication that our proposed method is most likely bad optimized to solve this certain data mining task and requires further exploration.

6.2.5. Computational complexity

Although our proposed method has shown no improvement in the overall classification accuracy compared to the literature method AVOCADO, which results in a rejection of one of our hypotheses, this is not the only hypothesis we are evaluating. We also propose our method to be faster than literature method. Although the variation in the number of samples per Time Series is not big enough to prove the asymptotic upper bound $O(n)$ of our proposed method, we can check if our proposed method is effectively faster than literature method, which uses Gaussian Process, a very expensive algorithm with a time complexity between $O(n^2)$ and $O(n^3)$ depending on the implementation. Experimental results show that our proposed method is 2 orders of magnitude faster than literature method to generate the features on PLaSTiCC test set, taking in average 24 milliseconds to process 6 Time Series in parallel when AVOCADO takes 256 milliseconds to process only one Time Series. These results directly validate our second hypothesis and open a different spectrum of possible applications to our proposed method.

For example, following the results from classification on a subset without Supernovae-type classes, we have a feature method 2 orders of magnitude faster than the state-of-the-art method that only losses 7 % accuracy when classifying the same dataset. If we include Supernovae-type classes, the difference in accuracy increases from 7 % to 17 % in the case of differentiating Supernovae-type astronomical events by taxonomy, with a value between 7 % and 17 % if we simplify this taxonomy, merging some similar Supernovae-type light curves in one class.

A final remark on the computing time experiments is that, even when the projected time required to process the 3.5 millions Time Series in the test set using our proposed method is around 4 hours, transforming 6 Time Series each 24 milliseconds, in practice the whole run will take more time due to a bias introduced by the sequential reads performed to the HDD and the pre-processing applied to the time Series data, required to generate the input for our proposed method. This bias also affects AVOCADO but at a minor scale since the method takes much longer to transform each Time Series.

6.3. Future work

Applying Information Retrieval theory to Time Series and adapting it to work on irregular Multivariate Time Series involves a wide variety of issues yet to explore. Although this thesis contributes with a new representation method for irregular Multivariate Time Series, there are a few possible lines of future research.

On the proposed method, the implemented code is not entirely optimized, having for-loops that can be reduced and code that can be transferred to C++ or Cython for faster computing times. On the method itself, we identify six possible research directions, described from less relevant to more relevant. The first is to directly explore the combination of our computed features with other method's features or additional metadata. We already shown that a direct combination of our computed features with AVOCADO's features does not really improve the performance compared to AVOCADO's results. However, we could evaluate the combination with other literature methods or the inclusion of more metadata to the classifier, trying to improve its performance.

The second research direction is to apply more complex statistical quantities to the Multi-Quantity extensions and not limit the algorithm to linear time statistical quantities, using the Special Character technique for those cases where the statistical quantity fails to be computed. Here the idea is to explore the trade-off between the time complexity of the algorithm and the classification performance, looking for an optimal depending on the application.

The third research direction is to directly explore the combination of our computed features with other method's features. For example, we could combine our proposed method features with AVOCADO features, where AVOCADO will dominate the computing times. However, depending on the relation of the features we would need to adapt the classifiers. For example, if we combine directly our 361 features with the 27 features of AVOCADO the LightGBM will be highly confused since the random sub-set of features used during the training phase will hardly be representative from both set of features. In order to work, we will need to adapt the classifier, use a different classifier or reduce our computed features to a similar scale compared to AVOCADO's features.

A fourth research direction is to handle the correlation between variables through synchronous Sliding Window. In our work, we already discarded the use of synchronous Sliding Window on the multivariate measures since they would produce a large number of empty-segments on complex discretization schemes for windows matching data on a variable but not on the others. However, during the experiments, we discovered that the representation method works better for complex Multi-Quantity schemes and simple discretization schemes, for which empty-segments are almost never present. This results directly opens the possibility of including the correlation between variables on our proposed method, which is our second proposed research direction. In particular, the correlation can be handled by using a simple discretization scheme, in combination with a complex Multi-Quantity scheme and the extraction of simultaneous sub-sequences through a synchronous Sliding Window.

The fifth proposed research direction is to apply our proposed method on outlier detection or event detection on streaming irregular Multivariate Time Series. Both applications do not require a complex classification scheme since they work on a more simpler problem. This is an advantage to our method, which works well on classification but fails when it tries to differentiate too similar classes. Furthermore, outlier detection and event detection on streaming irregular Multivariate Time Series require a fast computing algorithm or online algorithm that can work in real time. Since our proposed method has proven to be a linear-time complexity with fast computing times, it is a promising alternative to work on these problems. Furthermore, our method can directly work as an online method since for a new measure on the streaming irregular Multivariate Time Series, a new sub-sequence is extracted, for which a feature value is modified (add 1 on the BoW vector). For the compact representation, MANOVA can be used directly without any additional cost, while LSA would require to re-compute the feature vector.

The seventh and final identified research direction involves evaluating different parameter configuration through modifying the k-fold cross-validation steps. For example, we already discuss that our representation method is focusing on global structure, which could be related to large window widths. This can be further explored trying to use shorter window widths with a different compact technique that focuses more on local structures. Additionally, we can explore a customized parameter configuration for the specific data mining task of similarity search, using a different metric and repeating all of the k-fold cross-validation steps.

An extra and very interesting research direction based on recent work would be to use

Self-Attention mechanism [113]. In this approach, an encoder-decoder structure is used to learn latent space from textual data. Complex Attention-based mechanism, such as Multi-Head dot-Attention has been recently explored and applied to the astronomical Multi-Band Time Series [92] using the Transformer model [3]. A direct extension of our proposed work, which is related to this self-Attention mechanism, is to use our textual transformation on the self-Attention mechanism in replacement to the compact technique and the normalizing weighted scheme. With this, we would use this encoder-decoder structure to learn the latent space representation and also classify the features. Furthermore, this can be combined with improvements on the symbolic representation, exploring the use of codebooks instead of discretization or tokenization instead of Bag-of-Words. For codebooks, some clustering algorithms such as Variability trees [111] have already explored this approach for irregular Univariate Time Series. They could be extended to multivariate irregular Time Series as long as we propose or apply a similarity metric that can handle this specific type of Time Series. On tokenization, many different alternatives already exist in the Natural Language Processing (NLP) field such as N-grams [19], WordPiece Tokenization [104, 105] or Subword Tokenization [29, 60]. The key factor to this approach is to design an architecture based on self-attention mechanism that can effectively learn a latent space for astronomical Time Series.

For the experiments, the evaluation with only 1 dataset is not wide enough to prove that our proposed method could work on different application domains. For that, we require to evaluate the classification performance of our algorithm on different datasets collected from different fields. This will also allow to validate our proposed method to work not only on irregular Multivariate Time Series but also on any kind of Time Series, regular or irregular sampled, with one or many variables.

Bibliography

[1] Aghabozorgi, S., Shirkhorshidi, A.S., Wah, T.Y., 2015. Time-series clustering–a decade review. Information Systems 53, 16–38.

[2] Agrawal, R., Faloutsos, C., Swami, A., 1993. Efficient similarity search in sequence databases, in: International conference on foundations of data organization and algorithms, Springer. pp. 69–84.

[3] Allam Jr, T., McEwen, J.D., 2021. Paying attention to astronomical transients: Photometric classification with the time-series transformer. arXiv preprint arXiv:2105.06178 .

[4] André-Jönsson, H., 2002. Indexing Strategies for Time Series Data. Department of Computer and Information Science, Linköpings universitet.

[5] Andreatta, M., Corria-Osorio, J., Müller, S., Cubas, R., Coukos, G., Carmona, S.J., 2021. Interpretation of t cell states from single-cell transcriptomics data using reference atlases. Nature communications 12, 1–19.

[6] Assent, I., Krieger, R., Afschari, F., Seidl, T., 2008. The ts-tree: efficient time series search and retrieval, in: Proceedings of the 11th international conference on Extending database technology: Advances in database technology, pp. 252–263.

[7] Aßfalg, J., Kriegel, H.P., Kröger, P., Kunath, P., Pryakhin, A., Renz, M., 2008. Similarity search in multimedia time series data using amplitude-level features, in: International Conference on Multimedia Modeling, Springer. pp. 123–133.

[8] Badhiye, S.S., Chatur, P.N., 2018. A review on time series dimensionality reduction. HELIX 8, 3957–3960.

[9] Bailly, A., Malinowski, S., Tavenard, R., Chapel, L., Guyet, T., 2015. Dense bag-of-temporal-sift-words for time series classification, in: International Workshop on Advanced Analysis and Learning on Temporal Data, Springer. pp. 17–30.

[10] Ball, N.M., Brunner, R.J., 2010. Data mining and machine learning in astronomy. International Journal of Modern Physics D 19, 1049–1106.

[11] Bankó, Z., Abonyi, J., 2007. Correlation based dynamic time warping, in: 8th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, CINTI 2007, pp. 295–306.

[12] Bar-Joseph, Z., Gerber, G., Gifford, D.K., Jaakkola, T.S., Simon, I., 2002. A new approach to analyzing gene expression time series data, in: Proceedings of the sixth annual international conference on Computational biology, pp. 39–48.

[13] Baydogan, M.G., Runger, G., Tuv, E., 2013. A bag-of-features framework to classify time series. IEEE transactions on pattern analysis and machine intelligence 35, 2796–2802.

[14] Bhaduri, K., Zhu, Q., Oza, N.C., Srivastava, A.N., 2010. Fast and flexible multivariate time series subsequence search, in: 2010 IEEE International Conference on Data Mining, IEEE. pp. 48–57.

[15] Bianchi, F.M., Mikalsen, K.Ø., Jenssen, R., 2017. Learning compressed representations of blood samples time series with missing data. arXiv preprint arXiv:1710.07547 .

[16] Boone, K., 2019. Avocado: Photometric classification of astronomical transients with gaussian process augmentation. The Astronomical Journal 158, 257.

[17] Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, pp. 144–152.

[18] Cai, Q., Chen, L., Sun, J., 2015. Piecewise statistic approximation based similarity measure for time series. Knowledge-Based Systems 85, 181–195.

[19] Cavnar, W.B., Trenkle, J.M., et al., 1994. N-gram-based text categorization, in: Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval, Citeseer.

[20] Chan, K.P., Fu, A.W.C., 1999. Efficient time series matching by wavelets, in: Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337), IEEE. pp. 126–133.

[21] Charnock, T., Moss, A., 2017. Deep recurrent neural networks for supernovae classification. The Astrophysical Journal Letters 837, L28.

[22] Chen, L., Ng, R., 2004. On the marriage of lp-norms and edit distance, in: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30, pp. 792–803.

[23] Chen, L., Özsu, M.T., Oria, V., 2005. Robust and fast similarity search for moving object trajectories, in: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 491–502.

[24] Chouakria-Douzal, A., Nagabhushan, P.N., 2006. Improved fréchet distance for time series, in: Data Science and Classification. Springer, pp. 13–20.

[25] Ciaccia, P., Patella, M., Zezula, P., 1997a. M-tree: An effcient access method for similarity search in metric spaces, in: Proceedings of the 23rd VLDB conference, Athens, Greece, Citeseer. pp. 426–435.

[26] Ciaccia, P., Patella, M., Zezula, P., 1997b. M-tree: An efficient access method for similarity search in metric systems, in: Proc. of the 23rd Intl. Conf. on Very Large Databases (VLDB).

[27] Dan, J., Shi, W., Dong, F., Hirota, K., 2013. Piecewise trend approximation: a ratio-based time series representation, in: Abstract and Applied Analysis, Hindawi.

[28] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R., 1990.

Indexing by latent semantic analysis. Journal of the American society for information science 41, 391–407.

[29] Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 .

[30] Dhillon, I., Kogan, J., Nicholas, C., 2004. Feature selection and document clustering, in: Survey of text mining. Springer, pp. 73–100.

[31] Donoso-Oliva, C., Cabrera-Vives, G., Protopapas, P., Carrasco-Davis, R., Estevez, P., 2021. The effect of phased recurrent units in the classification of multiple catalogues of astronomical light curves. Monthly Notices of the Royal Astronomical Society 505, 6069–6084.

[32] Driemel, A., Krivošija, A., Sohler, C., 2016. Clustering time series under the fréchet distance, in: Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms, SIAM. pp. 766–785.

[33] Duque, A.F., Morin, S., Wolf, G., Moon, K., 2020. Extendable and invertible manifold learning with geometry regularized autoencoders, in: 2020 IEEE International Conference on Big Data (Big Data), IEEE. pp. 5027–5036.

[34] Eiter, T., Mannila, H., 1994. Computing discrete Fréchet distance. Technical Report. Citeseer.

[35] El Ouardighi, A., El Akadi, A., Aboutajdine, D., 2007. Feature selection on supervised classification using wilks lambda statistic, in: 2007 International Symposium on Computational Intelligence and Intelligent Informatics, IEEE. pp. 51–55.

[36] Esmael, B., Arnaout, A., Fruhwirth, R.K., Thonhauser, G., 2012. Multivariate time series classification by combining trend-based and value-based approximations, in: International Conference on Computational Science and Its Applications, Springer. pp. 392–403.

[37] Evangelopoulos, N., Zhang, X., Prybutok, V.R., 2012. Latent semantic analysis: five methodological recommendations. European Journal of Information Systems 21, 70–86.

[38] Faloutsos, C., 1996. Searching multimedia databases by content (vol. 3).

[39] Förster, F., Cabrera-Vives, G., Castillo-Navarrete, E., Estévez, P., Sánchez-Sáez, P., Arredondo, J., Bauer, F., Carrasco-Davis, R., Catelan, M., Elorrieta, F., et al., 2021. The automatic learning for the rapid classification of events (alerce) alert broker. The Astronomical Journal 161, 242.

[40] Frey, B.J., Dueck, D., 2007. Clustering by passing messages between data points. science 315, 972–976.

[41] Fu, T.c., 2011. A review on time series data mining. Engineering Applications of Artificial Intelligence 24, 164–181.

[42] Gabruseva, T., Zlobin, S., Wang, P., 2020. Photometric light curves classification with machine learning. Journal of Astronomical Instrumentation 9, 2050005.

[43] Gomez, S., Berger, E., Blanchard, P.K., Hosseinzadeh, G., Nicholl, M., Villar, V.A.,

Yin, Y., 2020. Fleet: A redshift-agnostic machine learning pipeline to rapidly identify hydrogen-poor superluminous supernovae. The Astrophysical Journal 904, 74.

[44] Hetland, M.L., 2004. A survey of recent methods for efficient retrieval of similar time sequences, in: Data mining in time series databases. World Scientific, pp. 23–42.

[45] Hu, Z., Tak, H., 2020. Modeling stochastic variability in multi-band time series data. arXiv preprint arXiv:2005.08049 .

[46] Hung, N.Q.V., Anh, D.T., 2008. An improvement of paa for dimensionality reduction in large time series databases, in: Pacific Rim International Conference on Artificial Intelligence, Springer. pp. 698–707.

[47] Itakura, F., 1975. Minimum prediction residual principle applied to speech recognition. IEEE Transactions on acoustics, speech, and signal processing 23, 67–72.

[48] Ivezić, Ž., Kahn, S.M., Tyson, J.A., Abel, B., Acosta, E., Allsman, R., Alonso, D., AlSayyad, Y., Anderson, S.F., Andrew, J., et al., 2019. Lsst: from science drivers to reference design and anticipated data products. The Astrophysical Journal 873, 111.

[49] Jamal, S., Bloom, J.S., 2020. On neural architectures for astronomical time-series classification. arXiv preprint arXiv:2003.08618 .

[50] Johnston, K.B., Peter, A.M., 2017. Variable star signature classification using slotted symbolic markov modeling. New Astronomy 50, 1–11.

[51] Kamath, U., Liu, J., Whitaker, J., 2019. Text and speech basics, in: Deep Learning for NLP and Speech Recognition. Springer, pp. 87–138.

[52] Kane, A., Shiri, N., 2017. Multivariate time series representation and similarity search using pca, in: Industrial Conference on Data Mining, Springer. pp. 122–136.

[53] Kent, J., Bibby, J., Mardia, K., 1979. Multivariate analysis. Academic press Amsterdam.

[54] Keogh, E., 2002. Exact indexing of dynamic time warping, in: VLDB'02: Proceedings of the 28th International Conference on Very Large Databases, Elsevier. pp. 406–417.

[55] Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S., 2001. Dimensionality reduction for fast similarity search in large time series databases. Knowledge and information Systems 3, 263–286.

[56] Keogh, E., Lin, J., Fu, A., 2005. Hot sax: Efficiently finding the most unusual time series subsequence, in: Fifth IEEE International Conference on Data Mining (ICDM'05), Ieee. pp. 8–pp.

[57] Keogh, E., Ratanamahatana, C.A., 2005. Exact indexing of dynamic time warping. Knowledge and information systems 7, 358–386.

[58] Keogh, E.J., Pazzani, M.J., 1998. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback., in: Kdd, pp. 239–243.

[59] Keogh, E.J., Smyth, P., et al., 1997. A probabilistic approach to fast pattern matching in time series databases., in: Kdd, pp. 24–30.

[60] Kudo, T., Richardson, J., 2018. Sentencepiece: A simple and language independent

subword tokenizer and detokenizer for neural text processing. arXiv preprint arXiv:1808.06226 .

[61] Labarre, A., 2013. Lower bounding edit distances between permutations. SIAM Journal on Discrete Mathematics 27, 1410–1428.

[62] Landauer, T.K., 2007. Lsa as a theory of meaning. Handbook of latent semantic analysis 3, 32.

[63] Larson, M.G., 2008. Analysis of variance. Circulation 117, 115–121.

[64] Li, H., 2015. Piecewise aggregate representations and lower-bound distance functions for multivariate time series. Physica A: Statistical Mechanics and its Applications 427, 10–25.

[65] Li, S., Zhu, Y., Zhang, X., Wan, D.s., 2009. Borda counting method based similarity analysis of multivariate hydrological time series. Journal of Hydraulic Engineering 40, 378–384.

[66] Li, X., Lin, J., 2017. Linear time complexity time series classification with bag-of-pattern-features, in: 2017 IEEE International Conference on Data Mining (ICDM), IEEE. pp. 277–286.

[67] Li, X., Wang, S., Cai, Y., 2019. Tutorial: Complexity analysis of singular value decomposition and its variants. arXiv preprint arXiv:1906.12085 .

[68] Liao, T.W., 2005. Clustering of time series data—a survey. Pattern recognition 38, 1857–1874.

[69] Lin, J., Keogh, E., Lonardi, S., Chiu, B., 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, pp. 2–11.

[70] Lin, J., Keogh, E., Wei, L., Lonardi, S., 2007. Experiencing sax: a novel symbolic representation of time series. Data Mining and knowledge discovery 15, 107–144.

[71] Lin, J., Khade, R., Li, Y., 2012. Rotation-invariant similarity in time series using bag-of-patterns representation. Journal of Intelligent Information Systems 39, 287–315.

[72] Liu, L., Kang, J., Yu, J., Wang, Z., 2005. A comparative study on unsupervised feature selection methods for text clustering, in: 2005 International Conference on Natural Language Processing and Knowledge Engineering, IEEE. pp. 597–601.

[73] Lkhagva, B., Suzuki, Y., Kawagoe, K., 2006. Extended sax: Extension of symbolic aggregate approximation for financial time series data representation. DEWS2006 4A-i8 7.

[74] Mackenzie, C., Pichara, K., Protopapas, P., 2016. Clustering-based feature learning on variable stars. The Astrophysical Journal 820, 138.

[75] Malinowski, S., Guyet, T., Quiniou, R., Tavenard, R., 2013. 1d-sax: A novel symbolic representation for time series, in: International Symposium on Intelligent Data Analysis, Springer. pp. 273–284.

[76] Malz, A., Hložek, R., Allam Jr, T., Bahmanyar, A., Biswas, R., Dai, M., Galbany, L., Ishida, E., Jha, S., Jones, D., et al., 2019. The photometric lsst astronomical time-series

classification challenge plasticc: Selection of a performance metric for classification probabilities balancing diverse science goals. The Astronomical Journal 158, 171.

[77] Manning, C.D., Schütze, H., Raghavan, P., 2008. Introduction to information retrieval. Cambridge university press.

[78] Marteau, P.F., 2008. Time warp edit distance with stiffness adjustment for time series matching. IEEE transactions on pattern analysis and machine intelligence 31, 306–318.

[79] Megalooikonomou, V., Li, G., Wang, Q., 2004. A dimensionality reduction technique for efficient similarity analysis of time series databases, in: Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 160–161.

[80] Megalooikonomou, V., Wang, Q., Li, G., Faloutsos, C., 2005. A multiresolution symbolic representation of time series, in: 21st International Conference on Data Engineering (ICDE'05), IEEE. pp. 668–679.

[81] Minnen, D., Isbell, C.L., Essa, I., Starner, T., 2007. Discovering multivariate motifs using subsequence density estimation and greedy mixture learning, in: Proceedings of the National Conference on Artificial Intelligence, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999. p. 615.

[82] Mislis, D., Mancini, L., Tregloan-Reed, J., Ciceri, S., Southworth, J., D'Ago, G., Bruni, I., Baştürk, Ö., Alsubai, K., Bachelet, E., et al., 2015. High-precision multiband time series photometry of exoplanets qatar-1b and tres-5b. Monthly Notices of the Royal Astronomical Society 448, 2617–2623.

[83] Mokbel, M.F., Ghanem, T.M., Aref, W.G., 2003. Spatio-temporal access methods. IEEE Data Eng. Bull. 26, 40–49.

[84] Montgomery, D.C., Runger, G.C., 2010. Applied statistics and probability for engineers. John Wiley & Sons.

[85] Morinaka, Y., Yoshikawa, M., Amagasa, T., Uemura, S., 2001. The l-index: An indexing structure for efficient subsequence matching in time sequence databases, in: Proc. 5th PacificAisa Conf. on Knowledge Discovery and Data Mining, pp. 51–60.

[86] Morrison, D.F., Marshall, L.C., Sahlin, H.L., 1976. Multivariate statistical methods .

[87] Muthukrishna, D., Narayan, G., Mandel, K.S., Biswas, R., Hložek, R., 2019. Rapid: early classification of explosive transients using deep learning. Publications of the Astronomical Society of the Pacific 131, 118002.

[88] Naul, B., Bloom, J.S., Pérez, F., van der Walt, S., 2018. A recurrent neural network for classification of unevenly sampled variable stars. Nature Astronomy 2, 151–155.

[89] Niennattrakul, V., Ruengronghirunya, P., Ratanamahatana, C.A., 2010. Exact indexing for massive time series databases under time warping distance. Data Mining and Knowledge Discovery 21, 509–541.

[90] Park, S., Kim, S.W., Chu, W.W., 2001. Segment-based approach for subsequence searches in sequence databases, in: Proceedings of the 2001 ACM symposium on Applied computing, pp. 248–252.

[91] Peña, D., Poncela, P., 2006. Dimension reduction in multivariate time series, in: Advances in distribution theory, order statistics, and inference. Springer, pp. 433–458.

[92] Pimentel, Ó., Estévez, P.A., Förster, F., 2022. Deep attention-based supernovae classification of multi-band light-curves. arXiv preprint arXiv:2201.08482 .

[93] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., Keogh, E., 2012. Searching and mining trillions of time series subsequences under dynamic time warping, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 262–270.

[94] Ratanamahatana, C., Keogh, E., Bagnall, A.J., Lonardi, S., 2005. A novel bit level time series representation with implication of similarity search and clustering, in: Pacific-Asia conference on knowledge discovery and data mining, Springer. pp. 771–777.

[95] Sainburg, T., McInnes, L., Gentner, T.Q., 2021. Parametric umap embeddings for representation and semisupervised learning. Neural Computation 33, 2881–2907.

[96] Salvador, S., Chan, P., 2007. Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis 11, 561–580.

[97] Sanchez, H., Bustos, B., 2018. A multi-resolution approximation for time series. Neural Processing Letters , 1–22.

[98] Sánchez Enríquez, H.Y., 2017. Anomaly detection in streaming multivariate time series .

[99] Sayal, M., 2004. Detecting time correlations in time-series data streams. Hewlett-Packard Company 12.

[100] Schmidhuber, J., Hochreiter, S., et al., 1997. Long short-term memory. Neural Comput 9, 1735–1780.

[101] Seijo-Pardo, B., Porto-Díaz, I., Bolón-Canedo, V., Alonso-Betanzos, A., 2017. Ensemble feature selection: homogeneous and heterogeneous approaches. Knowledge-Based Systems 118, 124–139.

[102] Shatkay, H., Zdonik, S.B., 1996. Approximate queries and representations for large data sequences, in: Proceedings of the Twelfth International Conference on Data Engineering, IEEE. pp. 536–545.

[103] Shieh, J., Keogh, E., 2008. i sax: indexing and mining terabyte sized time series, in: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 623–631.

[104] Song, X., Salcianu, A., Song, Y., Dopson, D., Zhou, D., 2020a. Fast wordpiece tokenization. arXiv preprint arXiv:2012.15524 .

[105] Song, X., Salcianu, A., Song, Y., Dopson, D., Zhou, D., 2020b. Linear-time wordpiece tokenization. arXiv e-prints , arXiv–2012.

[106] Soraisam, M.D., Saha, A., Matheson, T., Lee, C.H., Narayan, G., Vivas, A.K., Scheidegger, C., Oppermann, N., Olszewski, E.W., Sinha, S., et al., 2020. A classification algorithm for time-domain novelties in preparation for lsst alerts. application to variable stars and transients detected with decam in the galactic bulge. The Astrophysical

Journal 892, 112.

[107] Sorzano, C.O.S., Vargas, J., Montano, A.P., 2014. A survey of dimensionality reduction techniques. arXiv preprint arXiv:1403.2877 .

[108] St, L., Wold, S., et al., 1989. Analysis of variance (anova). Chemometrics and intelligent laboratory systems 6, 259–272.

[109] Team, P., Modelers, P., 2019. Unblinded data for plasticc classifi cation challenge, doi: 10.5281/zenodo. 2539456.

[110] Timm, N.H., 1975. Multivariate analysis, with applications in education and psychology. Technical Report. Brooks/Cole Publishing Company Monterey, California.

[111] Valenzuela, L., Pichara, K., 2018. Unsupervised classification of variable stars. Monthly Notices of the Royal Astronomical Society 474, 3259–3272.

[112] VanderPlas, J.T., Ivezic, Ž., 2015. Periodograms for multiband astronomical time series. The Astrophysical Journal 812, 18.

[113] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems 30.

[114] Verleysen, M., François, D., 2005. The curse of dimensionality in data mining and time series prediction, in: International work-conference on artificial neural networks, Springer. pp. 758–770.

[115] Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E., 2003. Indexing multi-dimensional time-series with support for multiple distance measures, in: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 216–225.

[116] Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E., 2006. Indexing multidimensional time-series. The VLDB Journal 15, 1–20.

[117] Wang, H., 2007. All common subsequences., in: IJCAI, pp. 635–640.

[118] Wang, H., Lin, Z., McClean, S., Liu, J., 2010. Measuring similarity for multidimensional sequences, in: 2010 IEEE International Conference on Data Mining Workshops, IEEE. pp. 281–287.

[119] Wang, J., Liu, P., She, M.F., Nahavandi, S., Kouzani, A., 2013a. Bag-of-words representation for biomedical time series classification. Biomedical Signal Processing and Control 8, 634–644.

[120] Wang, J., Zhu, Y., Li, S., Wan, D., Zhang, P., 2014. Multivariate time series similarity searching. The Scientific World Journal 2014.

[121] Wang, X., 2011. Two-phase outlier detection in multivariate time series, in: 2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), IEEE. pp. 1555–1559.

[122] Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E., 2013b. Experimental comparison of representation methods and distance measures for time series data. Data Mining and Knowledge Discovery 26, 275–309.

[123] Yang, K., Shahabi, C., 2004. A pca-based similarity measure for multivariate time series, in: Proceedings of the 2nd ACM international workshop on Multimedia databases, pp. 65–74.

[124] Zhang, D., Zuo, W., Zhang, D., Zhang, H., Li, N., 2010. Classification of pulse waveforms using edit distance with real penalty. EURASIP Journal on Advances in Signal Processing 2010, 1–8.

[125] Zhang, P., Huang, Y., Shekhar, S., Kumar, V., 2003. Correlation analysis of spatial time series datasets: A filter-and-refine approach, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer. pp. 532–544.

[126] Zhang, X., Wu, J., Yang, X., Ou, H., Lv, T., 2009. A novel pattern extraction method for time series classification. Optimization and Engineering 10, 253–271.

[127] Zhou, J., Ye, G., Yu, D., 2012. A new method for piecewise linear representation of time series data. Physics Procedia 25, 1097–1103.

Annexed

Additional classification figures

This annex includes additional Confusion Matrices and ROC-AUC curves figures from the different classification experiments described in Chapter 5.

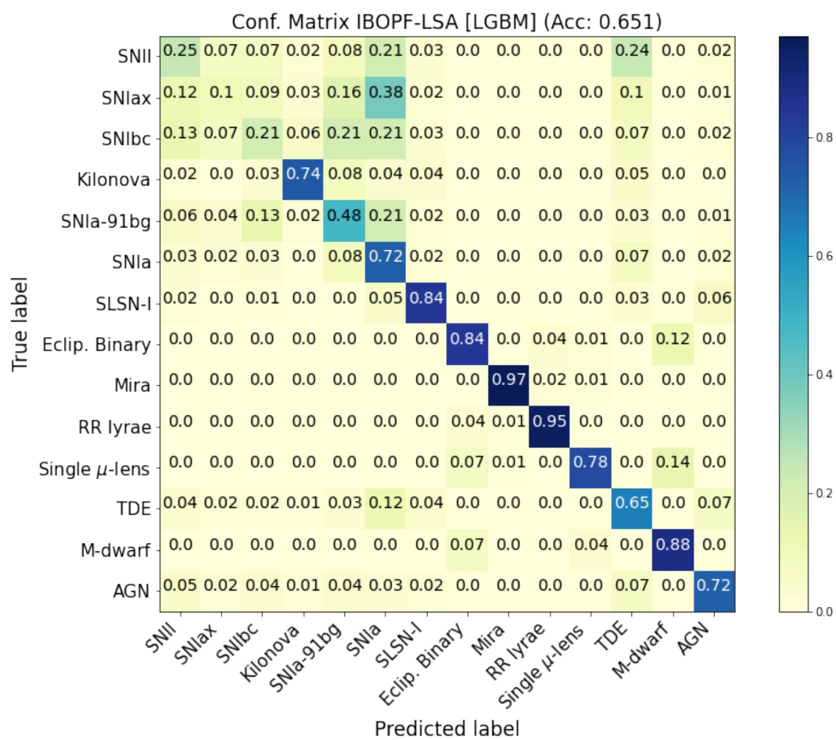


Figure A.1: Confusion matrix for IBOPF-LSA on full dataset using LightGBM classifier.

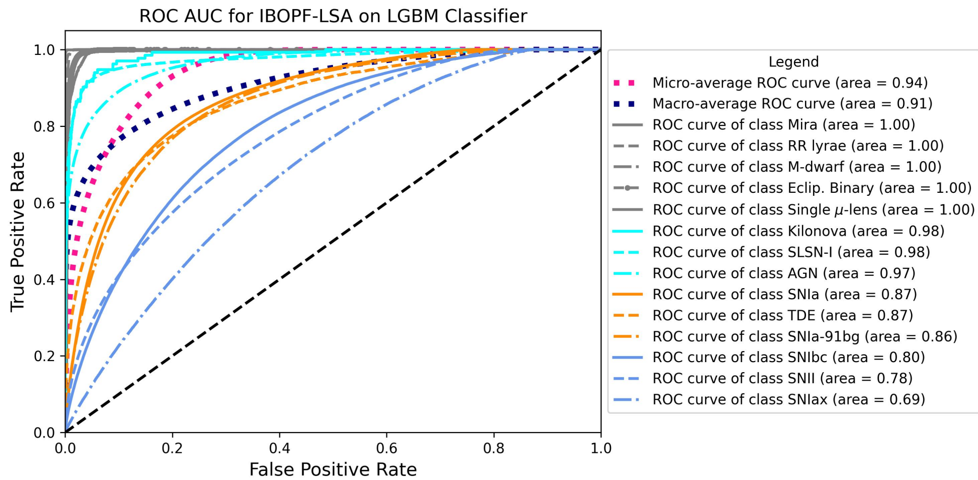


Figure A.2: ROC-AUC curve for IBOPF-LSA on full dataset using LightGBM classifier.

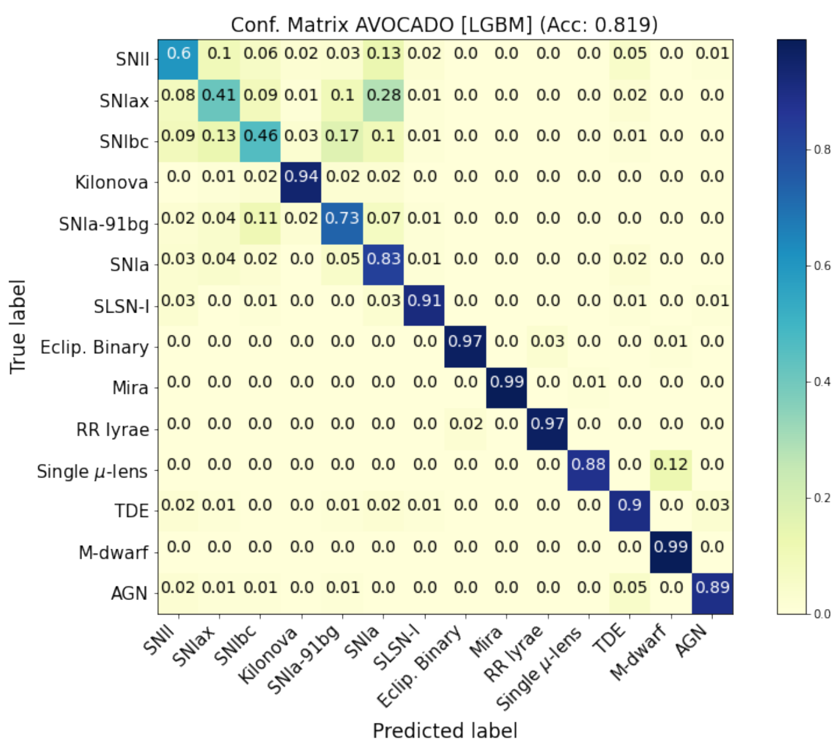


Figure A.3: Confusion matrix for AVOCADO on full dataset using LightGBM classifier.

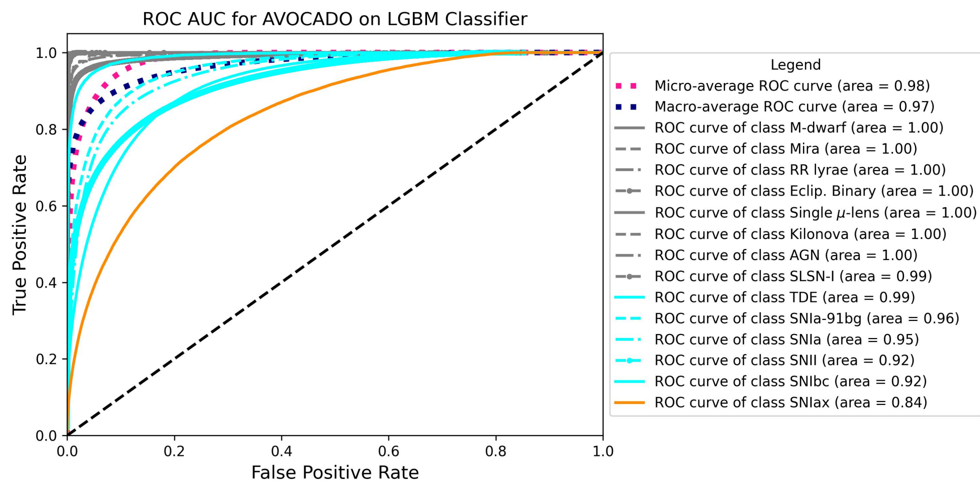


Figure A.4: ROC-AUC curve for AVOCADO on full dataset using LightGBM classifier.

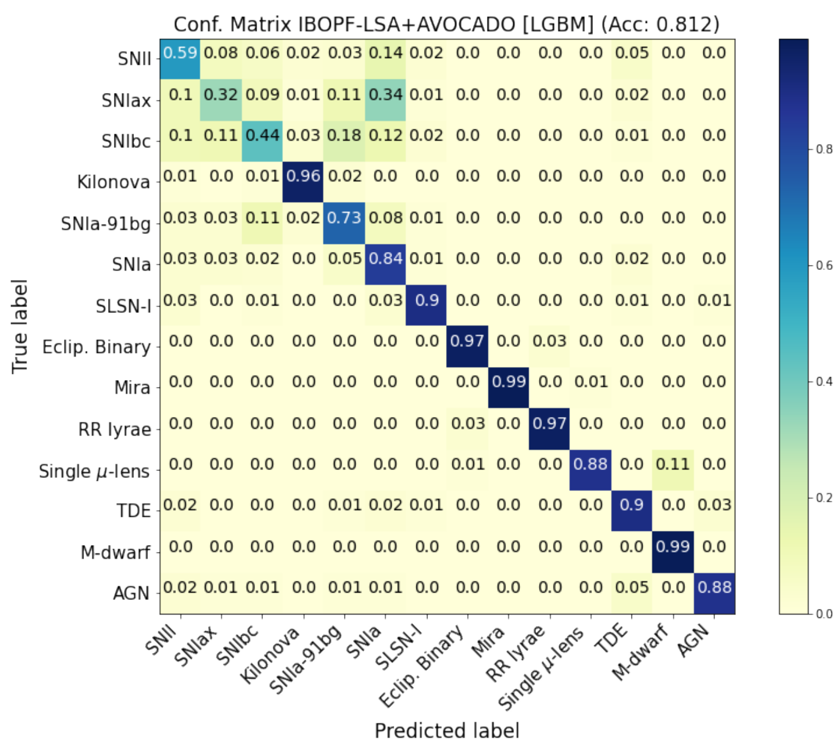


Figure A.5: Confusion matrix for IBOPF-LSA combined with AVOCADO on full dataset using LightGBM classifier.

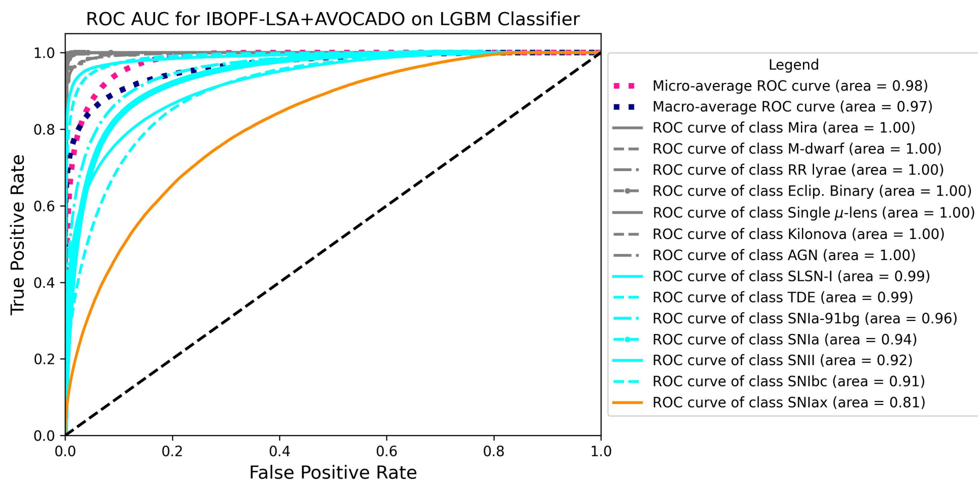


Figure A.6: ROC-AUC curve for IBOPF-LSA combined with AVOCADO on full dataset using LightGBM classifier.

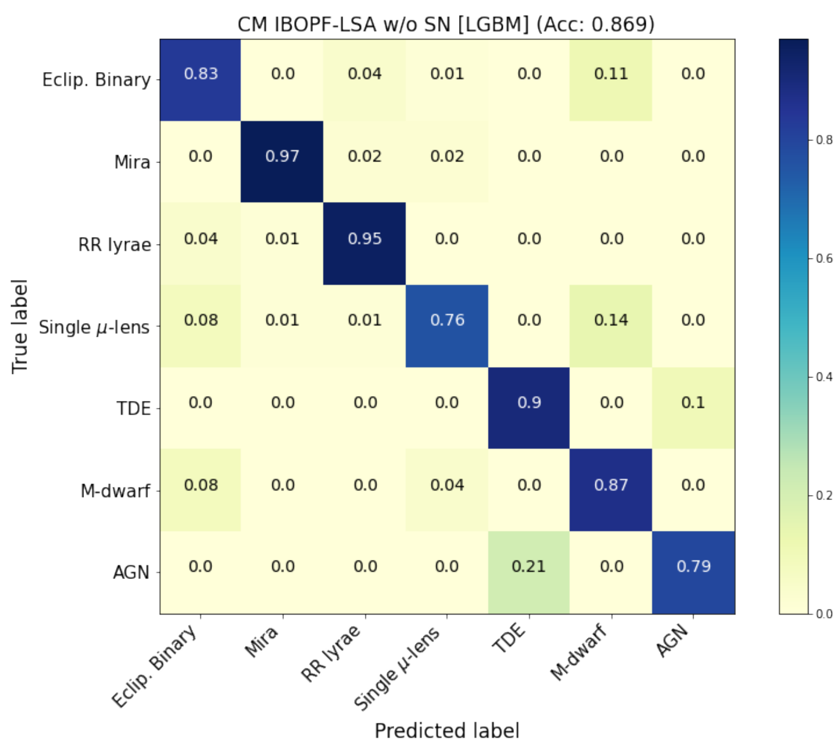


Figure A.7: Confusion matrix for IBOPF-LSA on sub-dataset without Supernovae-type classes using LightGBM classifier.

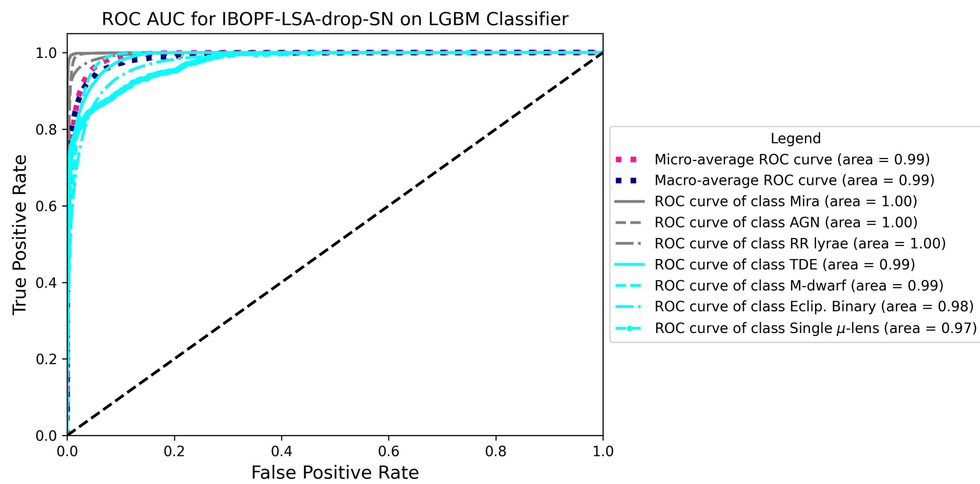


Figure A.8: ROC-AUC curve for IBOPF-LSA on sub-dataset without Supernovae-type classes using LightGBM classifier.

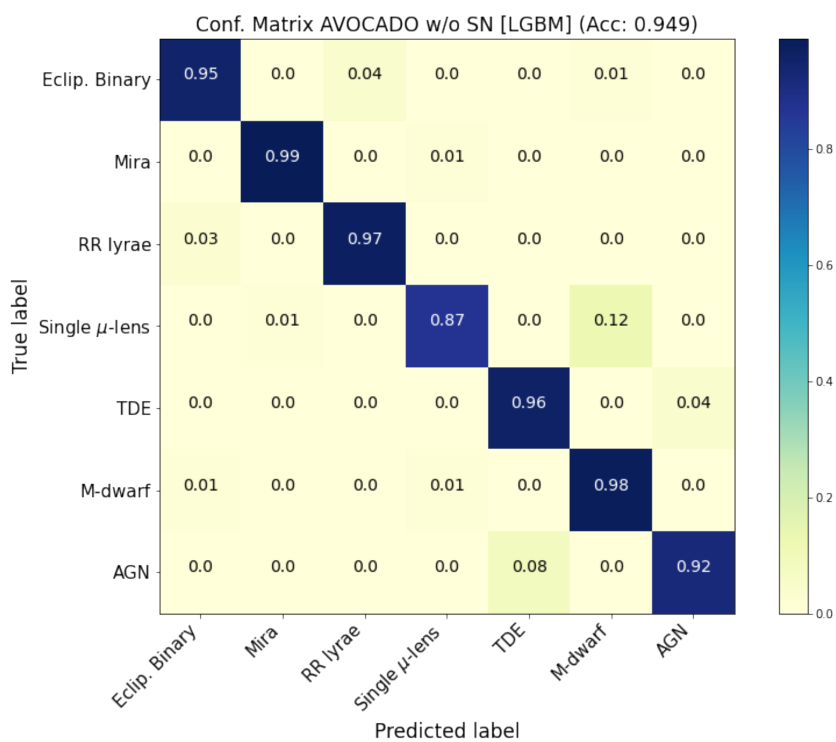


Figure A.9: Confusion matrix for AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier.

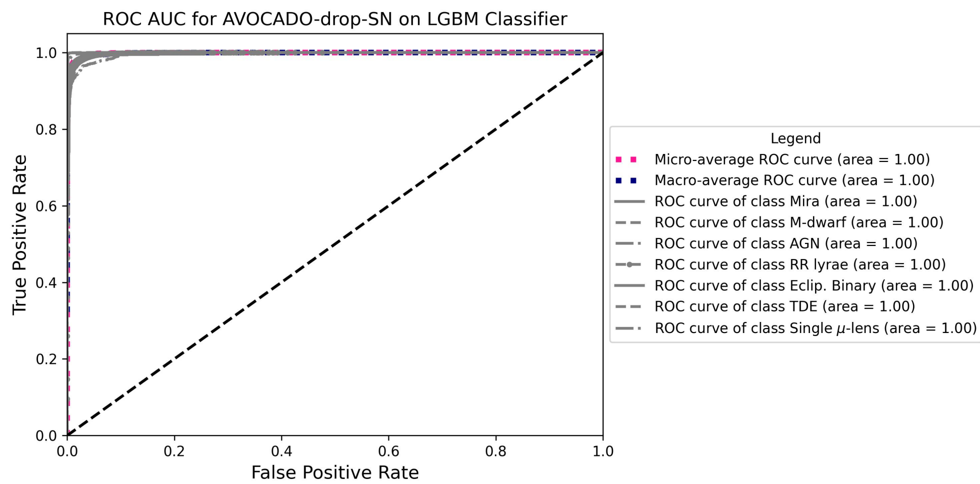


Figure A.10: ROC-AUC curve for AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier.

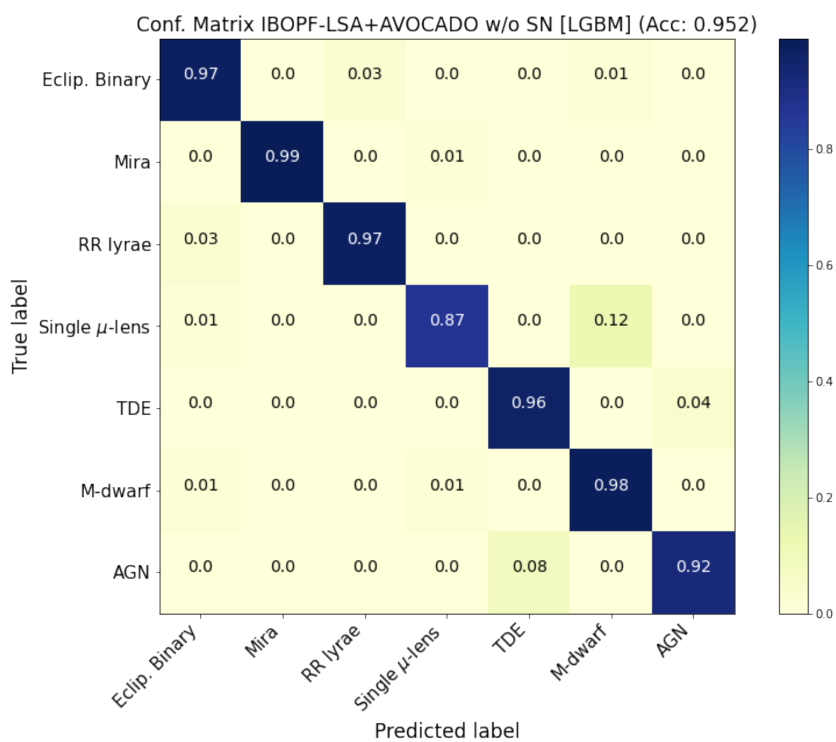


Figure A.11: Confusion matrix for IBOPF-LSA combined with AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier.

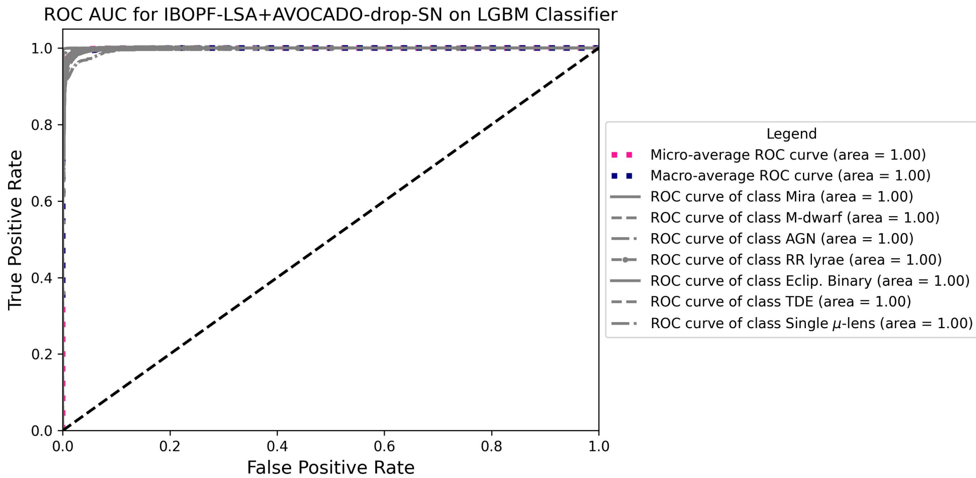


Figure A.12: ROC-AUC curve for IBOPF-LSA combined with AVOCADO on sub-dataset without Supernovae-type classes using LightGBM classifier.

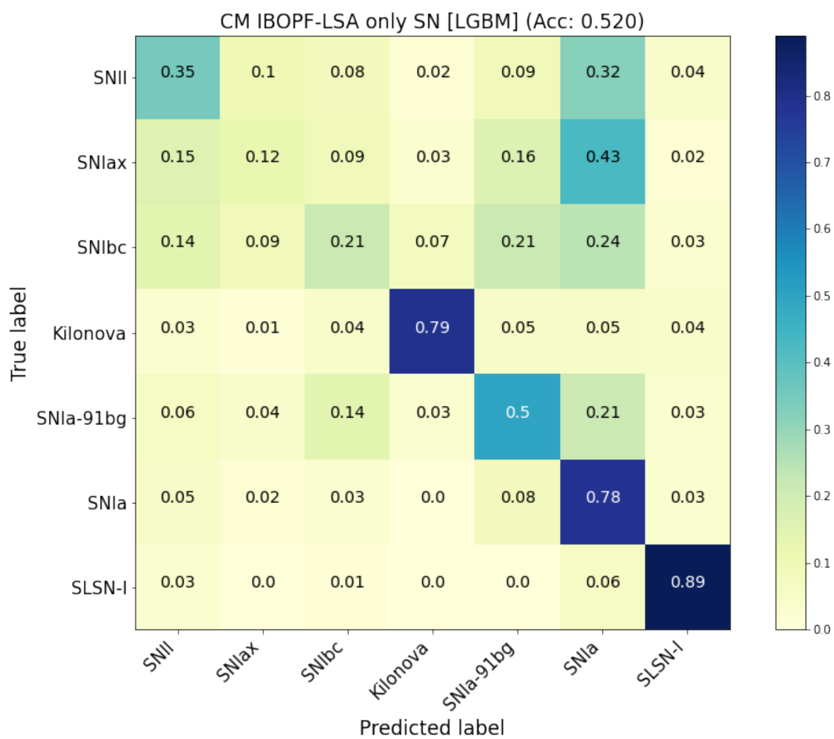


Figure A.13: Confusion matrix for IBOPF-LSA on sub-dataset with only Supernovae-type classes using LightGBM classifier.

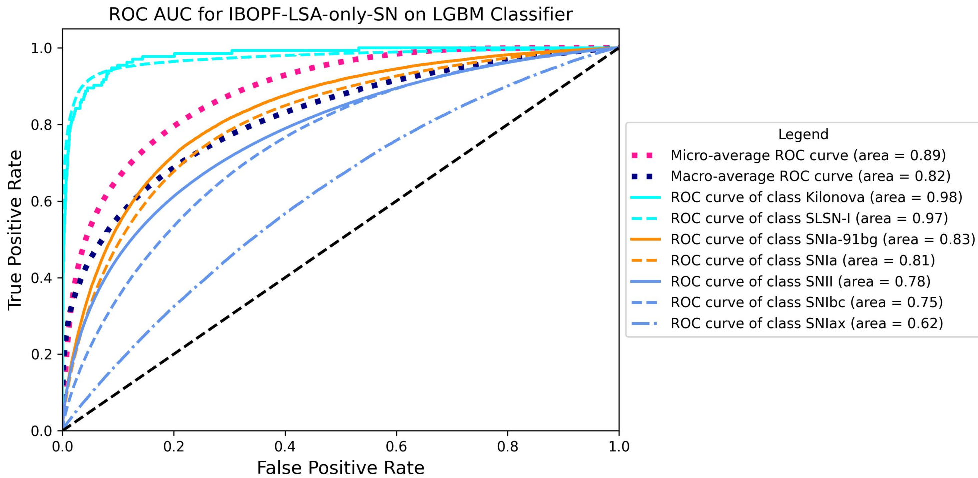


Figure A.14: ROC-AUC curve for IBOPF-LSA on sub-dataset with only Supernovae-type classes using LightGBM classifier.

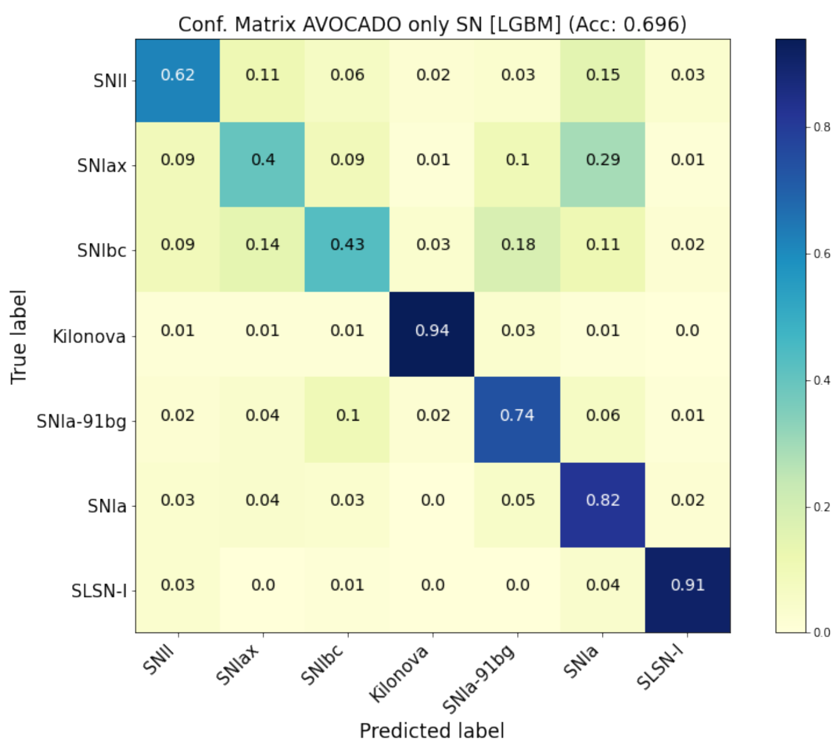


Figure A.15: Confusion matrix for AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier.

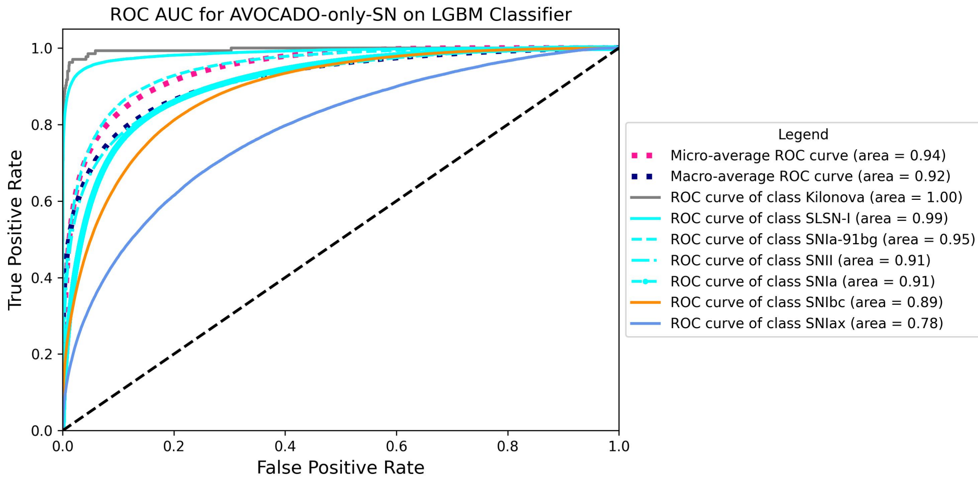


Figure A.16: ROC-AUC curve for AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier.

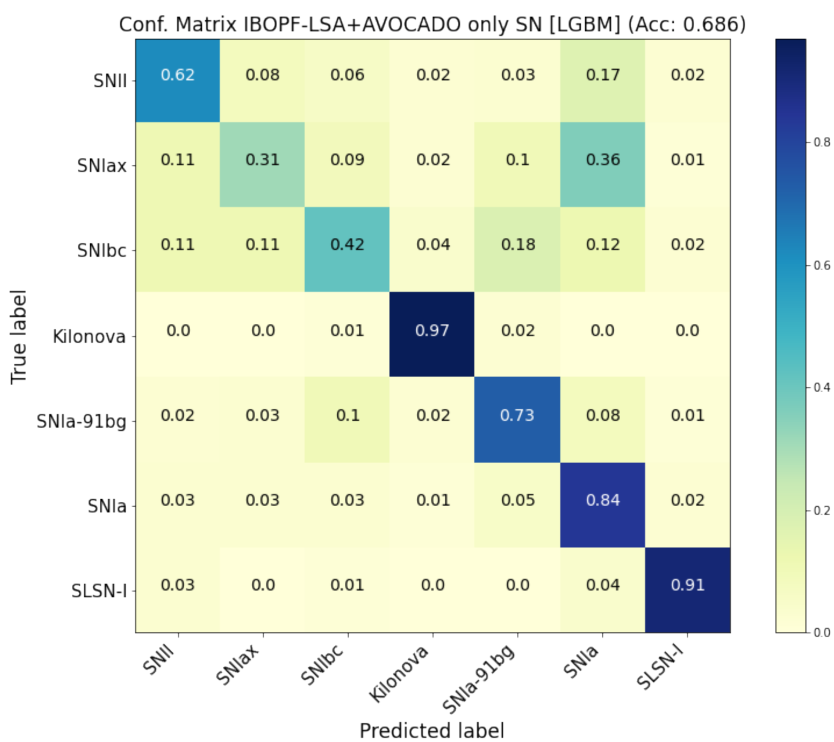


Figure A.17: Confusion matrix for IBOPF-LSA combined with AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier.

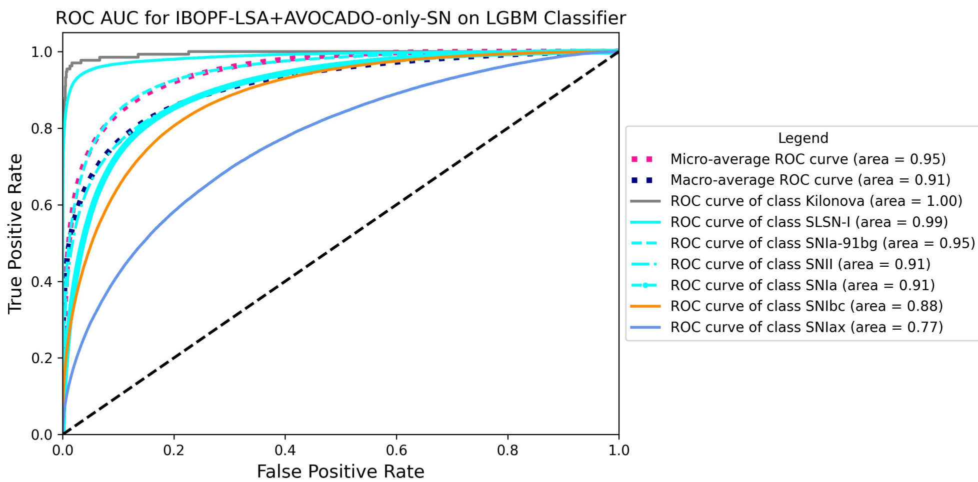


Figure A.18: ROC-AUC curve for IBOPF-LSA combined with AVOCADO on sub-dataset with only Supernovae-type classes using LightGBM classifier.