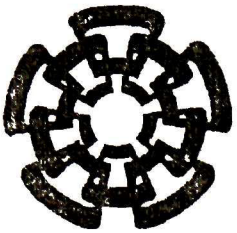
CT-745

Don. – 2013



Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

Aplicaciones de Redes Neuronales dentríticas a Análisis de Conceptos Formales

Tesis que presenta:

David Ernesto Caro Contreras

para obtener el grado de:

Maestro en Ciencias

en la especialidad de:

Ingeniería Eléctrica

Director de Tesis

Dr. Andrés Méndez Vázquez

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Febrero de 2013.

210387.1

Aplicaciones de Redes Neuronales dentríticas a Análisis de Conceptos Formales

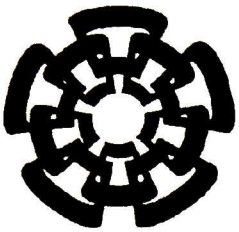
Tesis de Maestría en Ciencias
Ingeniería Eléctrica

Por:
David Ernesto Caro Contreras
Licenciatura en Informática
Universidad de Guadalajara 2002-2007

Becario de CONACyT expediente no. 243197

Director de Tesis
Dr. Andrés Méndez Vázquez

CINVESTAV del IPN Unidad Guadalajara, Febrero de 2013.



Centro de Investigación y de Estudios Avanzados

del Instituto Politécnico Nacional

Unidad Guadalajara

Dendritic Neuronal Network Applications to Formal Concept Analysis.

A thesis presented by:
David Ernesto Caro Contreras

to obtain the degree of:
Master in Science

in the subject of:
Electrical Engineering

Thesis Advisors:
Dr. Andrés Méndez Vázquez.

CINVESTAV del IPN Unidad Guadalajara,Guadalajara, Jalisco, February,2013.

Dendritic Neuronal Network Applications to Formal Concept Analysis.

Master of Science Thesis
In Electrical Engineering

By·
David Ernesto Caro Contreras
Degree in Informatics
Universidad de Guadalajara 2002-2007

Scholarship granted by CONACYT, No. 243194

Thesis Advisors:
Dr. Andrés Méndez Vázquez.

CINVESTAV del IPN Unidad Guadalajara, February, 2013.

ABSTRACT

Since the appearance of the theory of Formal Concept Analysis (FCA) have been published a lot of material and diversified application of this theory. FCA, has been primarily presented as a conceptual hierarchical model of a set of objects and attributes and a binary relation. The term was introduced by Rudolf Wille in 1984, and is based on Lattice Theory and order theory that was developed by Birkhoff and others in the 1930s. The Concept Lattice is the main tool that offers this theory, each concept in the hierarchical network represents the set of objects that share the same values for a given set of attributes, and every concept that is related down the hierarchy is a subset concepts that are above.

Many algorithms for generating the lattice of concepts have been developed since the creation of the foundations of the theory. This thesis presents the research results of this theory.

In this thesis we present initially, the fundamentals of the theory and some of its history, applications and mathematical concepts, which will serve to define specific applications of the theory of FCA.

We also review algorithms essential for generating the concept lattice and classify the algorithms with respect to their characteristics. Similarly, we present a new algorithm for generating the lattice of Concepts using dendritic computing theory. Finally, an application in the context of learning and computer vision is presented. Finally, in this document, we provide a formal categorical classification method using the notions researched in the field of FCA and Theory of Lattices. Finally, an application in the context of learning and computer vision is presented.

...

ABSTRACTO.

Desde la aparición de la teoría de Formal Concept Analysis (FCA) se ha publicado mucho material y se ha diversificado la aplicación de dicha teoría. FCA, es la derivación de un modelo jerárquico conceptual de un conjunto de objetos y atributos en una ontología formal. El término fue introducido por Rudolf Wille en 1984, y se basa en la teoría aplicada de las lattice y teoría de la orden que fue desarrollado por Birkhoff y otros en la década de 1930. La Lattice Conceptual, es la principal herramienta que ofrece dicha teoría, Cada concepto en la red jerárquica representa el conjunto de objetos que comparten los mismos valores para un conjunto determinado de atributos, y cada concepto que se encuentre relacionado abajo de la jerarquía contiene un subconjunto de los objetos de los conceptos por encima de ella.

Muchos algoritmos para la generación de la Lattice de Conceptos han sido desarrollados desde la creacion de los fundamentos de la teoría. Esta tesis presenta los resultados de investigación de esta teoría.

En esta tesis presentamos, inicialmente, los fundamentos de la teoría, así como algo de su historia, aplicaciones y conceptos matemáticos, que nos servirán para definir aplicaciones concretas de la teoría de FCA.

También navegaremos algoritmos escenciales para la generación de la lattice de conceptos y clasificaremos los algoritmos con respecto a sus caracteristicas, Del mismo modo, presentaremos un algoritmo nuevo para la generación de la Lattice de Conceptos basado en la búsqueda de máximos rectángulos. En este documento, tambien encontrarás un proceso formal de clasificación categorica usando las nociones investigadas en el ámbito de FCA y Teoría de Lattices. Finalmente, una aplicacion en el contexto del aprendizaje y visión por computadora es presentada.

"Nos resignamos al momento único y feliz. Preferimos perderlo, dejarlo transcurrir sin siquiera hacer el razonable intento de asirlo. Preferimos perderlo todo, antes que admitir que se trata de la única posibilidad y que esa posibilidad es solo un minuto y no una larga e impecable existencia." — Mario Benedetti. (Gracias por el Fuego)

Dedicated to the loving memory of Diego Argenis Caro Contreras and Ricardo Melendez Contreras.

Thanks for so many good memories.

"The question of whether computers can think is like the question of whether submarines can swim."
Edsger W. Dijkstra.

— ?

ACKNOWLEDGMENTS

Apart from the efforts of myself, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project. I would like to show my greatest appreciation to P.H.D. Andres Mendez. Without his encouragement and guidance this project would not have materialized. The guidance and support received from all the members who contributed and who are contributing to this project, was vital for the success of the project.

No one walks alone on the journey of life. So, perhaps this thesis and it's pages will be seen as a "thanks" to the tens of thousands of you who have helped make my life what is today. Much of what I have learned over the years came as the result of my family. I would like to express my gratitude towards my parents and friends, I dont know, even I cant understand, why do you always trust me, but that confidence is some of the most important things in my life. My thanks and appreciations also go to my colleague and life teacher, in special to CINVESTAV.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Part I

INTRODUCTION

This first part is a brief introduction to the Lattice Theory and FCA subjects. Chapter 1 is a brief on history of Lattice Theory and FCA. In this chapter we also define the purpose and the goals of this document. Chapter 2 contains the necessary background knowledge in order to achieve the proposed goals.

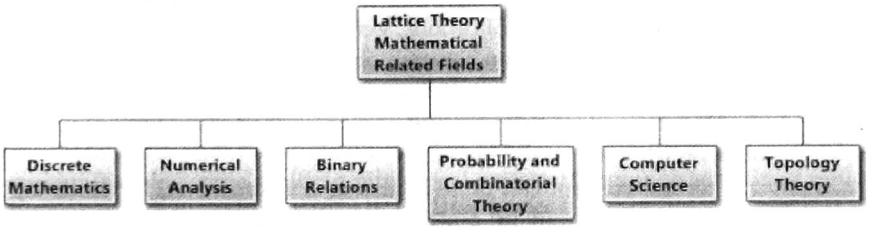
1

LATTICE THEORY AND FCA.

1.1 INTRODUCTION

Lattice Theory is growing popularity since it was first proposed as mathematical framework. Basically, **Lattice Theory** is the abstract study of a set of elements known as lattices. It is an outgrowth of the study of Boolean Algebras and Abstract Algebras, and provides a framework for the study of ordered sets in mathematics[6]. Because of its robustness, at first, this theory took much relevance, especially after G.Birkhoff[14] wrote the first textbook about the topic. Immediately, the history registered an increase of papers, publications and new elements of the subject. Figure 1 shows some fields of study in mathematics, that have been related to the theory of lattices. As can be seen in that figure, Discrete Mathematics[6], Numerical Analysis [74], Binary Relations issues[16, 38], Probability and Combinatorial Theory[68], Computer Science[104], Topology Theory[15], among others studies have a strong relationship with this subfield of Abstract Algebras.

Figure 1: Math Fields Related with Lattice Theory



Thousands of studies have been generated from the field of Lattice Theory, many of them mature, in both, the theoretical and application sense. In this work we will focus mainly in the theory of lattices in the field of **computer science** mentioned early[104]. In recent decades, the use of the theory of lattices has also spread, with surprising speed, in classical and emerging fields of computer science. The state of the art, presented in this thesis, only includes some topics of interest for the development and achievement of the objectives that we seek.

The concept tree of Figure 2 shows strongly related Lattice Theory with Computer Science fields. In this thesis, we are specially interested in **Lattice Based Neural Networks, Prediction** and **Classification**, and **Formal Concept Analysis**.

Formal Concept Analysis (FCA) emerged in the early years of the decade of the 80s by Wille. Wille first defined a Lattice Restructuring as: "Restructuring lattice theory is an attempt to reinvigorate con-

nections with our general culture by interpreting the theory as concretely as possible, and in this way to promote better communication between lattice theorists and potential users of lattice theory"[101].

Galois Lattices were later widely studied by the large body of work done by Wille and Ganter[99, 38] and the many researchers who worked with them. Their early work, named Formal Concept Analysis, deals with Concept Lattice idea, in a much more general context, and it was supported in the field of binary relations. These binary relations, as description of the world can be found in many areas of human interest. There exists several domains, including medicine, psychology, chemistry, biology, and many others, where we can find this kind of relations.

In addition, there exists a Philosophical perspective that defines a concept as a unit of thoughts. According Ganter[37], Philosophy has identified two parts over the term concept. The **extension** of the concept as the cover of all objects belonging to the concept and **intention** of the concept in which is included all attributes shared by the extension set.

R. Willie in[101], using lattice algebra[14] approximation, defines a structured way of deriving a concept hierarchy. Using Galois Connectors, a join and a meet operation over a binary context, generating lattice order. These operators with the concept idea rise to the Concept Lattice Structure.

Basically, FCA, refers to a mathematized and formal human centered way to analyze data. It is described as an ontology based on an algebraic space. This algebraic space is usable for covering and visualizing patterns, implications, generalizations and dependencies. Hence, the binary relation between objects and attributes plays a fundamental role in the understanding of human conceptualization models. FCA formalizes this idea and gives a mathematical framework to work on it.

A formal concept can be seen also as a data cluster, in which all properties are shared by a set of objects, and dually, all those objects share those represented properties. Those clusters, named also formal concepts, has the property to be closed. The set of all concepts sorted by inclusion order is called the Concept Lattice, some

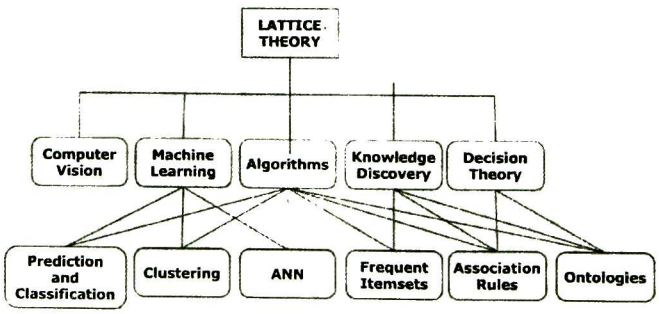


Figure 2: Lattice Theory And Computer Science

authors[39, 40, 55] describe the concept lattice as all the clusters living in a binary context.

1.2 PROBLEM DEFINITION.

The main drawback of concept lattices is that they may be of exponential size. It is known that a context of only modest size can produce hundreds of thousands formal concepts. However, with the knowledge of the number of concept as restriction, many algorithms have been proposed Concept Lattice Generation in polynomial delay time. A **polynomial delay** time means that the time elapsed between two consecutive outputs is polynomial in the input size.

Even, taking account of the polynomial delay analysis. The possibility of exponential size of the lattice makes impossible, in practice, to compute and span the entire lattice structure. Thus, it is outmost importance to be able to navigate the lattice efficiently, get reduced representations, or be able to define a polynomial sized sub-lattice which contains the right information.

Many challenges have been raised from the perspective of Lattice theory and FCA. In this thesis we work some of them in particular. The first is the generation of concept lattices. In this sense, an algorithm for generation of concept lattices is presented. The other problems are in the machine learning approach. In which, we present a formal classifier method. Additionally, this classifier is able to integrate many methods of pattern recognition, threadhold and binary functions in the context of categorical classification.

1.3 PURPOSE AND GOAL.

This thesis deals with the problems of generating concept lattices and machine learning, specifically in categorical pattern recognition sense. The first purpose is to introduce the reader to the theoretical framework of Lattice Theory and FCA. Other important background knowledge is the Lattice Based Neural Networks. This method will be used as a basis for our design. To that effect, the reader will be also introduced to the theory of Lattice Based Neural Network, in specific the Single Layer Lattice Perceptron.

To solve the problem of the generation concept is proposed a lattice based neural network for find maximal rectangles. In this sense, to find the complete concept a lattice, a maximal rectangles search is applied recursively on each maximum rectangle found. With respect to our proposed classifier, a Lattice Based Neural Network on minimum and maximum anti-chain is applied on binary contexts.

The goal of this thesis is to prove that both proposals are possible and feasible. That's why we are attached to each of the proposals, an experimental analysis and conclusions.

BACKGROUND KNOWLEDGE

2.1 INTRODUCTION.

This section deals with the background knowledge and it is organized as follows: In section 2.2 You will find a brief review on Machine Learning and Categorical Classification. You will find a Supervised/Unsupervised categorical classification description and the definition of binary and threadhold functions. In section 2.3 you will find formal and mathematized FCA notions used in this thesis. Section 2.4 is dedicated to review properties of some of the algorithms for generating Concept Lattice. Section 2.5 is a brief about known applications of FCA. A Lattice Based Neural Network introduction is exposed at Section 2.6. At the end of this chapter you must be able to understand the given notation and the main terminology used in this thesis.

2.2 MACHINE LEARNING AND CATEGORICAL CLASSIFICATION.

The essence of machine intelligence is **Machine Learning** or Computer Based Learning. When we have system with learning capability, we will have a real artificial intelligence. Due there exists many machine learning strategies and studies, our first step is review the state of the art in machine learning and provide a general idea of how this chapter understand this approach.

Machine learning has been widely studied by several authors, many subtopics have emerged, which are included in this sub-branch of artificial intelligence. In fact, it has been developed so much that have given rise to excellent methods that simulate both the acquisition of new knowledge and the separation of the classes contained in a analyzed data set. In that sense, **Pattern Recognition** emerged as a new branch of machine learning.

Pattern recognition is one of the cores on which the machine learning theory is supported to achieve two main objectives. A particular area of pattern recognition, named categorical prediction, focuses on assigning a category to a given input value. It means, models for prediction of categorical labels, in which, given an input set, a value label (class) is assigned to each element of that input set. **Real values** and **Series** prediction, works with the idea of get more complex outputs. In this thesis we works with **prediction of categorical labels**.

To understand this, suppose the following scenario: You have a certain family of objects, each of these objects have some properties, described by sensors or mathematical analysis. The pattern recognition, **supervised learning** case, related issue is to program the machine from several examples. The program must learn those examples and

then it must correctly infer which class belongs to a new entry, using the experience gained by the provided examples. Being the special case of **unsupervised learning** in which the same model is responsible for finding existing classes. In this second case, similarity functions, distances or some other techniques are used to group data that has something in common to each other, using those clusters on consequential extrapolations and always adapting the clusters to the new entries.

2.2.1 *Supervised learning to predict categorical labels.*

This learning technique consists of presenting to learning model, repetitively, stimuli input patterns belonging to the training game. The training game or training data consist of pairs "stimulus pattern right answer" and should be chosen carefully. Each pair is called a fact. In the training game should be represented evenly all the information you need learning model to extrapolate, once the training is finished, system should be capable of handle new information. A formal definition of that state is:

Definition 1. Let X be a input domain and let Y be a set of categorical labels. A set of pairs with the form $\{(x_1, y_1), ..., (x_n, y_n)\}$ is named training set T, where $x_i \in X$ is a feature vector, and $y_i \in Y$ is an assigned category. A Supervised Learning Algorithm to predict categorical labels search a function $f : X \to Y$ from a set of possible functions in a space F, using the training set T as main reference.

In this sense, several methods exists in the literature[29]. Table 1 shows some methods ordered by the framework in which they are supported. We present three kinds of frameworks to work. the first one, Kernel Methods (KM's), is considered a set of algorithms for pattern analysis. KM's Algorithms can operate on very general types of data and can detect very general types of relations. Correlation, factor, cluster and discriminant analysis are just some of the types of pattern analysis tasks that can be performed on data as diverse as sequences, text, images, graphs and of course vectors[83]. Statistical framework, in which supervised learning to predict categorical labels gains importance. Statistical algorithms are, mainly, categorized as generative or discriminative. Logistic regression, also known as maximum entropy classifiers, and Bayes Theory are the principal support of Statistical Algorithms. Neural Networks are also a good framework to work. These networks have been demonstrated an excellent performance in some specially solutions. Inspired in a biological sense, some of these Artificial Neural Networks are linked with the categorical classification trouble.

To fully understand the underlying problem, suppose that you want to predict whether a person will have a heart attack within a year. You should have a training game with information about people; records about their condition, age, sex, blood pressure, weight, symptoms, etc.. You should to know those people who had a heart attack about a year after the measurement data. Hence the problem is

Table 1: Categorical Supervised Learning.

	Supported by
Kernel Methods	Support Vector Machine (SVM)[25], Fisher's linear discriminant analysis (LDA)[32, 67], Principal Component Analysis(PCA).
Statistical Methods	Bayes Theory[11], Maximum Entropy Classifiers[26, 50]. Learning Vector Quantization[51]
Neural Networks	Backpropagation, Radial Basis Neural Networks, Fuzzy Based Neural Network, Dendritical Neural Networks[17, 47, 5].
Lattice Based	Lattice Based Neural Networks, Fuzzy Lattice Classifiers.[79, 80, 77, 51]

This table shows some of the supported classifiers in KM's, SM's, NN's and Lattice Based Methods.

to combine all available information with a learning model that when presented with data from a new person can predict whether this will have a heart attack within a year with a reasonable accuracy, that means the model predicts two categories, "have heart attack within a year" or "does not have heart attack within a year" What we are trying to say is that the main feature of this example of supervised learning is the use of a training set for categorical prediction. There exists a lot of system that deals with this particular issue.

2.2.2 *Unsupervised learning to predict categorical labels.*

In unsupervised learning, unlike supervised learning, there is no a priori knowledge, that is, there is no training set as a basis for adapting the model and then exploit it. In this method of learning, which is considered automatic method, the model is adapted to the observations. Initially, we can infer that this kind of algorithms don't require an external teacher. Thus, they show some degree of self-organization. Then, the model discovers the input data behavior autonomously using features regularities, correlations, similarity measures, distances, dissimilarities etc. you can realize that there are many algorithms to achieve this goal. The model discover the cluster model, in which, the data categories are finally used for an extrapolation phase.

With this technique it is possible, on each new input, generate or adjust the clusters and then classifying those new entries generated from the model. That task will be mainly studied in this chapter of the dissertation. It is important to note that there exist a lot of information and many techniques for unsupervised learning, but we are only interested in the part where the new information is assigned to a known category.

2.2.3 *Other Learning methods.*

Other learning methods, such as the prediction of real-valued labels and learning of categorical or real valued sequences are not considered in this thesis. However this does not mean they cannot be included in a future as part of the framework.

This work is delimited to use learning and prediction of categorical labels models. And it can be extended to semi-supervised learning to predict categorical labels, which mixes both supervised learning models as a the unsupervised machine learning models.

2.2.4 *Threadholds and binary functions.*

A threadholding method is an operation over a domain, in which, a threadhold is used to separate that domain in areas of interest. In many applications, it is useful to be able to separate out the regions of the data corresponding to objects in which we are interested. Thresholding often provides an easy and convenient way to perform this segmentation on the basis of the different intensities or scale values in the data set. This method have been successfully applied in a lot of Machine Learning related troubles. There also exists a lot of method for threadholding.

2.3 BASIC FORMAL NOTIONS OF FCA.

In literature of FCA exists a lot of notations. We started with Ganter work and we decide to use Ganter notation for avoid confusion. You will find an alternative notation in Appendix A.1. As an observation, in the literature there are thousands of publications and each proposes new definitions, properties, theorems, algorithms and capabilities that enrich the terminology. Formal Concept Analysis home page was an excellent way to start with this subject. ([http://www.upriss.org.uk/fca

This section, as the name implies, is a set of notions used as basis in this thesis.

Definition 2. *A **binary formal context** $\mathbb{K}$ is a triple (G, M, I). In this data structure G and M are two finite sets, called objects and attributes respectively, and $I \subseteq G \times M$ is a binary realtion over G and M, named the incidence of $\mathbb{K}$.*

Table 2 has an example of a Formal Context $\mathbb{K} := (G, M, I)$ as a binary relation between objects and attributes. There $G := \{g_1, g_2, g_3, g_4, g_5, g_6\}$, $M := \{m_1, m_2, m_3, m_4, m_5, m_6, m_7\}$ and I is a binary relation where $I \subseteq G \times M$.

Another useful and intuitively representation is **Binary Matrix Representation.** This representation can be seen too as a set of rows described with zeros and ones, where the ones describe that an object poses an attribute and zero on the contrary. Next Table 3 is representing the same context shown in Table 2. It is another representation of this data structure used to extract concepts in FCA.

Table 2: Example of Formal Context $\mathbb{K}$

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
g_1	x	x		x		x		x
g_2	x		x		x		x	x
g_3	x		x				x	
g_4		x			x		x	
g_5	x			x				
g_6								x

The mathematical structure which is used to describe formally this table with crosses is called a formal context (or briefly a context).

Table 3: Matrix representation of a Binary Context.

1	1	0	1	0	1	0	1
1	0	1	0	1	0	1	1
1	0	1	0	0	0	1	0
0	1	0	0	1	0	1	0
1	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1

Matrix representation of the binary relation in Table2.

In order to define formal concepts of the formal context (G,M,I), it is possible to define two derivation operators. As we mentioned earlier, these operators, named **Galois Connectors** or **Derivator Operators** or simply **Derivates**, are basic to understanding the main definitions of FCA.

The following definition contains the statement and rules that give form to these two operations.

Definition 3. *For an arbitrary subsets:*
$A \subseteq G$ *and* $B \subseteq M$*:*

- $A' := \{m \in M \mid (g,m) \in I, \forall g \in A\}$

  $B' := \{g \in G \mid (g,m) \in I, \forall m \in B\}$

These two **Derivation Operators** satisfy the following three conditions over arbitrary subsets $A_1, A_2 \subseteq G$ and $B_1, B_2 \subseteq M$:

1. $A_1 \subseteq A_2$ then $A_2' \subseteq A_1'$ **dually** $B_1 \subseteq B_2$ then $B_2' \subseteq B_1'$
2. $A_1 \subseteq A_1''$ and $A_1' = A_1'''$ **dually** $B_1 \subseteq B_1''$ and $B_1' = B_1'''$
3. $A_1 \subseteq B_1' \iff B_1 \subseteq A_1'$

Thus, as mentioned above, a concept is constituted into two parts, a set of irreductible objects and a set of irreductible attributes. Those

two parts form a dual isomorphism between those two closure systems. The mathematized idea of formal concept, and also mentioned earlier, represents the unit of FCA:

Definition 4. *Let* $\mathbb{K}$ *be a formal context,* $\mathbb{K} := (G, M, I)$, $A \subseteq G$ *and* $B \subseteq M$.

A **Formal Concept** C of $\mathbb{K}$ is defined as a pair $C = (A, B)$ where the following conditions are satisfied.

$A = B'$ and $A' = B$; where A is named the **extent** and B is named the **intent** of the formal concept (A, B).

Next, separating ourselves slightly of FCA topic, we will show from **Lattice Theory** some useful notions to understand the algebraic structure generated by those derivation operators and the formal concept idea.

Definition 5. A **Partially Ordered Set** or abbreviated, poset, is a system X in which a binary relation $x \geqq y$ is defined and which satisfies:

1. $\forall x,\ x \geqq x$ (reflexive).
2. if $x \geqq y$ and $y \geqq x$, then $x = y$ (Antisymmetric).
3. if $x \geqq y$ and $y \geqq z$, then $x \geqq z$ (Transitive).

From here, is easy to define our algebraic structure, the Concept Lattice.

Definition 6. The concepts of a given $\mathbb{K} := (G, M, I)$, and denoted by $\mathfrak{B}(\mathbb{K})$ can be regarded as the **Concept Lattice** of $\mathbb{K}$, where $\mathfrak{B}(\mathbb{K})$ naturally ordered by the sub-concept/super-concept relation is mathematized by:

Let (A_1, B_1), *and* $(A_2, B_2) \subseteq \mathfrak{B}(\mathbb{K})$

$(A_1, B_1) \leq (A_2, B_2) \iff A_1 \subseteq A_2 (\iff B_1 \subseteq B_2)$

However, using the derivation operators we can derive formal concepts from our formal context. The whole set of formal concepts together with the order relation, mentioned early, is called Concept Lattice or, also Galois Lattice, because it verifies the lattice properties. The relation $\leq$ presented above is clearly a partially order relation.

Theorem 1. ***Basic Theorem on Concept Lattices,*** *Wille [101]. Let* $\mathbb{K} := (G, M, I)$ *be a formal context. Then* $\mathfrak{B}(\mathbb{K})$ *is a* ***Complete Lattice,*** *called the Concept Lattice of (G, M, I), in which infimum and supremum exists and they are given by:*

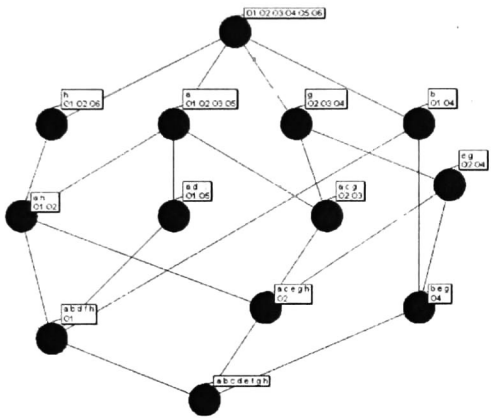
$$\bigwedge_{t \in T}((A_t, B_t) = (\bigcap_{t \in T} A_t, (\bigcup_{t \in T} B_t)'')$$

$$\bigvee_{t \in T}((A_t, B_t) = ((\bigcup_{t \in T} A_t)'', \bigcap_{t \in T} B_t)$$

Figure 3 shows a typical lattice diagrams, from the context in Table 2. The partial order with the meet and join operations generate the lattice graph (named Hasse Diagram) by removing transitivity and reflexivity. This means, given the concepts $C_1, C_2 \in \mathfrak{B}(\mathbb{K})$ there is an edge from C_1 to C_2 if $C_1 < C_2$ and there does not exist another element $C_3 \in \mathfrak{B}(\mathbb{K})$ such that $C_1 < C_3 < C_2$.

A claim and a proof that a complete lattice L is isomorphic to $\mathfrak{B}(\mathbb{K})$ is inGanter [38]. In particular $L \cong \mathfrak{B}(L, L, \leq)$.

Figure 3: Concept Lattice from Formal Context $\mathbb{K}$ in Table 1.



This graph, named Hasse diagram, represents generalization and specialization relationship for all concepts contained in $\mathfrak{B}(\mathbb{K})$.

The following definition is a preamble to explain an alternate way to understand the formal concepts. It deals with formed rectangles in binary matrices, as presented in Table 3. This is a problem which has been extensively studied. One of its definitions, maximum rectangles, is equivalent to the definition of formal concepts as we shown in a subsequent theorem.

Definition 7. Rectangles in $\mathbb{K} := (G, M, I)$. Let $A \in G$ and $B \in M$. A rectangle in $\mathbb{K}$ is a pair (A, B) such that $A \times B \subseteq I$.

From the example in Table 2, the pair $A := \{g_2, g_3\}$ and $B := \{m_1, m_3\}$ as (A, B) is a valid rectangle. But the pair (g_1, g_2, m_3) is not a valid rectangle, because g_1 and m_3 does not satisfies definition 7.

Definition 8. Maximal Rectangles in $\mathbb{K}$. A rectangle (A_1, B_1) is maximal if and only if there does not exist another valid rectangle (A_2, B_2) in $\mathbb{K}$. In which $A_2 \supset A_1$ or $B_2 \supset B_1$.

A clear example, using Table 2, is: $A := \{g_2, g_3\}$ and $B := \{m_1, m_3, m_7\}$ where (A, B) is a rectangle and it is maximal. $(\{g, g_3\}, \{m_1\}$ and $(\{g_3\}, \{m_1, m_3, m_7\})$ they are not maximal rectangles since they are covered by the previously maximal rectangle example.

Theorem 2. ***Formal Concept as maximals rectangles.*** *(A, B) is a formal concept of $\mathbb{K}$ if and only if (A, B) is a maximal rectangle in $\mathbb{K}$.*

Proof. We must to prove two cases, when we have a formal concept it must be a maximal rectangle, and when we have a maximal rectangle then it must be a formal concept. These cases are defined as follows: □

Case 1. Let (A_1, B_1) be a formal concept. $A_1 = B_1'$, $A_1' = B_1$, $A_1 = A_1''$ and $B_1 = B_1''$ suppose that (A_1, B_1) is not a maximal rectangle then there exists a maximal rectangle (A_2, B_2) such that or $A_2 \supset A_1$ or $B_2 \supset B_1$. That means, or $A_1'' = A_2$ or $B_1'' = B_2$, in which (A_1, B_1) is not a formal concept which lies in a contradiction.

Case 2. Let (A_1, B_1) be a maximal rectangle. So, there does not exists another valid rectangle (A_2, B_2) in which or $A_2 \supset A_1$ or $B_2 \supset B_1$. We will say that (A_1, B_1) is not a formal concept. Then, by derivatives properties we know that or $A_1 \subset A_1''$ or $B_1 \subset B_2''$. From this statement we can infer that a valid rectangle (A_1'', B_1) or (A_1, B_1'') exists, being a contradiction the claim that (A_1, B_1) is a maximum rectangle.

This theorem, will allow us to use the matrix representation presented below. This representation gives us another idea of the problem and a different perspective to propose solutions. As we will see in futhers chapters, the use of Lattice Based Neural Network Theory in FCA context is one of the possible perspectives. The following are three definitions related to the topology of the algebra of lattices.

Anti-chains, Upward and Downward Closed Sets definitions are related to Lattice Algebra topology. First notion, anti-chains relies in a set of incomparable elements.

Definition 9. Anti-chains in $\mathbb{K} := (G, M, I)$. Let be $A \subseteq G$. A is said to be a derivative anti-chain set if and only if $A_1' \nsubseteq A_2'$ and $A_2' \nsubseteq A_1'$ for any two distinct $A_1, A_2 \in A$.

All sets in which the super/subconcept order is not satisfied for any two elements of the set is called antichain set. Next, we define two closures systems on any complete lattice.

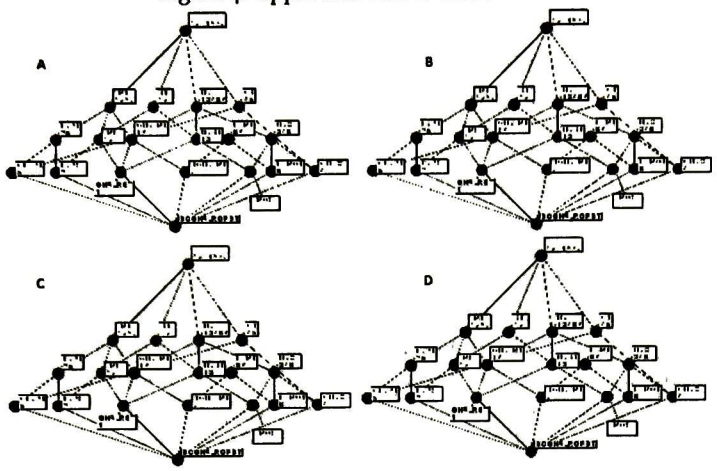
Definition 10. Upward Closed Set. Let $(L, \leq)$ be a poset P. A set $S \subseteq L$ is said to be upward closed if $\forall x, y \in S$ with $y \geq x$ implies that $y \in S$.

So, given a set $S \in L$ there is a smallest upward closed subset of L, which is denoted by $\uparrow S := \{x \in L | x \geq y \text{ for some } y \in S\}$ We can see that $S = \uparrow S$ if, and only if, S is a downward set.

A dually formal definition is the downward closed set using $S \subseteq L$ and $\downarrow S := \{x \in L | x \leq y \text{ for some } y \in S\}$. So, S is a downward closed set if, and only if $S = \downarrow S$.

Those closed set, also named upper and lower, exemplified in figure 4 have the following properties:

Figure 4: Upper and Lower Closed Sets.



sub-Figure A is the Union of two upward closed sets of , it is also the upward closed set. sub-Figure B is the union of two downward closed sets. sub-Figures C an D are their respectives intersections.

- Every partially ordered set is an Upward/Downward Closed Set of itself.
- The intersection($\bigcap$) and the union($\bigcup$) of Upward Closed Sets is always an Upward Closed Set.
- The intersection ($\bigcap$) and the union ($\bigcup$)of Downward Closed Sets is always an Downward Closed Set.
- The complement of any Upwards Closed Set set is a Downward Closed Set, and vice versa.

  Given a partially ordered set $(L, \leq)$: The family of Downward Closed Sets of L ordered with the inclusion relation is a complete lattice, the down-set lattice $O(L)$. Dually in the case of Upper Closed Sets.

2.4 ALGORITHMS FOR CONCEPT LATTICE GENERATION.

Now, we will discuss two main issues, the generation and navigation of Concept Lattices. Once, we briefly review the basic algorithms from the literature, we extract some properties for these algorithms. For Analysis of Algorithms theory was taken into account many ideas from literature[55, 56, 41]. There is no consensus for the analysis of algorithms that generate the Concept Lattice. We use information recorded in some publications that deal with that analysis[19]. The first comparative study of several algorithms constructing the concept set and diagram graphs can be found in [43].

The first algorithm that generates the lattice is attributed to **Malgrange***[62]* and dates from 1962. Few years later, Chein[21] refined

Malgrange algorithm and creates a *bottom-up algorithm* [1]. The algorithm build from the bottom and it goes up level by level. The first level contains all pairs $L_1 := \{(x, x') \mid x \in G\}$. Means that each object $g_i \in G$ as (g_i, g_i') is considered as a first layer. Iteratively, we can build new pairs in L_{k+1} by combining all pairs of L_k. In this process some elements of the layer k are generated again in a top layer in that case, the elements are deleted from the previous layer k and stored as elements of the new layer $k+1$. A simple formal description of Chein algorithm is:

Given two elements of L_k: (X_i, X_i'), (X_j, X_j') so if $X_i' \bigcap X_j' \notin L_{k+1}$ then $(X_i \bigcup X_j, X_i' \bigcap X_j')$ is a new element of L_{k+1} otherwise merge all the pairs with $X_i' \bigcap X_j'$. if $X_i' \bigcap X_j' \in L_{k+1}$ it must be deleted from L_k. Original version of **Chein Algorithm** looks through the current layer each time a new concept is generated and has a exponential worst-time complexity. Some authors have proposed a canonicity test based on the lexicographical order to improve the efficiency of the algorithm [55, 56]. The time complexity reported is $O(\mid G \mid^3 \mid M \mid\mid L \mid)$. And the algorithm has a polynomial delay of $O(\mid G \mid^3 \mid M \mid)$ [56]. Originally, this Algorithm does not generate the Associated Hasse Diagram, but with fewer modifications, some authors reports that it can do Hasse Diagram Generation. [2]

Some algorithms from the early 80's are Next-Closure Algorithm[35], Bordat Algorithm[16], MI-tree[106], Norris Maximal Rectangles Computation[70]. **Next-Closure** algorithm was proposed by Ganter, and it computes closures for only some of subsets of G using an efficient canonicity test, which does not look backwards to the list of generated concepts. It produces the set of all concepts in time $O(|G|^2|M||L|)$ where $L = \mathfrak{B}(\mathbb{K})$ and has polynomial delay $O(|G|^2|M|)$[56]. However, Next Closure Algorithm does not compute the associated Hasse Diagram, instead it generates the concepts in lexicographic order[3]. Next, **Bordat Algorithm [16]** generates the associated Hasse Diagram. This algorithm computes all the concepts of L by computing $cover(A, B)$ for each concept (A, B), starting from the bottom concept, until all concepts are generated (A Bottom Up Algorithm). The time complexity of Bordat is $O(|G||M|^2|L|)$. Moreover, this algorithm has a polynomial delay $O(|G||M|^2)$[56]. The inclusion-maximal subsets problem in Bordat Algorithm is known to be resolved using sophisticated data structures such as a Tree-Set. **MI-Tree Algorithm** searches over a set of previously generated concepts and it does not consider the Hasse Diagram Generation. The original version of the **Norris Algorithm** does not construct the diagram graph. But with fewer modifications, it produces the diagram graph without effort. This Algorithm is also the first Incremental Algorithm [4]. The algorithm of Norris, in processing a concept (A, B), checks whether $B \bigcap g' \subseteq h'$ for any $h \in G_i \setminus A$

1 **A Bottom Up Algorithm** construct the lattice from the bottom, generating first, the concepts with fewer objects.

2 An interesting property of these algorithms is that some of them have the ability to generate a structure that represents: the **Associated Hasse Diagram.**

3 Some Algorithms construct the lattice in lexicographic order.

4 Incremental Algorithms updates the lattice when a new relation between one object and the set of attributes is inserted.

Table 4: Properties of Algorithms for Concept Lattice Generation

	BU	TD	L	B	I
Chein		X		X	
Next-Closure			X	X	
Bordat	X			X	
MI-Tree			X	X	
Norris					X

BU = **Bottom-Up**, TD = **Top-Down**, L = **Lexicographical**, B = **Batch** and I = **Incremental**.

where G_i is the set of previously processed objects. Then A is not the maximal extent, and hence (A, B) is not the most general concept capable of generating B set[70]. Table 4 shows the Algorithmic properties proposed by FCA Theory over the previously algorithms mentioned.

After the decade of the 80's, many algorithms have been proposed for generate Concept Lattice. Concept Lattice Algorithms can be divided into two categories: Batch and Incremental algorithms, as we have seen before. Nourrine Algorithm[71], LingLing Algorithm[60], are two important batch generation approaches. In the other hand, Norris algorithm, Godin Algorithm[39], Carpineto Algorithm[19, 20], Dowling Algorithm and, recently Add Intent Algorithm[66] and Valtchev work[95, 96, 94] are some examples in the incremental sense. Also, some new ideas has emerged, from the Matrix Based Approach for Concept Lattice Generation[59], Farach-Colton, et al.[31], and a Concept Lattice Generation using RDF are examples of these new approach[27].

There are many algorithms in the literature and in time, more are added. Not surprisingly, the processing time of the concept lattice and complexity analysis of the new algorithms with respect to the former ones has decreased a lot. In this section, we mention only some algorithms. The literature is rich, if the reader wants to follow the new algorithms related with Concept Lattice Generation and Navigation.

2.5 FCA APPLICATIONS.

FCA comes from data analysis requirements. From FCA, early apparition, has been strongly linked with the theory of Knowledge Discovery[93, 85]. There are even papers, which recursively examines the concepts generated by FCA using the Lattice of Concepts as ontology[75]. **Ontology** in its original sense is a philosophical discipline dealing with the potentialities and conditions of being. **Parmenides** was among the first to propose an ontological characterization of the fundamental nature of reality[69]. Within Computer Science, ontologies have been used for more than, they represent a method for formally represent knowledge.[42]. There are a lot of proposals for standard knowledge representation format which are

independent of the content of knowledge being exchanged or communicated, establishing agreements about knowledge, such as shared assumptions and models of the world.

Ontologies play a significant role in software source structure analysis for example, studies for features searching in source code and source structure analysis has been researched[24, 30]. In this sense, many methods have been proposed in FCA, expressing the concept lattice as ontology. They use the inherent logic of that structure to understand and extract independent content knowledge[9, 48, 85, 8, 100].

As we have just seen, ontologies have the status of a model. Their purpose is to understand a shared interpretation of the reality. Formal Concept Analysis, on the other hand, plays other role with the same finality. Concept lattices are not understood as modeling some part of the reality, but rather it is an artifact to do domain analysis using a given data.[81]. While ontologies can be established without any given data, FCA relies always on some set of objects. Thus, in FCA, extensional and intentional aspects are equally important, while ontologies emphasize on the intentional part[42]. FCA can be used as a technique for Ontology Engineering. It supports the structuring of some given data by construction of concept lattices. In addition, it can be used to extract, from a given dataset, a conceptual hierarchy which may serve as a basis for the manual or semi-automatic development of an ontology. In this sense, Biological[86], Medical[3, 61] and Chemical[36] Ontologies have been reported as part of the FCA study.

Other area of interest to FCA is Databases systems. To access data stored in databases and using the information that emerges from those databases, new techniques are developed to automatically discover knowledge. The subfield of Datamining, to find the useful knowledge in databases, is named Knowledge Discovery in Databases (KDD). It is referred as the process to find outstanding rules, trends, patterns and relations, focusing on reusable information that is useful for predict behaviors. Association rule mining and learning is a popular and well researched method for discovering interesting relations between variables. KDD is another area that has been strongly attracted to FCA. In the case of **Mining of association rules** and **dependencies**[58] KDD has been strongly supported by FCA. The main aspects in which, FCA has been involved with KDD is the calculation of frequent itemsets[96], minimal generators[28, 85] and closed sets[49, 76], among others[64]. FCA has focused on decrease computing costs of mining of association rules, using lattices theory to reduce the search space and find better representations of these sets, making the search more efficient.

The first time FCA contributes to the field of association rules was with the development of new algorithms for frequent itemsets discovery. These results only took into account the closed set, pruning of the lattice was not considered. As we seen before Algorithms to find all closed itemsets (Formal Concept Set) were developed and those algorithms were used for find frequent closed itemset. Later, Duquenne Guigues described a minimum set of exact rules from which can

be derived all other rules[44]. Connections between this statement minim generators and FCA were published.[34] And although up to that point both fields were independent, several research groups bet on this combination. Since then, the attention of FCA has increased within the data mining community collaboration. Some work, of FCA in KDD, focuses on reduced representations or minimal generator of all rules.

Frequent itemset mining is one of the main issues discussed in the scope of FCA and is a crucial step in find Association Rules. The problem definition of association rule mining was introduced in 1993 by Agrawal, R. et. al.Agrawal [2]. Some years later, the research linked directly this definition with FCA. The approach of the problem of association rules, considered important part of Knowledge Discovery in Databases, as we mentioned above, requires detecting frequent patterns. The most frequent patterns, called Frequent Itemset, are the engine of this paradigm.

One of the main problems concerning the algorithms to compute frequent closed itemsets is the number of disk accesses required. The first algorithms were inspired by the Apriori method[1]. All these algorithms have in common the level-wise search and the generic closed itemsets search. for example, ChARM[107] also follows a bottom-up approach. Contrary to In-Close Algorithm[4] it performs a depth-first search in the powerset of itemsets. It stores the frequent itemsets in a prefix tree in main memory. And then, the algorithm traverses both the itemset and transaction search spaces. As soon as a frequent itemset is generated, the set of corresponding transactions is compared with those of the other itemsets having the same parent in the prefix tree. If they are equal, then their nodes are merged in the prefix tree, as both generate the same closure. Closet Algorithm[73] also compute the frequent closed sets and their supports in a depth-first manner, storing the transactions in a tree structure, called FP tree, inherited from the FP Growth algorithm[82]. They use a similar ChARM merging strategy. MaFia Algorithm[18] is an algorithm which mainly is intended for computing all maximal frequent itemsets (which, by basic results from FCA, are all close). It also has the option to compute all frequent closed itemsets. Bamboo [98] mines closed itemsets following a length decreasing support constraint, which states that large itemsets need a lower support to be considered relevant than small ones.

The area of marketing and corporate decision has been benefited from these algorithms. There are many software packages in the market that are especially dedicated to this type of data mining. They use as background FCA algorithms.

Other common application area of FCA is **Machine Learning**. For example, in cooperative e-Learning and semantic web Beydoun [13, 12] FCA is used for knowledge acquisition and as a system to process and exploit dependencies between nodes to yield subsequent and more effective focused search results. One model of machine learning that also uses closure systems in machine learning is the JSM-method[106]. Several versions of JSM-method has been developed.

There exist a JSM Method FCA based. This method uses positive and negative hypotheses, and search conditions in the intersections of these hypotheses. They first search them individually, then more inferences and constrains are found by hypothesis intersection[57]. Given a negative and a positive hypotheses, JSM-method are suitable to classify undetermined examples. In[106, 97] you can found a complete research about JSM-method techniques. Some applied work have been published for **Toxicology analysis**[36], **Carcinogenicity of Chemical Compounds**[54] and Prediction of the Toxicity of Chemical Compounds[92]. This applications provides an excellent compilation for discover and understand the reasoning technique on JSM-method. There exists, a lot of applications of JSM-method related with **Bayesian inference**[11] , **Decision Trees**[10, 22], and **inference** in general. **Spam Filtering**[105], **Software Patterns Analysis**[23, 24], **Structured Pattern Analysis**[33, 72], **Software Engineering** are some examples of applications of this theory.

The second model of machine learning, **Concept Lattice-based Artificial Neural Network** (CLANN) uses constraints transformations for supervised classification[87]. The first step of this classifier, learns relevant concepts from the data. The second step is build a join semi-lattice of concepts applying constraints on those relevant concepts. Finally, it convert the semi-lattice to a Artificial Neural Network topology. The proposed algorithm uses frequency and the validity of a concept, to select them. Then, a mapping from a join semi lattice is performed. After that mapping, a connection weights and threadholds are initialized. Finally the backpropagation algorithm is used to find those weights for an extrapolation phase. **M-CLANN: Multiclass Concept Lattice-based Artificial Neural Network** (M-CLANN) is an extension of CLANN in order to threat multi-class data[65].

The third studies related with machine learning in FCA are **composite classifiers based on Concept Lattice**[103][45, 63].

Other example of FCA application is image learning, also named, **Image characterization**[108]. In this case this image characterization is performed using image features and object class partitions. During extrapolation time, the system reviews if new data is a superset of the conditions that belong to a class.

As we have seen FCA is still growing. The previous literature review proves the feasibility of the field. All these topics were reviewed in terms of verify the scope of FCA in applications. Many of these papers served as inspiration of this thesis.

2.6 LATTICE BASED NEURAL NETWORKS.

Artificial Neural Network (ANN) is a paradigm of learning and automatic processing inspired by the nervous system. Figure 5 shows standard neuron structure. This structure is the inspiration of LBNN presented in this chapter.

The features of ANN make them quite suitable for applications where there is no a priori of an identifiable pattern that can be pro-

grammed, but there is a basic set of input examples (previously classified or not). They are also highly robust to noise so as to dysfunction of concrete elements and, moreover, ANN are parallelizable. As we said early, our work is related with LBNN, which is also considered an Artificial Neural Network. ANN merges Computer Science with Neuroscience, in order to take advantage of recent advances in neurobiology and the biophysics of neural computation.

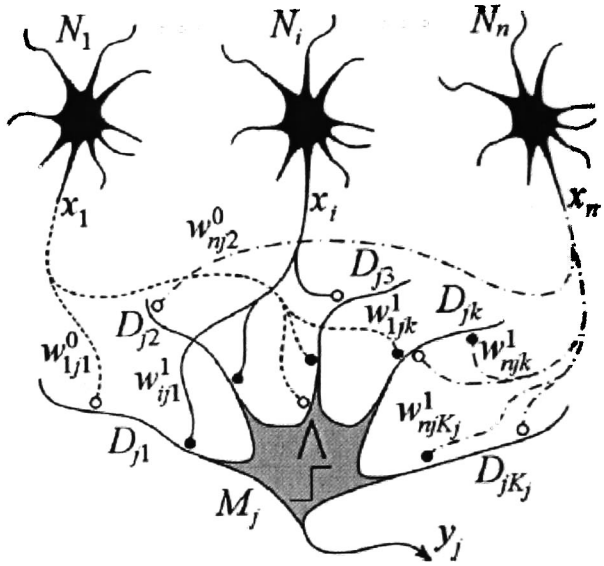
The theory of Lattice Based Neural Networks (LBNN) is actively used in classifiers, Ritter and Urcid [77], Urcid et al. [89], Barmpoutis and Ritter [7]. clustering, Kaburlasos [51], associative memories, Aldape-Perez et al. [3], Ritter et al. [80], among others[102, 90]. Some characteristics of these networks is their analogous morphology to the structure of the brain[78]. In this analogy, dendrites are considered an essential part of the computation. In biological neurons, the terminal axonal branches make contact with the soma and the many dendrites of other neurons [84]. Dendrites form the major receiving part of neurons. It is within these highly complex, branching structures that the real work of the nervous system takes place. The dendrites of neurons receive thousands of synaptic inputs from other neurons. However, dendrites do more than simply collect and funnel these signals to the soma and axon. They shape and integrate the inputs in complex ways. Despite being discovered over a century ago, dendrites received little research attention until the early 1950s. Over the past few years there has been a dramatic explosion of interest in the function of these structures. New research has increased out understanding of the properties of dendrites, and their role in neuronal function, Stuart et al. [84]. Most of the synapses occur on the dendritic tree of the neuron, and some researchers claims the information is processed in those dendrites. Part of this is due to the fact that pyramidal cell dendrites span all cortical layers in all regions of the cerebral cortex, Arbib [5].

Basicly, in LBNN model, an input layer receives external data. Subsequent layers perform the necessary functions to generate the desired outputs. This can range from determine which class belongs given input to the reconstruction of a pattern. Noise removal and image segmentation are two examples of applications of this kind of ANN[88, 91, 102]. Applications in a variety of disciplines have employed LBNN. Some aspects of LBNN have attracted considerable attention which is partially due to their useful applications in a variety of disciplines[53, 46].

Single Lattice Layer Perceptron, also named Dendritic Single Layer Perceptron, is basically a classifier in which exists a set of input neurons, a set of output neurons, and a set of dendrites, growing from the output neurons. Those dendrites are connected with the input set by some axonal branches from those input neurons. A training set configure those outputs based on the maxima $\bigvee$ and minima $\bigwedge$ operations. They are derived from the algebra $(\mathbb{R}, +, \bigvee, \bigwedge)$. Figure 5 represents this structure.

Lattice Based Neural Network Theory comes mainly from Lattice and Fuzzy Theory,Kaburlasos [51]. In this section, we will show only

Figure 5: LBNN Basic Structure



Published at Computational Inteligence Based on Lattice Theory[52].

how to do computation from dendrites to neuron and how to compute the output of that neurone in SLLP. Advanced information and a complete introduction in the subject is in Computational Inteligence Based on Lattice Theory,Barmpoutis and Ritter [7].

In SLLP, a set of n input neurons N_1, N_n accepts input $x = (x_1, ..., x_n) \in \mathbb{R}^n$. An input neuron provides information through its axonal branches to the dendritic trees of the set output neurons. A set of O output neurons is represented by $O_1, ..., O_m$. The weight of an axonal branch of neuron N_i connected to the k_{th} dendrite of the O_j output neuron is denoted by w^ℓ_{ijk}, in which, the superscript $\ell \in \{0,1\}$ represents an excitatory $\ell = 1$ or a inhibitory $\ell = 0$ input to the dendrite. The k_{th} dendrite of O_j will respond to the total value recived from the N input neurons set, and it will accept or reject the given input. Dendrite computation is the most important operation in LBNN. The following equation $\tau^j_k(x)$, from **SLLP**, corresponds to the computation of the k_{th} dendrite of the j_{st} output neuron[77].

$$\tau^j_k(x) = p_{jk} \bigwedge_{i \in I(k)} \bigwedge_{\ell \in L} (-1)^{1-\ell}(x_i + w^\ell_{ijk})$$

Where x is the input value of neurons $N_1, ..., N_n$ and x_i is the the value of the input neuron N_i. $I(k) \subseteq 1, ..., n$ represents the set of all input neurons with synaptic connection on the k_{th} dendrite of O_j. The number of terminal axonal fibers on N_i that synapse on a dendrite of O_j is at most two, since $\ell(i) \subseteq \{0,1\}$ Finally, the last one involved value is $p_{jk} \in \{-1,1\}$ and it denotes the excitatory ($p_{jk} = 1$) or inhibitory ($p_{jk} = -1$) response of the k_{th} dendrite of O_j to the received input.

All the values $\tau_k^j(x)$ are passed to the neuron cell body. The value computation of O_j is a function that computes all its dendrites values. The total value received by O_j is given by[77]:

$$\tau^j(x) = p_j * \bigvee_{k=1}^{K_j} \tau_k^j(x)$$

In this SLLP model, K_j is the set of all dendrites of O_j, $p_j = \pm 1$ represents the response of the cell body to the received input vector. At this point, we know that $p_j = 1$ means that the input is accepted and $p_j = -1$ means that the cell body rejects the received input vector.

The last statement related with O_j correspond to an activation function f, namely $y_j = f[\tau^j(x)]$.

$$f[\tau^j(x)] = \begin{cases} 1 \iff \tau(x) \geq 0 \\ 0 \iff \tau(x) < 0 \end{cases} \tag{1}$$

As, we mention early, dendrites configurations is computed using a training set. There are in SLLP, two algorithms in order to complete the training task: **Merge and Elimination** methods. Present section only lies with the basic structure of the SLLP. Those methods, and a lot of applications can be reviewed in literature. Basically the computation of a neuron, which represents a class, is made by checking if the element is classified within the regions formed by, excitatory and inhibitory hyper-boxes.

2.7 CONCLUSION.

As we have seen, Lattice Theory has a strong presence in the world of mathematics, engineering and recently Industrial Research. It is a fact that Lattices Theory still moves on, and continues growing with respect to their applications and base. More and better, algorithms and applications emerge to solve more interest problems every day. We can see that this theory, which, naturally, studies a topological space associated to an abstract world, is an excellent tool for solving many important problems in Computer Science.

Section 2 showed how FCA born and evolution from Lattice theory. Subsection 2.3 formally presents definitions used in this thesis scope. A section dedicated to show the state of FCA applications was presented in the state of art in FCA.

Practical applications, such as finding the set of frequent items, association rules, classifiers, cryptography, network analysis, semantic web, analysis of DBPL, marketing demonstrate the ability and power of FCA.

Chapter 3 of this work is focused in develop our own algorithm for generating Concept Lattice.

Part II

APPLICATIONS

Concept lattice generation algorithms plays an essential role for the application of FCA. As we studied before, many algorithms have been proposed for generating the Concept lattice from a binary relation, each one with their own properties and structures. This Part presents two applications of research scope. Chapter 3, proposes a nobel method for Concept Lattice Generation. The construction of this method is highly related with the Lattice Based Neural Networks Theory. You will find also, an experimental analysis on our Algorithm for Concept Lattice Generation. In chapter 4 we propose a classifier. In this chapter you will find the notions of how to extrapolate knowledge, understand the learning process using the power of the definitions of FCA and Lattice Theory. That, in order to achieve the goal of formal classification on binary relations.

3

COMPUTING CONCEPT LATTICE USING LATTICE BASED NEURAL NETWORK THEORY

3.1 INTRODUCTION.

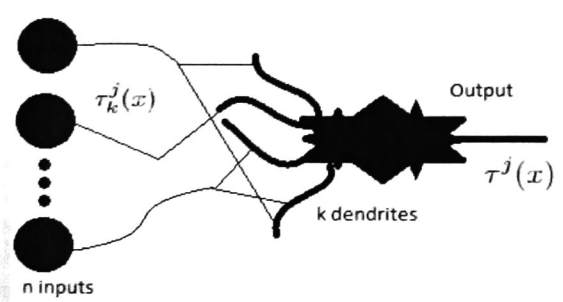
This chapter proposes and algorithm for the generation of all formal concepts. This is achieved through the use available FCA theory[101, 99, 38, 37], lattice tools[6, 14] the search of the Maximal Rectangles Set, Intent Cardinality and **Lattice Based Neural Networks** (LBNN)[79, 78]. Also, we will use the intervals between those maximal rectangles sets for the construction of the Concept Lattice. In addition, the proposed algorithm can compute the diagram graph without extra effort.

As discussed above, there are many algorithms dedicated to the generation of this structure. But, none of them have used Lattices Based Neural Networks theory to achieve their goal.

As we have seen early, computing unit of the LBNN is the dendrite. The dendrites in the LBNN split into two types: positives and negatives dendrites. The combination of positive and negative dendrites generates rectangles. Similarly those rectangles can be classified in two types, those in which the input is accepted and rectangles which reject inputs. It has also been explored their ability to orthonormalize this structure for better results[7]. In this chapter we include theory in order to find the maximal rectangles in a binary formal context. Definitions on LBNN are bounded by this goal. Our idea is to use a similar method to compute the maximal rectangles, given a set of binary vector and using the cardinality notion to extract those maximal rectangles.

The chapter is organized as follows. Next, section .Section 3.3 contains the complete description of the Concept Lattice Generation LBNN based. In section 4.6 you will find the experimental analysis,

Figure 6: Standard Dendrite Neuron



in which we compare our algorithm with some other existent algorithms. At the end of this chapter you will find the conclusion and some notes and observations.

3.2 REVISITING UPWARD AND DOWNWARD CLOSED SETS.

Before of LBNN, in this section, we define two notions. First one is the minimum coverage set in a binary context $\mathbb{K}$. The second definition, is the maximal coverage set. We will work on their properties, and we will define their usages over the binary context.

Let $G' := \{m \subseteq M | m = g', g \in G\}$ be the objects derivative set.
Let $M' := \{g \subseteq G | g = m', m \in M\}$ be the attributes derivative set.

Definition 11. The **Minimum Coverage set** D^-(or Negative Dendrite Set) over $\mathbb{K}$, for object derivative set, is described as follows:

$$D^- := \{d \in G' \mid \forall m \in G', m \neq d, d \not\supseteq m\}$$

This is, all elements d in G such that d is not a superset of any other element m in G'.

A dually definition on the attributes derivative set is defined.

The following definition is focused on, the maximal coverage set associated with a the formal context. In this case, we got the maximal rectangles in a given context.

Definition 12. The **Maximal Coverage Set** D^+(or Positive Dendrite Set) over G' is defined as:

$$D^+ := \{d \in G' | \forall m \in G', m \not\models d, d \not\subseteq m\}.$$

It means, all elements d in G' such that d is not a subset of any other element m in G'.

A dually definition on the attributes derivative set is also defined.

Definition 13. As the reader may guess, the alternative name in these definitions shows the purpose for which this pair of sets will be used. But that will be done later. Here, we explain and exemplify some basic notions about these definitions.

Claim 1. The conditions sets D^-, D^+ are anti-chains sets over G derivatives.

Since, any poset is reflexive and transitive we can do five more claims:

Claim 2. Let $x \subseteq M$ and let $\uparrow$ be defined in the M Power Lattice Set. If $x \supseteq d$ for some $d \in D^-$, then $x \in\uparrow G'$. It means: $x \in\uparrow D^- \iff x \uparrow G'$

Example 1. Let $x := \{abc\}$, $G' := \{\{abce\}, \{abd\}, \{ab\}, \{cd\}, \{abde\}\}$, then $D^- : \{\{ab\}, \{cd\}\}$ where $x \in\uparrow D^-$ and $x \in\uparrow G'$.

Claim 3. Let $x \subseteq M$ and let $\downarrow$ be defined in the M Power Lattice Set. If $x \subseteq d$ for some $d \in D^+$, then $x \in\downarrow G'$.

Example 2. Let $x := \{abc\}$, $G' : \{\{abce\}, \{abd\}, \{ab\}, \{cd\}, \{abde\}$, then $D^+ : \{\{abce\}, \{cd\}, \{abde\}\}$ with $x \in\downarrow D^+$ You can easily verify that $x \in\downarrow G'$.

Claim 4. Let $x \subseteq M$. If $x \subset d$ for any element $d \in D^-$, then $x \notin G'$

Example 3. Let $x := \{a\}$, $G' : \{\{abce\}, \{abd\}, \{ab\}, \{cd\}, \{abde\}$, then $D^- : \{\{ab\}, \{cd\}\}$. You can easily verify that $x \neq m$ for any $m \in G'$.

Claim 5. Let $x \subseteq M$. If $x \not\subseteq d$ for any element $d \in D^+$, then $x \not\subseteq m, \forall m \in G-'$

Example 4. Let $x := \{be\}$, $G' : \{\{abce\}, \{abd\}, \{ab\}, \{cd\}, \{abde\}$, then $D^+ : \{\{abce\}, \{cd\}, \{abde\}\}$. You can easily verify that $x \not\supseteq m$ for any $m \in G'$

Claim 6. The cardinality of the minimum and maximal coverage sets is bounded by $|G|$ assuming G' is itself, a set of anti-chains.

Thus, these sets are bounded by $|G|$.

All these claims are defined and exemplified in the object derivative set.

Next claim, defines a relation between a lattice infimum and the set D^+.

Claim 7. Let (A_t, B_t) be the infimum element in $\mathfrak{B}(\mathbb{K})$. Any element (d', d), where $d \in D^+$ for G', is closed and there is not any other formal concept $(A_j, B_j) \in \mathfrak{B}(\mathbb{K})$ such that $d \subseteq B_j \subset B_i$ and $d' \supseteq A_j \supset A$.

Based, in our last claims, next section 3.3 construct an algorithm for Concept Lattice Generation.

3.3 COMPUTING CONCEPT LATTICE WITH A LBNN MODEL.

In our LBNN model for find maximal rectangles set. A set of binary patterns are represented by G' So the binary representation of a formal context, is itself a set of patterns, in which the derivative of each object is an element $x \in G'$, as we seen before. We can define $x = (x_1, ..., x_n) \in \mathbb{B}^n$ as a vector. In the same way as SLLP, a set of n input neurons $N_1, ..., N_n$ is directly associated with the x input. We also know that the value of each x_i in x is binary, so we can say that each x_i is an input for N_i and $x_i \in \{0, 1\}$. A simple class formulation is used for learning the maximal rectangles set, in these terms one models of dendrite computation are proposed. Those two proposes are based in claim 7 and uses the set D^+ for objects derivatives. The first model is based on an LBNN that classify upper closed set members on this set. The second, is based in the an operation that represents the lower closed set on the same set. A unique output neuron j with k dendrites is provided with information from an axonal branches from the input neurons. In this section we works with the second propose.

Other particular thing is the fact that for us is sufficient to use only the notion of maximal or minimum to generate rectangles, it means our superscript ℓ will be a constant $\ell = 1$ or $\ell = 0$ depending on the perspective. In the same way we can use only excitatory or inhibitory dendrites. So, p_{jk} is also a constant, $p_{jk} = 1$ or $p_{jk} = -1$ and it denotes the excitatory or inhibitory response of the k_{th} dendrite of M_j to the received input, another remarkable thing is the fact that we only need to connect maximal or minimum axonal branches which of depends on our algorithm perspective. But the simplest way to compute the value of the k_{th} dendrite, using claim 2, derived from the SLLP equations, is:

$$\tau_k^j(x) = \bigvee_{i \in I(k)} (x_i) \tag{2}$$

Where $\tau_k^j(x)$ is the value of the computation of the k_{th} dendrite of the j_{th} output neuron given a input x, $I(k) \subseteq \{1,...,n\}$ is the set of input neurons with terminal fibers that synapse the k_{th} dendrite of our output neuron. We realize that all weights w_{ijk}^{ℓ} are equal to zero, this is, for our maximal rectangles classifier, we only need to store zero values from the input patterns at training step, our goal is that the output of our classification neuron be $x \in C_1$ if the input x is an maximal rectangle given the patterns that currently holds our dendritic neural network. Later, in this chapter, you will find a detailed discussion about the goal and the learning step.

Specifically, each dendrite k corresponds with one maximal rectangle intent to be tested, $I(k)$ is the incidence set of positions where the value of the maximal rectangle is zero for the pattern represented by the k dendrite.

We get the state value of M_j computing the minimum value of all it's dendrites. Again, as the SLLP, each $\tau_k^j(x)$ is computed, and it is passed to the cell body of M_j. Then we can get the total value received by our output neuron as follows:

$$\tau^j(x) = \bigwedge \tau_k^j(x) \tag{3}$$

It is easy to realize that the activation function is not required since $f[\tau^j(x)] = \tau^j(x)$ where $\tau^j(x) = 1$ if $x \not\leq y$ for all $y \in C_1$ and $\tau^j(x) = 0$ if $x \leq y$ for some $y \in C_1$. As we mentioned above x is a maximal rectangles if and only if $x \not\leq y$ and $y \not\leq x$ for all $y \in C_1$. Using the previous statement we can ensure that half of the work is done, the second test, $y \not\leq x$, will be performed processing data in a particular order. In our case, we use cardinality order, objects with more attributes first. Cardinality order, previously defined notion, ensures that each new computed row is minor than the previously computed rows.

As we said before, the idea is to use our LBNN structure to classify maximal rectangles. When we start to compute a formal concept, our structure is empty, it means, it hasn't any dendrite or any ax-

Algorithmus 3.1 addDendrite

```
INPUT: NeuralNetwork P,Pattern x
OUTPUT: Updated P
    Dendrite k = addNewDendrite(P)
    FOR EACH element in x
        IF getValue(element) = 0
            i = getPosition(element)
            addAxonalBranch(k,i)
        END
    END
END
```

This function adds, as a dendrite, a new Pattern to a LBNN.

onal branch connection, so the first step is add each element of the maximal cardinality as a pattern to learn.

Algorithm 3.1 shows how an element is added to our lattice neural network for maximal rectangles learning.

First, Algorithm 3.1 receives as parameter the LBNN which is being trained and a binary vector. As we will see below, this binary vector has been proven as a maximal rectangle. The Algorithm 3.1, first grows a new dendrite k_{th} in our output neuron O_j. Every column in x is checked, if that property is not contained by the object x, then an axonal branch grows from the i position of the input neuron set to the new dendrite. This operation is represented by addAxonalBranch calling.

Now that we know how a dendrite grows in our neural network, we must ensure that only maximal rectangles are presented as a part of the training.

Algorithm 3.2 shows how to, compute the Concept Lattice, computing **Maximal Rectangle Sets Recursively**.

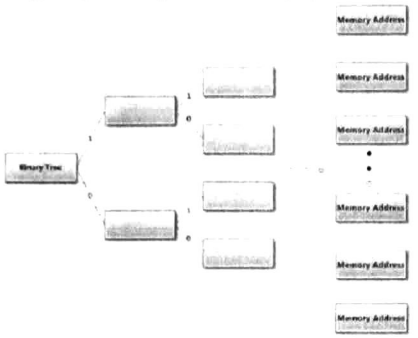
In algorithm 3.2: **First step** is create a new dendritic neural network. It must be initialized with one output neuron, $n = |intent|$ and $k = 0$. Where the number of input neurons is n, and each one neuron represents one attribute element in intent. **Second step** is get G set ordered by G' attribute cardinality. **Next**, third step, if the maximal cardinality of the elements in G' is equal to the K-Supremum intent, we add a link between Supremum and Infimum and stop. Otherwise, we must add to our dendritical neural network structure all the elements in G' with the maximal cardinality, those elements are maximal rectangles in the given binary context. Once all elements, with maximal cardinality, were processed. **Fourth step** is to check remaining elements. If an element derivative does not exists already, as an intent, and the evaluation 3 says that it is a maximal rectangle then we must to add the object and his derivative as a new maximal rectangle. Otherwise, if the element derivative already exist we will add it to the previously formal concept as extent. **Last step**, we will to process each maximal rectangle founded. If that element is already contained in Lattice, we add a link between K-Infimum and the previously element created in Lattice. Otherwise, we add that link, we process that formal con-

Algorithmus 3.2 Compute Maximal Rectangles

```
INPUT: A Binary Context K:(G,M,I), K-Supremum, K-Infimum, Lattice
OUTPUT: Intent HashSet Maximal_Rectangles
STEP 1:
        Init:
        LBNN Upward Structure
        Maximal_Rectangles
STEP 2:
        Sort G by derivative higher to lower Cardinality
STEP 3:
        IF maximal cardinality is equal to K-Supremum intention
            cardinality
                Add Link from K-Supremum to K-Infimum
                RETURN
        Add, as positive dendrites, all elements with maximal
            cardinality in G' to LBNN.
STEP 4:
        Foreach remaining element in G
                IF Maximal_Rectangles does not contains element'
                    AND Upward evaluation element' is 1 then:
                        add, as dendrite, element to LBNN
                        create Formal Concept with:
                                element union K-infimum as extent
                                element' as intent
                        add this new formal concept to:
                                Maximal_Rectangles
                OTHER IF Maximal_Rectangles contains element
                        add element to previously Formal Concept
                            created.
STEP 5:
        Foreach Rectangle in Maximal_Rectangles
                IF Lattice does not contains Rectangle
                        add Rectangle to Lattice
                        add a Link from Rectangle to K-Infimum
                        Compute Maximal Rectangles with:
                                G = G / Rectangle extent
                                M = Rectangle intent
                                I = Projection
                                K-Supremum
                                Rectangle as a K-Infimum
                                Lattice
                ELSE
                        add a Link from previously created
                            Rectangle in Global_Maximal_
                            Rectangles to K-Infimum
```

This recursive algorithm is used to compute the concept lattice.

Figure 7: Binary Tree for $\mathfrak{B}(\mathbb{K})$ storage.



Each formal concept added is represented by his intent part. the binary tree is used to Link a set of formal concepts input to a consecutive memory address.

Algorithmus 3.3 Main Function

```
INPUT: BinaryContext(G,M,I)
OUTPUT: Lattice L
Step 1: Get Maximum and Infimum elements.
    FormalConcept max = getMax(G,M,I)
    FormalConcept min = getMin(G,M,I)
    addConcept(L,max), addConcept(L,min)
STEP 2: Get maximal Rectangles From min
    MaxRectangles = Compute Maximal Rectangles with:
        G/min.extent,
        min.intent,I-Proj,
        max,
        min,
        L
```

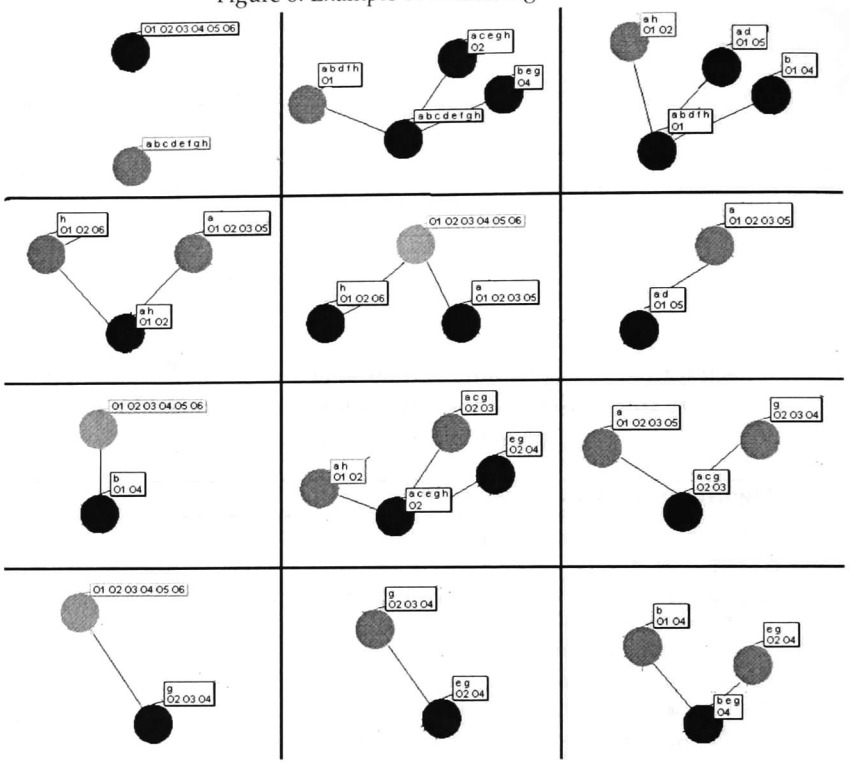
This algorithm triggers the process for compute the concept lattice.

cept recursively. And, finally, we add this new element to the Lattice Structure.

Algorithm 3.3 shows how the recursive process is triggered from the maximal element of the lattice.

At this point, we can compute all the concepts given by the binary context. But, as we seen before, $\mathfrak{B}(\mathbb{K})$ cardinality is bounded by an exponential number. A search in $\mathfrak{B}(\mathbb{K})$ exploit the algorithmic complexity to exponential delay time. A simple way to avoid this issue is to use a **Binary Tree** to store and recover all elements in$\mathfrak{B}(\mathbb{K})$. Basically, this represents binary tree serves as hash function. Basically, this binary tree serves as hash function. The idea is to use the concept of intent in their binary representation. Figure 7 represents a binary tree in which the leaf nodes are linked to memory addresses which is stored in the formal concepts. Using this structure we do a clean and unbounded cardinality of $\mathfrak{B}(\mathbb{K})$ search, finding and adding any intent in $|M|$ steps.

Figure 8: Example of Processing



This computations is done in a depth search way. All those concepts are finally stored in a binary tree structure.

Using the formal context 2 presented in the first chapter, Figure X shows the process and the order in which the elements are computed from the lattice. In this process, we can observe that the formal concepts are not processed twice. In addition, the processing order enables us to generate the edges of the Hasse diagram without additional computing steps. The process of generating the concept lattice and Hasse diagram presented in this paper meets the basic rules to be considered an expected polynomial algorithm. That is, there is no exponential search or update the lattice. Although it is true, it is a batch algorithm, we can prove a good efficiency and performance of the algorithm.

Now that we have described the process of generating the concept lattice, in the next section we will make an experimental analysis.

3.4 EXPERIMENTAL ANALYSIS.

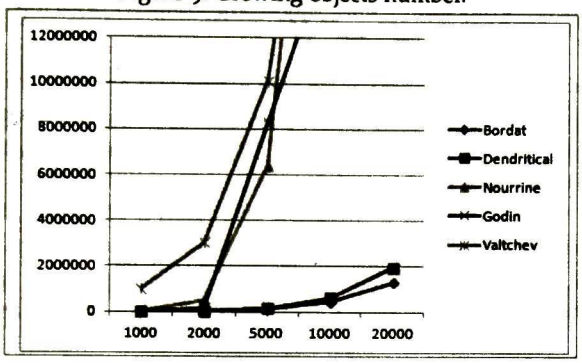
As shown in Sergei O. Kuznetsov et. al.[56] many parameters are involved in the performance and running time of an algorithm. For our tests, it was considered, the number of objects, the number of attributes, the density of the context and the worst case for contexts of $M * M$, also called diagonal context. Those parameters were tested

independently, and four algorithms were selected to compare our algorithm performance. In experimental analysis charts our algorithm is named Dendritical.

With the exception of the contexts for M * M, the datasets were taken from the website fcarepository, a collection of binary contexts available to the general public.

The algorithms were implemented in java environment and the tests were run on the same machine with the same O.S.

Figure 9: Growing objects number.



How grows execution time when the number of objects grows, from 1000 to 20000, with 100 attributes and 10% of density.

Is important to understand that each of these algorithms proved here, has independent properties, which emphasizes: Valgrin, Nourrine and Godin incremental processing, which increases the execution time of these algorithms with respect to the Bordat and our algorithm.

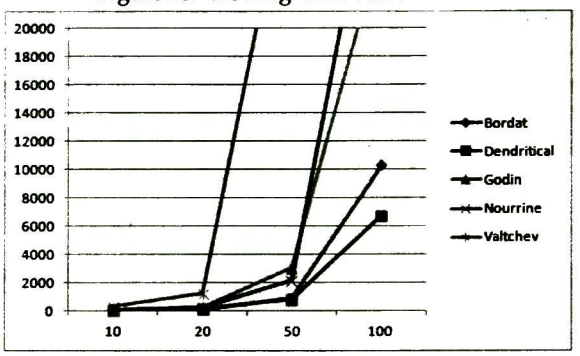
Figure 9 shows the **execution time** behavior for a formal context with 10% density and 100 attributes. The number of objects is growing from one thousand to twenty thousand, obviously, with more objects the number of formal concepts grows when the density is constant. We can verify that Bordat algorithm has the better performance when the number of objects grows.

Figure 10 shows how grows the time execution when the number of attributes grows, all datasets for this test has a 10% density and 1000 objects. the number of attributes is growing from ten to one hundred, and, the number of formal concepts is growing too. Here we can notice that Bordat and Dendritical algorithms, are faster than Godin, Nourrine and Valtchev algorithms because Bordat and Dendritical algorithms are not incremental. We can also notice that Dendritical execution time behavior is faster when the number of objects grows.

Figure 11 shows how grows the time execution when density becomes higher, all datasets has 1000 objects and 100 attributes and obviously when the density grows the number of formal concepts grows too. Density is mainly the percentage of attributes for all the objects in the formal context. Here, we notice that when the density grows Dendritical is closer and even faster than the other algorithms.

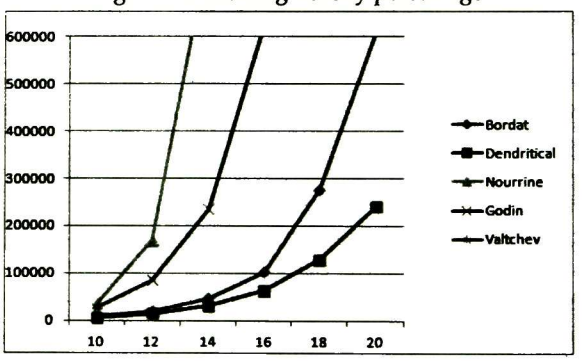
Figure 6 shows running time for diagonal contexts where $|G| = |M|$ and yields the complete lattice, it means $2^{|M|}$ formal concepts.

Figure 10: Growing attributes number



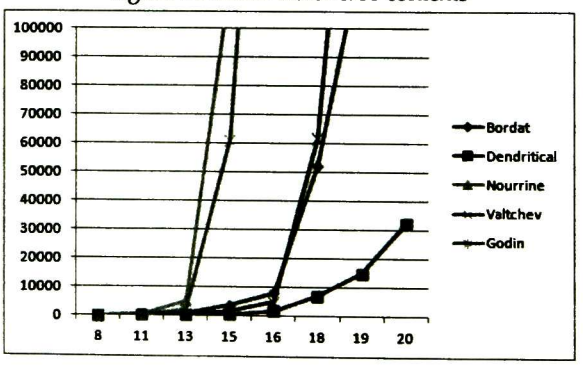
How grows execution time when the number of attributes grows from 10 to 100 with 1000 objects and 10% density case.

Figure 11: Growing density percentage



How grows execution time when density percentage grows from 10 to 20 percent. Here, 1000 objects and 100 attributes case.

Figure 12: MxM worst case contexts



Algorithms running diagonal contexts in which number of attributes is growing from 8 to 20 attributes. This kind of context generates $2^{|M|}$ formal concepts.

Table 5: Execution time of the dendritical algorithm.

G	M	Density	QT	Ordering	Dendritical	Total
1000	*36*	*17*	722ms	20ms	49ms	791ms
1000	*49*	*14*	924ms	67ms	101ms	1092ms
1000	*81*	*11*	1853ms	124ms	201ms	2178ms

G is the number of objects. M is the number of attributes. QT is the time of query's over each formal concept process projection. Ordering is the time in which elements are sorted. Dendritical is the time in which dendritical classifier goes on. Finally, total is the running time.

In this kind of formal contexts we can see a clear running time superiority of dendritical algorithm. We can also notice that Dendritical algorithm mantains some stability with the execution time relation.

Table 3 shows how our implementation grows in each dendritical algorithm step. As a note, this dendritical implementation handles sets and binary representation for each stage. Removing sets structures. In this case using a purely binary representation, we got a better results.

3.5 CONCLUSION.

In this Chapter, we presented an algorithm based on the idea of Maximal Rectangles, cardinality notion and LBNN theory. We have also compared it with some known algorithms for the construction of Concept Lattice.

From the tests presented in this paper, we can see that when the number of objects grows, our algorithm time execution grows more than Bordat or other methods, but it could be a better choose when the density or the number of attributes is high. Also, the results shows a good performance for the $M \times M$ contexts worst case.

Another interesting feature that will be studied in future work is the possibility of parallelization of this LBNN based Algorithm. Similarly, using this algorithm idea a bottom-top approach (for intents) must be performed.

It is also necessary a wider study of Dendritical Algorithm for lattice generation, and a comparisson with more and sofisticated new algorithms for lattice construction. However, in this paper we demonstrate that using our algorithm is feasible and scalable to large datsets.

SUPERVISED LEARNING USING FCA.

New ideas, theories, methodologies and engineering techniques have emerged from the Machine Learning Theory. In this field, many algorithms related to self-organization and extrapolation of knowledge have gained importance. Those algorithms, also named as algorithms for classification and prediction, have been used for design systems where the learning of the environment is the main objective. With the learning idea in mind, individually algorithms have demonstrated certain skills and more than acceptable results in industry and research field.

This chapter proposes a method for the use of FCA in supervised learning. The goal of this method is based in the use of Lattice Theory as driver between different threadholds, binary functions, conceptual scaling generating a descriptor and a classification rule. We show how in this process is possible a clean and clear design. With those goals, we will concentrate on methods published in the machine learning literature, as well as methods from other fields that have had considerable impact on the machine learning literature. We will also use as example a basic binary function over the Computer Vision Environment, that can be used to serve a higher classification purpose.

The main objective of this chapter is to show the proposed method to classify, using categorical classifiers, threadholds or binary functions. Here, we prove that our generated classifier is feasible. Note that the experiment show the reader how this method works, which include the application of classify from binary threadholds.

This chapter is organized as follows: In the previous section 2.2 is an overview of what is regarded by our method, in addition you will find a basic review of learning models that concern us for this work. It focuses especially in the categorical output learning.

Section 4.1 shows the definition on which the classifier presented in this chapter works. Section 4.2 defines the Classification Rule used in this chapter. It defines a Downward/Upward membership function over Maximum and Minimum Coverage. Section 4.3 defines in equations terms the proposed classification rule. In this section, we also define a simple extension to solve the multiclass problem using the classification rule previously presented and the formal context partition notion. Section 4.4 defines in lattice terms the proper binary definition, which is useful for data revision. Section 4.5 explain the dataset in which our classifier will be analyzed. Next, in section 4.6 you will find the results of the proposed experiment together with the observations. Finally, you will find, in section 4.7 conclusions and remarks.

4.1 BINARY CONTEXTS AND LATTICE BASED NEURAL NETWORK

As we have seen, Lattice Based Neural Network (LBNN) are a very specific class of algorithms. They are characterized by the use of rectangles, the use of inhibitory and excitatory classification regions and the absence of local minima. After the introduction of , an increasing number of researchers have worked on both the algorithmic and theoretical analysis of these systems, creating in just on a few years what is effectively a new research direction in its own right, merging concepts mainly from lattice theory. In practice, LBNN theory can solve the problem of identifying to which category a new observation belongs. Means, LBNN theory can solve the classification problem. Where the term classifier refers, mainly, to the mathematical function mapping from the input data to the categories. Known LBNN works with real-value and discrete input terms. Those LBNN models may be less effective when it comes to classification of binary inputs.

There exists many other algorithms to solve classification problem. This work is exclusively focused on classification for binary contexts using LBNN theory. The advantage of working in a binary context classification binary, is the ease of use of the binary input method classification in the area of information fusion[refs]. In appendix section A.2, you will find a brief and a simple description of how our classifier can work on information fusion terms.

As we defined previously, a binary context can be represented in matrix terms. Moreover, each object or set of objects has its derived attributes. Similarly, a set of attributes has its derived objects. With these notions previously defined, the following section presents a classification rule in a context as a training space. This competition aims to define borders, using the maximum and the minimum coverage, to allocate a new element to a class. Using the definition 1, in which is specified a mapping $f : X \to Y$, we restrict $X \in \mathbb{B}^n$, with these ideas in mind, the following section defines a classification rule for our work.

4.2 A FORMAL CLASSIFICATION RULE.

In this section, we will define a simple classification rule that can be used in our supervised classifier model. The proposed classification rule uses notions of upward and downward closed sets, which is represented in the next statement. Since we are not interested in intersections, we will work over formal contexts. For practical purposes, we initially define the two class case. Later, we will define the multiple partition case.

Definition 14. This **formal classification rule** is a mapping from $x \subseteq M$ to our formal context class ϕ, and it is defined as follows :

Given $x \subseteq M$, x will be associated to ϕ if and only if

- $x \in \uparrow D^-$ and $x \in \downarrow D^+$ where $(\uparrow, \downarrow)$ are defined in the M power set domain.

This rule is the main driver of our learning method, it is also necessary to complete the goal set in this chapter.

In order to build our classifier, using LBNN structures, we will define a computation way for the minimum and maximum coverage's sets, associated with a binary context..

We will define algorithms to find the minimum and maximum coverage set in G. Even more, we will define a Lattice Based Neural Network structure that uses D_{ϕ}^{-},D_{ϕ}^{+} to training and extrapolation phases.

4.2.1 *Computing Negative Dendrites Set.*

In this particular case, we need the negative dendrite set that is present in the formal context class description. The main idea over this structure will be to discover all the elements in D^{-} which have the minimum conditions to associate a set of attributes to the class ϕ. The following algorithm, Algorithm 4.1, must not be confused with the one presented in Chapter 3. First, This algorithm classifies positive when the evaluated member is in the upward, computing from the minimum to maximum cardinality.

Thus, we will process each element $g \in G$ from lower to the higher derivative cardinality. The final output will be the set D^{-} We also will construct a LBNN P which will be used for the extrapolation phase.

In this LBNN based model, a set of binary patterns are represented by X, and it is used as a training set. Thus, the binary representation of a formal context, is itself a set of patterns, in which the derivative of each object is an observation $x \in G$ We can define $x' = (x_1, ..., x_n) \in \mathbf{B}^n$ as a binary vector. A set of n input neurons $N_1, ..., N_n$ is directly associated with the x input. In order to compute a output neuron O_j value, we compute each dendrite individually. Formally, in this case, k_{th} dendrite computation is given by:

$$\tau_k^j(x)^{-} = \bigwedge_{i \in I(k)} (x_i) \tag{4}$$

All the values $\tau_k^j(x)^{-}$ are passed to the neuron cell body. The computation value of the output neuron M_j, for negative dendrites is given by the Equation 5. And it computes all the negative dendrites values.

$$\tau^j(x)^{-} = \bigvee \tau_k^j(x)^{-} \tag{5}$$

Then, first step is to sort G' by cardinality from lower to higher cardinality. Second step will be add all the elements with lower cardinality and without repetitions as dendrites of our LBNN. Algorithm 4.1 is in charge to add individually minimum elements to the LBNN structure. Basically, Algorithm 4.1 receives an $x \subseteq M$, and grows a new dendrite. Therefore, for each attribute $m_i \in M$, if $m_i \in x$, it grows an axonal branch from the input neuron N_i to the new dendrite.

The third step will be process, the remaining sorted elements. Those elements must be evaluated in the LBNN P. Equation 5 represent the

Algorithmus 4.1 Add Negative Dendrite

```
INPUT: NeuralNetwork P,Pattern x
OUTPUT: Updated P
    // add a new dendrite
    Dendrite k   addNegativeDendrite(P)
    FOR EACH element in M
        IF getValue(element) = 1
            i = getPosition(element)
            addAxonalBrach(k,i)
        END
    END
END
```

This Algorithm shows how a new Binary Pattern x is added to a Neural Network P. This algorithm is used for the computation of Minimum Coverage Set or Negative Dendrite Set.

Algorithmus 4.2 Computing Minimum Coverage Set.

```
INPUT A Binary Context (G,M,I)
OUTPUT an LBNN Structure
STEP 1:
Sort G' by lower to higher Cardinality
STEP 2:
Add to LBNN, as negative dendrite, all elements with lower
    cardinality and no repetition in G'
STEP 3:
Each remaining element is evaluated for D- set.
        If evaluation is 0 then add, as negative dendrite,
            element to LBNN.
```

evaluation function. This evaluation together with the cardinality notion are based on the claim 2. Then, in this training phase, $\tau^j(x)^- = 0$ means that $x \in D^-$ And finally, Algorithm 4.1 adds processed elements with $\tau^j(x)^- = 0$ to the LBNN structure. Algorithm 4.2 shows this process.

So far we have obtained the minimum dendrites, to declare that an object is associated with a class. We can see from the output given by the negative output for neuron j, that it is needed that x be a superset of only one element in D^- to say that the attribute set x satisfies the minimum coverage of the context $\mathbb{K}$.

Remark 1. The evaluation function previously defined, together with the set D^- , gives us an upward closed set membership function on the power set lattice of M.

4.2.2 *Computing Positive Dendrites Set.*

Using the same notions of the last subsection, now we will define a LBNN structure in order to find all the Maximum Coverage Set, D^+ Algorithm 4.3 adds a new positive dendrite to our LBNN. In order to complete the algorithm, we must find all the maximal elements and

Algorithmus 4.3 Add Positive Dendrite

```
INPUT: NeuralNetwork P,Pattern x
OUTPUT: Updated P
        // add a new dendrite
        Dendrite k = addPositiveDendrite(P)
        FOR EACH element in x
                IF getValue(element) = 0
                        i = getPosition(element)
                        addAxonalBrach(k,i)
                END
        END
END
```

Axonal branches grows from positions with o value.

add them to the LBNN model. First step, showed in Algorithm 4.2, is to define a function in which those elements are computed. This time, we will process elements from higher to lower cardinality. This computation is based in the Claim 3.

In this case, the k_{th} **positive dendrite** state for the j output neuron computation is given by:

$$\tau_k^j(x)^+ = \bigwedge_{i \in I(k)} (0^{x_i}) \tag{6}$$

The difference with the use of the D^+ set, given by the Dendritical Algorithm in 3, is: Meanwhile Dendritical Algorithm classifies positive for elements in which $x \supset d$ for all $d \in D^+$ This one, classifies positive for elements in which $x \subseteq d$ for some $d \in D^+$.

As you can see, one of the ways to maintain the consistency of the output, is turning the x_i values. The reason for this is to assess the closed downward set membership having a positive output if the membership function is satisfied.

Again, all the values $\tau_k^j(x)^+$ are passed to the neuron cell body. The computation value of the output neuron M_j is given by;

$$\tau^j(x)^+ = \bigvee \tau_k^j(x)^+ \tag{7}$$

The trick is the same presented in the negative dendrites. Vector x is a subset of a member of a k_{th} member of D^+ if and only if each $x_i = 0$ for all $i \in I(k)$, with $I(k)$ growing from o's. It means, x does not has attributes that the k_{th} member of D^+does not has. So, it is only necessary satisfies one k_{th} element to make positive the Equation 7. It is enough to be considered as an element that does not exceed the maximum coverage set. It is very important to see that the output of the classifier or binary function, generated in the training stage described above, is equivalent to these statements. The output is $\tau^j(x)^+ = 1$, when the element x is member of the class and zero otherwise. Algorithm 4.4 shows how to compute the maximum coverage set for this case.

Algorithm 4.4 Computing Maximum Coverage Set

```
INPUT A Binary Context (G,M,I)
OUTPUT an LBNN Structure
STEP 1:
Sort G' by higher to lower Cardinality
STEP 2:
Add, as positive dendrites, all elements with maximum cardinality
     in G' to LBNN.
STEP 3:
Each remaining element is evaluated.
        If evaluation is 0 then add element to LBNN.
```

Here, our result is an LBNN Structure with a downward membership function and D^+ coverage.

Remark 2. This evaluation function, gives us an downward closed set membership function, defined on the power set lattice of M, for our formal context.

4.3 EXTRAPOLATION PHASE AND MULTI-CLASS.

The final computation at extrapolation phase for the j_{st} neuron is the minimum $\bigwedge$ of the maximum and minimum condition set evaluation.

$$\tau^j(x) = \tau^j(x)^- \bigwedge \tau^j(x)^+ \tag{8}$$

This function represents an intersection on those membership functions, and it matches with the formal classification rule previously defined. Next, we will see, a simple way to get a multiclass classifier.

In order to simplify the multiclass version of this algorithm. We will define Context Partitions. This definition is used to multi-class classification over the same context.

Definition 15. Let $\mathbb{K} = (G, M, I)$ be a formal context. An element $\phi \in \Psi$ is called a class and Ψ is the set of classes. Let each $g \in G$ be a member of one class ϕ . G_ϕ is the set over G_ϕ derivatives, of elements associated to the class ϕ It is represented by the context partition $\mathbb{K}_\phi$.

Table 6 shows a formal context particioned in classes.

Table 6: Formal context with $|\Psi|$ classes.

Class	Sample	f_1	...	f_{n-1}	f_n
ϕ_1	$O_{1.1}$	1	...	0	1
	$O_{1.2}$	0	...	1	1
	...	...	...	...	...
	$O_{1.\|G_{\phi_1}\|}$	0	...	1	0
...	...	...	...	...	...
$\phi_{\|\Psi\|}$	$O_{2.1}$	0	...	1	0
	$O_{2.2}$	1	...	0	0
	...	...	...	...	...
	$O_{2.\|G_{\phi_{\|\Psi\|}}\|}$	1	...	0	1

This table shows, how a set of classes $|\Psi|$ is separating a binary context. Each partition has $|G_{\phi_j}|$objects. Each Object $O_{j.k}$ belongs to the partition j and is the element k of that partition.

Thus, is easy to see that the context can be processed by the proposed partition. Achieving a separation of $|\Psi|$ classes.

Next section deals with an attribute reduction and a levelwise approximation on the reduction of rules.

4.4 HANDLING ATTRIBUTES.

The smaller extent of all element of $\mathfrak{B}(\mathbb{K})$ is the set of objects that share all features of M. So it is not generally the empty set $\emptyset$. Actually, the formal definition of a context that is given above is too general and may include unnecessary redundancies. It is possible to cancel non discriminant attributes or process them early, which are useless for class separation. The following definition, **proper context**, is the main definition of this possible reduction of rules.

Definition 16. Let $\mathbb{K}$=(G,M,I) be a context. If $(\emptyset, M)$ and $(G, \emptyset)$ are concepts, then the context $\mathbb{K}$ is said **proper**.

If the context is not proper, at least one attribute is present in all objects, or one attribute is not related with any object or, maybe simultaneously, at least one objects has all or none attributes. Thus, the set of attributes or objects must be studied.

If an attribute m is present in all object on the learning set, it is not a discriminative attribute. This does not mean that m is not useful, cause maybe it could be used to discriminate from external contexts. But if it is not the case, either attribute m must be removed. At the other side, it could happen that instead $(\emptyset, m)$ the context contains another $(\emptyset, m)$ with $m \subset M$, in this case it means that for some values, in this case $M \backslash m$, are not related to any object.

It is important to know that if one object of any class has all attributes then classification will not work properly, and the set of attributes has to be enlarged.

If the context is proper, $(\emptyset, M)$ is the smaller concept, and $(G, \emptyset)$ is the greater concept, then the context has not an unnecesary attributes for classification. At this step, it is important to considerate this annotation.

To understand how to use these statements and annotations to try to reduce the numbers of dendrites, we will define an abstract example.

4.4.1 *Level wise attribute handling.*

At this point, assume that both D^+ and D^-, have not been computed for each class. Our purpose will be through an abstract example, using the previously defined statement, proper context, reduce the attributes for computation of those sets. Our goal is to handle the levels of information and try to reduce the number of dendrites in our classification. As we are working on the intentional part of the formal concept, we will work only with the attributes set M.

Proposition 1. *Let the greater concept of an a context* $\mathbb{K}be$*:* (G, m)*, where* $m \subset M$*. As explained above, this means that the set of attributes* m *is related to all objects that exist in our formal context.*

We can define a Formal Context Level wise Negative Dendrite, in this case m is the training pattern. After that, we can ignore all the attributes that belong to m when we compute the dendrites for the classes.

In a dual way we can reduce the set of positive dendrites. So, we can work with the formal context infimum, as follows.

Proposition 2. *Let our smaller concept of an a context* $\mathbb{K}be$*:* $(\emptyset, m)$*, where* $m \subset M$*. In this case, the difference* $c = M \setminus m$*, tells us that the set of attributes* $c \in M$*, is not related to any object in the formal context.*

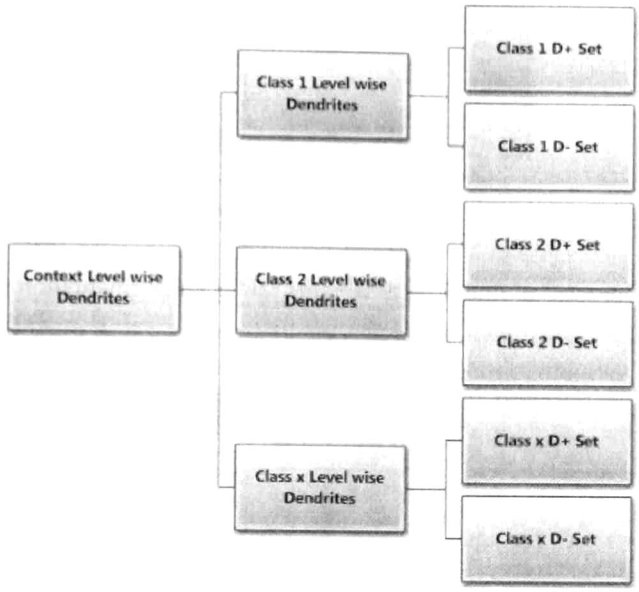
Here, we can define a Formal Context Level wise Positive Dendrite, with the attributes set c as a training pattern. After that we can ignore all the attributes that belong to c when we compute the dendrites for the classes.

Once we have calculated the pair of dendrites for the formal context level wise, we can also define dendrites for Classes Partitions Level wise. In the following proposition, which works with formal context class partitions, we define also a infimum and a supremum elements. We will work on those elements to generate a reduction in the number of attributes for both positive and negative dendrites.

Proposition 3. *Let* $\phi \in \Psi$ *be a class. Let* (G_ϕ, m)*, with* $m \subset M$*be the greater formal concept of the formal class context for* ϕ*. Here,* m *is the set of common attributes for all the objects that belongs to the class* ϕ*.*

For this case, we define a Class Level wise Negative Dendrite, with m as a training pattern. Again, attributes in m can be ignored when we compute D^+ and D^- for the class ϕ. Next, we defina a Class Level wise Positive dendrite.

Figure 13: Level wise attributes computation.



Processing from the rigth to the left. First compute the Context Level wise dendrite, if that test is passed, compute the class level wise dendrites. Last, if tge Levelwise condition are satisfied, then the set D^+ and D^- computation will be done.

Proposition 4. *Let $\phi \in \Psi$ be a class. Let $(\emptyset, m)$ be the smaller concept of the formal class context. In this case, the differencec $= M \setminus m$, tells us that the set of attributes $c \in M$, is not related to any object in the class ϕ.*

Finally, we add a Class Level wise Positive Dendrite with c as a training pattern. We will, also, ignore those c attributes at $D+$, $D-$ computation. Figure X shows how to process dendrites to get the final classifier computation.

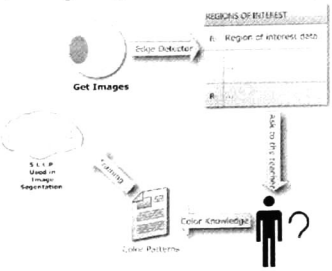
Reduce the number of attributes or grouping them by level gives us a better and a cleaner set of dendrites to work with. The main idea of this, is get a reduced computation of the class membership functions. This was achieved with the notion of supreme and infimum, both context and class level. This was achieved with the notion of supreme and infimum, both context and class level. Next, we will define a dataset to work with the experimental analysis.

4.5 COMPUTER VISION DATASET: REGIONS OF INTEREST.

. In this dataset objects were selected with different shapes and colors. The number of classes that depend on the shape are three (Cone, Sphere, Square). While the color-dependent classes are five (Green, Purple, Orange, Gray, Black). Figure 15 shows objects used for this example.

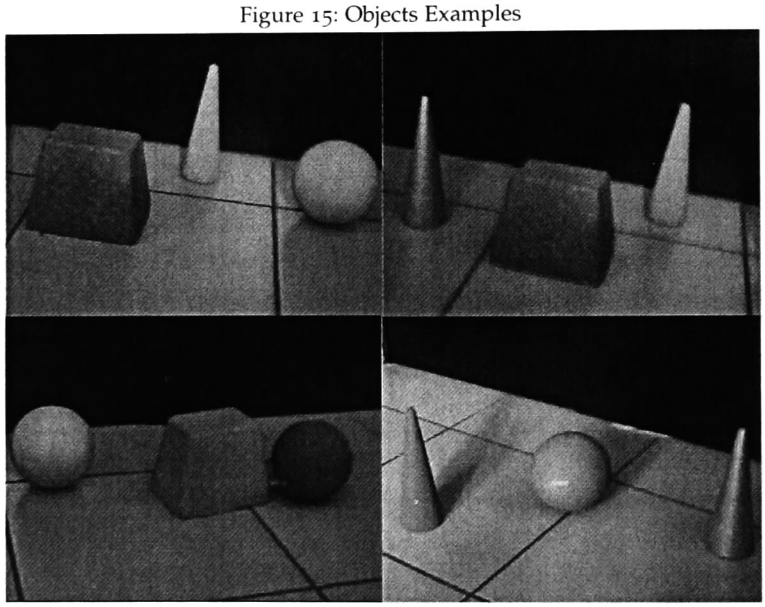
We used a SLLP for color recognition. The image segmentation was carried out with the purpose of simplifying the dataset creation

Figure 14: Image Segmentation Learning Process.



The process of find S.L.L.P weights for image segmentation.

Figure 15: Objects Examples



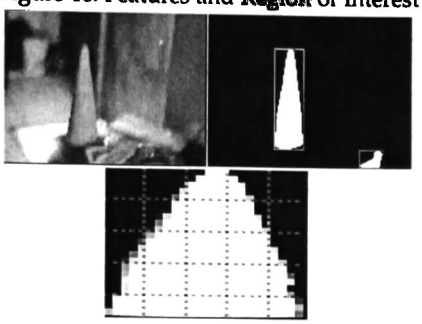
Cone, Box and Sphere are shapes. Gray, Purple, Green and Orange are Colours.

task. Two tasks were considered. Color segmentation and shape detection. The first task was carried out using a set of images for the training step. In each image, using the algorithm of Canny; we detected regions of interest. For each region of interest, manually, we classified the color. Finally, a file containing the characteristics of the region of interest was generated. Figure 14 is a diagram representing this process.

The use of a detector of edges and regions of interest obtained is a first step for image segmentation. We manually classify the colors in order to reduce the cost of obtaining the training samples. Once the samples were collected and trained, we were focused on selecting the characteristics of features used.

In the next chapter we will se the proposed features for the shape classification task. At this point, our work is focused on generating, based on this segmentation algorithm, a set of data, represented by

Figure 16: Features and Region of Interest



Sub-Figure A: is an arbitrary image from the webcam taken in the enviroment. Sub-Figure B represents the image segmentation using an SLLP., in which, red bounded region of interest is classified as Cone by the supervisor. Sub-Figure C represents that region scaled and a croped for feature extraction. In this case we have 25 features it means, each section in the cropped scaled sample is qualified using a threadhold.

the regions of interest for shapes. The next part focused on getting the attributes, threadhold takes several crops of that scaled region of interest. Thus, this dataset is justified as the elements generated by the segmentation algorithm presented. Which is noisy and looks classify shapes and contours.

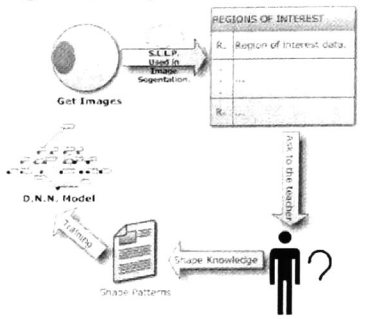
4.6 EXPERIMENTAL ANALYSIS.

Once the color system were specified, the next stage is the shape detection. Here, we propose a set of features and a test bed for the experimental analysis. This stage was performed using an LBBN, specifically the SLLP-Merge algorithm. As in image segmentation training, we use an contour detector, on the segmented image. Each region of interest was also classified manually. Figure 16 shows the cuttings of an area of interest and illustrates the ability to get features using individual cropped regions.

With the purpose of challenging the ability of our classifier, we use very simple features and a noisy camera. As we have seen before, these features are mainly a dynamic threshold over croped regions. With dynamic we mean a threadhold based on the image density, given preference to find discriminative values. Nothing stop the reader to use other kinds of features. In Appendix A.3, we present some reviewed features.

Once, all the feature extraction process has finished, we have a file containing rows of features associated to classes. This file is used to train our classifier. Figure 17 shows the learning process of shapes using the SLLP for image segmentation. In this case, the final result is a shape classifier, using the model proposed in the previous section. Even though, when this process some noise features are produced.

Figure 17: Shapes Learning Process



Process of construction for shapes classifier.

Table 7: Number of Dendrites

CUTS	$\lvert D^+_{Box}\rvert$	$\lvert D^-_{Box}\rvert$	$\lvert D^+_{Cone}\rvert$	$\lvert D^-_{Cone}\rvert$	$\lvert D^+_{Sphere}\rvert$	$\lvert D^-_{Sphere}\rvert$	Total
5x5	4	17	54	31	12	43	195
6x6	9	32	95	50	30	35	215
7x7	15	50	129	85	31	49	248
8x8	19	84	176	108	39	52	296

Number of Positive and Negative dendrite per class.

One concern is the system ability to add new color classes or even, new shapes. It is possible add new colors and extrapolate known shapes, using those new colors and viceversa. You can also add new shapes and extrapolate them with the known colors. Thus, you can individually modify our S.L.L.P., adding or removing output neurons, or modify, even, the shape classifier.

For the training set was used: 500 samples of cones, 500 samples of sphere, 500 samples of boxes and 500 unknown elements. For each test set 700 samples were taken. Those samples represent objects in various different locations. The next step was reduce the number of attributes and separate all dendrites in the levelwise system presented below. The discriminative attributes for the class for a training on 5x5 cuts is bounded by 25. All this information is presented in table 7. This table shows the ability for good extrapolation of the proposed algorithm.

Next step was to find the percentage of the classifier performance. Experimental Analysis over this classifier were simple and classical. We calculated misclassified, false positives and correct classifications. This on a 10-fold cross validation test on binary classes[refs...]. In this k-fold cross-validation, our binary context is randomly partitioned into k equal size subsamples. One partition is used as testing data, meanwhile the other k-1 partitions are used as a training set. This process is repeated k times, in our case, it is ten partitions and ten times. The main advantage of k-fold cross validation is that it works over repeated random sub-sampling and it use all observations for training exactly k-1 times, and each observation is used for validation

Table 8: Classification Rate.

Class	Missclassified%	False Positive%	Accuracy%
Cone	0.0421	0.0900	0.9442
Sphere	0.0280	0.0750	0.9585
Box	0.0420	0.1200	0.8785

64 features over scaled regions of interest with size 40x40

Table 9: Comparing classifiers.

Class	Missclassified	False Positive%	Accuracy%
Dendritical	0.0240	0.0850	0.9585
SVM	0.1600	0.0280	0.9057
Linear Classifier	0.4250	0.448	0.5585

100 features for cone class scaled regions of interest with size 40x40.

exactly once. Table 8 shows the classification performance results in our experimental analysis framework.

As we said those results consider several conditions and even incomplete patterns, we discovered at this point a good performance over our classifier. Even more, it shows that is a good method for shape recognition.

This shows the system's response to an instantiation, in which figures were sought in the environment. The result of the classification is showed from 25 to 64 features, given the simplicity of the problem. The study of features for classification are many. We proposed a simple dataset with equally simple features. There exists a lot of studies of image features for classification. We proposed a simple dataset with equally simple features. Most of the time, features modeled in a computer vision system depend on the specific addressed problems. In this case, we only took the classification of shapes to generate a dataset. However, in Appendix A.3, you will find a brief on features images.

Our final experiment is to compare our classifier performance with other algorithms for classification. At this point, we chose two key classifiers to make the comparison. The former, a discriminator, to show the non-linear separability nature of our dataset. The second, a Support Vector Machine using a radial basis function for classification. These two classifiers are considered sufficient to demonstrate the quality of our classifier in these terms. Table 9 shows the obtained experimental results.

One significant difference is that for this test we used 100 crops, in images of 30x30. Furthermore, single cone dataset was used to execute the operation, using the same system of k-fold crossover, with $k = 10$.

In the next and final section of this chapter, you will find the conclusions.

4.7 CONCLUSION

Machine learning is a growing area. We demonstrate in this chapter, the possibility of using the lattice theory in this context. The ability on algorithm to learn, to discriminate objects in specific, helps us to understand the environment. Similarly, based on the same precepts of the theory of dendritical neural networks. The experimental results were collected and explained in the previous section. They showed an excellent performance on a known problem.

Recognition of multiple classes in computer vision. It is not a trivial problem and there are many methods to be compared with ours. However, the scope of our work, achieved through an example demonstrates the feasibility and capability of our classification rule. This first attempt, from our point of view successful, links the machine learning theory with FCA. Addition of more and better features can be tested in the model. The purpose of this chapter, which was to show the reader through an example, the method we developed for classification. Understanding topological operations in lattices a good start to continue the theme and further develop more comprehensive improvements and ideas.

We have defined the basic theoretical framework for this dissertation. We showed several ways on how theory of lattices can be used as a system for design models in pattern recognition and machine learning. In section [sec:Experimental-Analysis.], which serves as a test bed, we shown how to design a classifier for shape recognition with a simple cam, combining several binary functions over the image. Furthermore, this design allows structural organization: the detection of features needed for an output that specializes the description of the object in each layer.

Part III

APPENDIX

APPENDIX A

A.1 FCA ALTERNATIVE NOTATION.

This Appendix Section introduces basic notions of formal concept analysis, among which are the fundamental notions of a formal context, formal concept, and concept lattice. The chapter uses an alternative notation which represents the same definitions given at the **Chapter 1**.

Definition 17. (Formal Context). A formal context is a triplet (X, Y, I) where X and Y are non-empty sets and I is a binary relation between X and Y , i.e., $I \subseteq X \times Y$.

Like its definition used in this thesis, this, first, alternative definition of the definition 2, reflects the concept of a formal context.

Definition 18. (Concept-forming Operators). For a formal context (X, Y, I), operators $\uparrow: 2^X \to 2^Y$ and $\downarrow: 2^Y \to 2^X$ are defined for every $A \subseteq X$ and $B \subseteq Y$ by:
$A \uparrow = y \in Y | for\, each\, x \in A : (x, y) \in I$,
$B \downarrow = x \in X | for\, each\, y \in B : (x, y) \in I$

Every formal context induces a pair of operators, so-called concept-forming operators. Those concept-forming operators are equivalent to galois connector from the Galois Connectors definition 3.

Definition 19. (Formal Concept). A formal concept in (X, Y, I) is a pair (A, B) of $A \subseteq X$ and $B \subseteq Y$ such that $A \uparrow = B$ and $B \downarrow = A$.

This definition, equivalent to definition 4 says mathematically, that an (A, B) is a formal concept if and only if $(A, B)$is a fixpoint of the pair $(\uparrow, \downarrow)$ of concept-forming operators.

Definition 20. (Subconcept-superconcept Ordering). For formal concepts (A_1, B_1) and (A_2, B_2) of (X, Y, I), put $(A_1, B_1) \leq (A_2, B_2)$ iff $A_1 \subseteq A_2$(iff $B_2 \subseteq B_1$).

the above definition together with the following one is an equivalent to the definition 6 of chapter 1.

Definition 21. (Concept Lattice). Denoted by $\mathfrak{B}(X, Y, I)$ the collection of all formal concepts of (X, Y, I), ordered by definiton is named the Concept Lattice.20.

As the reader will see, the differences between these two notations are minimal, and those differences relapse basically in the symbols and the style used to define the same set of operations, definitions and properties. Our purpose was to simplify and make pleasant the reading of this thesis. In that sense any notation is fine to reach that goal. However, the combined notations, or not maintain a prewritten

order usually causes complications for the reader. The latter is the main reason for decide to skip this notation and use only the one presented in the first chapter.

A.2 INFORMATION FUSION.

Although the idea of fusion of information is very intuitive. There are several definitions of this notion. Many definitions have been proposed [refs], with some more general or more abstract and inclusive than others have been defined in some specific fields of study. A more general and detailed work on information fusion could be found at Andres Mendez[refs]. One of the more general and accepted definition was given by Walden[refs] is:

Definition 22. "**Information fusion** is a formal framework in which are expressed means and tools for the alliance of data originating from different sources. It aims at obtaining information of greater quality ; the exact definition of "greater quality" will depend upon the application."

The study of information fusion is mainly divided into three categories [refs]:

1. Data level fusion, which fuses data acquired from multiple sources, such as images acquired in different spectral bands, directly.
2. Feature level fusion, which combines features calculated on data acquired from multiple sources.
3. Decision level fusion, which combines decision statistics and confidence base information derived from algorithms applied to multiple sources.

In this appendix section we works with the second and the third case. Next, we will define useful notions to plot the outputs of several classifiers. Even when we are talking about classifiers. It is remarkable, that this idea could be extended to fuzzy and binary threadholds. Last one, presented at chapter 4.

Definition 23. Concept Lattice in Machine Learning: Let Y be a set of categorical output spaces such that $Y := Y_1, ..., Y_n$, in which, each $y_j \in Y_i$ is a categorical label. By the same way, Let Z be a set of categorical classificators such that $Z := Z_1, ..., Z_n$ each one handling a map function from $Z_i \to Y_i$, and $X := \{X_1^i, ..., X_{m^i}^i\}$ is a binary vector output for the classifier Z_i, over our set of objects to conceptulize, those vectors will serve as our formal context $\mathbb{K}$. We can note that m^i is the cardinality of Y_i.

Last definition allows us to formally identify elements that form the basis of our framework intended to be applied to conceptualize or fuse the output of a set of classifiers. In the table 10, is illustrated in the abstract way, how different classifiers, for different or even the same features, can generate the set of attributes M, feeding binary contexts.

	$Classifier_1$			$Classifier_2$			...	$Classifier_n$		
G/M	$Y_{1,1}$	...	Y_{1,m^1}	$Y_{2,1}$	...	Y_{2,m^2}	...	$Y_{n,1}$	...	Y_{n,m^n}
O_1	...			...			...	...		
...	...			...			...	...		
$O_{\|G\|}$	...			...			...	...		

Table 10: A Set of n classifiers feeding M set of $\mathbb{K}$

This formal context, as reviewed in chapter 3, generates a Concept Lattice representing all clusters present in the context. Section 4.2 defines a formal classification rule for this kind of binary formal contexts.

A.3 IMAGE FEATURES .

One of the first publications on image features appeared after the observation on the importance of corners and junctions in visual recognition[refs]. Since then, a large number of algorithms and techniques have been proposed to extract and detect points or regions of interest in an image. Interest points are the preferred strategy for solving a wide variety of problems, from wide baseline matching and the recognition of specific objects to the recognition of object classes. Additionally, similar ideas have been applied to robot navigation, texture recognition, scene classification, visual data mining, and symmetry detection, image segmentation to name just a few application domains. In this appendix section we present some known characteristics on those features.

A characteristic feature of the type are given by the detector response obtained. The main branches, with respect to the output type are:

Edges - Edges based characteristics works with boundaries. This kind of response is frequently described as a point in the image which have a strong gradient magnitude.

Interest point - Mainly, those interest point are defined on measurements of one point with their neighboorhood. They are a two dimension features, and the features detector of this kind, are expected to get descriptors of the image.

Blobs - An blob output, is a complementary image descriptor. In this kind of features, several regions of the images are grouped to get those areas as a feature.

Other measures to the features are given by the quality response of a kind of feature to lighting, scale, rotation variation. The following lines describes those kind of variations and how they are defined by an specific goal.

Quality - the quality of detected features is a measurement on the stability of the detected points or region of interest, given similar images. So, those features must to have a good characteristic with the

good quality with a good level of consistency. They are able to obtain information that will separate the classes that are being sought.

Lighting invariant Is an special characteristic on some features. Features detector are expected to detect the same features even when the light is variant. Sight bright and contrast fluctuations will not affect significantly on the feature detection.

Scale invariant This is, in this case, features detection are expected to detect the same features on a class even when the scale is large or tiny. A robust detection is critical for processing classes recognition in computer vision.

Rotation invariant - The position or rotation of one object is considered in this kind of features. The objective is detect the common features when the image is rotated.

Feature confidence - The confidence of one feature is an information about the certainty about the value of some kind of feature.

An important problem in pattern analysis is the automatic rec- ognition of an object in a scene regardless of its position, size, and orientation. They arise in a variety of situations such as inspection and packaging of manufactured parts. Features on image literature are rich and it is growing fast. In table 11 we present some known features.

Table 11: Features

Feature	Output	Invariant	Observation
FAST	Corners	None	First feature based on Accelerated Segment Test.
Harris operator	Corners	Gray	Works over local neighborhod using image derivatives.
Canny-edges	Interest point	Gray	Gives a good edge descriptor for several objects on the image.
Sobel	Interest point	Gray	An discrete differentiation operator to detect edges of an image.
LoG	Blob	Scale	The image is is convolved by a Gaussian kernel.
DoG	Blob	Gray Scale	Involves the subtraction of one transformed version of an original grayscale with other image less transformed.
Zernike Moments	Interest point	Rotation	Use the notion of regular orthogonal moments to handle rotation variation.

Some features studies example, each one with particular properties and an own mathematical framework.

BIBLIOGRAPHY

[1] Rakesh Agrawal. Fast algorithms for mining association rules in large databases. *Proceedings of the 20th International Conference on Very Large Data Bases,*, 1:487–499, 1994. (Cited on page 19.)

[2] T.; Swami A. Agrawal, R.; Imielinski. Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 1:207, (1993). (Cited on page 19.)

[3] Mario Aldape-Perez, Cornelio Yanez-Marquez, Oscar Camacho-Nieto, and Amadeo J.Arguelles-Cruz. An associative memory approach to medical decision support systems. *Comput. Methods Prog. Biomed.*, 106(3): 287–307, 2012. doi: 10.1016/j.cmpb.2011.05.002. URL `http://dx.doi.org/10.1016/j.cmpb.2011.05.002`. (Cited on pages 18 and 21.)

[4] Simon Andrews. In-Close, a Fast Algorithm for Computing Formal Concepts. In *the Seventeenth International Conference on Conceptual Structures*, 2009. (Cited on page 19.)

[5] M.A. Arbib. *The Handbook of Brain Theory and Neural Networks: Second Edition*. Bradford Books. MIT Press, 2002. (Cited on pages 9 and 21.)

[6] H. A. Priestley B. A. Davey. *Introduction to lattices and orders*. Press Sindicate H. Cambridge University, 2nd edition, 2002. (Cited on pages 3 and 27.)

[7] A. Barmpoutis and G. X. Ritter. Orthonormal Basis Latice Neural Networks. *In Computational Intelligence Based on LatticeTheory, V. Kaburlasos and G. X. Ritter*, 1:43–56, Springer-Verlag, Heidelberg, Germany, 2007. (Cited on pages 21, 22, and 27.)

[8] R. Belohlavek and Ghassan Beydoun. Formal Concept Analysis With Background Knowledge: Attribute Priorities. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 5465:109–117, 2009. (Cited on page 18.)

[9] Radim Belohlavek and Vilem Vychodil. Formal concept analysis with background knowledge: attribute priorities. *Trans. Sys. Man Cyber Part C*, 39(4):399–409, 2009. (Cited on page 18.)

[10] Radim Belohlavek, Bernard De Baets, Jan Outrata, and Vilem Vychodil. Inducing decision trees via concept lattices. (Cited on page 20.)

[11] James O. Berger. Bayesian Analysis: A Look at Today and Thoughts of Tomorrow. *Journal of the American Statistical Association*, 95(452):1269 – 1276, 2000. (Cited on pages 9 and 20.)

[12] Ghassan Beydoun. Formal concept analysis for an e-learning semantic web. *Expert Syst. Appl.*, 36(8):10952–10961, 2009. (Cited on page 19.)

[13] Ghassan Beydoun. Using Formal Concept Analysis towards Cooperative E-Learning. In Debbie Richards and Byeong-Ho Kang, editors, *Knowledge Acquisition: Approaches, Algorithms and Applications*, volume 5465 of *Lecture Notes in Computer Science*, pages 109–117. Springer Berlin Heidelberg, 2009. (Cited on page 19.)

[14] Garrett. Birkhoff. *Lattice Theory, Volumen 25*, volume 25 (3rd ed). American Mathematical Society Colloquium Publications., 1995. (Cited on pages 3, 4, and 27.)

[15] J. Blanck. Domain representations of topological spaces. *Theoretical Computer Science.*, 247:229???255, 2000. (Cited on page 3.)

[16] J. P. Bordat. Calcul pratique du treillis de Galois d une correspondance. *Informatiques et Sciences Humaines*, 96:31–47, 1986. (Cited on pages 3 and 16.)

[17] Martin D. Buhmann and M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, New York, NY, USA, 2003. ISBN 0521633389. (Cited on page 9.)

[18] Douglas Burdick, Manuel Calimlim, Jason Flannick, Johannes Gehrke, and Tomi Yiu. MAFIA: A Maximal Frequent Itemset Algorithm. *IEEE Trans. Knowl. Data Eng.*, 17(11):1490–1504, 2005. (Cited on page 19.)

[19] C. Carpineto and G. Romano. An order-theoretic approach to conceptual clustering. *Proc. of the 10th Conf. on Mach. Learning.*, 10:33–40, 1993. (Cited on pages 15 and 17.)

[20] C. Carpineto and G. Romano. Information retrieval through hybrid navigation of lattice representations. *International Journal of Human-Computer Studies,*, 45:553–578., 1996. (Cited on page 17.)

[21] Michel Chein. Algorithme de recherche des sous-matrices premieres d une matrice. *Bull. Math. Soc. Sc. Math. de Roumanie*, 1 (13):21–25, 1969. (Cited on page 15.)

[22] St?phanie Chollet, Vincent Lestideau, Philippe Lalanda, Yoann Maurel, Pierre Colomb, and Olivier Raynaud. Building FCA-Based Decision Trees for the Selection of Heterogeneous Services. In Hans-Arno Jacobsen, Yang Wang, and Patrick Hung, editors, *IEEE SCC*, pages 616–623. IEEE, 2011. (Cited on page 20.)

[23] Richard Cole and Peter Becker. Navigation spaces for the conceptual analysis of software structure. In *Proceedings of the Third international conference on Formal Concept Analysis*,

ICFCA'05, pages 113–128, Berlin, Heidelberg, 2005. Springer-Verlag. doi: 10.1007/978-3-540-32262-7_8. URL http://dx.doi.org/10.1007/978-3-540-32262-7_8. (Cited on pages 18 and 20.)

[24] Tilley T. Ducrou J. R. Belohlavek V. Snasel Cole, R. Conceptual Exploration of Software Structure: A Collection of Examples. In *CLA*, pages 135–148, 2005. (Cited on pages 18 and 20.)

[25] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 1 edition, March 2000. (Cited on page 9.)

[26] Hal Daumé, III and Daniel Marcu. Domain adaptation for statistical classifiers. *J. Artif. Int. Res.*, 26(1):101–126, May 2006. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=1622559.1622562. (Cited on page 9.)

[27] Alexandre Delteil, Catherine Faron, and Rose Dieng. Building Concept Lattices by Learning Concepts from RDF Graphs Annotating Web Documents. In Uta Priss, Dan Corbett, and Galia Angelova, editors, *Conceptual Structures: Integration and Interfaces*, volume 2393 of *Lecture Notes in Computer Science*, pages 191–204. Springer Berlin Heidelberg, 2002. (Cited on page 17.)

[28] Guozhu Dong, Chunyu Jiang, Jian Pei, Jinyan Li, and Limsoon Wong. Mining succinct systems of minimal generators of formal concepts. In *Proceedings of the 10th international conference on Database Systems for Advanced Applications*, DASFAA'05, pages 175–187, Berlin, Heidelberg, 2005. Springer-Verlag. doi: 10.1007/11408079_17. URL http://dx.doi.org/10.1007/11408079_17. (Cited on page 18.)

[29] P. E.; Stork D. H. Duda, R. O.; Hart. *Pattern Classification*. Wiley Interscience, 2000. (Cited on page 8.)

[30] Koschke R. Simon D. Eisenbarth, T. Locating Features in Source Code. *IEEE Transactions on software engineering.*, Vol. 29.(3):210–224, March. 2003. (Cited on page 18.)

[31] Martin Farach-Colton and Yang Huang. A Linear Delay Algorithm for Building Concept Lattices. In Paolo Ferragina and GadM. Landau, editors, *Combinatorial Pattern Matching*, volume 5029 of *Lecture Notes in Computer Science*, pages 204–216. Springer Berlin Heidelberg, 2008. (Cited on page 17.)

[32] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936. (Cited on page 9.)

[33] Boris Galitsky, Boris Kovalerchuk, and Sergei O. Kuznetsov. Learning Common Outcomes of Communicative Actions Represented by Labeled Graphs. In *ICCS*, pages 387–400, 2007. (Cited on page 20.)

[34] A. Galy, R. Medina, L. Nourine, and Y. Renaud. Uncovering and Reducing Hidden Combinatorics in Guigues-Duquenne Bases. In *Formal Concept Analysis*, pages 235–248. Springer Berlin / Heidelberg, 2005. (Cited on page 19.)

[35] Bernhard Ganter. Two basic algorithms in concept analysis. In *Proceedings of the 8th international conference on Formal Concept Analysis*, ICFCA-10, pages 312–340, Berlin, Heidelberg, 2010. Springer-Verlag. (Cited on page 16.)

[36] Bernhard Ganter and SergeiO. Kuznetsov. Hypotheses and Version Spaces. In Bernhard Ganter, Aldo Moor, and Wilfried Lex, editors, *Conceptual Structures for Knowledge Creation and Communication*, volume 2746 of *Lecture Notes in Computer Science*, pages 83–95. Springer Berlin Heidelberg, 2003. doi: 10.1007/978-3-540-45091-7_6. URL `http://dx.doi.org/10.1007/978-3-540-45091-7_6`. (Cited on pages 18 and 20.)

[37] Bernhard Ganter and Rudolf Wille. Applied Lattice Theory: Formal Concept Analysis. In *In General Lattice Theory, G. Gratzer editor, Birkhauser*. Preprints, 1997. (Cited on pages 4 and 27.)

[38] Wille R. Ganter, B. *Formal Concept Analysis, Mathematical Foundation*. Springer-Berlin, 1999. (Cited on pages 3, 4, 13, and 27.)

[39] R. Godin, R. Missaoui, and H. Alaoui. Incremental concept formation algorithms based on Galois (Concept) lattices. *Computational Intelligence*, 11(2):246–267, 1995. URL `http://www.bibsonomy.org/bibtex/26ab0b458c708c8ab14597749573e92b6/stumme`. (Cited on pages 5 and 17.)

[40] Saunders E. Godin, R. and J. Gecsei. Lattice Model of Browsable Data Spaces. *Information Sciences*, 40:89–116., 1986. (Cited on page 5.)

[41] Leslie Ann Goldberg. *Efficient algorithms for listing combinatorial structures*. Cambridge University Press, New York, NY, USA, 1993. (Cited on page 15.)

[42] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. In *International Journal of Human-Computer Studies*, pages 907–928. Kluwer Academic Publishers, 1993. (Cited on pages 17 and 18.)

[43] A. Guenoche. Construction du treillis de Galois d une relation binaire. *Math. Inf. Sci. Hum.*, 109:41–53, 1990. (Cited on page 15.)

[44] Duquenne V. Guigues, J.L. Familles minimales d'implications informatives resultant d'un tableau de donnees binaires. *Math. Sci. Hum.*, 95(1):5–18, 1986. (Cited on page 19.)

[45] Stephanie Guillas, Karell Bertet, and Jean-Marc Ogier. A generic description of the concept lattices' classifier: application to symbol recognition. In *Proceedings of the 6th international*

conference on Graphics Recognition: ten Years Review and Future Perspectives, GREC'05, pages 47–60, Berlin, Heidelberg, 2006. Springer-Verlag. (Cited on page 20.)

[46] Caudell T Healy M. Ontologies and worlds in category theory: implications for neural systems. *Axiomathes*, 16:165–214, 2006. (Cited on page 21.)

[47] Hecht-Nielsen. Theory of the backpropagation neural network. In *International Joint Conference on Neural Networks*, pages 593–605 vol.1. IEEE, June 1989. (Cited on page 9.)

[48] Joachim Hereth, Gerd Stumme, Rudolf Wille, and Uta Wille. Conceptual knowledge discovery: a human-centered approach. *Journal of Applied Artificial Intelligence*, 17:281–302, 2003. (Cited on page 18.)

[49] Sertkaya B. R. Medina Hermann, M. and S. Obiedkov. On the Complexity of Computing Generators of Closed Sets. *ICFCA*, LNAI 4933:Springer, 2008. (Cited on page 18.)

[50] Jorge Eduardo Hurtado-Gómez and Diego Andrés Álvarez-Marín. An optimization method for learning statistical classifiers in structural reliability. *Probabilistic Engineering Mechanics*, 25(1):26–34, 2010. (Cited on page 9.)

[51] Vassilis Kaburlasos. Granular Enhancement of Fuzzy ART-SOM Neural Classifiers Based on Lattice Theory. In Vassilis Kaburlasos and Ritter Gerhard, editors, *Computational Intelligence Based on Lattice Theory*, volume 67 of *Studies in Computational Intelligence*, pages 3–23. Springer Berlin / Heidelberg, 2007. (Cited on pages 9 and 21.)

[52] Vassilis G. Kaburlasos and Gerhard X. Ritter. *Computational Intelligence Based on Lattice Theory*, volume 67 of *Studies in Computational Intelligence*. Springer, 2007. (Cited on page 22.)

[53] James M. Keller and Douglas J. Hunt. Incorporating Fuzzy Membership Functions into the Perceptron Algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 7(6):693–699, 1985. doi: 10.1109/TPAMI.1985.4767725. URL `http://dx.doi.org/10.1109/TPAMI.1985.4767725`. (Cited on page 21.)

[54] N. Kharchevnikova, V. Blinova, D. Dobrynin, N. Fedorova, M. Novich, and M. Vrachko. Data mining on carcinogenicity of chemical compounds by the JSM method. *Automatic Documentation and Mathematical Linguistics*, 43:330–335, 2009. URL `http://dx.doi.org/10.3103/S000510550906003X`. 10.3103/S000510550906003X. (Cited on page 20.)

[55] Sergei O. Kuznetsov and Sergei A. Obiedkov. Algorithms for the Construction of Concept Lattices and Their Diagram Graphs. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 289–300,

London, UK, UK, 2001. Springer-Verlag. (Cited on pages 5, 15, and 16.)

[56] Sergei O. Kuznetsov and Sergei A. Obiedkov. Comparing performance of algorithms for generating concept lattices. *Journal of Experimental and Theoretical Artificial Intelligence*, 14(2-3):189–216, 2002. URL http://www.bibsonomy.org/bibtex/2582910e7a14a80469b2cb328fa0d9884/kde. (Cited on pages 15, 16, and 34.)

[57] S.O. Kuznetsov. Learning of Simple Conceptual Graphs from Positive and Negative Examples. *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery.*, 1704:284–391, 1999. (Cited on page 20.)

[58] Stumme G. Lakhal, L. Efficient Mining of Association Rules Based on Formal Concept Analysis. *Ganter et al. Springer.*, LNAI 3626:180–195, 2005. (Cited on page 18.)

[59] Kai Li, Yajun Du, Dan Xiang, Honghua Chen, and Zhenwen Liao. A Method for Building Concept Lattice Based on Matrix Operation. In De-Shuang Huang, Laurent Heutte, and Marco Loog, editors, *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, volume 4682 of *Lecture Notes in Computer Science*, pages 350–359. Springer Berlin Heidelberg, 2007. (Cited on page 17.)

[60] LingLing Lv, Lei Zhang, PeiYan Jia, and FuNa Zhou. A Bottom-Up Incremental Algorithm of Building Concept Lattice. In Yanwen Wu, editor, *Software Engineering and Knowledge Engineering: Theory and Practice*, volume 115 of *Advances in Intelligent and Soft Computing*, pages 91–98. Springer Berlin Heidelberg, 2012. (Cited on page 17.)

[61] Schnabel M. Representing and processing medical knowledge using formal concept analysis. *Methods Inf Med.*, 41(2):160–167, 2002. (Cited on page 18.)

[62] Y. Malagrange. Recherche des sous-matrices premieres d une matrice a coefficients binaires. *Deuxieme congress de L AFCLTI*, 1:Unknow, 1961. (Cited on page 15.)

[63] Nida Meddouri and Mondher Maddouri. Boosting Formal Concepts to Discover Classification Rules. In *Proceedings of the 22nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems: Next-Generation Applied Intelligence*, IEA/AIE '09, pages 501–510, Berlin, Heidelberg, 2009. Springer-Verlag. doi: 10.1007/978-3-642-02568-6_51. URL http://dx.doi.org/10.1007/978-3-642-02568-6_51. (Cited on page 20.)

[64] Raoul Medina and Lhouari Nourine. A Unified Hierarchy for Functional Dependencies, Conditional Functional Depen-

dencies and Association Rules. In *ICFCA*, pages 98–113, 2009. (Cited on page 18.)

[65] Engelbert Mephu Nguifo, Norbert Tsopze, and Gilbert Tindo. M-CLANN: Multi-class Concept Lattice-Based Artificial Neural Network for Supervised Classification. In *Proceedings of the 18th international conference on Artificial Neural Networks, Part II*, ICANN '08, pages 812–821, Berlin, Heidelberg, 2008. Springer-Verlag. (Cited on page 20.)

[66] Dean Merwe, Sergei Obiedkov, and Derrick Kourie. AddIntent: A New Incremental Algorithm for Constructing Concept Lattices. In Peter Eklund, editor, *Concept Lattices*, volume 2961 of *Lecture Notes in Computer Science*, pages 372–385. Springer Berlin Heidelberg, 2004. (Cited on page 17.)

[67] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers. Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing IX, 1999. Proceedings of the 1999 IEEE Signal Processing Society Workshop*, 1:41–48, August 1999. (Cited on page 9.)

[68] A. Nica and R. Speicher. *Lectures on the Combinatorics of Free Probability*. Number v. 13 in London Mathematical Society Lecture Note Series. Cambridge University Press, 2006. (Cited on page 3.)

[69] Bakalis Nikolaos. *Handbook of Greek Philosophy: From Thales to the Stoics Analysis and Fragments*. Trafford Publishing, 2005. (Cited on page 17.)

[70] E. M. Norris. An Algorithm for Computing the Maximal Rectangles in a Binary Relation. *Revue Roumaine de Mathematiques Pures et Appliquees*, 23(2):243–250, 1978. URL `http://www.bibsonomy.org/bibtex/241ffd27a2c9c70b7e1f8fa85c40240b8/stumme`. (Cited on pages 16 and 17.)

[71] Lhouari Nourine and Olivier Raynaud. A Fast Algorithm for Building Lattices. *Inf. Process. Lett.*, 71(5-6):199–204, 1999. (Cited on page 17.)

[72] Vera V. Pankratieva and Sergei O. Kuznetsov. Relations between Proto-fuzzy Concepts, Crisply Generated Fuzzy Concepts, and Interval Pattern Structures. *Fundam. Inform.*, 115(4):265–277, 2012. (Cited on page 20.)

[73] Jian Pei, Jiawei Han, and Runying Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30, 2000. (Cited on page 19.)

[74] John L. Pfaltz. Representing Numeric Values in Concept Lattices. In *CLA*, 2007. (Cited on page 3.)

[75] Jonas Poelmans, Paul Elzinga, Stijn Viaene, and Guido Dedene. Formal concept analysis in knowledge discovery: a survey. In *Proceedings of the 18th international conference on Conceptual structures: from information to intelligence*, ICCS'10, pages 139–153, Berlin, Heidelberg, 2010. Springer-Verlag. URL `http://dl.acm.org/citation.cfm?id=1881168.1881185`. (Cited on page 17.)

[76] Liu D.Y. Hu C.Q. Lu M. Zhao L. Qi, H. Searching for closed itemset with Formal Concept Analysis. In *3rd Int. Conf. on Machine learning and Cybernetics, Shanghai.*, 2004. (Cited on page 18.)

[77] Gerhard Ritter and Gonzalo Urcid. Learning in Lattice Neural Networks that Employ Dendritic Computing. In Vassilis Kaburlasos and Gerhard Ritter, editors, *Computational Intelligence Based on Lattice Theory*, volume 67 of *Studies in Computational Intelligence*, pages 25–44. Springer Berlin / Heidelberg, 2007. (Cited on pages 9, 21, 22, and 23.)

[78] Gerhard X. Ritter and Gonzalo Urcid. Lattice Neural Networks with Spike Trains. In *HAIS (2)*, pages 367–374, 2010. (Cited on pages 21 and 27.)

[79] Gerhard X. Ritter, Laurentiu Iancu, and Gonzalo Urcid. Neurons, Dendrites, and Pattern Classification. In *CIARP*, pages 1–16, 2003. (Cited on pages 9 and 27.)

[80] Gerhard X. Ritter, Darya Chyzhyk, Gonzalo Urcid, and Manuel Grana. A Novel Lattice Associative Memory Based on Dendritic Computing. In *HAIS*, pages 491–502, 2012. (Cited on pages 9 and 21.)

[81] Volker J. Hitzler Rudolph, S. Supporting Lexical Ontology Learning by Relational Exploration. *U. Priss, S. Polovina. And R. Hill ICCS*, LNAI 4604:488–491, 2007. (Cited on page 18.)

[82] P. V. G. D. Prasad Reddy Shashikumar G. Totad, R. B. Geeta. Batch incremental processing for FP-tree construction using FP-Growth algorithm. *Knowledge and Information Systems*, 1:1–16, 2012. (Cited on page 19.)

[83] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004. ISBN 0521813972. (Cited on page 8.)

[84] G. Stuart, N. Spruston, and M. Hausser. *Dendrites*. Oxford University Press, USA, 2007. URL `http://books.google.com.mx/books?id=hbn1RRrP004C`. (Cited on page 21.)

[85] Gerd Stumme, Rudolf Wille, and Uta Wille. Conceptual Knowledge Discovery in Databases Using Formal Concept Analysis Methods. In *Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD '98,

pages 450–458, London, UK, UK, 1998. Springer-Verlag. URL http://dl.acm.org/citation.cfm?id=645802.669204. (Cited on pages 17 and 18.)

[86] J. B. C. R. J.-F. B. Sylvain Blachon, RuggeroG. Pensa and O. Gandrillon. Clustering formal concepts to discover biologically relevant knowledge from gene expression data. *In Silico Biology*, 7: 467???483, 2007. (Cited on page 18.)

[87] Norbert Tsopze, Engelbert Mephu Nguifo, Gilbert Tindo, and Yaounde Cameroun. CLANN: Concept Lattice-based Artificial Neural Network for supervised classification. (Cited on page 20.)

[88] Gonzalo Urcid and Gerhard Ritter. Noise Masking for Pattern Recall Using a Single Lattice Matrix Associative Memory. In Vassilis Kaburlasos and Gerhard Ritter, editors, *Computational Intelligence Based on Lattice Theory*, volume 67 of *Studies in Computational Intelligence*, pages 81–100. Springer Berlin / Heidelberg, 2007. (Cited on page 21.)

[89] Gonzalo Urcid, Gerhard X. Ritter, and Laurentiu Iancu. Single Layer Morphological Perceptron Solution to the N-Bit Parity Problem. In *CIARP*, pages 171–178, 2004. (Cited on page 21.)

[90] Gonzalo Urcid, Jose Angel Nieves-Vazquez, Anmi Garcia-A., and Juan Carlos Valdiviezo-N. Robust image retrieval from noisy inputs using lattice associative memories. In *Image Processing: Algorithms and Systems*, 2009. (Cited on page 21.)

[91] Gonzalo Urcid, Juan Carlos Valdiviezo-N., and Gerhard X. Ritter. Lattice Associative Memories for Segmenting Color Images in Different Color Spaces. In *HAIS*, pages 359–366, 2010. (Cited on page 21.)

[92] V. K. Finn-S. O. Kuznetsov V. G. Blinova, D. A. Dobrynin and E. S. Pankratova. Toxicology analysis by means of the jsm-method. *Russia Institute for Scientific and Technical Information (VINITI)*, 13:1201–1207, 2002. (Cited on page 20.)

[93] Missaoui R. Godin R. Valtchev, P. Formal Concept Analysis for Knowledge Discovery and Data Mining: The New Challenges. *P. Eklund (Ed.): ICFCA*, LNAI 2961:352–371, 2004. (Cited on page 17.)

[94] Missaoui R. Godin R. Valtchev, P. A framework for incremental generation of closed itemsets. *Discrete Applied Mathematics*, 156: 924–949, 2008. (Cited on page 17.)

[95] P. Valtchev, R. Missaoui, and P. Lebrun. A partition-based approach towards constructing Galois (concept) lattices. *Discrete Math.*, 256(3):801–829, 2002. (Cited on page 17.)

[96] Petko Valtchev, Rokia Missaoui, Robert Godin, and Mohamed Meridji. Generating frequent itemsets incrementally: two novel approaches based on Galois lattice theory. *J. Exp. Theor. Artif. Intell.*, 14(2-3):115–142, 2002. (Cited on pages 17 and 18.)

[97] A. Volkova. Algorithmization of procedures of the JSM method for automatic hypothesis generation. *Automatic Documentation and Mathematical Linguistics*, 45:113–120, 2011. (Cited on page 20.)

[98] J. Wang, G. Karypis, J. Wang, and G. Karypis. BAMBOO: Accelerating Closed Itemset Mining by Deeply Pushing the Length-Decreasing Support Constraint. (Cited on page 19.)

[99] Rudolf Wille. Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies. In *Formal Concept Analysis*, pages 1–33, 2005. (Cited on pages 4 and 27.)

[100] Rudolf Wille. Methods of Conceptual Knowledge Processing. In *Formal Concept Analysis*. Springer Berlin Heidelberg, 2006. (Cited on page 18.)

[101] Rudolf Wille. Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts. In *ICFCA*, pages 314–339, 2009. (Cited on pages 4, 12, and 27.)

[102] Valrie Witte, Stefan Schulte, Mike Nachtegael, Tom Malange, and Etienne Kerre. A Lattice-Based Approach to Mathematical Morphology for Greyscale and Colour Images. In Vassilis Kaburlasos and Gerhard Ritter, editors, *Computational Intelligence Based on Lattice Theory*, volume 67, pages 129–148. Springer Berlin / Heidelberg, 2007. (Cited on page 21.)

[103] Zhipeng Xie, Wynne Hsu, Zongtian Liu, and Mong Li Lee. Concept Lattice based Composite Classifiers for High Predictability, 2002. (Cited on page 20.)

[104] X.S. Yang. *Introduction To Computational Mathematics*. World Scientific Publications, 2008. (Cited on page 3.)

[105] Seongwook Youn and Dennis McLeod. Improved spam filtering by extraction of information from text embedded image e-mail. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 1754–1755, New York, NY, USA, 2009. ACM. doi: 10.1145/1529282.1529677. URL `http://doi.acm.org/10.1145/1529282.1529677`. (Cited on page 20.)

[106] Ivashko V.G. Kuznetsov S.O. Mikheenkova M.A. Khazanovskii-K.P. Zabezhailo, M.I. and O.M. Anshakov. Algorithms and Programs of the JSM-Method of Automatic Hypothesis Generation. *Automatic Documentation and Mathematical Linguistics*, 21.5:1–14, 1987. (Cited on pages 16, 19, and 20.)

[107] Mohammed J. Zaki and Ching J. Hsiao. CHARM: An Efficient Algorithm for Closed Itemset Mining. (Cited on page 19.)

[108] Emmanuel Zenou and Manuel Samuelides. Characterizing Image Sets Using Formal Concept Analysis. *EURASIP J. Adv. Sig. Proc.*, 2005(13):1931–1938, 2005. (Cited on page 20.)

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.
UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Aplicaciones de Redes Neuronales dentriticas a Analisis de Conceptos Formales - Dentritic Neuronal Network applications to Formal Concept Analysis

del (la) C.

David Ernesto CARO CONTRERAS

el día 08 de Febrero de 2013.

Dr. Luis Ernesto López Mellado
Investigador CINVESTAV 3C
CINVESTAV Unidad Guadalajara

Dr. Félix Francisco Ramos Corchado
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara

Dr. Andrés Méndez Vázquez
Investigador CINVESTAV Guadalajara
2C
CINVESTAV Guadalajara