Universidade de Aveiro Departamento de Física
2018

Diana Barros
de Carvalho

Agrupamento de dados em modelos de Frustração Celular

Clustering in Cellular Frustration models

Diana Barros
de Carvalho

Agrupamentos de dados em modelos de Frustração Celular

Clustering in Cellular Frustration models

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requesitos necessários à obtenção do grau de Mestre em Engenharia Física, realizada sob a orientação científica de Fernão Abreu, Professor do Departamento de Física da Universidade de Aveiro

o júri / the jury

presidente / president

António Ferreira da Cunha
Professor Auxiliar com Agregação da Universidade de Aveiro

vogais / examiners committee

Fernão Rodrigues Vístulo de Abreu
Professor Auxiliar da Universidade de Aveiro (orientador)

Sara Guilherme Oliveira da Silva
Investigadora da Faculdade de Ciências da Universidade de Lisboa

agradecimentos / acknowledgements

Ao Professor Fernão Abreu, pela sua orientação, disponibilidade e colaboração na resolução dos muitos problemas que foram surgindo ao longo da realização desta tese.

Ao Zé Pedro, pela sua paciência, compreensão e apoio.

Aos restantes membros do júri por se terem disponibilizado a arguir esta tese.

palavras-chave exploração de dados, deteção de anomalias, frustração celular.

resumo Sistemas de frustração celular são modelos de interação de agentes que demonstram uma dinâmica complexa que pode ser utilizada para aplicações de deteção de anomalias. Na sua versão mais simples, estes modelos são compostos por dois tipos de agentes, designados de apresentadores e detetores. Os apresentadores exibem a informação das amostras. Os detetores leem essa informação e percecionam-na em sinais binários, dependendo da frequência com que são apresentados. O tipo de sinal percecionado terá impacto na dinâmica de decisões dos agentes. Em particular, a presença de anomalias produz uma dinâmica menos frustrada, i.e., mais estável.

Nesta tese é questionado se este mapeamento em sinais binários não poderá beneficiar do conhecimento da existência de grupos (*clusters*) nas amostras. Com esta finalidade, foi desenvolvida uma técnica de *clustering*, que dá particular atenção ao facto que os sistemas de frustração celular detetam as amostras dependendo do número de características que exibem valores extremos. Os *clusters* obtidos com esta técnica também são comparados com aqueles obtidos com técnicas conhecidas, como o k-*means* ou o *clustering* hierárquico aglomerativo. Nesta tese demonstra-se que a utilização de uma técnica de *clustering* antes da aplicação do sistema de frustração celular pode melhorar as taxas de deteção de anomalias. Contudo, também é demonstrado que dependendo do tipo de anomalias, esta alteração pode não ser benéfica, podendo ser mais vantajoso utilizar a técnica de frustração celular original, uma vez que é mais simples. Acredita-se que este estudo propõe direções claras sobre como se poderá vir a melhorar a técnica da frustração celular num contexto mais geral.

key-words data mining, anomaly detection, cellular frustration.

abstract

Cellular frustrated systems are models of interacting agents displaying complex dynamics which can be used for anomaly detection applications. In their simplest versions, these models consist of two agent types, called presenters and detectors. Presenters display information from data samples. Detectors read this information and perceive it in a binary signal, depending on its frequency of appearance. The type of signal perceived will have an impact on the agents' decision dynamics. In particular, the presence of anomalies leads to less frustrated dynamics, i.e., more stable.

In this thesis it is questioned if the mapping in binary signals could not benefit from the knowledge of the existence of clusters in the data set. To this end, a clustering technique was developed that gives particular attention to the fact that cellular frustrated systems discriminate samples depending on the number of features displaying rare values. The clusters obtained with this technique are also compared with those obtained using k-means or hierarchical agglomerative clustering. It is shown that using a clustering technique prior to application of cellular frustration system can improve anomaly detection rates. However, it is also shown that depending on the type of anomalies, this may not be generally the case, and therefore simpler cellular frustration algorithms may have the advantage of being simpler. It is believed that this study proposes new directions on how to improve the cellular frustration technique in a broader context.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Cellular frustrated systems are anomaly detection algorithms and originally were inspired by a model that proposes to describe how men and women choose their partners [1]. The most interesting new idea was however that men and women would frustrate each others pairing preferences instead of looking for a stable configuration. Frustration turns out to be important because it produces an amplification mechanism to build specific long lived interactions [2, 3].

The main goal of this thesis is to improve the anomaly detection rates of the cellular frustration system proposed in [4]. This will be achieved by adding a pre-processing stage with clustering algorithms to help agents process the information presented by each sample in a certain data set. Three different clustering techniques will be tested and compared: the k-means, the hierarchical agglomerative clustering and the extremes clustering technique. Being the latter, a technique created specifically for the needs of the cellular frustration system.

Anomaly detection algorithms solve the problem of finding patterns in data that do not adjust to the expected behaviour of normal data. These nonconforming patterns are called anomalies or outliers [5].

Anomalies in data are significant, and sometimes critical, information in a broad variety of applications. For example, anomalies in credit card transactions could indicate that the credit card was stolen [6], an anomalous pattern in a computer network could mean that the computer is hacked and sending out sensitive data to an unauthorised destination [7], an anomalous MRI image may indicate the presence of malicious tumours [8], or even anomalous readings from a space craft sensor could indicate a fault in some component [9].

The thesis is organised as follows. In Chapter 2, an introduction to data mining is made, as well as the description of relevant algorithms for this work. In Chapter 3, cellular frustration systems will be introduced and described. In Chapter 4, first, the extremes clustering technique will be described, and secondly, it is shown how clustering techniques can be integrated in the cellular frustration system. In Chapter 5, a practical application of the algorithm is going to be made with synthetic data and later with a real wine data set. The thesis ends with a conclusion and a discussion of how these results open new research directions.

Chapter 2

Data mining

In a modern society, computer systems are accumulating enormous amounts of data every second, starting from bank cash withdrawals or credit card transactions, to stock exchanges and satellites' Earth observation. Today the internet makes gigantic volumes of information available, making the world data rich but comparatively knowledge poor [10]. Indeed, the challenge is not anymore to gather data, but to gain knowledge from the available data, which still hide patterns that could be useful to achieve discoveries in science, to predict the weather and natural disasters [11], to find cures for illnesses or help physicians identify effective treatments and best practices [12] and also in customer relationship management [13].

Data mining is the process of discovering patterns in data sets involving methods of machine learning, artificial intelligence, pattern recognition, statistics and database systems. It was originally proposed to solve perceptual tasks like optical character recognition, face recognition and voice recognition. These are tasks that humans can perform many times easily but there is no mathematical model to describe them [14].

There can be two different goals to data mining: description or prediction of data. Description is finding human interpretable patterns that describe the data. Prediction is using the available data to predict values in future cases.

Usually, data sets are composed of several samples, each of which comprises the values of a number of variables, commonly called features. Samples can be labelled or not. If samples are labelled, the data mining used is referred as supervised learning where the goal is to predict the labels of new samples. If they are not labelled, it is referred as unsupervised learning and usually, the aim is to create groups of similar samples, i.e., clusters [10].

In supervised learning the algorithm uses a set of training samples to learn a function that maps feature values to labels. Depending on the type of label, categorical or numerical, the data mining tasks can be classification or regression [15]. Classification is one of the most common data mining tasks. Examples of well known classification algorithms are Artificial Neural Networks [16], Decision Trees [17], Naive Bayes [18] and Support Vector Machines [19].

In unsupervised learning the algorithm learns a function that describes the data, even though no explicit feedback is given [15]. The most common unsupervised algorithms are clustering techniques [20].

This chapter is organised as follows. In the next section I will explain one of the most powerful techniques in data mining: Support Vector Machines (SVMs). SVMs for anomaly detection are an extension of the original algorithm, derived for classification tasks. For this reason the next section discusses both techniques. Afterwards well known clustering techniques

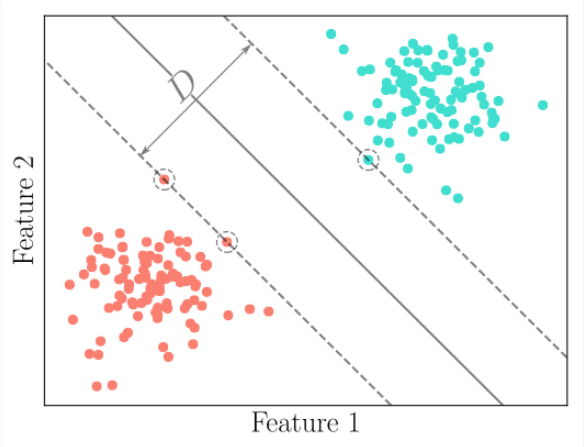


Figure 2.1: Linear separable data set where samples from two classes are represented in two colours, pink and blue. The solid line represents the hyperplane separating the two classes; dashed lines represent the maximum separating margins. The points that lie on the these margins are circled and are called the support vectors. The distance between the separating margins is called the gap, D.

- k-means and the hierarchical agglomerative clustering - are also introduced. They will be used to understand how they can improve anomaly detection rates of the cellular frustration algorithm and to compare with the new clustering technique developed here. To illustrate these techniques at play, we will use synthetic data sets with samples with only two features, generated using two Gaussian number generators, one for each feature.

2.1 Support Vector Machines

Support Vector Machines (SVMs) are one of the best known models used for classification and regression. SVMs are algorithms that proceed in two stages: training and testing. Training is used to find the model parameters that later, in the testing stage, will be used to classify a new sample.

Even though we are interested in anomaly detection algorithms, we are going to start off this discussion with SVMs that perform binary classification. In a latter sub-section, an extension of the SVMs will be described that has applications in anomaly detection.

2.1.1 Linear separable case

Consider the example in Fig. 2.1, where the classes are represented by colours. The main goal of the SVMs is to find a linear decision surface that can separate the two classes and has the maximum distance from the closest points of each class. The decision surface is called a hyperplane, represented by the solid line in Fig. 2.1. The samples that lie on the defined maximum separating margin, represented with circled points in Fig. 2.1, are called support vectors, thus the name of this algorithm, support vector machines. The hyperplane is a linear decision surface that splits the space into two parts. If the data set is 2-dimensional, the hyperplane is a line, if the data set is 3-dimensional, the hyperplane is a plane and so on.

In a data set with N samples, each sample is represented by a set of features $\vec{x_i}$, with $i = 1, 2, ..., n$ and has an assigned class $y_i = \pm 1$. The hyperplane is defined as a set of points that uphold the following equation

$$\vec{w}\vec{x} + b = 0. \tag{2.1}$$

The parameters that need to be found in order to the define the decision surface are $\vec{w}$ and b. This is an optimisation problem that can be solved efficiently with Quadratic Programming. Quadratic Programming is a kind of optimisation method that optimises an objective function with subject to linear constraints. The only requirement for it to work is that the objective function has to be convex, i.e., have only one global minimum.

The goal is to maximise the gap, D, delimited in Fig. 2.1 by the dashed lines. The dashed lines are parallel hyperplanes and can be defined as

$$\vec{w}\vec{x} + b = -1 \quad \text{and} \quad \vec{w}\vec{x} + b = +1 \tag{2.2}$$

so the gap is $D = \frac{2}{||\vec{w}||}$. In order to maximise D, $||\vec{w}||$ has to be minimised. Thus, our objective function is $\frac{1}{2}||\vec{w}||^2$. The linear constraints need to make sure that all the samples are correctly classified. The subspace where a sample has $y_i = -1$ is delimited by $\vec{w}\vec{x_i} + b \leq -1$ and the subspace where a sample has $y_i = +1$ is delimited by $\vec{w}\vec{x_i} + b \geq +1$. Both of these subspaces, as well their dependency on the class, can be defined as

$$y_i\big(\vec{w}\vec{x_i} + b\big) \geq 1 \tag{2.3}$$

Hence, to find the hyperplane, $\frac{1}{2}||\vec{w}||^2$ needs to be minimised subject to $y_i\big(\vec{w}\vec{x_i} + b\big) \geq 1$, for $i = 1, 2, ..., n$.

This is the primal formulation of the linear SVMs where the optimisation problem has n variables, being n the number of features.

The same problem can be formulated in a dual form where instead of having n variables there will be as many variables as training samples, N. This is done by solving for the Lagrangian dual, in which the optimal solution is found by maximising

$$\sum_{i}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j (\vec{x_i}^T \cdot \vec{x_j}) \tag{2.4}$$

being α the Lagrangian multipliers and they are only non-zero for the support vectors (points that are closer to the hyperplane). Eq. 2.4 is the objective function and it needs to be solved subject to the constraints $\alpha_i \geq 0$, and $\sum_i^N \alpha_i y_i = 0$. Once α_i are found, $\vec{w}$ can be computed as

$$\vec{w} = \sum_{i}^{N_s} \alpha_i \vec{x_i} \tag{2.5}$$

where N_s is the total number of support vectors. Since that for any support vector $\vec{x_s}$ the solution of $y_s(\vec{w}\vec{x_s} + b)$ is 1, the parameter b can be computed as

$$b = \frac{1}{y_s} - \vec{w}\vec{x_s} = \frac{1}{y_s} - \sum_{i}^{N_s} \alpha_i \vec{x_i}\vec{x_s} \tag{2.6}$$

After the training stage, where the optimal α and b values are stored, a new sample $\vec{z}$ can be tested as

$$h(\vec{z}) = sign\Big(\sum_{i}^{N_s} \alpha_i y_i (\vec{x_i}^T \cdot \vec{z}) - b\Big) \tag{2.7}$$

If the sign of $h(\vec{z})$ is positive, the sample z is going to be labelled as one of the classes, if it is negative, the sample is classified as a member of the other class.

The dual form is better because only the dot products between samples is needed, not the original data set. Also, the number of free parameters α_i is upper bounded by the number of samples and not features, this is particularly good in high-dimensional data sets. The complexity of the testing phase is only going to depend on the number of support vectors.

Nonetheless, sometimes the data set has noise or outliers so that finding the linear decision surface may not be an easy task. To work around problems like these, a new variable, $\xi \geq 0$ called a slack variable is going to be introduced. If a sample is misclassified, ξ_i can be thought of the distance from the separating hyperplane. If it is correctly classified, $\xi_i = 0$. By doing this, we are allowing the SVMs to consider that some training samples might be of the other class. This is called the Soft-Margin SVMs where, in the primal formulation the objective function,

$$\frac{1}{2}||\vec{w}||^2 + C\sum_{i}^{n} \xi_i, \tag{2.8}$$

needs to be minimised subject to the constraints $y_i\big(\vec{w}\vec{x_i} + b\big) \geq 1$, for $i = 1, 2, ..., n$. And in the dual form,

$$\sum_{i}^{N} \alpha_i - \frac{1}{2}\sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j (\vec{x_i}^T \cdot \vec{x_j}), \tag{2.9}$$

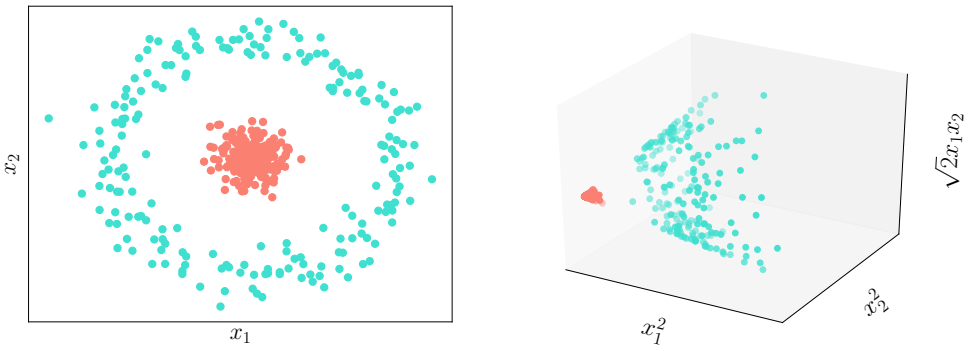
needs to be maximised subject to the constraints $0 \leq \alpha_i \leq C$, and $\sum_i^N \alpha_i y_i = 0$. C is a user defined parameter that penalises errors and controls the width of the gap. If C is very large, the soft-margin SVMs are equivalent to the previous approach where no errors in the training stage are allowed. If C is small, some misclassifications in the training data set are allowed since the gap is wide. C needs to be selected depending on the data set that is being used because different C will lead to different decision surfaces.

2.1.2 Non-linear separable case

The former SVMs approach works well for linearly separable cases, however the previous optimisation problem will find no feasible solutions for non-linearly separable data set. In order to solve problems like these, one must perform the kernel trick. The kernel trick maps the data set into a higher dimensional space where it is more likely for the data set to be linearly separable. Take the example in Fig. 2.2(a), in 2D the data set can not be separated by a hyperplane. Nevertheless, if the data set is mapped into a higher space, $F(\vec{x})$, with these particular transformations

$$f_1 = x_1^2 \quad f_2 = x_2^2 \quad f_3 = \sqrt{2}x_1 x_2, \tag{2.10}$$

the data set is now linearly separable in 3D, as seen in Fig. 2.2(b). In the SVMs algorithm, this is achieved by replacing the dot product $\vec{x_i} \cdot \vec{x_j}$ by $F(\vec{x_i}) \cdot F(\vec{x_j})$. However, this can be



(a) (b)

Figure 2.2: Non-linearly separable data set in 2D (a) and in 3D (b) after applying the transformations, $F(\vec{x})$, to the 2D case. In (a) it is clear that the data set can not be separated with a linear hyperplane. In order to overcome this problem, the data set can be mapped into a higher dimensional space, for example 3D, with the transformations described in Eq. 2.10. In 3D the data set is now linearly separable.

---

achieved in another way. There is no need to compute F for every point before making the dot product $F(\vec{x_i}) \cdot F(\vec{x_j})$. This can be achieved with a Kernel function $K(\vec{x_i}, \vec{x_j})$. This function computes the dot product and maps the data set into a higher dimension all at once. In this example, the kernel function is a polynomial function of second degree, $K(\vec{x_i}, \vec{x_j}) = (x_i \cdot x_j)^2$. So, rewriting Eq. 2.9 with the kernel trick, we need to maximise

$$\sum_{i}^{N} \alpha_i - \frac{1}{2} \sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j K(\vec{x_i}, \vec{x_j}) \tag{2.11}$$

subject to same linear constraints as before and the decision function for a new sample $\vec{z}$ is

$$h(\vec{z}) = sign\Big(\sum_{i}^{N_s} \alpha_i y_i K(\vec{x_i}, \vec{z}) - b \Big). \tag{2.12}$$

After applying the non-linear SVM, one can plot the supporting hyperplane in the original feature space (2D), as seen in Fig. 2.3.

In this example a fairly simple polynomial kernel function was used. Still, there are a lot of kernel functions that can be used, such as the linear kernel, the sigmoid kernel, the radial basis function (RBF) kernel, etc. Choosing the right kernel function for the data set constitutes the biggest difficulty on applying the SVMs algorithm.

2.1.3 One-class Support Vector Machines

As seen in the previous sub-sections, support vector machines work with the underlying assumption of the existence of, at least, two different classes. Nevertheless, the SVMs formalisation was extended to learn a descriptive model of a data set with only one class. In one-class problems, only normal samples are presented during training of the algorithm. In the testing stage, if a new sample is too different from the model created during training, the

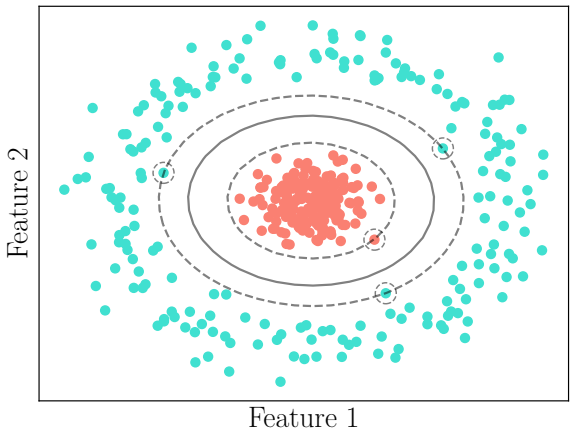


Figure 2.3: Final result in the original 2D space of the labelling of samples in two groups. The two classes are represented by different colours (blue and pink dat points). The support vectors are circled and the supporting hyperplane is represented by the solid line.

---

sample is labelled as an anomalous sample. An example of a one class problem is depicted in Fig. 2.4(a) where the blue points are the normal samples while the pink samples are the anomalous samples. In problems like these, only the normal samples are known, the anomalies can be anywhere in the feature space. Thus, the goal of a One-class SVMs is to define the normal space as better as possible, so most of the anomalies will be detected. This algorithm is unsupervised, since there are no labels in the data set and after the training stage the algorithm does not know how the other class (anomalous samples) looks likes. In the testing stage it classifies the new samples as similar or different to the training samples.

In the one-class SVM, the hyperplane separates the training examples with maximum margin from the origin of the feature space [21]. It can be obtained, again, by maximising

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j K(\vec{x_i},\vec{x_j}) \tag{2.13}$$

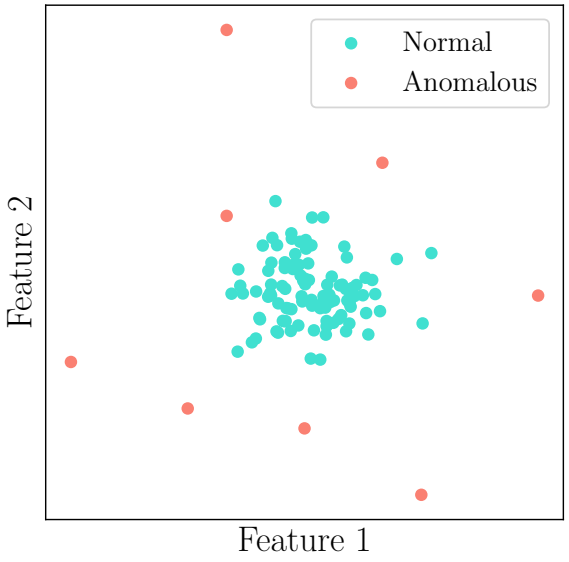
subject to the constraints $0 \leq \alpha_i \leq \frac{1}{\nu n}$, where α_i and α_j are the Lagrange multipliers, $K(\vec{x_i},\vec{x_j})$ is the kernel function , n is the number of training examples and ν sets an upper bound on the allowed number of training examples that are considered as outliers.

Solving Eq. 2.13, results in a binary function that returns $+1$ in a small region of the feature space where the training samples are located and -1 elsewhere. So, for a new sample $\vec{z}$, the function looks like

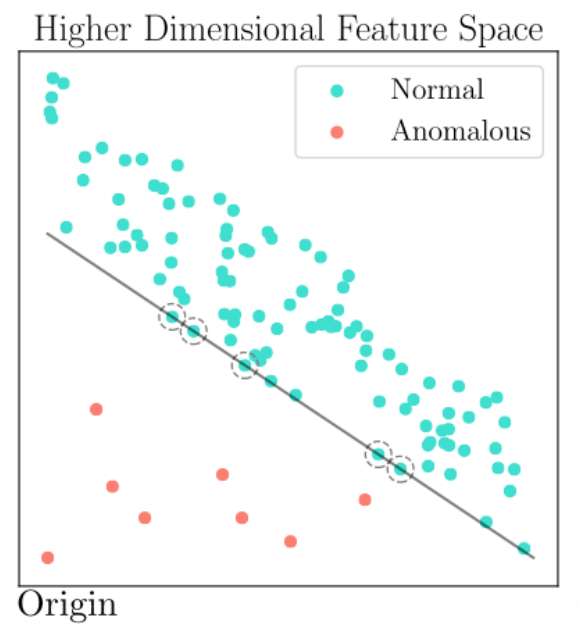
$$h(\vec{z}) = sign\Big(\sum_{i}^{N_s}\alpha_i K(\vec{x_i},\vec{z}) - b\Big) \tag{2.14}$$

where, similarly to before, $b = \sum_i^{N_s}\alpha_i K(\vec{x_i},\vec{x_s})$.

In this extension of the SVM, the kernel trick is needed not to linearly separate one class from the other (since there are no classes) but to separate the normal samples from the origin as seen in Fig. 2.4(b).



(a)



(b)

Figure 2.4: Example of an anomaly detection problem. In (a), the original data is displayed where blue samples are normal samples and pink are (possible) anomalous samples. In (b), the One-Class SVMs were applied and with the kernel trick the data set is now mapped into a higher dimensional feature space where it is possible to define a hyperplane (solid line) that separates the training samples from the origin. The support vectors are encircled with a dashed line. Hopefully, the anomalous samples will be located on the other side of the decision boundary.

The main difficulty when using SVMs is choosing the kernel function and several user-defined parameters like ν. When it is used to perform classification, the choice can be made by trying different kernel functions and parameters. This could be tiresome but usually works, since all classes are previously defined. With the One-Class SVMs this is tougher because we only know how the normal class looks like and the anomalies could be everywhere. This constitutes a problem that will be explored in Chapter 5.

2.2 Clustering Techniques

Clustering is an unsupervised algorithm that aggregates samples into groups (called clusters). Clustering algorithms have been implemented in several contexts and by researchers in many disciplines [22]. The goal is to define clusters where the samples withing a cluster are similar and samples from different clusters are different.

There are several clustering techniques, with different inclusion principles, that for the same data set, produce different outcomes. An important question is to know which technique is more suitable for each data type or purpose [23].

There are several types of clustering techniques such as partitional and hierarchical. A partitional clustering technique is when the data set is divided into non-overlapping clusters. In hierarchical clustering the clusters are organised as a tree. The root of the tree is a cluster with all the samples while the clusters on each node are the union of its children. Note that a hierarchical clustering can be viewed as a sequence of partitional clustering and a partitional clustering can be obtained by cutting the hierarchical tree at a certain level.

2.2.1 k-means

k-means is the best known and simplest partitional clustering technique. The user needs to choose the number of desired clusters k. Every cluster is defined by a centroid, which usually is the mean value of the samples belonging to the respective cluster [24]. In the basic k-means algorithm, the initial k centroids are chosen randomly from the samples. Then, every sample is assigned to the closest centroid. With the initial clusters defined, the centroids are updated to the mean value of the samples belonging to that cluster. Then the process repeats itself: the samples are assigned to the centroids and then they are updated. This continues until the centroids remain the same. This algorithm is described in Pseudo-code 1 and pictured in Fig. 2.5.

In order to assign the points to the clusters, there is the need to define a proximity measure. For real or integer data sets usually the Euclidean distance is used, however one can also use the Manhattan or the Minkowski distance [25].

The goal of the k-means it is usually defined by an objective function that depends on the distance between the points and centroids. If the Euclidean distance is chosen as a proximity measure the objective function is the sum of the squared error (SSE).

$$SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist(c_i, x)^2 \tag{2.15}$$

where C_i are the clusters, c_i the centroids of each cluster and $dist()$ is the Euclidean distance between two objects. This means that the goal is to minimise the squared distance of each point to the centroid.

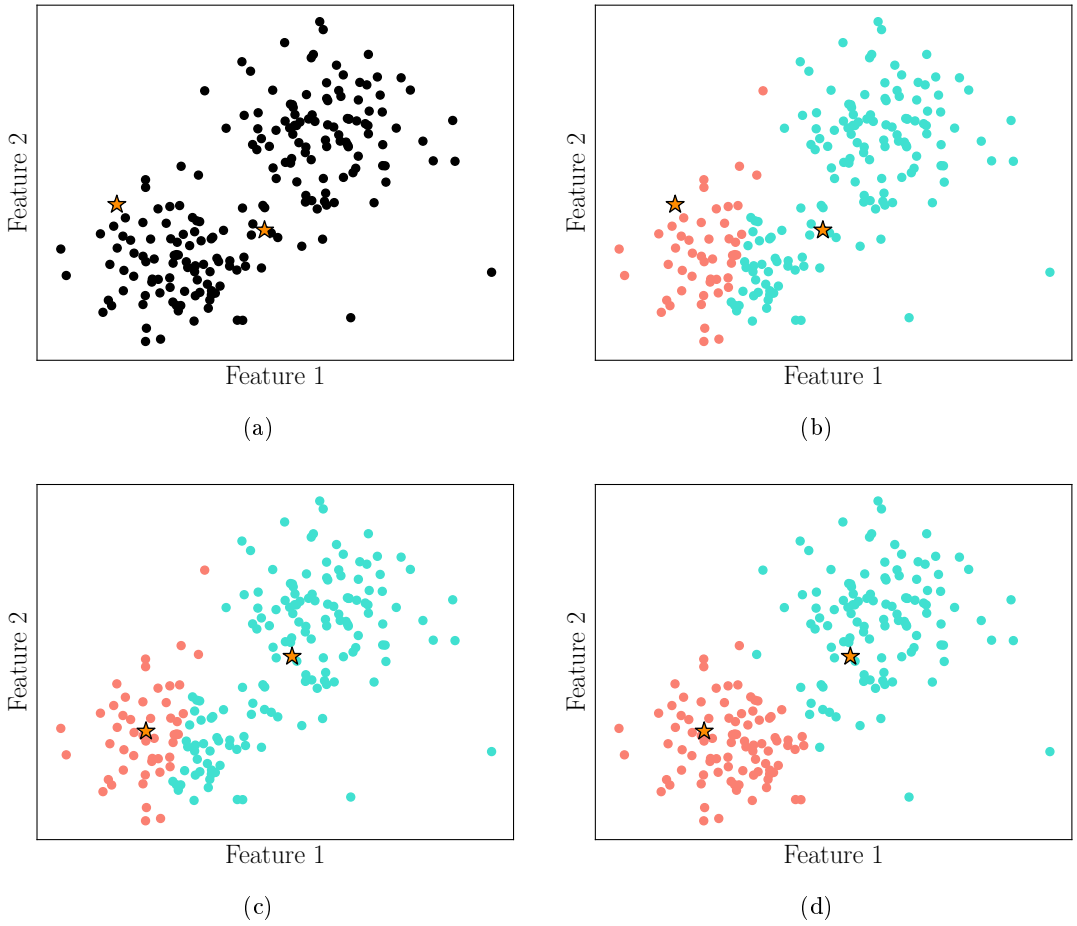


Figure 2.5: Application of the k-means clustering algorithm to find the two clusters in a toy model. The centroids of each cluster are represented by an orange star. In (a), the centroids are choosen randomly from the samples. In (b), the samples are assigned to the closest centroid. Samples in pink belong to one cluster and the ones in blue belong to the other. In (c), the centroids are updated to the mean value of the samples that belong to its respective cluster. In (d), the samples belonging to each cluster are updated regarding the new centroids. This process repeats itself until the centroids remain the same.

Since the k-means always converge to a solution, not necessarily an optimal one, for two runs of the algorithm on the same data set, we would prefer the one with the lowest SSE.

Algorithm 1 k-means algorithm

1: Select k initial centroids
2: **while** the centroids change **do**
3: Assign every sample to a centroid
4: Recompute the new k centroids

2.2.2 Hierarchical Agglomerative

Hierarchical algorithms are also very well known and there are two approaches to it: agglomerative and divisive [26]. With the agglomerative approach, the clusters are built by starting with every sample as an individual cluster and then merging the most similar clusters into bigger ones. The divisive approach works the other way around. It starts with a big all inclusive cluster and then starts to split them up. We are going to use and explain the agglomerative approach.

In this algorithm, described in Pseudo-code 2, there is the need to define the proximity between clusters so we can merge the similar ones. This definition can vary depending on the data set or on the type of clusters that are intended. There is the single link proximity measure, where the distance between the two closest points of different clusters is used. The complete link approach uses the distance between the two farthest points of different clusters.

An alternative technique is the Ward's method, where the clusters are represented by centroids and the proximity between two clusters is measured in terms of increase in the SSE that results from merging the two clusters.

Algorithm 2 Agglomerative Hierarchical clustering algorithm

1: Compute the proximity matrix between all clusters
2: **while** there are more than one cluster **do**
3: Merge the closest two clusters
4: Recompute the proximity matrix

Since hierarchical algorithms can be seen as a tree, a common visual representation of it is a dendrogram. Using the same toy model as before, the dendrogram is displayed in Fig. 2.6. In the dendrogram, the two closest samples are merged successively until there is only one cluster. One can see that the if the tree is cut above all the blue and pink nodes, we will get the similar clusters as in the k-means algorithm.

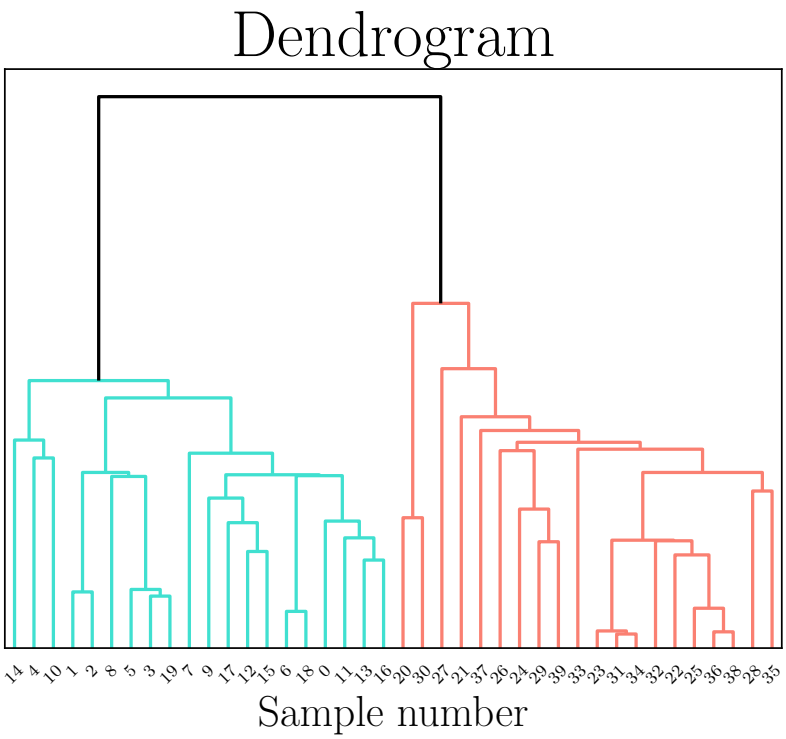


Figure 2.6: Dendrogram of the toy model with 40 samples. The samples are represented in the x axis. In the y axis, a measure of closeness of either individual data points or clusters is represented. In the bottom of the graph, each sample represents its own unique cluster, these are called the leaf nodes. Samples that are close to each other are merged in pairs forming a new cluster and a node in the graph. The lower the position of the node in the y axis, the closer together are the two samples or clusters that are being merged.

Chapter 3

Cellular Frustrated Systems

The Cellular Frustrated System (CFS) was inspired from a well-known problem in computational mathematics, the stable marriage problem (SMP). This problem was proposed by Gale and Shapley in 1962 and its aim is to efficiently match men and women in stable pairs [1]. Stable pairs are formed when both elements prefer their current partners relatively to any other individual. In this problem, men and women are the two types of interacting agents. Both, men and women, have different and complex preferences regarding agents of the opposite sex. Solving the SMP can be computationally hard in several variants of the problem [27] and therefore natural systems that could be modelled with this type of models, may not ever reach a final stable configuration with all agents matched. For this reason, it was also studied whether this type of models could be used to explore the dynamical properties of systems that attempt to reach the stable solution but never get there. It was found that these ideas could be applied in evolution to explain the emergence of new species [28] and immunology. As the immune system protects the host from invaders, an immune inspired computational system would be able to detect undesirable behaviours or, in the context of data mining, anomalies in data sets [29].

3.1 Motivation

The Cellular Frustration Framework (CFF) uses the ideas taken from the stable marriage problem to propose how protection systems or methods to detect anomalies should be built [4]. Contrary to the SMP, the CFF explores the complex dynamics emerging from the existence of potentially conflicting preferences among agents. It was noticed that they could change considerably the stability of pairings. If the stability of pairings could have a crucial impact on the system, for instance, to trigger a reaction, then it becomes clear that the analysis of the complex dynamics could have a strong impact in the system's response.

In the stable marriage problem, men and women have a list of preferences and interact with each other regarding their preferences lists. In Fig. 3.1, a population of men and women, as well as their preferences, are depicted. In this example, men's preferences depend only on women's hair colour and blond (brunet) men will prefer to be married with blond (brunette) women. On the other hand, women have more complex preferences list because they are interested in a combination of different features (hair colour, type of hair, if the man wears glasses or not, clothes, etc.). People will interact in marriages and if they both prioritise this interaction rather than all others, the marriage will last a long time. If not, the interaction

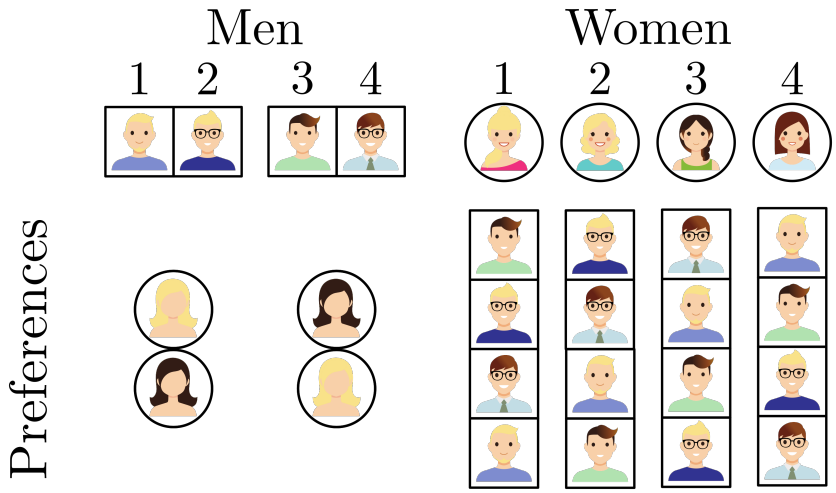


Figure 3.1: Example of a population with four men and women. In the top row are represented all men and women that could appear in the population. Below are represented their preference lists. There are two types of agents, men (squares) and women (circles) and two sub-types, blond and brunet. According to preference lists, men chose according to women hair colour. This defines two (sub-)types of men, those preferring blondes (blonde men) or those preferring brunettes (brunet men). Women's preferences are more complex since they take into account not only the hair colour but also whether the men wear glasses or not. A person will prefer to interact with another person that is on top of their preference list

---

terminates and new marriages are created.

Inside the population defined in Fig. 3.1 there are several possible combinations of men and women that will lead to different dynamics. In Fig. 3.2(a) the preferences of men 2 and 4 and women 2 and 3 are illustrated. If these people now interact, they will form two long lasting marriages, as seen in Fig. 3.2(b). Both men and both women prefer to be paired with their current partner rather than the others (within this choice of population).

If another sub-set of men and women is chosen, the result may be different. In Fig. 3.3(a), the men 1 and 3 and the womae 1 and 4 were selected and their preferences shown. In Fig. 3.3(b), their interactions are depicted and in this case the dynamics of their interactions is different than the one in Fig. 3.2(b). Starting with marriage 1, the blond man prefers the blond woman but she does not prefer him, so when the brunet man appears, she terminates her previous marriage and gets married with the brunet man, marriage 2. Now, the brunet man is the one that is not happy and he prefers the brunette woman, so he terminates his marriage and begins a new one with the brunette woman. This dynamics continues to happen indefinitely because at least one of the person interacting is going to prefer some other person instead of its current partner. In this case, all the marriages will be short lived and the dynamics is frustrated. However, if now a new man appears, one that none of the women have seen before, their preference towards him can be any. They may prefer to partnered with him rather than the other men or they may not. If they do prefer to form a marriage with this new man, the dynamics is disrupted and there will be a long lasting marriage. It can be easily seen, that a frustrated dynamics can be created by choosing carefully the population or by changing the women's preferences list.

A frustrated dynamics is exactly what we look for in cellular frustration systems. A

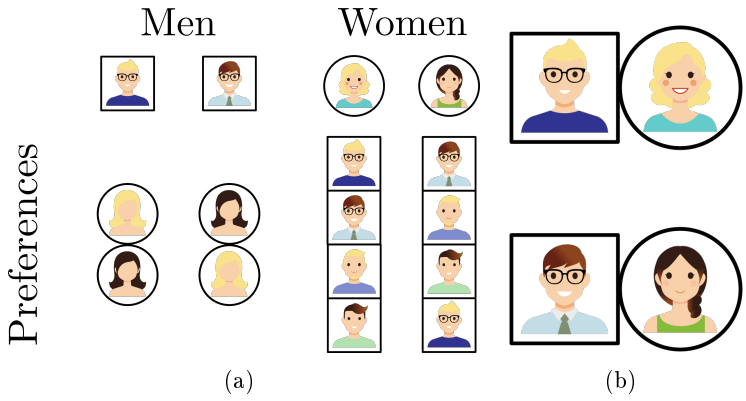


Figure 3.2: Particular instance of the population with only some of the elements that can exist in the population. According to the preference lists, stable marriages appear involving blond and brunet men and women. This is an example of a population that reaches a stable matching. In (a) their preferences list are displayed and in (b) their interactions. The blond man prefers to be married with the blond woman and the blond man is on top of her preferences list, so they form a long lasting marriage. The same is true for the brunet couple. With this choice of men and women, the solution is stable.

---

dynamics that is able to have short lived pairings but also long lived pairings depending on the agents that are presented. This is particularly useful in anomaly detection. Instead of using men and women as agents, we use presenters and detectors. Presenters display information from a sample in a data set, and detectors read this information and take pairing decisions accordingly. In a data set with normal and anomalous samples, the normal samples will have short lived interactions while the anomalous samples will have long lived interactions.

3.2 Cellular Frustration Algorithm for anomaly detection

In a data set with normal and anomalous samples, one sample is presented at a time to the system, each presenter displaying a feature value. For a data set with M samples and N features x_i, $i = 0, ..., N$, there will be N presenters P_i and N detectors D_i. If the number of features is too small to create the necessary dynamics, the number of presenters can be increased and different presenters would display same feature values. Also, if the number of features is too large, then a pre-processing can be applied for dimensionality reduction.

All the presenters will interact with all the detectors in pairs. When two agents, P_i and D_j, interact, they both make a decision: should they form a new pair -in which case they would terminate former pairings, if they existed- or should they stay in their current pairing. Each agent takes these decisions using an interaction lists (ILists) the equivalent to the preference lists in the SMP - where the signals delivered by agents of the opposite type are ranked in descending order of priority. All agents will continuously attempt to be paired with agents of the opposite type that deliver signals that are ranked in higher positions in their ILists.

The CFF divides presenters and detectors in two sub-types, I and II, with $\frac{N}{2}$ agents each. Going back to the SMP example, the sub-types were whether a person was blond or brunet. It is assumed that detectors only display a single signal: their sub-type. This establishes that there are only two types of detectors. Presenters of sub-type I (II) rank in the top position the

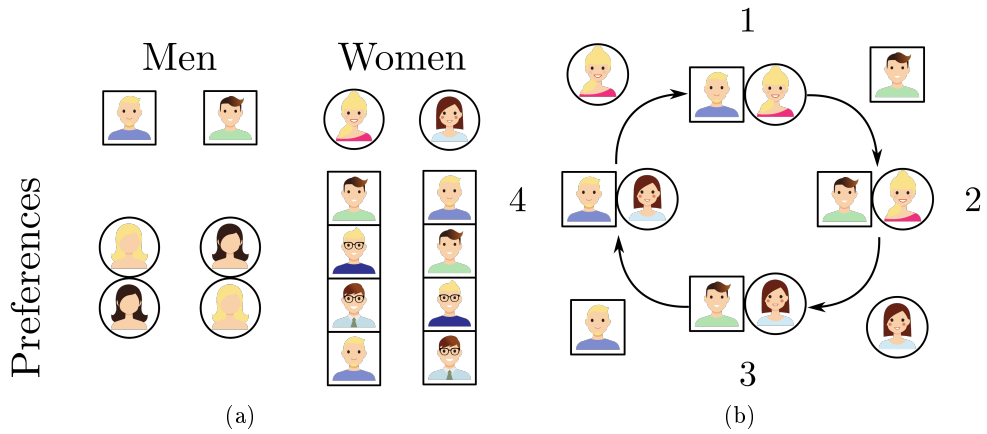


Figure 3.3: Particular instance of the population for which no stable matching can be achieved. In (a) their preferences list are displayed and in (b) the possible marriages within this population are represented. The pairing dynamics shows that for all couples there will always be an agent that can destabilise (frustrate) formed marriages. In marriage 1, the blond man prefers to interact with the blond woman, but the blond woman does not. When a brunet man appears, she is going to prefer to terminate her previous marriage and start a new one with the brunet man. However, now in marriage 2, the brunet man is the one that prefers to be with the brunette woman rather than the blond woman. So he is going to terminate the interaction with the blond woman and start a new one with the brunette woman. And so on, one of the agents in the interaction will prefer to interact with some other agent from the population instead of their current partner. This creates a frustrated dynamics where all the interactions are shortly lived.

---

signal delivered by sub-type I (II) detectors. By contrast, since the information displayed by presenters is larger, the way each detector prioritises its pairings is more complex, requiring a list with many positions to rank the different perceived signals. In Figs. 3.4 and 3.5 the agents are depicted, where the presenters are represented by squares and the detectors by circles. The agents of sub-type I are white and of sub-type II are grey.

Here we follow [29] and assume that each detector can only perceive a binary signal - r_i or f_i - from the information displayed by presenter i. However, before turning the signal into r_i or f_i, there is the need to guarantee that different presenters display distinct information. The information x_i is turned into a signal s_i:

$$s_i = i + \frac{x_i - min(x_i)}{max(x_i) - min(x_i) + \epsilon} \tag{3.1}$$

where $max(x_i)$ and $min(x_i)$ represent the maximum and minimum value of feature i in the entire data set, and ϵ is a small number to ensure that different presenters display disjoint information.

This simplifies ILists and their ordering. Still, the length of the detectors ILists will be $2N$, since each presenter can have two different signals. The mapping from the s_i signal to the f_i or r_i signal perceived by a detector is made so that signals f_i and r_i are perceived frequently and rarely, respectively. Hence, during training the signals perceived by detectors are mostly f signals.

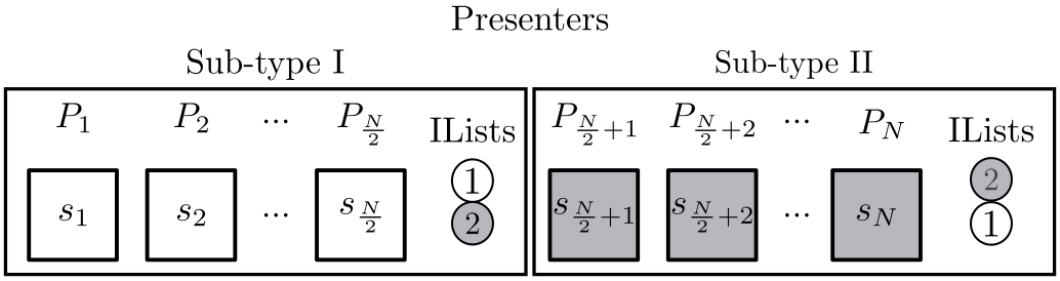


Figure 3.4: There are two sub-types of presenter agents. Presenters are represented by squares and agents of sub-type I are white and of sub-type II are grey. Sub-type I presenters prioritise interactions with detectors of the first sub-type. The opposite happens with presenters of the other sub-type. Each presenter displays a signal s_i. Each of these signals will be perceived by each detector as a binary signal, f or r, depending on how frequent they are displayed.

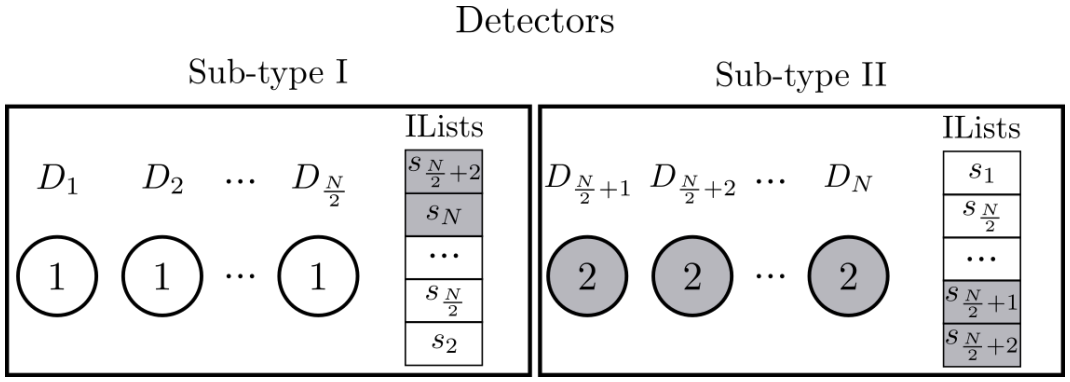
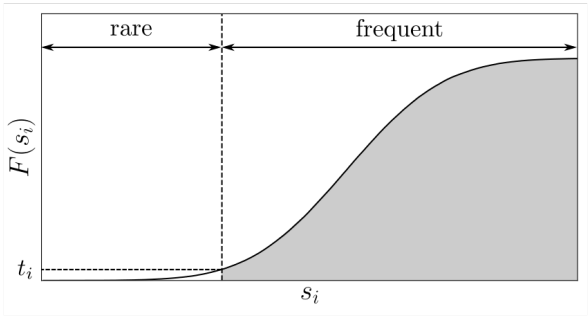
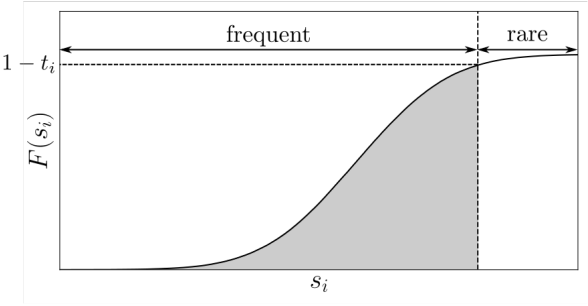


Figure 3.5: Detectors are represented by circles and agents of sub-type I are white and of sub-type II are grey. There can be many different kinds of detector's ILists. In this figure only two kinds are represented, one for each type. In these cases, the ILists already went through the training process and the IList from detectors of sub-type I will have on top the signals from presenters of sub-type II and the ILists from detectors of sub-type II will have on top the signals from presenters of sub-type I. Each detector displays as signal that depends only on their sub-type (I or II).



(a)



(b)

Figure 3.6: Mapping established by each detector between the signal displayed by each presenter and the signal perceived (f or r, for frequent or rare). The cumulative distribution $F(s_i)$ is computed for each feature with all the training samples. Then, the definition of a rare signal can be made either on the left side of the distribution (a), or on the right side (b). Each detector chooses randomly for each feature. The value of the threshold probability t_i is also random between a certain interval, usually $0 < t_i < 0.2$.

The discussion on how to create the f and r signals is the main subject of this thesis. It is proposed the usage of a clustering algorithm to do this mapping in order to improve the anomaly detection rates. However, for now, the original mapping will be presented [29]. The discussion on how to use clustering techniques and why is it relevant, will be done in next chapters.

Each detector first computes the cumulative distribution for each feature i, $F_i(s)$, with all the samples available for training. Then, a threshold probability is defined, t_i, where $0 < t_i < t_{max}$ and, usually, $t_{max} = 0.2$. The rare signals could be located on the left, Fig. 3.6(a), or right side, Fig. 3.6(b), of the distribution for which $F_i(s) < t_i$ or $F_i(s) < 1 - t_i$ respectively.

Note that, how the detectors map the samples' information into rare or frequent signals has an impact in the anomaly detection rate.

With all these definitions set, the frustrated dynamics can be easily understood. Presenters and detectors will interact with one another in pairs. When they interact, they need to make a decision, whether to create a new paring, and terminate an previous one, or to stay with their current pair. This decision is made with the aid of the ILists. Presenters will always prefer to interact with detectors of same sub-typeas themselves, while detectors will have more complex ILists. For example, presenter P_i and detector D_j are in a pair. When a presenter

P_k appears, such that its signal s_k is ranked higher in the D_j IList than the signal s_i, the detector D_j will terminate its current pairing and start a new pair with presenter P_k. The Pseudo-code for the decision function is shown in Pseudo-code 3.

Algorithm 3 Function that determines pairing decisions when presenter P_i and detector D_j interact. Both agents take a decision based on each agents' IList. The position of the signal s_i in agent j is denoted by $rank_j(s_i)$. Note that an agent prefers to interact with agents displaying signals that are higher on its IList and that a higher position means a lower rank number.

```
1: function DECISION(i,j,{P_n},{D_n},{s_i})
2:     if P_i and D_j are alone then
3:         Pair P_i and D_j
4:     else if P_i is paired with D_j then
5:         if P_k is alone and rank_j(s_k) < rank_j(s_i) then
6:             Unpair P_i and D_j
7:             Pair P_k and D_j
8:             Set τ_i and τ_j to 0
9:         else if D_l is alone and rank_i(s_l) < rank_i(s_j) then
10:            Unpair P_i and D_j
11:            Pair P_i and D_l
12:            Set τ_i and τ_j to 0
13:    else if P_i is paired with D_j and P_k is paired with D_l then
14:        if rank_i(s_l) < rank_i(s_j) and rank_l(s_i) < rank_l(s_k) then
15:            Unpair P_i and D_j
16:            Unpair P_k and D_l
17:            Pair P_i and D_l
18:            Set τ_i, τ_j, τ_k and τ_l to 0
19:        else if rank_j(s_k) < rank_j(s_i) and rank_k(s_j) < rank_k(s_l) then
20:            Unpair P_i and D_j
21:            Unpair P_k and D_l
22:            Pair P_k and D_j
23:            Set τ_i, τ_j, τ_k and τ_l to 0
```

3.2.1 Training

To obtain a system capable of performing accurate anomaly detection, the cellular frustrated system must go through a training stage that is composed by an education and a calibration stage. During training, only normal (non anomalous) samples are presented and detectors are changed to increasingly frustrate the system. The important quantity to measure, which is characteristic of the system's dynamics, is the agents' pairing lifetime. Nevertheless, measuring directly pairings lifetimes can be difficult, so the CFF proposes measuring instead the fraction of pairs that last longer than a certain time. In this stage, ILists are changed when a detector is paired for a time longer than an established threshold τ_n.

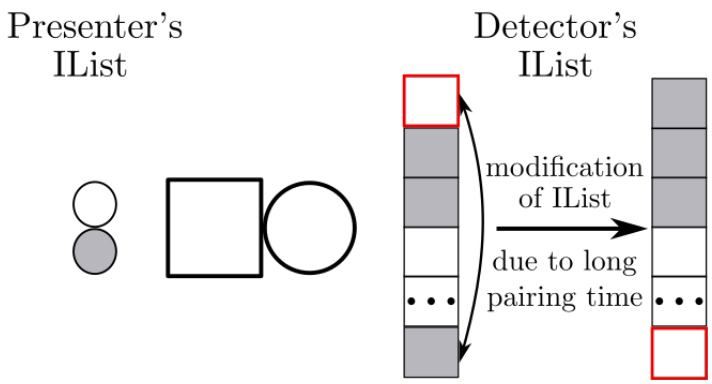


Figure 3.7: Representation of the corrections introduced in a detector's IList. A presenter and a detector of the same sub-type (in this case, sub-type I) create a pair that lasts longer than τ_n because they both prioritise this pairing. A modification in the IList should be introduced to avoid producing these long lived interactions, by swapping the signal that led to this stable pairing, represented by a red box, with a random signal ranked in any position below.

Education

Initially, frequent and rare signals from all presenters are randomly ranked in detectors ILists. Then, the iterative frustrated dynamics begins. Agents interact randomly with agents of the opposite type and a pair is formed if the two agents involved prioritise the interaction relatively to their current matches, as described in Pseudo-code 3. This process repeats itself until all agents are given a chance to choose an agent of the opposite type to interact with. Education (correction of a detector IList) takes place when a pairing lasts longer than τ_n. Then the signal from the detector's IList that led to this long pairing time is exchanged with a randomly drawn signal from a lower position in its IList, as represented in Fig. 3.7. This approach pushes signals from presenters of the same sub-type to lower positions on detectors ILists, since they are the ones that create the longest pairing durations.

Pseudo-code 4 describes how to educate ILists. Education stops when τ_n reaches a pre-defined minimum value, τ_{min}. The decision dynamics described in Pseudo-code 3 is run for every presenter and detector. When all agents have interacted with all agents of the opposite type, the duration of these interactions τ_i will be evaluated. If $\tau_i \geq \tau_n$ then the signal of the IList of detector i that led to the longer pairing will be swapped with a random signal that has a lower position in the IList. If after a certain number of iterations, W_e, none of the pairings exceed τ_n, thus $N_{subs} = 0$, then τ_n is updated to the largest pairing duration (line 22). Every T_s iterations, the sample displayed by presenters is also changed.

In this work, the same values for the user-defined parameters were used for all the cellular frustration systems. In the last iteration of the education stage, the minimum value for the largest paring time after education, τ_{min} is equal to 100 iterations. T_s should have a small enough value to guarantee that during education the agents see many samples. This way, they have to integrate the information of many samples at the same time. T_s was taken as 100 iterations. W_e should be sufficiently large to be immune to statistical fluctuations and has the value of 10000 iterations.

When the system is properly educated, on the top of the detectors ILists there will be mostly signals displayed by presenters of the opposite sub-type. Fig. 3.8 illustrates the

Algorithm 4 Education of detectors' ILists.

Initialise D_i with a random IList with f and r signals
Initialise $\{\tau_i\} = 0$
Initialise $\tau_n = W_e$
while $\tau_n > \tau_{min}$ **do**
    Initialise $N_{subs} = 0$
    Initialise $\tau_n^W = 0$
    for t_w in 1 to W_e **do**
        if t_w **mod** $T_s = 0$ **then**
            Display signals from anothe sample $\{s_i\}$
        for all P_i and D_j **do**
            P_k : randomly selected presenter
            DECISION(k,j,$\{P_n\}$,$\{D_n\}$,$\{s_i\}$)
            D_l : randomly selected detector
            DECISION(i,l,$\{P_n\}$,$\{D_n\}$,$\{s_i\}$)
        for all D_i **do**
            if $\tau_i \geq \tau_n$ **then**
                P_k : agent paired with D_i
                r = random integer larger than $rank_i(s_k)$
                In D_i IList swap the sinals with $rank_i(s_k)$ and $rank_i(s_r)$
                Unpair D_i
                Set τ_i to 0
                $N_{subs} = N_{subs} + 1$
        $\tau_n^W = max(\tau_i, \tau_n^W)$
    if $N_{subs} = 0$ **then**
        $\tau_n = \tau_n^W$

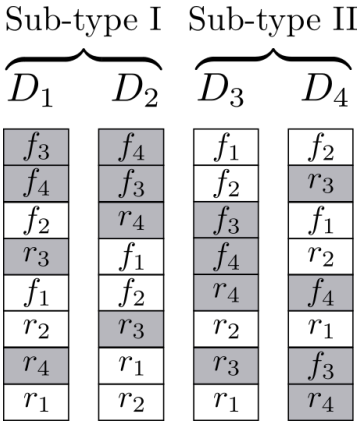


Figure 3.8: Detectors ILists after education, with $N = 4$. Detectors of sub-type I (D_1 and D_2 will have on top of their ILists frequent signals displayed by presenters of sub-type II (grey background). The detectors of the other sub-type (D_3 and D_4) will have on top frequent signals from presenters of sub-type I. Only the top tanked positions tend to be educated, i.e.,avoid having on top positions signals displayed by presenters of the same sub-type. However, there can be small mistakes, here represented by signal f_2 in D_1 IList or signal r_3 in D_4 IList.

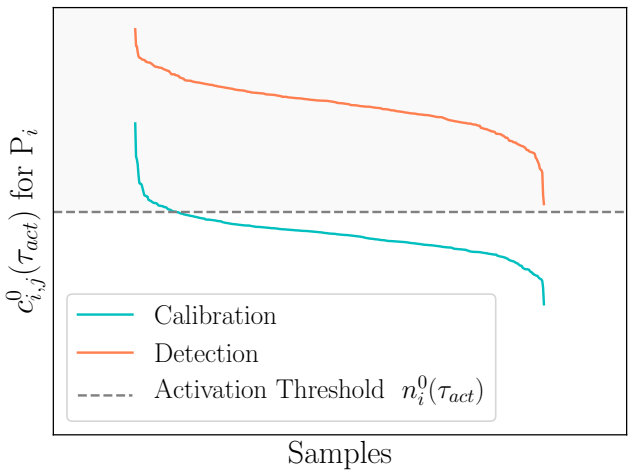


Figure 3.9: Representation of the ordered vector, $c^0_{i,j}(\tau_{act})$, of the number of pairs that last longer than τ_{act} for the same P_i and all the samples, for a typical cellular frustration system. In the figure, two ordered vectors are presented: one for the calibration samples (blue line) and the other for the detection anomalous samples (red line). In this example, the detection samples are mostly composed of anomalous samples. In the calibration stage the ordered vectors are calculated for each detector and the activation threshold is defined, defined by the dashed grey horizontal line. In a latter stage, the detection stage, all the samples that have an activation higher than the threshold (light grey area) will contribute for the final response of the system.

---

impact of training in ILists for a data set with only 4 features ($N = 4$).

Calibration

After education, there is a calibration stage. The decision dynamics is run as described in Pseudo-code 5. Pairings that last longer than a certain amount of time τ_A are terminated. The decision dynamics is run for W iterations for each sample. Afterwards, the number of pairs, $c^0_{i,s}(\tau_{act})$, that last longer than an activation lifetime τ_{act}, involving the presenter P_i when sample s is displayed, is registered. Here τ_{act} is chosen to be equal to the largest pairing time in the calibration, so $\tau_{act} = \tau_A$.

An ordered vector, $c^0_{i,j}(\tau_{act})$, of the number of pairs that last longer than τ_{act} for the same P_i and all the samples can be defined as $c^0_{i,j}(\tau_{act}) \geq c^0_{i,(j+1)}(\tau_{act}), \forall j$. With this ordered vector, an activation threshold can be established as $n^0_i(\tau_{act}) = c^0_{i,x}(\tau_{act})$ where $x = N_s \times \nu$ with N_s being the number of samples used during training and ν a real number between 0 and 1. The ordered vector $c^0_{i,j}(\tau_{act})$ is represented in Fig. 3.9 by the blue line and the activation threshold $n^0_i(\tau_{act})$ by the dashed grey line.

W needs to be large such that it has robust statistics. In this work it has a value of $W = 10000$ iterations. τ_A needs to be sufficiently large to guarantee that a long lived interaction took place. Still, $\tau_A = 5$ is enough, because during 5 iterations, on average, each agents has 10 encounters and a decision has to be taken to terminate or not its current pairing. The probability that a pairing lasts 5 iterations is already small and therefore the number of pairings that last this long can be considered as long lived [2]. Usually $\nu = 0.1$, thus the

Algorithm 5 Calibration and detection stage. The same algorithm is run for both stages with few modifications. For the detection stage the ordered vector of the number of pairings that last longer than τ_{act} is $c_{i,s}(\tau_{act})$ and not $c^0_{i,s}(\tau_{act})$ as in the calibration stage. In the calibration stage only the training samples are presented. In the detection stage both normal and anomalous samples from the testing set are presented. Normal samples will lead to low pairing times while anomalous samples will lead to high paring times.

Initialise $\{\tau_i\} = 0$
Initialise $c_{i,s} = 0$
for t_w in 1 to W **do**
    for all P_i and D_j **do**
        P_k : randomly selected presenter
        DECISION(k,j,$\{P_n\}$,$\{D_n\}$,$\{s_i\}$)
        D_l : randomly selected detector
        DECISION(i,l,$\{P_n\}$,$\{D_n\}$,$\{s_i\}$)
    for all D_i **do**
        if $\tau_i \geq \tau_{act}$ **then**
            P_k : agent paired with D_i
            Unpair D_i
            Set τ_i and τ_k to 0
            $c^0_{i,s}(\tau_{act}) = c^0_{i,s}(\tau_{act}) + 1$
            $c^0_{k,s}(\tau_{act}) = c^0_{k,s}(\tau_{act}) + 1$

calibration threshold is the 10% largest number of pairings that last a time longer than τ_{act}.

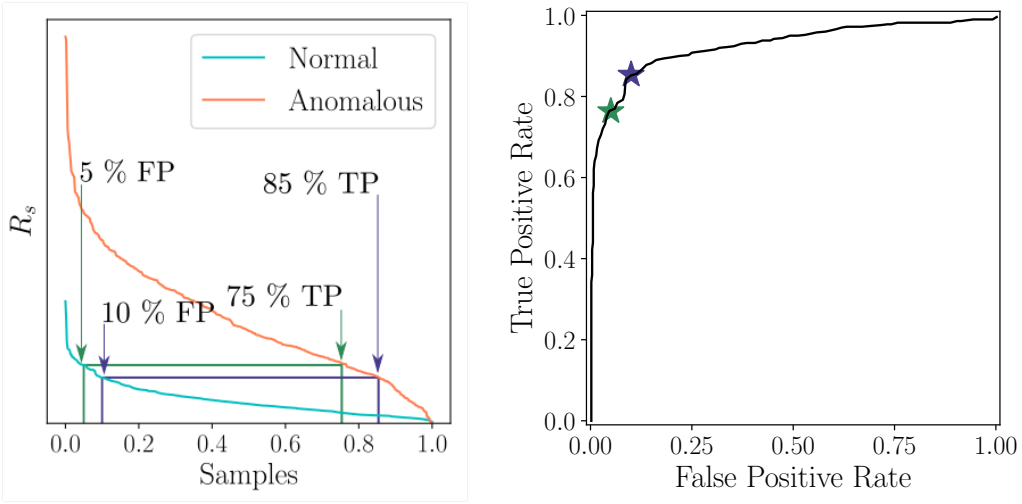
3.2.2 Detection

In the detection stage, both normal and anomalous samples are presented and the decision dynamics is the same as the one used in the calibration stage, that is described in Pseudo-code 5. The only difference is that in the detection stage the ordered vector of the number of pairings that last longer than τ_{act} is $c_{i,s}(\tau_{act})$ and not $c^0_{i,s}(\tau_{act})$ as in the calibration stage. The total response to the information displayed by a sample s from the system is computed using the normalised number of pairings $\tilde{c}_{i,s}(\tau_{act}) = \frac{c_{i,s}(\tau_{act})}{c_{i,s}(0)}$ and the normalised activation threshold $\tilde{n}_i(\tau_{act}) = \frac{n^0_i(\tau_{act})}{n_i(0)}$. The response of the cellular frustration system is given by the sum of the deviations to the typical responses estimated in the calibration stage, being given by:

$$R_s = \sum_i^N \Big(\tilde{c}_{i,s}(\tau_{act}) - \tilde{n}_i(\tau_{act}) \Big) \times \theta\big(\tilde{c}_{i,s}(\tau_{act}) - \tilde{n}_i(\tau_{act})\big) \tag{3.2}$$

where θ is the Heaviside function. The region where the Heaviside function is non-zero is shown in Fig. 3.9 by the light grey area above the activation threshold. All samples that have an activation lower than the threshold will not contribute to the global response. Thus, the CFS response sums the increments on the number of long pairings relatively to the calibration stage.

The goal of a good anomaly detection system is to produce extremely different responses depending on whether samples are normal (similar to those used for training; these samples



(a) (b)

Figure 3.10: An example on how to create a ROC curve from the total response of a cellular frustration system. In (a), the total responses for the testing set are represented by a blue line for normal samples and a pink line for anomalous samples. In the x axis of this plot is the percentage of samples. By choosing the total response of the system for the normal samples at a certain percentage, one can find the percentage of how many anomalous samples would be rightly classified as anomalous. This is the methodology used to create a ROC curve. The green (purple) lines represent this method for 5% (10%) of false positive rate and the corresponding true positive rate. In (b), the corresponding ROC curve is pictured. The green and purple stars represent the true positive rate at 5% and 10% respectively.

---

should be tolerated, i.e., not trigger any effect) or anomalous (very different from any sample seen during training; these samples should be pointed by the detection system). In CFSs, normal samples should frustrate considerably the dynamics, and therefore the total response of the system should be weak, compared to the response produced when anomalous samples are presented. Fig. 3.10(a) shows and example where this discrimination is clear. In the x axis is the number of samples in percentage. Notice that the anomalous samples (pink line) have higher total responses than the normal samples (blue line). In the same figure, 3.10(a), it is illustrated how to create a ROC (Receiver Operating Characteristic) curve. A ROC curve is a plot that illustrates the anomaly detection rate of an algorithm as a threshold is varied. In the x axis has the false positive rate and in the y axis the true positive rate. In our case, for 5% of false positive, i.e.,for 5% of the normal samples being classified as an anomalous sample, there is 75% of true positive rate. In our data set, for 5% of misclassifications of the normal samples there are 75% of rightly classified anomalous samples. This is represented in Fig. 3.10(a) by the green lines. The same is demonstrated for a 10% of false positive rate. In this case, there is a 85% true positive rate and it is depicted in the figure by the purple lines. In Fig. 3.10(b), the whole ROC curve is depicted. This curve was created by varying the percentage of the false positive rate and finding the correspondent true positive rate.

Mechanisms of detection

CFSs detect 3 types of anomalous patterns when an anomalous sample is displayed to the cellular frustration system. The appearance of rare signals that were not seen during training, the absence of frequently displayed signals during training and the absence of combinations of frequently displayed signals during training.

In Fig. 3.11, the three mechanisms of detection are displayed. In Fig. 3.11(a), the ILists from detectors D_1, D_2 and $D_{\frac{N}{2}}$ are presented. These ILists already went through the training stage. In Fig. 3.11(b), the anomalous sample in question, has a rare signal that was never seen during training, s_3. The detectors do not know in which position they should rank this signal, so they rank it randomly. It could happen that it will be on top of their IList, D_1 and $D_{\frac{N}{2}}$, or not, D_2. If is in high positions, this signal will lead to longer pairings, will flag this sample as anomalous [2, 29]. In Fig. 3.11(c), this anomalous sample, does not have a frequent signal that was seen by the detectors during training, $f_{\frac{N}{2}+16}$. Since this signal is not present, all the other signals that have a lower position will go up a position [29]. This upward shift may cause signals from the opposite sub-type to be in higher positions, leading to larger pairing times [29]. In Fig. 3.11(d), two frequent signals that appeared during training are both missing, $f_{\frac{N}{2}+3}$ and $f_{\frac{N}{2}+6}$. This may happen because of strong correlations between features in the data set. Since the two signals are missing, the other signals will, again, be pushed upwards in the detectors ILists [29]. Signals from the opposite sub-type will now be on higher positions, which will lead to longer pairings.

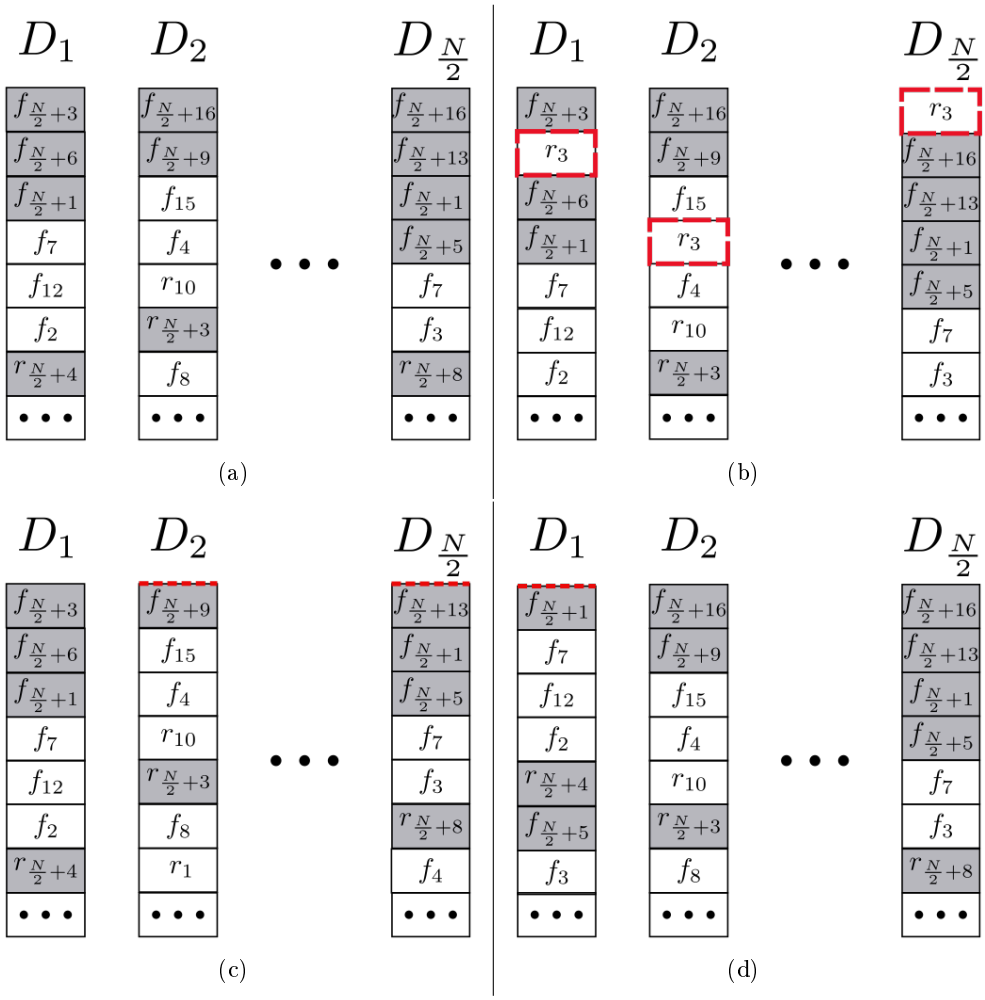


Figure 3.11: Representation of the ordering of detector's interaction lists when anomalies are presented. In Fig. (a), the already trained ILists for detectors 1, 2 and $\frac{N}{2}$ are presented. Notice that these detectors are of sub-type one, so on the top of their ILists are mostly signals from sub-type II (grey). In (b), (c) and (d) the 3 mechanisms of detection are presented. In (b), a rare signal, r_3, that was not present during training is ranked by detectors in random positions. These random signals are represented by the dashed red boxes. Detectors D_1 and $D_{\frac{N}{2}}$ will be able to establish long lived interactions, detecting this way the anomaly. In (c), the presented anomalous sample displays fewer frequent signals than the ones seen during training. Because of this absent signal, $f_{\frac{N}{2}+16}$, the detectors ILists can have on top positions less signals from presenters of the opposite sub-type, leading to longer interactions. The absent signal is represented by a dashed red line. Note that the signals that are bellow the missing one shift upwards. This is a mild effect but it happens on several detectors at a time. In (d) the detection is triggered if the frequent signals that are absent were never absent together during training, $f_{\frac{N}{2}+3}$ and $f_{\frac{N}{2}+6}$. The shifts towards are stronger in this case, even though it affects a smaller number of detectors. Again, the absent signals are represented by dashed red lines.

Chapter 4

Clustering techniques and the Cellular Frustration system

In the previous chapter the mapping from the values displayed by presenters, to the binary (frequent or rare) signals perceived by detectors was defined. This mapping used distributions of the whole data set. In this chapter a new mapping will be presented using clustering techniques. The premise is that even inside the set of normal samples there could be patterns (or clusters). The application of a clustering technique to map feature values into rare and frequent signals could help the cellular frustration algorithm achieve better anomaly detection rates because the space where anomalous samples could lay would be enlarged.

In the 2D example of Fig. 4.1, the normal samples are the blue data points and the anomalous, the red points. If the CFS without clustering is applied, the mapping using tails with 5% of the samples ($t_i = 0.05$) is pictured in solid black lines and it contains most of the anomalous samples. Within this kind of mapping, usually, there are big regions without any normal samples and where anomalies could arise.

If a clustering technique is performed prior to the CFS, the clusters will be defined and then, the mapping of signals into rare or frequent signals will be performed twice, once for each cluster, obtaining the dashed black lines in Fig. 4.1. Notice that now all the anomalous samples are outside the defined normal space (composed by the two sub-spaces).

With this in mind, a new clustering technique was created to suit the needs of the CFS. In this chapter, the extremes clustering technique will be explained as well how to integrate the clustering with the cellular frustration algorithm.

4.1 Extremes Clustering Technique

In this thesis, a new clustering technique is discussed, denominated by extremes clustering technique (ECT). This technique was created with the underlying assumption that it should improve an algorithm that discriminates samples that have more extreme feature values comparing to samples considered normal.

Very often different features exhibit correlations characteristic of the different states or configurations of a system. These correlations should be particularly noticeable when features take extreme values. The idea of the ECT is to use these extreme values to obtain information about how different samples should be grouped in clusters.

The ECT starts by using the marginal cumulative distribution function $F(x_i)$ on feature

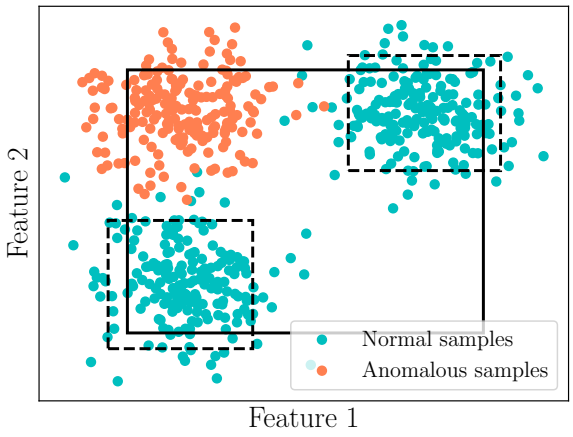


Figure 4.1: A 2D example where implementing a clustering technique prior to the CFS would be beneficial. The normal samples (blue points) distinctly have two clusters, so mapping the space with accumulative distributions of the whole normal sample space would create a mapping that most of the anomalous samples (red points) belong to (solid black lines). Using a clustering technique first, the space can be divided into two sub-spaces (dashed black lines) and the anomalous samples are on the outside of it.

---

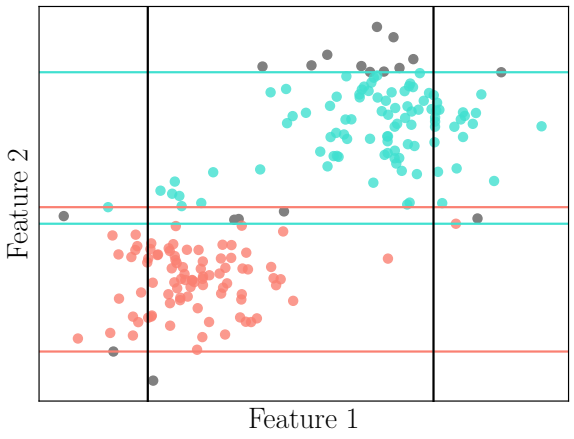
i, for the samples displayed during training and finds which samples are in a right or left tail with a pre-defined probability, p, typically from 0% to 10% (see the vertical black lines in Fig. 4.2(a)). With these samples, two disjoint sets, one for each cluster, $S_{1,i}$ and $S_{2,i}$, are created in feature i. Samples belonging to each set will, typically, have values in ranges not shared by another features. All the samples that are inside the range defined by the previous samples in other feature will be added to the corresponding set, $S_{1,ij}$ or $S_{2,ij}$ (blue and pink samples in Fig. 4.2(a)). If the two sets overlap, $S_{1,ij} \cap S_{2,ij} \neq \varnothing$, only the unique part of each set will be considered. When more features exist, this procedure is repeated for all pairs of features producing potentially $2(N(N-1))$ sets of samples. To properly relate the several sets, each time a couple of new sets is found, they are labelled with the appropriate set number using the information from a correlation matrix of the features. Afterwards, a score is associated to each sample for each set, $c_{1,s}$ and $c_{2,s}$, as follows:

$$c_{n,s} = \sum_i \sum_j \#\{s \wedge S_{n,ij}\} \quad n = 1, 2 \tag{4.1}$$

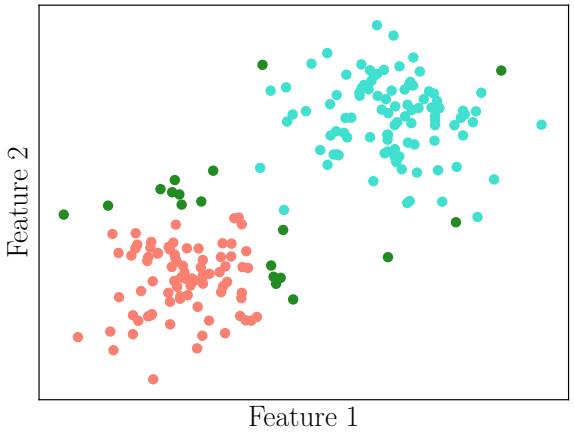
For a sample s, this score represents how many times it has been associated to each cluster. Depending on the samples score, the two clusters are defined. If a sample as null score, the algorithm can not assign it to any of the clusters (green samples in Fig. 4.2(b)). The Pseudo-code of this clustering algorithm is in 6 and the final result is depicted in Fig. 4.2(b).

4.2 Integrating responses from two CFSs

With the clustering technique described in the previous section, samples were divided in two different sub-sets (clusters). A CFS was associated to each sub-set, and frequent and rare signals were defined for all detectors and each system separately using only samples from its



(a)



(b)

Figure 4.2: Extremes clustering technique with the toy model. In (a), the procedure of the clustering technique is presented. The black lines represent the limits of left and right tail of the cumulative distribution function in feature 1. Then in each tail, the sample with minimum and maximum value in the second feature will impose the limits of the set. In this figure they are represented by the blue and pink horizontal lines. All the samples that are within these limits and belong to only one interval will be added to the corresponding set. In (b), the final cluster are displayed with different colours, blue and pink. The samples that the algorithm was not able to assign to a cluster are depicted in green. This is a fairly simple example with two features, so only four sets were created. In a data set with more features the extremes clustering algorithm would be able to detect more correlations between features and produce better clusters.

Algorithm 6 Extremes clustering algorithm

Compute the correlation matrix between all features $corr(x_i, x_j)$
Identify the feature X with stronger correlations
for i in N **do**
    Compute $F(x_{i,s})$ with all the training samples
    for s in training samples **do**
        if $corr(X, x_i) > 0$ **then**
            if $0 < F(x_{i,s}) < p$ **then**
                Add s to $S_{1,i}$
            else if $1 - p < F(x_{i,s}) < 1$ **then**
                Add s to $S_{2,i}$
        else if $corr(X, x_i) < 0$ **then**
            if $0 < F(x_{i,s}) < p$ **then**
                Add s to $S_{2,i}$
            else if $1 - p < F(x_{i,s}) < 1$ **then**
                Add s to $S_{1,i}$
    for j in N **do**
        for s in training samples **do**
            if $min(x_{j,S_{1,i}}) < x_{j,s} < max(x_{j,S_{1,i}})$ **and** $\big(x_{j,s} < min(x_{j,S_{2,i}})$ **or** $x_{j,s} > max(x_{j,S_{2,i}})\big)$ **then**
                Add s to $S_{1,ij}$
            else if $min(x_{j,S_{2,i}}) < x_{j,s} < max(x_{j,S_{2,i}})$ **and** $\big(x_{j,s} < min(x_{j,S_{1,i}})$ **or** $x_{j,s} > max(x_{j,S_{1,i}})\big)$ **then**
                Add s to $S_{2,ij}$
for s in training samples **do**
    for k_1 in $S_{1,ij}$ **do**
        if s in k_1 **then**
            $c_{1,s} = c_{1,s} + 1$
    for k_2 in $S_{2,ij}$ **do**
        if s in k_2 **then**
            $c_{2,s} = c_{2,s} + 1$
for s in training samples **do**
    if $c_{1,s} - c_{2,s} > 0$ **then**
        Add s to Cluster 1
    else if $c_{2,s} - c_{1,s} > 0$ **then**
        Add s to Cluster 2

cluster. Likewise, each CFS was educated using samples from its sub-set. Both CFSs were run until τ_n reached the same value. Accordingly, samples from cluster 2 will be seen by system 1 as anomalous and vice-versa.

The goal is that the global system - formed by the two CFSs - should produce a weak response, whenever any of the two CFSs produces a weak response (it would mean that the sample is seen as belonging to the normal sub-set of samples from that CFS). If none of the two CFSs produces a weak response then this would imply that the sample is seen as anomalous by both CFSs and therefore should produce a strong response of the global system.

However, since both CFSs were educated separately, their dynamics can be considerably different. Consequently, it is expectable that their responses can be different in magnitude and variability. In order to make the two responses comparable we introduced a normalisation of both responses as given by:

$$\tilde{R}_{s,i} = \frac{R_{s,i} - \mu_i}{\sigma_i} \quad i =, 1, 2 \tag{4.2}$$

where μ_i is the mean and σ_i the standard deviation of the responses obtained by evaluating the samples of the other cluster rather than the cluster that the system was trained with.

In order to define a global response with the properties defined above, we established that the global response, R_s^g, of the system is given by the minimum of the responses from each CFS:

$$R_s^g = \min(\tilde{R}_{s,1}, \tilde{R}_{s,2}) \tag{4.3}$$

In this way, whenever normal samples are presented, one of the two systems produces a small response due to the frustrated dynamics, as required. In principle, anomalous samples will not frustrate the dynamics of any system and therefore producing strong response results.

The whole procedure is summarised in Pseudo-code 7.

Algorithm 7 Entire procedure

1: Use a clustering technique to compute the two clusters
2: Create two separate systems
3: **for** each system **do**
4: Map the space with accumulative distributions and define frequente and rare signals
5: Run the cellular frustration algorithm
6: Normalise the response of both systems and for each sample chose the smallest one

Chapter 5

Results

In this chapter the cellular frustrated algorithms will be tested using two types of data sets. Initially we will consider synthetic data sets: data sets generated with known distributions. Synthetic data sets have several virtues. First, they can be manipulated and therefore, the anomaly detection performance of the algorithm can be tested in very different scenarios. This is particularly relevant when algorithms have an intrusion detection application in mind. In this case, any vulnerability could be explored by the intruder, and hence it becomes crucial to gain knowledge of how the detection system behaves in every specific context. A second set of tests will use a real data set. In the studies included here, the chosen data set is related to quality wine evaluations, and it was chosen for two main reasons: the significant number of samples (≈5000) and its practical (and meaningful) relevance. This data set is also interesting because anomalies can be defined in different ways. For instance, one can be interested in detecting excellent wines or, on the contrary, below quality wines.

In all cases, the anomaly detection performance is going to be evaluated with ROC (Receiver Operating Characteristic) curves. In order to evaluate whether the changes in the algorithm improve previous results, the ROC curve for a CFS operating without using any clustering technique is also presented. To further compare the CFS results with other anomaly detection techniques, the ROC curve for a One-Class SVMs is also presented.

5.1 Synthetic data sets

The goal of this section is to demonstrate that the extremes clustering technique improves anomaly detection rates in cellular frustrated systems. The data sets used in this section were generated using combinations of multi-varied Gaussian functions:

$$f(x_i|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \tag{5.1}$$

where μ represents the mean and σ the standard deviation.

Five different sets of samples were generated. All the sets have 11 features. In each feature, three Gaussian functions were used: two for the clusters in the normal samples and one for the anomalous samples. In different features, the parameters for the Gaussian functions were always the same, creating a direct correlation between features. The parameters used for each Gaussian function can be seen in Table 5.1. Each set of samples has 1000 normal samples

Table 5.1: Values of μ and σ used for each Gaussian function and each case.

Case	Samples					
	Normal ($N_n = 1000$)				Anomalous ($N_a = 500$)	
	Cluster 1 ($N_1 = 500$)		Cluster 2 ($N_2 = 500$)			
	μ	σ	μ	σ	μ	σ
1	0.3	0.1	0.4	0.1	0.7	0.1
2	0.48	0.06	0.54	0.06	0.51	0.12
3	0.25	0.06	0.5	0.06	0.75	0.06
4	0.38	0.06	0.64	0.06	0.51	0.12
5	0.25	0.06	0.75	0.06	0.5	0.06

and 500 anomalous samples. Inside the normal samples, there are two clusters, each with 500 samples.

The training set is composed of 500 randomly drawn samples from the normal set while the testing set has the remaining 500 normal samples and the 500 anomalous samples. The five sets of samples are depicted in Fig. 5.1. The normal samples are the coloured data points, pink and blue, where different colours represent different clusters.

Two clusters are found by using the extremes clustering technique. Two separate systems are defined and the definition of rare and frequent signal is made for each detector by using the feature values that lie on the tails of the distributions. For each system, the CFS is run. The responses of each CFS are normalised and the minimum response for each sample is chosen. The procedure used is the one described in Pseudo-code 7. In order to have a comparison, a simple CFS without clustering will also be run for each set of data.

In Fig. 5.1, the mappings for a CFS without clustering (solid black lines) and for a CFS with clustering (dashed black lines) are represented. Both of these mappings have have tails of the same size, i.e., containing the same fraction of the samples available for education. To ease the reading, from now on the mapping made with all the normal samples is going to be called normal mapping and the mapping established with the clusters is going to be called clusters mapping.

To have a comparison with other anomaly detection algorithms, the results for the One-Class Support Vector Machines are also presented. The One-Class SVMs was implemented using the *Scikit Learn* library [30]. As it was seen in Chapter 2, choosing the kernel function can be tricky. Several kernel functions and parameters were tested. The kernel function that produces better results, in all the synthetic models, is the RBF kernel, which is defined with a Gaussian distribution

$$K(\vec{x_i}, \vec{x_j}) = e^{\gamma ||x_i - x_j||^2}. \tag{5.2}$$

The parameter γ defines the width of the Gaussian. Increasing γ, in these toy models, produces better results. However, higher values of γ produce poorly robust results that highly depend on the number of training samples. In order to avoid problems like these, the value of $\gamma = 1/N_f$, where N_f is the number of features, was used. The parameter ν that sets an upper bound on the allowed number of training examples that are considered as outliers, was taken as $\nu = 0.05$. The presented One-Class SVMs ROC curves in this section were all achieved with this kernel function and these parameters.

The first synthetic data set has the two clusters close together while the anomalous and normal samples are disjoint, Fig. 5.1(a). As expected, the mapping with all the normal samples and the mapping with the clusters are similar. Since the anomalous samples are distinct from the normal ones, the anomaly detection rate is very good in both CFSs (with and without clustering). In Fig. 5.2(a), the ROC curves for the CFS with clustering (solid line), CFS without clustering (dashed line) and the One-Class SVMs (dash-dotted line) are presented. In this case, both the cellular frustrated algorithms and the One-Class SVMs perform perfectly and can detect all the anomalous samples.

In the second data set, Fig. 5.1(b), the clusters are identical to the ones from the previous case. The anomalous samples have similar values to the normal samples but with a bigger variance. At first glance, both mappings may seem similar. However, there are some regions, represented in red areas in Fig.5.3, where the anomalous samples are, that will lie outside of the clusters mapping and inside of the normal mapping. Consequently, the true positive rate will be slightly higher in the case of the CFS with clustering.

The resulting ROC curves, Fig. 5.2(b), are in agreement with this hypothesis. The curve for the CFS with clustering is slightly better than the one without. At the same time, the One-Class SVMs anomaly detection rate is identical to the detection rate of the CFS with clustering.

With respect to the third data set, Fig. 5.1(c), the normal samples are composed of two disjoint clusters and the anomalous samples are also segregated from the normal ones. This is a trivial example where all the algorithms should perform excellently. As it was mentioned before, the aim of this section is not only to show in what cases the CFS benefits from having clusters but also that it does not produce worse results than the CFS without clustering. Observing Fig. 5.2(c), it can be verified that both the CFSs and the One-Class SVMs have a perfect detection rate.

In the fourth data set, Fig. 5.1(d), the normal samples have also two disjoint clusters while the anomalous samples have similar feature values as the normal samples but with a bigger variance. It is clear, that the normal mapping contains a bigger number of anomalous samples than the clusters mapping. The consequences of these mappings can be seen in the ROC curves presented in Fig. 5.2(d). The CFS without clustering can barely detect any sample correctly while the CFS with clustering has a higher detection rate. This is a case where the clustering is critical to obtain good anomaly detection rates. The One-Class SVMs has a better performance than the CFS without clustering but still not as good as the performance from the CFS with clustering.

The fifth and last data set, Fig. 5.1(e), is the most drastic one. The normal samples have also two disjoint clusters but now the anomalous samples have feature values between the two clusters. Since the normal mapping contains all the anomalous samples, the anomalous samples will lead to a frustrated dynamics and the CFS without clustering will not be able to detect anything at all. Looking at Fig. 5.2(e) this can be verified. The ROC curve for the CFS without clustering shows that the algorithm can not only not detect anything but it also classifies wrongly normal samples as anomalous. On the contrary, the CFS with clustering has a perfect detection rate as well as the One-Class SVM. This is due to the fact that both of the algorithms take advantage of the existence of clusters and that the normal samples are disjoint with the anomalous samples.

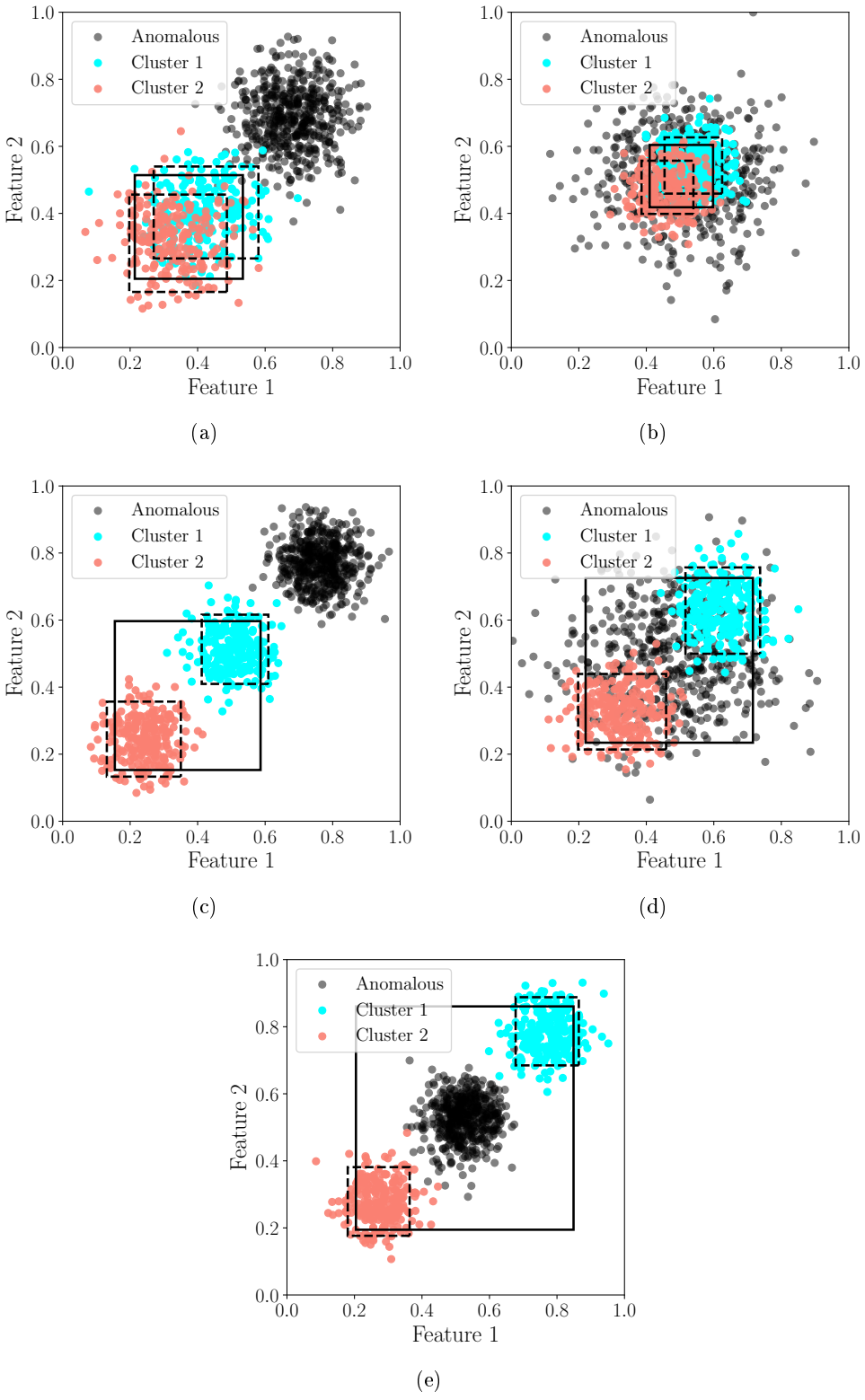


Figure 5.1: Representation of samples from the five sets of synthetic data sets used in this section. In red and blue are represented the sample values for two features from normal samples. Black points show 500 samples values for anomalous samples. Also shown are the tail borders using the original CFS approach (solid black lines) and the CFS with ECT (dashed lines). In this case, tails contain 5% of samples.

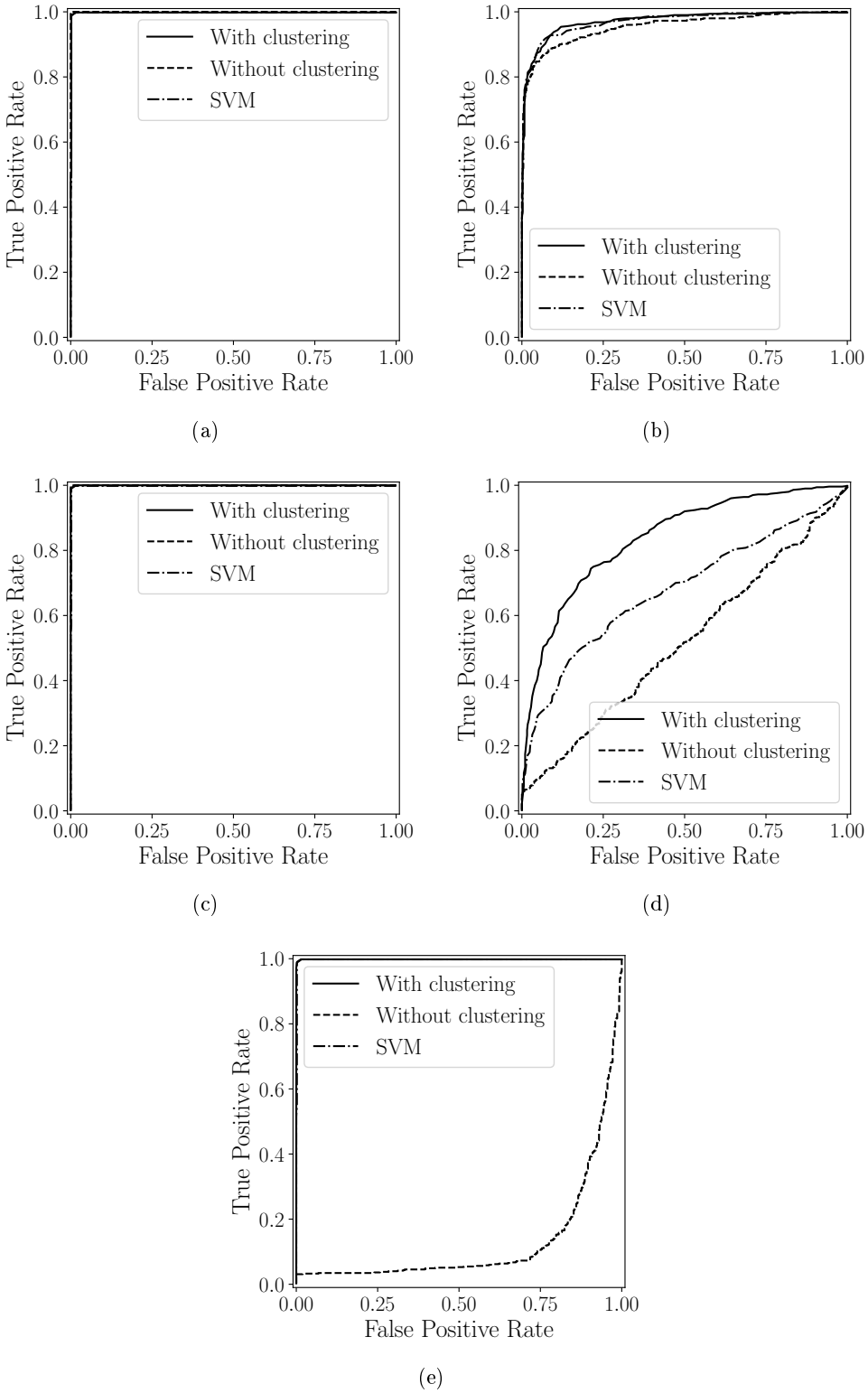


Figure 5.2: Resulting ROC curves for the cases displayed in Fig. 5.1. The solid line represents the ROC curve obtained with the CFS with clustering, the dashed line with the CFS without clustering and the dash-dotted line with the One-Class SVMs.

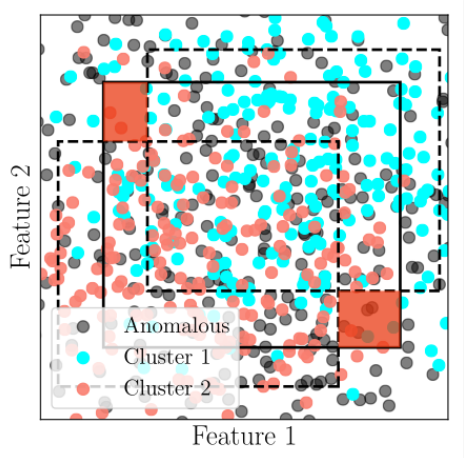


Figure 5.3: Representation of case 2, Fig. 5.1(b), closer and with $t_i = 0.1$. The red highlighted regions show how the clusters mapping can have slightly better anomaly detection rates. The anomalous samples that are inside these regions will not be detected with the normal mapping.

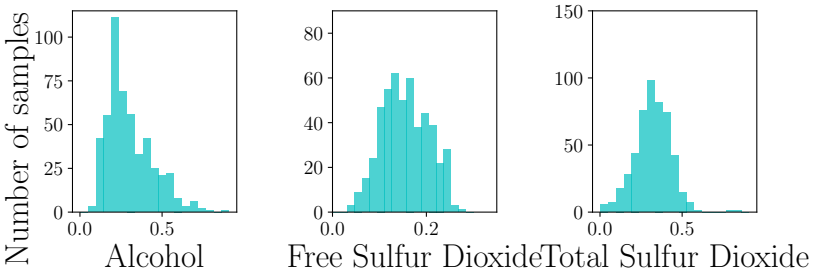
Table 5.2: Number of samples from each training and testing set and from each group.

Number of samples in the set			
Group	Training	Testing	
	Normal	Normal	Anomalous
3,4,5	500	1140	3258
4,5,6	500	3318	1080
5,6,7	500	4035	363
6,7,8	500	2753	1645
7,8,9	500	560	3838

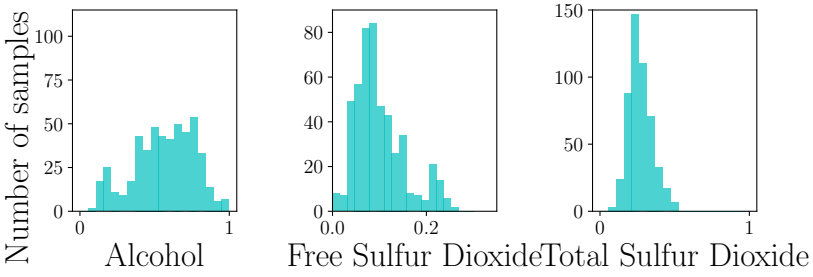
5.2 Results obtained with a real data set

Our method was applied to a data set with 4898 samples related to the evaluation of wine quality [31]. This data set has 11 features: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates and alcohol. It also has a classification from 3 to 9, assigned to each wine quality assessment (3 for very bad; 9 for excellent). It is important to notice that each class does not have the same number of samples each. Wines evaluated with scores 3 and 9 are only a few: 20 and 5 respectively. Wines with a score of 4 and 8 represent a small portion of the whole data set ($\approx 3\%$ each). Wines evaluated with the scores 5, 6 and 7 are the majority, appearing, respectively, 30%, 45% and 18% of the times in the data set.

To evaluate the anomaly detection algorithm it is necessary to establish which sub-set of wines characterises the normal class. To avoid having normal classes with too few samples, groups were defined with wines scoring 3, 4 and 5, or 4, 5 and 6, or 5, 6 and 7, etc, defining in this way 5 different groups, see Table 5.2. In each group, only 500 of normal samples were used in training and the remaining, along with the anomalous samples, for testing.



(a)



(b)

Figure 5.4: Histograms for (a) bad wines and (b) good wines for 3 features with re-scaled values between 0 and 1.

5.2.1 Comparison of clustering techniques

In this section, since the data is not synthetic anymore and the clusters are not obvious, the three clustering techniques presented throughout this thesis, k-means, hierarchical agglomerative (HA) clustering and the extremes clustering technique, are going to be compared. First, a graphic comparison of the clusters produced by each technique is going to be made for the two groups of normal samples that are most different from each other. Then, the performances of the CFS with the 3 clustering techniques, for all the 5 groups, are going to be compared.

The graphic comparison is going to be shown only for two groups. In one case, the normal wines are made of wines classified with 3, 4, and 5 (bad wines) while in the other group, the normal wines are those classified with 7, 8 and 9. In Fig. 5.4 histograms for some features in the data set are shown, with re-scaled values between 0 and 1.

The three different clustering techniques were applied to both groups of normal wines. The contributions to each histogram can be appreciated in Fig. 5.5. The k-means and the hierarchical agglomerative clustering were implemented using the *Scikit Learn* library [30]. The k-means was used with $k = 2$ and with an Euclidean distance. The hierarchical agglomerative clustering was used with a Ward linkage, that minimises the variance of the clusters being merged, and with an Euclidean distance as well. By observing Fig. 5.5 one can notice that the clusters produced by the extremes clustering technique and by the k-means are very similar. Clusters produced by the hierarchical agglomerative are slightly different from the previous two. In any case, results clearly show that meaningful clusters are produced, involving many features, because, for instance, low alcohol wines tend to be clustered together with wines with high sulfur dioxide.

It is also interesting to remark that our technique produces clusters with some similarity to those obtained with the k-means algorithm. However, some difference exists (see Fig. 5.5) as cluster edges are much sharper in k-means clusters. This is understandable since the k-means algorithm uses a metric that translates directly on the histograms axis directions, whereas our technique looks at the number features with rare values.

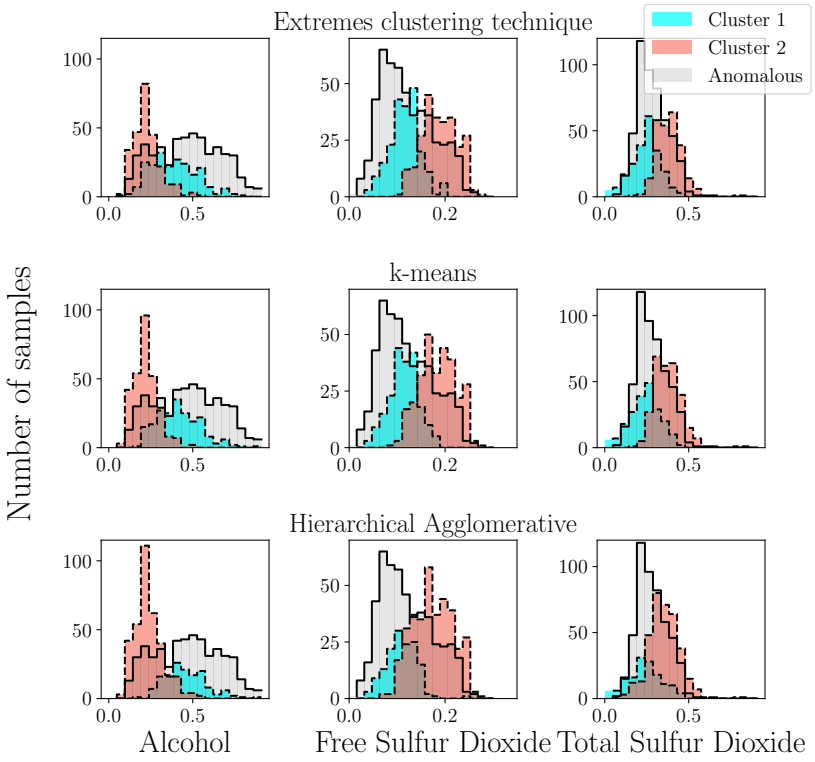
By creating clusters in the data set we expect to better divide the normal sample space and in this way to increase the anomaly detection performance of CFSs. This can be specially done in the case where anomalous samples would not share the same correlations as those present in the training data set. As an example, it can be seen in Fig. 5.5(b) that samples with high Alcohol content have low Sulfur Dioxide content. So a sample with high Alcohol and high Sulfur Dioxide has a high probability of being anomalous. However, not every violation of correlations will necessarily be present, because they also need to be physically or chemically possible.

After realising that the samples that compose the 5 different normal groups have indeed meaningful clusters, the entire cellular frustration algorithm was run 3 times, once with the results of each clustering technique to map the wine features onto f and r signals perceived by the detectors. For each group and each clustering technique, the algorithm was run 5 times with different random initial 500 normal samples and for each group of 500 normal samples it was run 3 times.

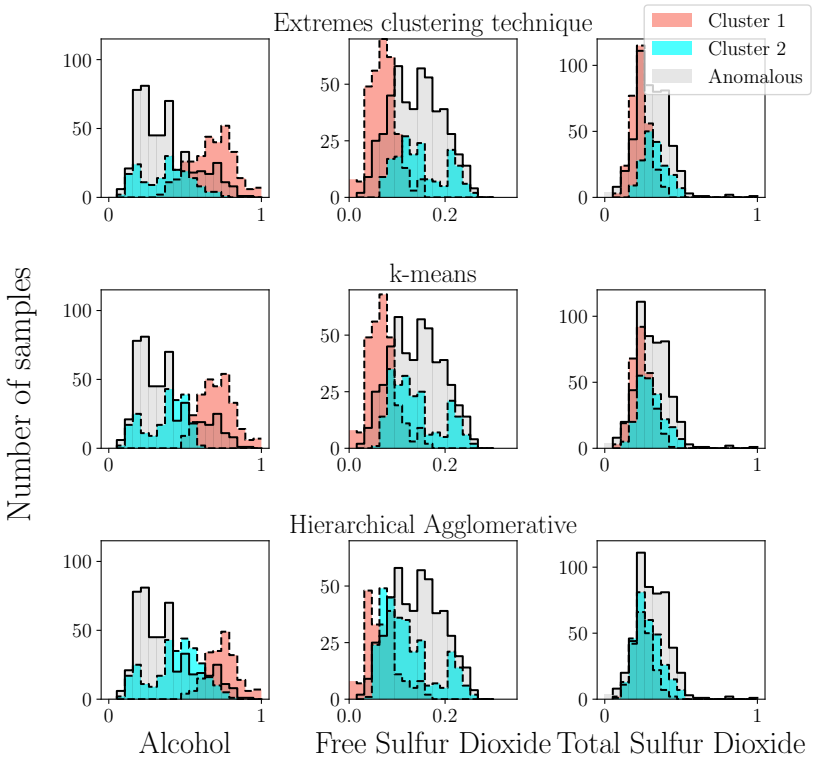
The final results of each group are shown in Fig. 5.6. In order to evaluate whether the changes in the algorithm improve previous results, the ROC curve for a CFS operating without using the clustering technique is also presented.

In Fig. 5.6(a), the results when the normal samples are samples belonging to the classes 3, 4 and 5 are shown. In this case, one can see that when the false positive rate has low values $(0 - 0.1)$ the CFS with the extremes clustering technique performs slightly better than the CFS without clustering. However, for larger values of false positive rate, the CFS without clustering performs better. Both CFS with k-means and hierarchical agglomerative perform worse than the CFS with ECT. For the case depicted in Fig. 5.6(b), where the normal samples belong to the classes 4, 5 and 6, none of the CFS with clustering can surpass the results of the CFS without clustering. However, in Fig. 5.6(c), where the normal samples belong to 5, 6 and 7, the results for the CFS without clustering and the one for the CFS with the ECT are similar until a false positive rate of 0.25. The CFS with k-means and HA have worse anomaly detection rates than the CFS with ECT. In Fig. 5.6(d), where the normal samples are the samples belonging to the classes 6, 7 and 8, both the CFS with the ECT and the k-means perform better than the CFS without clustering. While CFS with the hierarchical agglomerative clustering has an equivalent performance of CFS without clustering. In Fig. 5.6(e), where the normal samples belong to the classes of 7, 8 and 9, all the three CFS with clustering out perform the CFS without clustering, being the HA the best one. Overall, the best clustering technique for the CFS is the extremes clustering technique.

In the case where the normal samples belong to the classes 5, 6 and 7, i.e., average wines, we can see how the algorithm detects really bad (3 and 4) wines and really good (8 and 9) wines. In Fig. 5.7, the ROC curve for the CFS with the ECT is shown with all the anomalous samples (solid line), with only samples belonging to the classes 3 and 4 (dashed line) and with only samples belonging to the classes 8 and 9 (dash-dotted line). The anomaly detection rate when the anomalous samples are bad wines is better than when the anomalous samples are good wines. This result shows that the correlations between average wines and bad wines are more different than the correlations between average and good wines.



(a)



(b)

Figure 5.5: Histograms obtained after applying three clustering algorithms on 3 features of (a) bad wines and (b) good wines. First row, our algorithm; second row, k-means; third row, hierarchical agglomerative clustering. In these plots, histograms for 500 randomly drawn anomalous samples (not bad wines in (a). and not good wines in (b)) are also presented in grey.

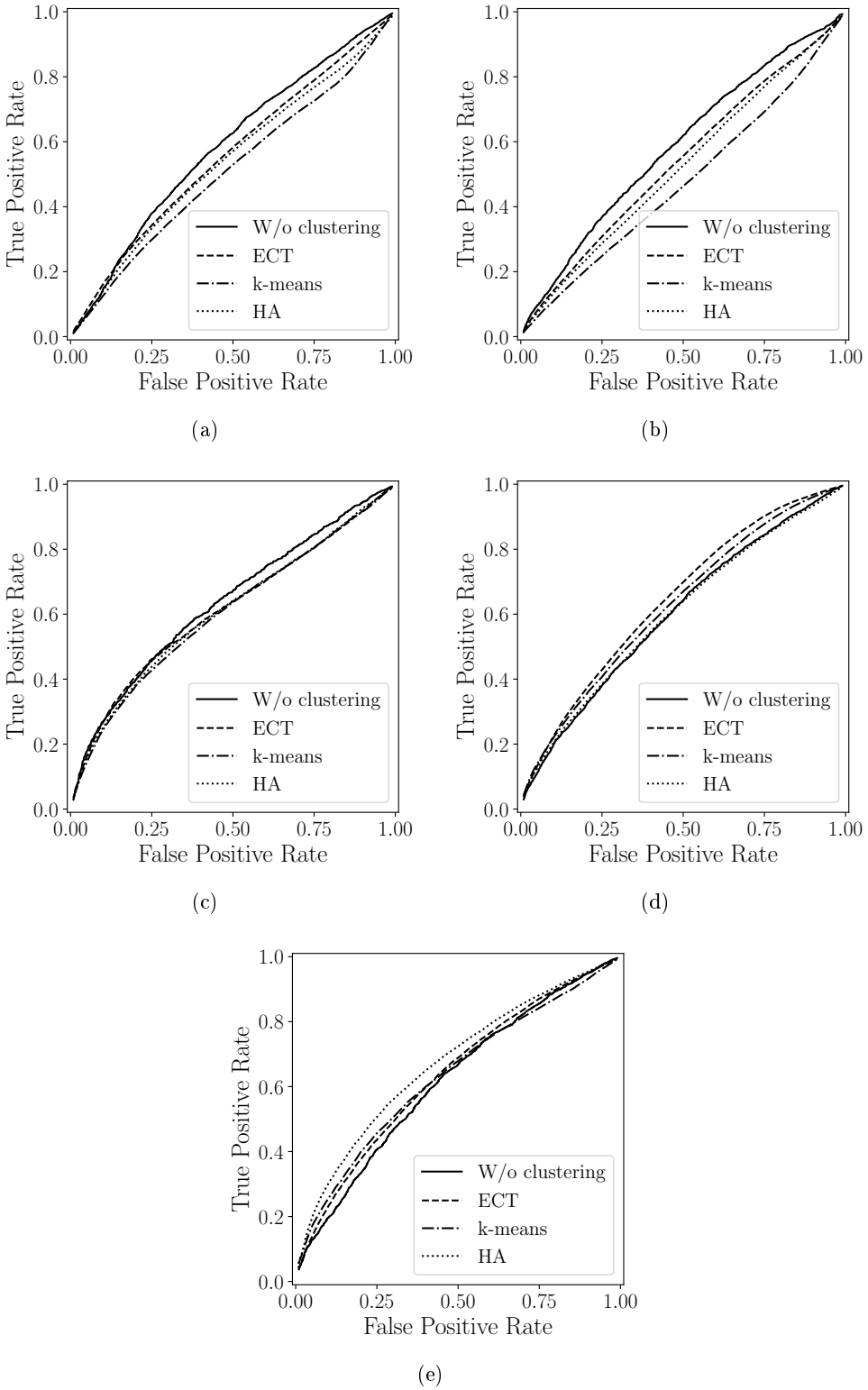


Figure 5.6: ROC curves for wine data set. The solid line represents the ROC curve obtained with the CFS without clustering, the dashed line with the CFS with the Extremes Clustering Technique (ECT), the dash-dotted line with the CFS with the k-means and the dotted line with the CFS with the hierarchical agglomerative clustering (HA). The normal samples in (a) are the samples belonging to the classification 3,4 and 5; in (b) belonging to 4,5 and 6; in (c) to 5,6 and 7; in (d) to 6,7 and 8 and in (e) to 7,8 and 9.

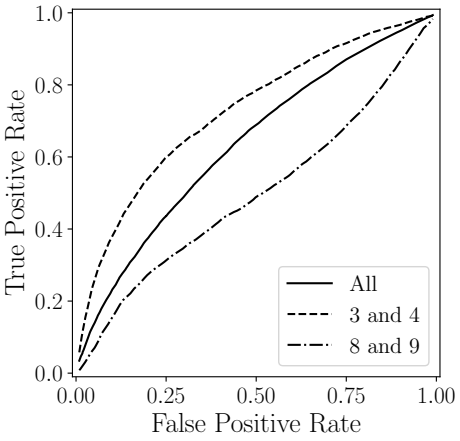


Figure 5.7: ROC curves for the CFS with ECT when the normal samples belong to classes 5,6 and 7. The solid line shows the ROC curve with all the anomalous samples; the dashed line with only samples from the classes 3 and 4; and the dash-dotted line with samples from classes 8 and 9.

In summary, when bad wines are used in training (3, 4 and 5; 4, 5 and 6), ROC curves do not show an improvement when clustering is used. However, improvements can be noted when good wines are used for training (6, 7 and 8; 7, 8 and 9). Therefore, this result confirms that it is possible to improve the anomaly detection rate of the cellular frustration algorithm by better delimiting the sample space with clusters. However, results with the bad wines show that, in order to be used in general, our technique still requires future refinements so that even when the anomaly detection rate does not improve with clustering it will not worsen it either.

5.2.2 Comparison with the One-Class SVMs

In order to validate our results, a comparison with a One-Class SVMs is going to be made. However, as it was previously noted, the SVMs have several parameters that can be tuned. The variability of the SVMs when the parameters are varied will be shown next.

In Table 5.3, the true positive rate for a false positive rate of 10% is shown for each group of normal samples and for different anomaly detection methods. The One-Class SVMs were tested for two distinct kernels: the Radial Basis Function (RBF) and the polynomial kernel. For the RBF, the parameter that was tuned was the γ and for the polynomial kernel it was its degree. To compare, the true positive rate for 10% of false positive rate for the CFS with the ECT is also shown.

When normal samples belong to classes 3, 4 and 5, and to classes 5, 6 and 7 the RBF kernel performs better than the polynomial kernel and the results are similar with the different γ. The CFS with ECT achieves a similar result for normal samples of classes 3, 4 and 5 and better results for normal samples of classes 5, 6 and 7. For normal samples belonging to classes 4, 5 and 6, the RBF kernel is also better than the polynomial, however, with different γ, different results are achieved. The best result is for a $\gamma = 1/N_f$, which is the default value of the library. In this case, the CFS results are worse than the One-Class SVMs with a RBF. On the contrary, with samples belonging to classes 6, 7, the best result with the SVMs is achieved with a RBF kernel but with $\gamma = 10$ and the result with the CFS is better than

Table 5.3: Values of True Positive Rate for 10% of False Positive Rate for each group of normal samples and different anomaly detection techniques. The results for the One-Class SVMs were achieved with a $\nu = 0.1$. Two kernels were tested: the RBF and the polynomial. For the RBF kernel, the user-defined parameter γ was tested with different values, from the default value $1/N_f$ up to 10. For the polynomial kernel, the results shown were obtained for a degree of 2 and 3.

Normal samples	One-Class SVMs					CFS
	RBF			Polynomial		
	$\gamma = 1/N_f$	$\gamma = 1$	$\gamma = 10$	degree= 2	degree= 3	
3,4,5	16.2	16.2	16.1	7.9	8.0	16.2
4,5,6	19.5	18.8	14.2	5.5	5.2	14.1
5,6,7	23.8	23.7	24.8	12.2	12.3	26.7
6,7,8	15.2	16.1	19.1	15.5	15.4	22.3
7,8,9	12.2	14.5	32.3	23.9	24.1	23.1

Table 5.4: Values of True Positive rate for 10% of False Positive rate for normal samples belonging to classes 7, 8 and 9 and for One-Class SVMs with a RBF kernel with $\gamma = 10$ and a polynomial kernel of degree 3.

Number of training samples	RBF $\gamma = 10$	Polynomial degree=3
10	20.6	25.1
20	21.4	24.5
50	21.2	24.5
100	21.2	24.6
200	28.0	23.7
300	29.8	24.5
400	30.8	24.4
500	32.3	24.1

any of the tested One-Class SVMs. When normal samples belong to classes 7, 8 and 9, the best performance of the One-Class SVMs is with a RBF kernel and $\gamma = 10$. However, as it was previously stated, a high value of γ produces unstable results that highly depend on the number of training samples. In Table 5.4, the performances of a One-Class SVMs with a RBF kernel with $\gamma = 10$ and with a polynomial kernel of degree 3 are evaluated for different numbers of training samples. The results for the RBF kernel depend extremely on the number of samples, varying from a value of 20.6% to 32.3%, while the results with the polynomial kernel are more stable and independent of the number of samples, varying from 23.7% to 25.1%. In this case, when samples belong to classes 7, 8 and 9, even though the RBF kernel with $\gamma = 10$ produces better results for 500 samples, the polynomial kernel of degree 3 is the most trustworthy result. With this in mind, the results for CFS with ECT, in this case, are slightly worse than the results of the One-Class SVMs with a polynomial kernel of degree 3.

Choosing the most appropriate parameters or even the right kernel function to use in an One-Class SVMs imposes a major difficulty. On the other hand, results of the cellular frustration algorithm tend to be consistent overall.

Chapter 6

Conclusions

In this thesis it was shown that algorithms of agents engaging in a frustrated dynamics (cellular frustration models) can be used to perform the complex computation task of detecting anomalies in data sets.

The main goal of this thesis was to show that anomaly detection rates could be improved if cellular frustration models used a clustering technique to define how the different agents processed the information presented in each sample. Some synthetic data set were presented where the application of a clustering technique prior to the cellular frustration system was beneficial and anomaly detection rates were better than without clustering. It was also shown, with synthetic data sets, that the application of the clustering technique did not worsen results.

Three different clustering techniques were presented and compared: the $k-$means, the hierarchical agglomerative clustering and the extremes clustering technique. The latter is a technique devised in this thesis that is the most suited to CFSs. This clustering technique assumed that samples with opposite extreme values on a feature should belong to different clusters provided other features had values in non-overlapping intervals. We showed that in a real data set (on wine quality) all the clustering techniques produced well defined clusters which, in some cases, helped improving the anomaly detection rate of cellular frustrated algorithms. In most cases that were studied, the anomaly detection rates of the CFS with each clustering technique were similar. However, the extremes clustering technique was, overall, the best clustering technique. It also has an advantage over the other techniques. When there are no meaningful clusters in the data set, the extremes clustering technique will not produce any clusters (the sets will be empty), while the k-means and the hierarchical agglomerative clustering will always produce any number of clusters.

The results of the cellular frustration system with the extremes clustering technique were compared with the One-Class support vector machines. In this, it was shown how much the results of the One-Class SVMs vary with the change of the user defined parameters.

These results are significant because they show directions on how cellular frustrated algorithms could be improved. However, more extensive studies using other data sets should be conducted to appreciate the extension of the improvements.

Bibliography

[1] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[2] Abreu F. V. Mostardinha, P. Positive and negative selection, self-nonself discrimination and the roles of costimulation and anergy. *Scientific Reports*, 2:769, 2012.

[3] Andre M. Lindo, Bruno F. Faria, and F. V. de Abreu. Tunable kinetic proofreading in a model with molecular frustration. *Theory in Biosciences*, 131(2):77–84, Jun 2012.

[4] F. Vistulo de Abreu and et al. Cellular frustration: A new conceptual framework for understanding cell-mediated immune responses. In Hugues Bersini and Jorge Carneiro, editors, *Artificial Immune Systems*, pages 37–51, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[6] E. Aleskerov, B. Freisleben, and B. Rao. Cardwatch: a neural network based database mining system for credit card fraud detection. In *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*, pages 220–226, Mar 1997.

[7] V. Kumar. Parallel and distributed computing for cybersecurity. *IEEE Distributed Systems Online*, 6(10), 2005.

[8] Clay Spence, Lucas Parra, and Paul Sajda. Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA'01)*, MMBIA '01, pages 3–, Washington, DC, USA, 2001. IEEE Computer Society.

[9] Ryohei Fujimaki, Takehisa Yairi, and Kazuo Machida. An approach to spacecraft anomaly detection problem using kernel feature space. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pages 401–410, New York, NY, USA, 2005. ACM.

[10] Max Bramer. *Principles of Data Mining*. Springer Publishing Company, Incorporated, 2nd edition, 2013.

[11] Saptarsi Goswami, Sanjay Chakraborty, Sanhita Ghosh, Amlan Chakrabarti, and Basabi Chakraborty. A review on application of data mining techniques to combat natural disasters. *Ain Shams Engineering Journal*, 2016.

[12] Hian Koh and Gerald Tan. Data mining applications in healthcare. 19:64–72, 02 2005.

[13] E.W.T. Ngai, Li Xiu, and D.C.K. Chau. Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Systems with Applications*, 36(2, Part 2):2592 – 2602, 2009.

[14] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Advances in knowledge discovery and data mining. chapter From Data Mining to Knowledge Discovery: An Overview, pages 1–34. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

[15] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2015.

[16] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[17] Leo Breiman. *Classification and regression trees*. Routledge, 1984.

[18] Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[19] V. Vapnik and A. Lerner. Pattern Recognition using Generalized Portrait Method. *Automation and Remote Control*, 24, 1963.

[20] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[21] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.

[22] A. K. Jain and et al. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, September 1999.

[23] Amit Saxena, Mukesh Prasad, and et al. A review of clustering techniques and developments. *Neurocomputing*, 267:664 – 681, 2017.

[24] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[25] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[26] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms which use cluster centers. *Comput. J.*, 26:354–359, 1984. cited By 3.

[27] David F Manlove, Robert W Irving, and et al. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1):261 – 279, 2002.

[28] Catarina R Almeida and Fernão Vistulo de Abreu. Dynamical instabilities lead to sympatric speciation. *Evolutionary Ecology Research*, 5(5):739–757, 2003.

[29] Bruno Filipe Faria and et al. Can the immune system perform a t-test? *PLOS ONE*, 12(1):1–35, 01 2017.

[30] F. Pedregosa and et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[31] P. Cortez and et al. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems, Elsevier*, 47(4):547–553, 2009.