

POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica (Computer Engineering)

Tesi di Laurea Magistrale

**Novel Neural Techniques for Gene
Expression Analysis in Cancer
Prognosis**



Relatore

prof. Elio Piccolo

Correlatori:

prof. Giansalvo Cirrincione

prof. Andrea Bertotti

Gabriele CIRAVEGNA

matricola: 234726

Pietro BARBIERO

matricola: 252818

ANNO ACCADEMICO 2017 – 2018

Summary

This manuscript summarises two years of analyses, experiments and developments in the machine learning field. During that period, authors have collaborated in devising novel ideas and applying them to real world problems.

The main application setting is related to the analysis of patient derived xenografts (PDXs) of metastatic colorectal cancer (mCRC). PDXs are obtained by propagating surgically derived tumor specimens in immunocompromised mice. Through this procedure, cancer cells remain viable *ex-vivo* and retain the typical characteristics of different tumors from different patients. Hence, they can effectively recapitulate the intra- and inter-tumor heterogeneity that is found in real patients. During the last decade, the Candiolo Cancer Institute (Italy, IRCC) has been assembling the largest collection of PDXs from mCRC available worldwide in an academic environment. Such resource has been widely characterized at the molecular level and has been annotated for response to therapies, including cetuximab, an anti-EGFR antibody approved for clinical use. The mCRC PDX samples analyzed in this manuscript were kindly provided by IRCC in the form of microarray data, i.e. a large table containing the gene expression levels of tumor cells. Indeed, the medical objectives of the analyzes described in this work concern, on the one hand, the extraction of gene expression patterns useful for the instruction of therapies and further clinical experiments, and, on the other hand, the creation of models capable to correctly classify unlabeled data according to the cancer response to drugs.

From a statistical and machine learning point of view, the main difficulty in dealing with such data is the so-called curse of dimensionality. Indeed, only few hundreds of PDXs (samples) were provided against tens of thousands of gene expressions (features). Preliminary analyses performed with state-of-the-art techniques perform poorly when dealing with this problem, reporting limited effectiveness and opaque models. Thus, the machine learning objectives were to improve the effectiveness of existing models and designing ad-hoc techniques to deal with high-dimensional data.

Analyses have been performed through both supervised and unsupervised methods. In particular, two different and delimited directions of work were carried out according to the typology of methods. This choice was made at the beginning of this work in order to also sub-divide the medical objectives to pursue. Roughly speaking, in fact, the extraction of gene expression patterns is commonly achieved through unsupervised techniques, while the creation of a classifier model can be obtained only with supervised techniques. Nonetheless, meaningful insight into gene behaviours were provided also through supervised learning. Authors equally contributed in the development of all analyses presented by alternating on the two paths.

Chronologically, the research was carried out in two different stages: initially, the high dimensional problem was addressed through the development of feature selection algorithms. Moreover, during this phase, the dataset was analyzed with state-of-the-art techniques for data visualization, like parallel coordinates diagrams, as well as feature extraction and manifold analysis, like CCA. Results of these preliminary analyses were published in two different chapters as book chapter

of the Springer series “Quantifying and Processing Biomedical and Behavioral Signals”: "Unsupervised gene identification in colorectal cancer" and "Supervised gene identification in colorectal cancer". In the second stage, the attention of this research, from the unsupervised point of view, focused on the development of a biclustering framework capable of extracting useful gene correlations. In order to do that, a completely novel neural clustering technique called GH-EXIN was devised. Further details about the main features of this technique will be given in the following. The results were presented in "Neural biclustering in gene expression analysis", in the proceedings of the CSCI conference, Las Vegas 2017. This biclustering framework was successfully tested also on an external dataset for face detection problem and the results were presented in "Assessing discriminating capability of geometrical descriptors for 3d face recognition by using the GH-EXIN neural network", in the proceedings of the 2018 WIRN conference in Vietri sul Mare. From the supervised point of view, instead, different states of the art classifiers were compared in order to create the best classifier model. Finally, the best performer was a shallow neural network, with hyper-parameters tuned through an evolutionary algorithm. At the same time, another feature selection algorithm based on shallow neural network weight analysis was developed and, compared with ANOVA, performs a more effective selection for instructing a successive neural network model. The works were proposed and accepted at the 2018 WIRN conference, in two conference articles entitled "Evolutionary optimization of neural network hyperparameters" and "Understanding cancer phenomenon at gene-expression level by using a shallow neural network chain".

The main innovation proposed in this manuscript is represented by GHEXIN, a novel neural based algorithm for hierarchical clustering. The proposed approach builds a divisive hierarchical tree in an incremental and self-organized way. Indeed, it is a top-down technique which divides data at deeper and deeper levels. It is data driven (self-organization), in the sense that the final tree is automatically estimated from data. Also, it does not require a predefined number of units and levels (incremental with pruning phase). With regard to state-of-the-art neural based algorithms (as GHNG and DGSOT), GH-EXIN shows remarkable innovations:

- It performs a semi-isotropic quantization of the input space, by exploiting both isotropic and anisotropic criteria. The isotropic criterion is based on the extent of the neuron neighborhood, like in most neural based approaches. The anisotropic criterion (unique feature of GH-EXIN) is topological, as it is modelled by the convex hull generated by the weight vector of the winner neuron and the weight vectors of its topological neighbours (i.e. those connected through edges).
- It exploits sophisticated data reallocation and outlier detection methods. Data reallocation is the mechanism by which data associated to pruned nodes (orphan data) are possibly reassigned to other neurons. In GH-EXIN all orphan data are labelled as potential outliers at the end of each epoch. For each potential outlier, GH-EXIN looks for a new winner among all leaf nodes. Only when the winner belongs to the same basic neural unit of the pruned node and the potential outlier is outside its hypersphere, is datum definitely marked as outlier and is not reassigned.
- It exploits a simultaneous vertical and horizontal growth in order to represent abstract characteristics of the observed phenomenon. At the end of the training process of a basic neural unit, the resulting graph is analyzed by searching for connected components. If more than one connected component is detected, the algorithm tries to abstract a representation from the observed phenomenon. Hence, each connected component, representing a cluster of data, is associated with a novel abstract neuron. The reference vectors of abstract neurons are placed in the centroids of the respective clusters. The tree structure is modified by inserting abstract neurons between the leaf nodes and the father node, resulting in a simultaneous vertical and horizontal growth.

The comparison with aforementioned techniques shows how GH-EXIN is typically more efficient, as it reaches similar performances in terms of peak-signal to noise ratio (PSNR) by using fewer neurons. Moreover, qualitative evaluation of the resulting topology shows how GH-EXIN is much more elegant in connecting neurons, providing superior manifold representations. Finally, the restricted number of user-dependent parameters makes the tuning process of GH-EXIN very easy. The GH-EXIN source code was fully developed by authors in MATLAB and is publicly available on BitBucket¹. The application of the biclustering framework integrated with GH-EXIN on the biological dataset revealed some interesting gene correlation patterns. These results have been submitted to the attention of IRCC doctors, who are currently analyzing them for possible scientific implications.

The results above are promising and highlight the potential for future work. From the point of view of the biological advances, the outcomes of both the unsupervised and the supervised path are promising yet opaque: while the models can be used effectively, the results are difficult to interpret from a human point of view. As for the unsupervised direction of work, the GH-EXIN neural network resulted to be effective and easy to use as aforementioned; results provided by the biclustering framework, instead, are rather difficult to interpret without a statistical knowledge, since biclusters need several additional tools to be correctly evaluated. As for the supervised direction, the major advances with respect to previous analyses are achieved by exploiting the shallow neural network model. Indeed, the simplicity of such model makes it easier to handle and interpret. On limiting the number of features used, however, the accuracy drops significantly.

Moreover, major limitations of our work directly derive from the analyzed data. On the one hand, the analyzed data represent an estimate of the amount of times each gene is transcribed in a tumor xenograft. However, gene replication does not always result in protein generation. Indeed, this kind of data may not represent cell behaviour correctly. On the other hand, the restricted amount of samples was the most serious issue, since machine learning reliability is directly related to the amount of data provided.

Hence, within the biological domain, future developments will involve the use of up-to-date data (e.g. representing proteins instead of gene expressions) and the integration with other sources of data, such as image samples. Besides, from a machine learning point of view, models easier to interpret may be developed in order to provide more reliable and human-understandable outcomes. Future research in this field will consist in devising new algorithms overcoming the intrinsic weaknesses of machine learning, above all understandability. To this purpose, novel algorithms which integrate classic symbolic artificial intelligence with machine learning techniques seem to be very promising.

¹Gabriele Ciravegna and Pietro Barbiero. GH-EXIN (version 1.0.1). https://bitbucket.org/machine_learning_research/ghexin/src/master/, 2018

Acknowledgements

We have to thank few people who personally contributed to the production of this work:

- *Our supervisors, Elio Piccolo and Giansalvo Cirrincione, who followed the development of this work with interest and passion far beyond institutional duties;*
- *Andrea Bertotti, from Istituto per la Ricerca e la Cura del Cancro (IRCC), who instructed us about the cancer phenomenon and provided us with the dataset;*
- *Giovanni Squillero, from Politecnico di Torino, who helped us in the development of the evolutionary algorithms;*
- *Alberto Tonda, from Institut National de la Recherche Agronomique (INRA), who contributed to the development of the supervised neural networks.*

The following acknowledgments are related to Gabriele Ciravegna only.

I want to thank many people who have always supported me during these years and in particular during the last months. They have contributed to the achievements of this result too.

Just to name a few:

- *The co-author Pietro Barbiero, endless source of ideas and projects;*
- *My mother, pure love*
- *My father, my fundamental Talking Cricket*
- *My uncle Fabio, who drove me to choose Computer Science Engineering*
- *My grandmother, my brother and my sister-in-law, whom I can always count on*
- *Silvia, who first believed in me*
- *My friends, among others, Amedeo, Stefano, Costanza, Bianca, Matilde, Francesca, Matteo, Guglielmo, Fabio, Giuseppe, Francesca, Luca*

Thank you from the bottom of my heart.

Contents

| | | |
|-----------|---|-----------|
| I | Introduction | 9 |
| 1 | Biological Background | 11 |
| 1.1 | Colorectal Cancer | 11 |
| 1.1.1 | What Is Cancer? | 11 |
| 1.1.2 | Why Does Cancer Arise? | 12 |
| 1.1.3 | CRC | 13 |
| 1.2 | Xenografts | 14 |
| 1.3 | Gene Expression Analysis | 14 |
| 2 | Machine Learning Background | 17 |
| 2.1 | History | 17 |
| 2.2 | Achievements | 18 |
| 2.3 | Deep Learning Limits and Weaknesses | 19 |
| 2.3.1 | Brief Deep Learning Theory | 20 |
| 2.3.2 | Limits | 20 |
| II | Preliminary Data Analysis | 23 |
| 1 | The High Dimensional Problem | 25 |
| 2 | Unsupervised Manifold Analysis | 27 |
| 2.1 | Unsupervised Feature Selection | 27 |
| 2.2 | Parallel Coordinate Plot | 28 |
| 2.3 | Sub-Manifold Analysis | 29 |
| 2.4 | Biological Feedback | 32 |
| 3 | Supervised Manifold Analysis | 33 |
| 3.1 | Supervised Feature selection | 33 |
| 3.2 | Sub-Manifold Analysis | 33 |
| 3.3 | Lasso regression | 35 |
| 3.4 | Biological Feedback | 36 |
| 3.5 | Classification | 36 |

| | | |
|------------|--|-----------|
| III | Advanced Data Analysis Through Neural Techniques | 39 |
| 1 | Unsupervised Neural Techniques | 41 |
| 1.1 | Introduction | 41 |
| 1.2 | Biclustering | 41 |
| 1.2.1 | Neural Framework | 43 |
| 1.3 | Clustering State-of-the-art | 45 |
| 1.3.1 | GHNG | 45 |
| 1.3.2 | DGSOT | 46 |
| 2 | GH-EXIN | 49 |
| 2.1 | A Novel Neural Technique: GH-EXIN | 49 |
| 2.1.1 | Self-Organizing and Data Driven Approaches | 49 |
| 2.1.2 | Basic Neural Units | 49 |
| 2.1.3 | Tree Building | 54 |
| 2.1.4 | The GH-EXIN Algorithm | 58 |
| 2.1.5 | User-Dependent Parameters | 59 |
| 2.2 | Benchmark Comparison | 59 |
| 2.3 | Gene Analysis | 69 |
| 2.3.1 | Neural Biclustering with GH-EXIN | 69 |
| 2.3.2 | Validating Techniques | 70 |
| 2.3.3 | Experimental Findings | 72 |
| 2.4 | An Application to Face Recognition | 74 |
| 3 | Supervised Neural Techniques | 81 |
| 3.1 | Introduction | 81 |
| 3.2 | Neuroevolution | 81 |
| 3.2.1 | Mathematical Model of the Shallow Neural Network | 81 |
| 3.2.2 | Individuals | 84 |
| 3.2.3 | Generator | 84 |
| 3.2.4 | Evaluator | 85 |
| 3.2.5 | Selector | 85 |
| 3.2.6 | Variator | 85 |
| 3.2.7 | Replacer | 86 |
| 3.2.8 | Terminator | 86 |
| 3.2.9 | Comparison of the Results | 86 |
| 3.3 | Transparent Neural Based Model for Feature Selection | 88 |
| IV | Conclusion and Future Developments | 91 |
| 1 | Critical Analysis | 93 |
| 2 | Future Works: Towards Artificial General Intelligence | 95 |
| | Bibliography | 97 |

Part I

Introduction

Chapter 1

Biological Background

1.1 Colorectal Cancer

Before starting any analysis, a brief but fundamental study of the matter has been necessary. As computer scientist and even more as data scientist, it is always important to spend some time trying to deeply understand the characteristics of the problem you are analyzing. Producing a deep statistical analysis without being able to understand the results is pointless most of the times. Nevertheless, in the case at hand, the problem is such complex that a medical response is still necessary. The following, indeed, is only a rough summary of main cancer features without claiming to be complete.

1.1.1 What Is Cancer?

Under the word cancer, a collection of related diseases is grouped. Common traits to these diseases are the abnormal growth of the cells and their capacity of spreading to surrounding tissues.

A list of different anomalies allows cancer cells to grow abnormally. Firstly, it is possible to notice that old and damaged cells don't die, differently from normal cells. This characteristic is caused by the capacity of tumor cells to ignore apoptosis signals. These signals are normally sent by the system to get rid of unneeded cells. In cancer cells, instead, cycle-regulating genes as RAS and p-53 result to be altered or often deactivated, causing apoptosis pathways not to be followed. Further signals sent by the immune system are also ignored, as those preventing cells from replicating when its DNA is altered. Furthermore, the immune system generally stops sending nutrients to unneeded cells causing target cells to starve. In cancer cells, however, gene responsible to autophagy are also inhibited and cells do not get destroyed even though they are starving. Cancer cells are also able to influence surrounding micro-environment to avoid this phenomenon: blood vessels are created nearby tumoral cells in order to feed them of oxygen and nutrients, necessary elements for cell lives. These vessels are also used by the cells to throw away cellular waste products [1]. Secondly, but not least, new cells are formed even though they are not needed. These new cells keep dividing by mitosis which allows cancer to grow more and more.

Cancer spreading to other tissues, instead, may occur at different levels. At first, it is possible to notice that tumor has affected nearby tissues; afterwards, it may be possible

to notice that tumoral cells are growing on distant organs. This phenomenon is called metastasis which from the Greek literally means "transposition" of a disease: in this case, the tumor. It is worth to notice that cells found inside a brain metastasis are colon tumoral cells, not brain tumoral cells. Original cells, in fact, spread through blood or through lymph system, attach to existing tissue of distant organs and start growing forming the metastasis, as shown in figure 1.1. Among many reasons, this occurs because cancer cells are less specialized than normal cells and, hence, they are less likely to be rejected by new tissues.

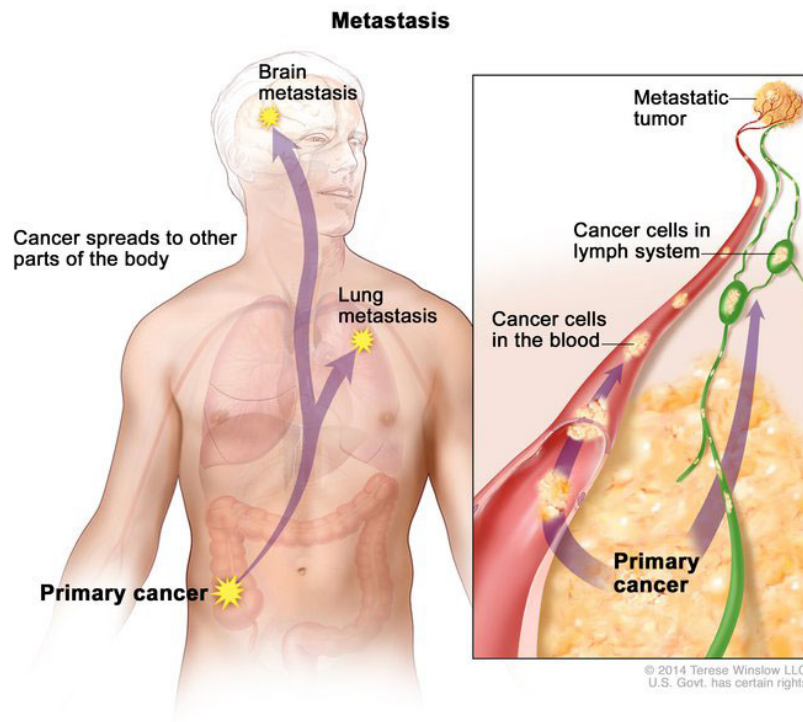


Figure 1.1: Cancer spreading from its original site to distant organs

The capacity of spreading to surrounding tissues is the main difference between malignant and benign tumors. Another characteristic typical of malignant tumor is the capability of growing back after being extracted.

1.1.2 Why Does Cancer Arise?

Fundamentally, cancer is a genetic disease and its appearance is due to some gene behaviour changes. These changes are generally either inherited from our descendants or caused by environmental factor.

Going into further details, there are three main types of gene whose mutation is critical for cancer growth.

1. Proto-oncogenes: involved in cell growth or inhibition of apoptosis, these genes may turn into oncogenes after some genetic mutations and, therefore, promote cancer

emergence.

2. DNA-repair genes: their scope is to repair damaged DNA after a mutation occurred, caused either by an environmental factor or a metabolic activity.
3. Tumor-suppressors: their role is to protect cell from alterations either by repressing genes that promote cell division when not needed or when DNA is damaged, or by starting apoptosis if DNA damage could not be fixed. Furthermore, they prevent tumor cells from dispersing (and creating metastasis) by blocking locomotion when DNA is irreparably damaged.

As first noticed in the *two-hit hypothesis* [2], cancer is the result of many mutations to cell's DNA: in particular, it is the result of both the mutation of proto-oncogenes and the deactivation of tumor-suppressor genes.

What is really important to notice here is that each cancer is a unique combination of genetic changes. Because of this inter-patient heterogeneity, research in these years has focused on personalized therapies to increase their efficiency [3]. Nevertheless, to achieve this goal further studies are needed, in particular on bio-marker discovery, which is the scope of this work. In fact, many gene functionalities and how different genes are co-regulated under specific circumstances is still unknown []. For personalized therapies, in particular, it is important to know a priori whether the patient will heal using a certain drug or not, given the genetic expression of the cancer tissue.

1.1.3 CRC

Colorectal cancer (CRC) is a particular type of cancer that develops in the large-intestine. It generally starts as an adenomatous polyp which may turn into an adenocarcinoma as shown in figure 1.2.

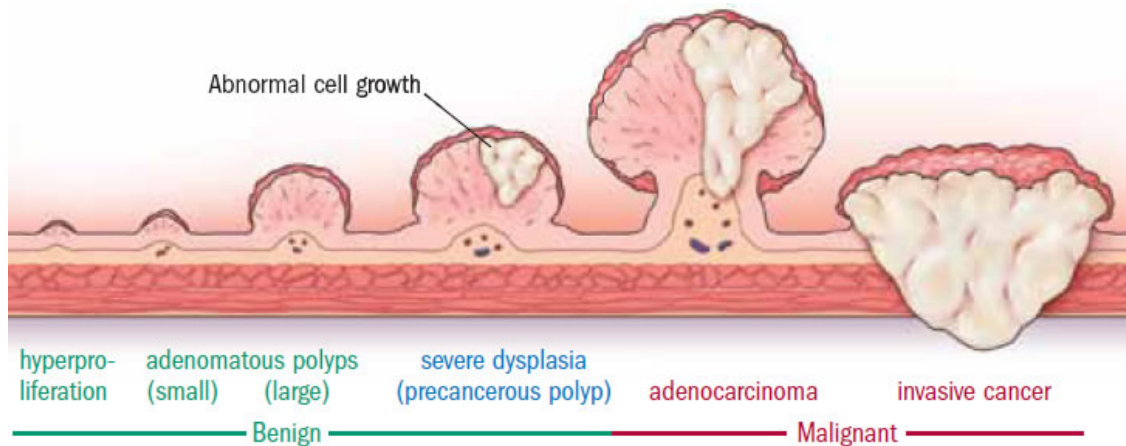


Figure 1.2: Colorectal cancer evolution

Recently, computer-aided diagnosis researches obtained good results in the classification of colorectal polyps through Convolutional Neural Network (CNN) system [4]. These systems are used in order to determine whether analyzed cells belong to a healthy tissue,

or to an adenoma (which may be a signal of potential successive cancer), or to an adenocarcinoma (which is already cancer). The results are commonly used to create an attention map of possible cancer areas, which may drive doctors during a prognosis.

This type of cancer has as main causes the advanced age of the patients and lifestyle factors: among others, diet (in particular excessive consumption of red meat and alcohol), pollution, smoke, obesity and lack of physical activity play a key role.

Colorectal cancers are the third most common cancer globally as reported by the 2014 World Cancer Report [5]. There are about 1.4 million new cases and 694 thousand deaths each year because of colorectal cancer. It is more likely to appear in man than in woman and it is more present in the developed countries than backward countries.

1.2 Xenografts

Medical treatment of cancer is an extremely complex problem. Due to intra- and inter-tumor heterogeneity, the same drug may have different levels of effectiveness on patients with the same type of cancer. Therefore, personalized approaches are required to increase the reliability of prognostic predictions and the efficacy of therapies. Recently, new powerful tools have been developed for biomarker discovery and drug development in oncology, which rely on a technology called Patient-Derived Xenografts (PDXs). A Xenograft, in general, is a cell, tissue or organ that is transplanted from one species to another. PDXs, in this particular case, are surgically-extracted tumor tissue specimens from patient affected by colorectal cancer. These tissues are then transplanted into immunocompromised mice as shown in fig 1.3.

Through this, cancer cells remain viable ex-vivo and retain the typical characteristics of different tumors from different patients. Hence, they can effectively recapitulate the intra- and inter-tumor heterogeneity that is found in real patients. Based on this idea, the PDX technology has been employed to conduct large-scale analyses which, as in this work, try to identify reliable correlations between genetic or functional traits and sensitivity to anti-cancer drugs. In this context, metastatic colorectal cancers (mCRC) have been collected for the last ten years and have generated the largest PDX biobank available worldwide in an academic environment. This collection has been already characterized at the molecular level and has been exploited to identify clinically relevant biomarkers for prediction of therapeutic efficacy [6].

1.3 Gene Expression Analysis

Transcriptional data were obtained from mCRC PDXs through the Illumina microarray technology [7]. The microarray technology is based on thousands of DNA microscopic probes placed on a solid surface such as a piece of glass, plastic or a chip, forming an array. These arrays allow to measure the gene expressions on a tissue sample.

The word gene expression refers to the quantity of Messenger RNA (mRNA) produced at a certain moment in a certain cell. The RNA synthesis (transcription) is a transfer of the information from the DNA where it is stored into mRNA which can be transported and interpreted. Later, mRNA moves the information to the ribosomes to enable the

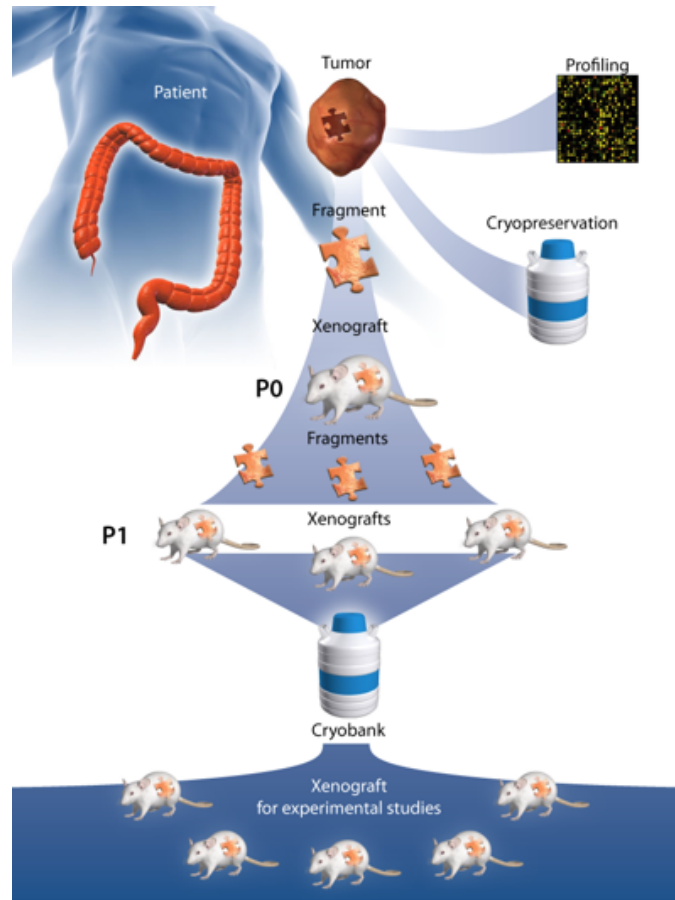


Figure 1.3: Patient-Derived Xenografts

production of protein (translation). Protein will finally respond to cellular needs for which transcription started.

In theory, genes contain the information to produce all kind of mRNA. Nonetheless, in the biological samples, genes are transcribed into mRNA sequence in different quantities. This occurs for some reasons:

- Cell typology, different type of cells (brain cell, epithelial cells, liver cells) express different genes: among other factors this is what makes them different;
- Environmental factors such as the time of the day, whether the cell was proliferating or not and the presence of signals sent from other cells: according to cell needs, different gene are transcribed.

Gene expression analyzes often require measuring mRNA quantity in different conditions. High mRNA levels of a sequence under specific condition may imply, indeed, a cellular need for the protein coded by the sequence. This kind of need may suggest a pathological condition when found on sick patients and not on healthy patients. As an example, if cancer cells express higher mRNA levels associated to a specific receptor, it is possible to infer that the receptor play a role in that type of cancer.

The focus of this work is about predicting drug sensitivity in PDXs and human patients affected by CRC. Hence, the different condition under which gene expression levels are evaluated are the different responses to drugs. In particular, this research tries to find out which genes discriminate whether a drug is capable to heal the patient or not. Further details will be given from part [II](#).

Chapter 2

Machine Learning Background

In the last decade, Machine Learning proved to be the enabling technology for many innovations in different fields. Nowadays, it is the most important trend in computer science according to Gartner Hype Cycle [8]. Nonetheless, it is possible that the expectation created among investors, who are putting billions of dollars in AI research, and among customers will not be completely satisfied in the next future, as also machine learning has some intrinsic limits.

2.1 History

The term *artificial intelligence* (AI) was coined by John McCarthy during a conference at Dartmouth College in 1956. During his talk, he proposed to design an artificial machine able to learn and reason like human beings. In the same years, Allen Newell and Herbert Simon developed Logic Theorist, the first program able to apply basic reasoning functionalities. However, until 70's the advances in artificial intelligence were purely academic. Indeed, the proposed approaches were unable to tackle real world problems, due to computational power and memory limits, and reasoning deficiencies [9]. The first turning point in AI occurred when Feigenbaum introduced DENDRAL and MYCIN, the first expert systems, based on symbolic approaches. These algorithms were able to incorporate human knowledge and manipulate basic logic principles in order to provide reliable inferences in specific domains. The reliability guaranteed by expert systems allowed the spread of such algorithms for commercial use.

The second turning point happened in 1986, thanks to the work of Rumelhart, Hinton, Williams and McClelland on the backpropagation algorithm. They showed how to apply efficiently this algorithm for training multi-layer perceptron neural networks, overcoming the deficiencies pointed out by Minsky [9]. With backpropagation the learning problem was shifted as an optimization problem, where the model fits its parameters directly on data, without being explicitly programmed. This change of perspective generated an AI sub-field picturesquely called machine learning.

Nevertheless, until few years ago it was not feasible to effectively employ complex (or deep in case of neural networks) models as they require large amount of both computational power and data to be optimized. Everything changed in 2012 when different works

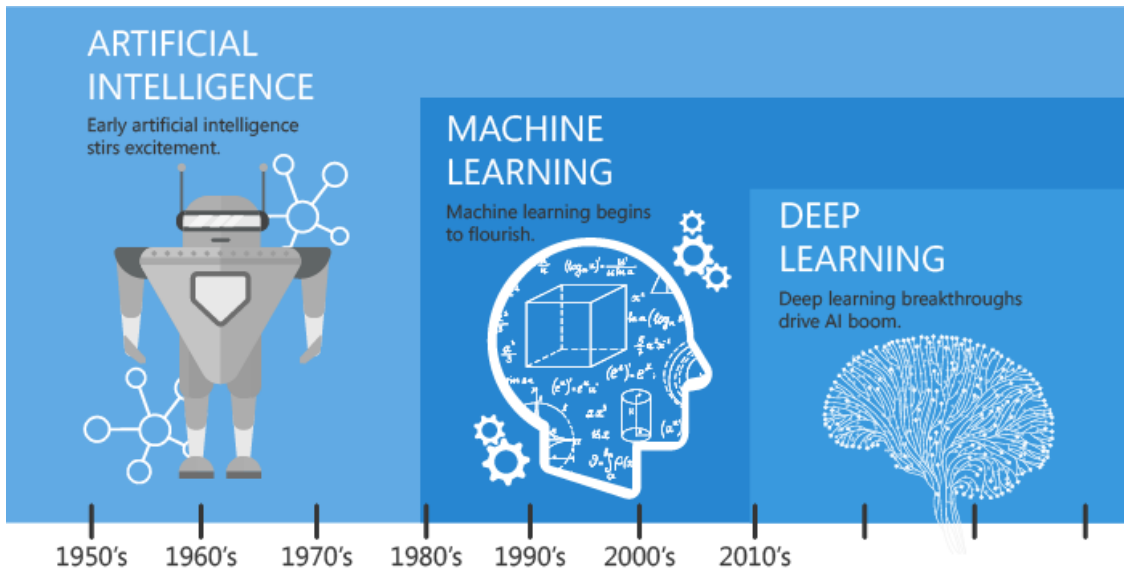


Figure 2.1: Artificial Intelligence progresses

achieved important results using Deep Neural Networks in challenges such as ImageNet in object recognition [10]. The technological progress provided scientists with ever more powerful processors, while the advent of Big Data and IoT at the beginning of the 2010s started producing the quantity of data required by models to be effective. It is possible to state that the advances in the field of deep learning will be strictly correlated in the future to the ones of IoT and Big Data; the information extraction process always starts from data. As it will be possible to apply sensors on even more objects which in the past were not accessible (e.g. human organs), new applications of machine learning will be discovered.

2.2 Achievements

Machine learning is nowadays already changing our daily life in many different fields:

- Preventive diagnosis: machine learning is achieving good results in healthcare, in particular for heart diseases and cancer diagnosis. In some conditions, an early diagnosis can save a patient's life. Statistics state that 17.9 million people die every year because of a heart attack [11] and 9.6 million because of cancer [12]: together they account for about half of the global deaths. A similar idea is also applied in preventive maintenance for machines: it allows companies to save thousands of dollars, avoiding replacing parts unless it is necessary.
- Autonomous car: in the very next years we will be relieved of the chore of driving, resulting in a significant reduction of traffic collisions and enhanced mobility for the elderly, children and disabled. Nowadays, around 1.3 million people die every year in a car accident [13]: hypothetically, this number could be reduced to zero with autonomous cars.

- Security: face recognition algorithms are widely used in video surveillance to find wanted criminals. Thousands of cameras have been deployed in our cities, it would require hundreds of people to check them all. Furthermore, emotion recognition and other algorithms are used to preempt crimes from taking place by analyzing people unusual behaviours.
- Language translation: results provided by Google Translate are astonishing, in many languages it performs almost as well as a human translator.
- Virtual assistants: many people nowadays are used to speak to their smartphones to save appointment on the agenda or to check weather conditions. This has become possible thanks to advances in speech recognition.
- Fraud detection: machine learning is used in this field for analyzing mails, detecting fishing attempts and generally spam. Banks and other financial actors are also using it for analyzing bank transactions, in order to detect illegitimate ones and money laundering.
- Product recommendation: Amazon completely transformed online shopping by suggesting people items they may be interested in. Google AdWords did the same with personalized advertising: nowadays it is the only competitor for publishing commercials online.

Most of these achievements have been feasible due to deep learning. Nowadays, it is the most popular e probably powerful technique. Because of this reason, in the following it will be analyzed in further details trying to highlight what are its limits. Anyway, much of the below mentioned applies also for machine learning in general.

2.3 Deep Learning Limits and Weaknesses

Deep Learning advances in recent years have brought great excitement in AI world. Andrew Ng, one of the pioneers of Deep Learning and founder of Google Brain, asserted: “If a typical person can do a mental task with less than one second of thought, we can probably automate it using AI either now or in the near future” [14]. Is it true? Is current artificial intelligence (mainly based on deep learning) already able to carry out now or in the next future whatever human mental task? Other researchers expressed a different opinion about it: "Scaling up current deep learning techniques by stacking more layers and using more training data can only superficially palliate some of these issues. It will not solve the more fundamental problem that deep learning models are very limited in what they can represent, and that most of the programs that one may wish to learn cannot be expressed as a continuous geometric morphing of a data manifold" [15] As partial proof of it, an important project as the full-service chatbot M developed by Facebook already failed to achieve its initial goals because they were beyond deep learning current capacities and eventually has been closed.

2.3.1 Brief Deep Learning Theory

In order to better understand strengths and limits of deep learning, it is good to know how it works. Basically, it is a statistical technique that performs a mapping between an input space and an output space. Learning is performed in a supervised way with thousands of labelled data supplied to the algorithm to build a robust model. Internally, neural networks are made of nodes which exchange information. They are grouped into layers with an input layer which represents the input space, an output layer with as many nodes as the elements of the output space are and several hidden layers: the more the number of hidden layers the deeper is the network. Nodes are linked to each other and a weight is associated to each connection. These networks are generally told to represent neural cortex although in a very simplistic way. Nodes can be thought as neurons and connections between nodes may stand for the synapses.

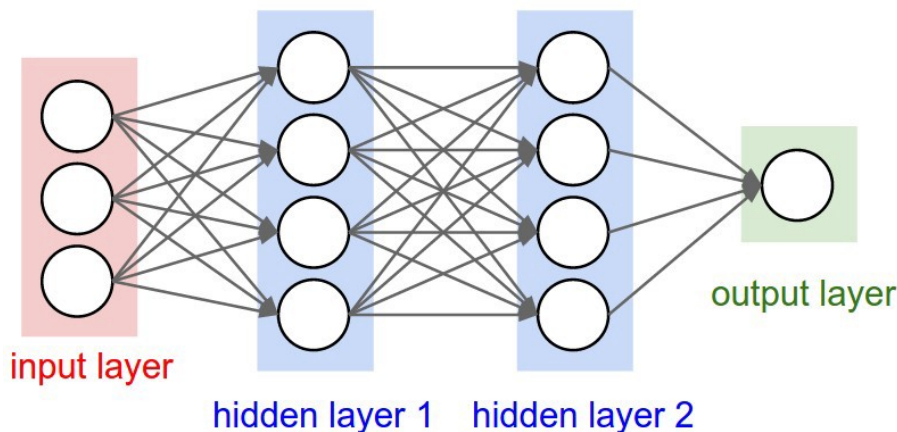


Figure 2.2: Classic Neural Network Representation

2.3.2 Limits

The incredible strength of deep learning is that “In principle, given infinite data, deep learning systems are powerful enough to represent any finite deterministic ‘mapping’ between any given set of inputs and a set of corresponding outputs” [16]. Nevertheless, this statement also reveals some of its limits:

- Generalisation: we will never be able to provide ‘infinite data’ to the systems. This implies deep neural networks will always be brittle, failing as fast as the context on which the model was trained change a little. An example may be the DeepMind’s Atari game work [17]: the system learns in 240 minutes of training to play Breakout. Nonetheless, it has been proved [18] that the same previously trained system fails on a transfer test where a minor number of variations are made in the game such as inserting another wall in the middle. This implies that the system has not really learnt to break the wall. It has only acquired a sequence of moves to be performed to break that single wall in a very narrow context.

- Abstraction: deep learning systems are only able to perform this specific mapping operation. They are built without any abstract representation of the world. They have no idea of concepts such as causality or composition, that all humans come to the world with, neither do they have the possibility to learn it, as it is impossible to represent these concepts in terms of features.
- Understandability: deep learning is usually considered and treated as a black-box because the way in which this ‘mapping’ is produced is so complicated that it is not explainable in form of rules. This introduces several limits to the use of this technique in critical applications such as medicine where the reason for which a decision needs to be taken is almost as important as the decision itself. Doctors cannot decide not to prescribe a saving-life drug to a patient only because a deep learning system suggested that it would not be effective on him, without really having the possibility to check why.

Hence, it seems possible that even with all conceivable computational power and data, new applications will be discovered, but deep learning’s effective capabilities may not grow any further.

Part II

Preliminary Data Analysis

Chapter 1

The High Dimensional Problem

The dataset stemming from the DNA microarrays is composed of the expression of 15396 genes in 203 Colorectal Cancer (CRC) murine tissues. For each tissue two additional quantities are available.

1. A discrete variable describing the cancer response to drugs, whose values are chosen as:
 - +1 (regressive cancer);
 - 0 (stable cancer);
 - -1 (worsening cancer).
2. A continuous variable representing the cancer response to drugs after three weeks, estimated as the difference in size of the tumor.

Data are preprocessed by the z-score technique in order to work on the same range. All analyses in this part have been done by considering the genes as variables. This is a very challenging problem because of analyzing very high dimensional data by means of a small training set. The only possible way to overcome this difficulty is the dimensionality reduction, even if it is data-driven too. As a consequence, the Principal Component Analysis (PCA in an under-determined framework) has been performed both for searching a first rough estimation of the intrinsic dimensionality of data and, above all, to test the non-linearity of the problem at hand. However, it results that at least 100 principal components are needed in order to explain the 90% of the data variance as shown in fig 1.1. This number has the same order of magnitude of the training set size and is a consequence of the high dimensionality and, probably, of the fact that the manifold is nonlinear. Other tools are needed in order to check if the manifold is linear or not.

In the following chapters two different kind of manifold analysis are shown that approach the high dimensional problem in two different ways. In chapter 2 the high dimensional problem is approached through an unsupervised feature selection which make use only of gene expressions. In chapter 3, thanks to the discrete label assigned to each patient, the feature

selection is performed in a supervised way. In both chapters, further analyses are then conducted with different techniques according to the path: unsupervised or supervised.

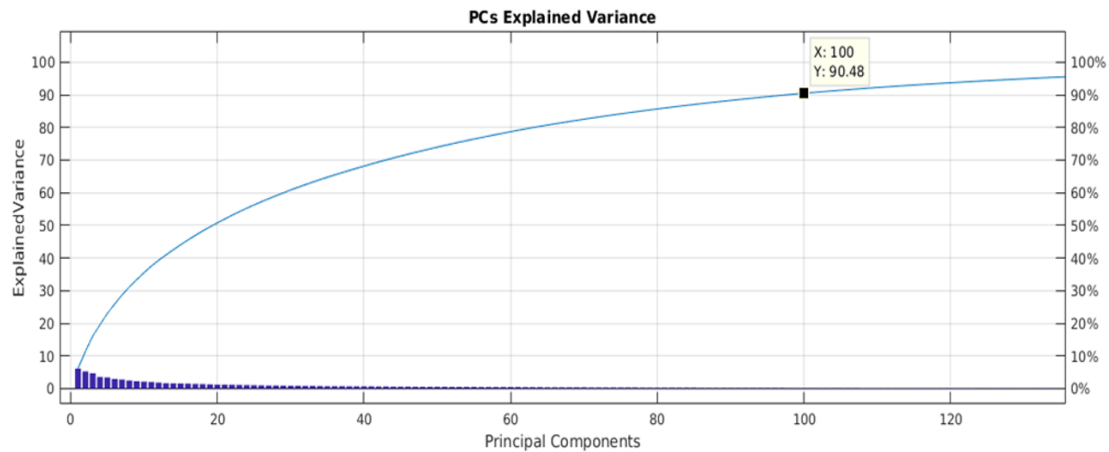


Figure 1.1: Pareto diagram of PC's explained variance

Chapter 2

Unsupervised Manifold Analysis

The unsupervised path makes use of a scoring algorithm of features based on their clustering ability to bypass the high dimensionality issue. Traditional methods of dimensionality reduction and projection are then used on subset features with high discriminant power in order to better analyze the data manifold. This chapter has been extracted from a work presented at the 2017 "Workshop Italiano sulle Reti Neurali" (WIRN2017) conference [19].

2.1 Unsupervised Feature Selection

Initially, tissues have been grouped considering all the available genes (features) with the Unweighted Pair Group Method with Arithmetic Mean (UPGMA), an agglomerative hierarchical clustering approach [20]. This bottom-up approach finds and merges the nearest pair of clusters r and s according to their mutual average distance:

$$d(r, s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} d(x_{ri}, x_{sj}) \quad (2.1)$$

The average distance algorithm has been chosen because it is robust against noise and outliers. Among different metrics, the Minkowski distance of order $p = 1$ has been found to have the highest cophenetic correlation coefficient (here 0.8392), therefore it is used in the following analyzes [21]. However, this approach has not provided satisfactory results; indeed, it was not able to find meaningful groups. For this reason, tissues have been processed using a different procedure based on the Ward's minimum variance method [22]. This algorithm finds and merges the pair of clusters that leads to minimum increase in the total within-cluster variance after merging. The within-cluster variance increment due to the merging of r and s is proportional to the distances of the resulting cluster objects from the resulting cluster centroid:

$$d(r, s) = \sqrt{\frac{2n_r n_s}{(n_r + n_s)} \|x_r - x_s\|_2} \quad (2.2)$$

Differently from the first approach, the clustering algorithm is applied using one feature at a time, determining a one-dimensional clustering which yields the individual ability of discrimination of genes. This property is evaluated using the Calinski-Harabasz index (also called Variance Ratio Criterion, VRC, [23]):

$$VRC_k = \frac{SS_B(N - k)}{SS_W(k - 1)} \quad (2.3)$$

where N is the number of samples, k is the number of clusters, SS_B is the between cluster variance and SS_W is the within cluster variance. Well defined clusters tend to have a high VRC. Therefore, genes are ranked according to this index. By defining a threshold in advance, several genes can be extracted. In other words, genes are selected according to their ability to discriminate tissues: this is estimated by checking the best possible separation (with regard to several choices of the number of clusters by means of the parameter $K \in [2,6]$) in terms of quality of the groupings by using an index (see Fig. 2.1).

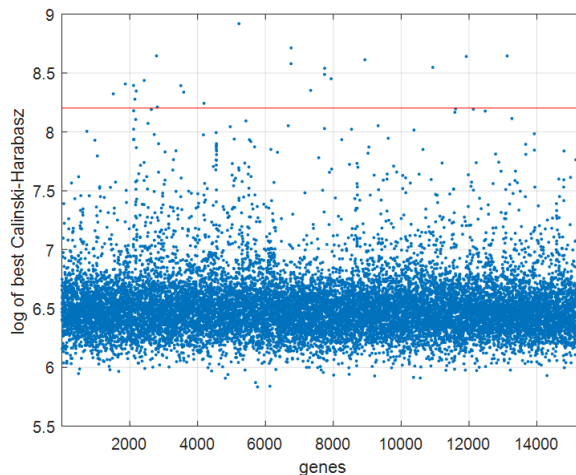


Figure 2.1: Cluster quality evaluation. The red line is the threshold.

2.2 Parallel Coordinate Plot

Parallel coordinates are a powerful way of visualizing high-dimensional data. This kind of data visualization was invented during the 19th century and sharpened by Wegman in 1990 [24].

A point in n -dimensional space is represented as a polyline with vertices on equally spaced parallel axes each of one representing a feature; the position of the vertex on the i -th axis corresponds to the i -th coordinate of the point. This plot is used in order to understand deeply the gene capacity of discrimination, by visualizing the distribution of the murine tissues (colored polylines) along all the dimensions (genes) represented as parallel vertical axes. In this figure blue lines represent worsening cancers, red lines stable cancers

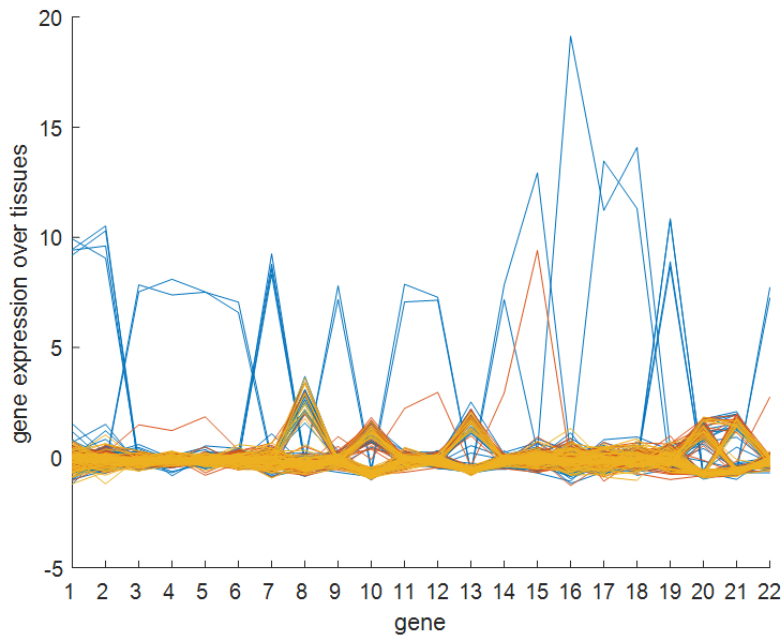


Figure 2.2: Parallel coordinate plot.

and yellow lines regressive cancers, respectively. The intersections of the polylines with the vertical axes show there are some genes highly discriminating the three colors, which means that some mice have particular expression levels for some genes. Also, the colored grouping of polylines show coherency, which means there is discrimination for tissues. Hence, these genes can be used as markers of CRC subtypes. This tool has been used as a visualization tool for the validation of the previous gene selection, based on the cluster quality evaluation. This technique has confirmed the selection of 22 genes (see Fig. 2.2). This allows the study of the data manifold representing the tissues in a lower dimensional space, just alleviating the problem of the curse of dimensionality.

2.3 Sub-Manifold Analysis

The reduced submatrix is composed of 203 rows, i.e. the tissue values (samples) and 22 columns, i.e. the selected genes (features). In order to check the intrinsic dimensionality and the linearity of the data manifold, the Principal Component Analysis (PCA, [25]) has been performed. The plot of the variance explained by the principal components shows an intrinsic dimensionality of about 5 (see Fig. 2.3), corresponding to 90% of variance explained. This result suggests that tissues belonging to the 22-dimensional space lay on a 5-dimensional hyperplane. The remaining 10% can be justified by either noise or small departure from linearity, that is nonlinearity only on a large scale, but not locally. In order to confirm this hypothesis, the Curvilinear Component Analysis (CCA, [26]) has been used. CCA is a neural technique for dimensionality reduction which projects points by preserving as many distances as possible in the input space. However, CCA is here used not for the

exploitation of the projection, but for the information that can be derived from its dy - dx diagram (see Fig. 2.4). This plot represents distances between pairs of points in the input space (the dx value) and in the reduced (latent) space (the dy value) as a pair (dy, dx) . If a distance is preserved in the projection, the corresponding pair is on the bisector (indicated in the figure). If the pair is under the bisector, it represents the projection as an unfolding of the input data manifold.

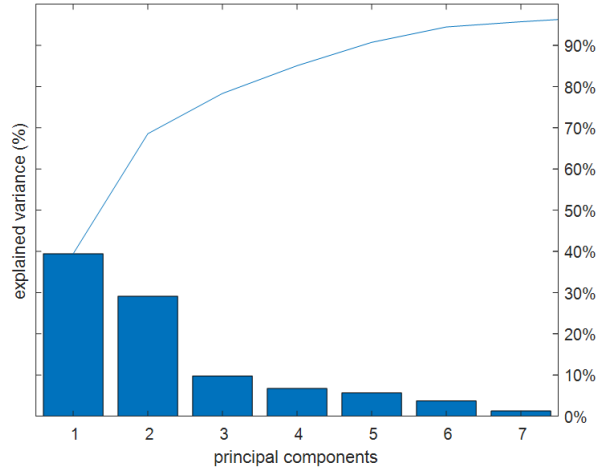


Figure 2.3: Variance explained by principal components.

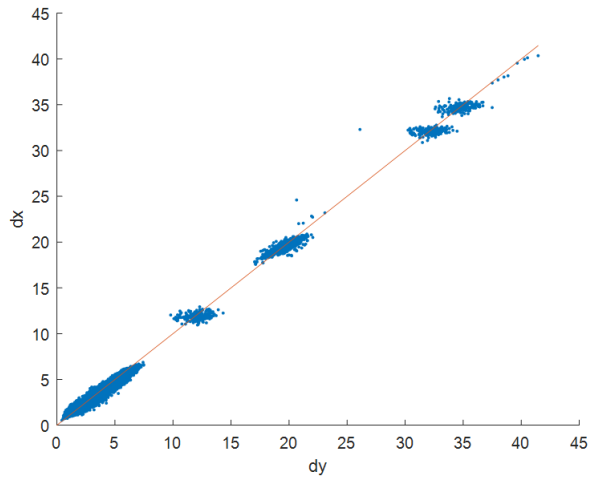


Figure 2.4: CCA dy - dx diagram.

If the manifold is linear, all points tend to lie on or around (because of noise) of the bisector. Clusters of points on the bisector, but far from the origin, represent large inter-cluster distances, and, hence, reveal the presence of clusters. Fig. 2.4 shows this plot for a five-dimensional latent space (this choice is suggested by the previous PCA). The smaller

grouping near the origin represents the intra-cluster distances and suggests the idea of one or several hyperplanes as data manifold. This is confirmed by the other groupings at larger distances. They represent the inter-distances and suggest the presence of at least four distinct clusters. The fact that, above all, the groupings of farthest distances have the biggest departure from the bisector, yields the idea of a curvature at large scale. Resuming, the data manifold in the space of the selected genes is composed of several well distinct nearly-flat submanifolds. This confirms the validity of our approach, in the sense that the extracted features discriminate well with regard to tissues.

Biplots [27] are now used in order to understand the reciprocal behavior (in statistical terms) between all tissues and the selected genes. They are a generalization of scatter plots. A biplot allows information on both samples and variables of a data matrix to be displayed graphically. Samples are displayed as points while variables are displayed as vectors. Fig. 2.5 shows the biplot over the first three principal components. With regard to tissues (red points), there are only few data along the first and the second principal component. Instead, the third component has a good discriminant capacity over tissues. With regard to genes (blue vectors), most of them are strongly related to the first or the second principal component. However, five genes (in the figure, represented as 8, 10, 13, 20 and 21) stay along the third axis, thus explaining the variance of tissues along this direction. The biplot shows that a combination of these genes has a bimodal behavior along the third principal component. Fig. 2.6 shows a parallel coordinate plot of these five genes. This graph points out relationships between these genes and their bimodal behavior. Blue lines represent worsening cancers, red lines stable cancers and yellow lines regressive cancers, respectively. The most interesting gene is shown in the first vertical axis because it marks a coherent bundle of segments which represents a set of worsening cancers.

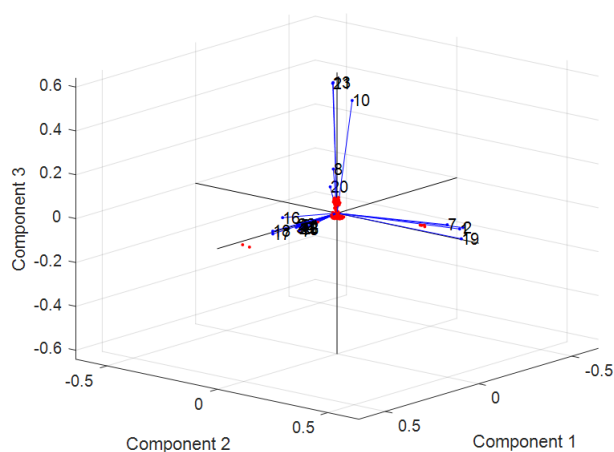


Figure 2.5: Biplot over the first three principal components.

2.4 Biological Feedback

The previous analysis shows that the selected genes, whose biological names are CRMP1, CSAG1, EIF1AY, PRAC1 and RPS4Y1 have a high discriminant power. From a biological point of view, some of these genes are strongly related with cancer. In particular:

- CRMP1 is supposed to be related to inhibition of metastasis [28];
- CSAG1 is supposed to be related to squamous cell carcinoma [28];
- PRAC1 is supposed to be related to human prostate and colorectal cancer [28].

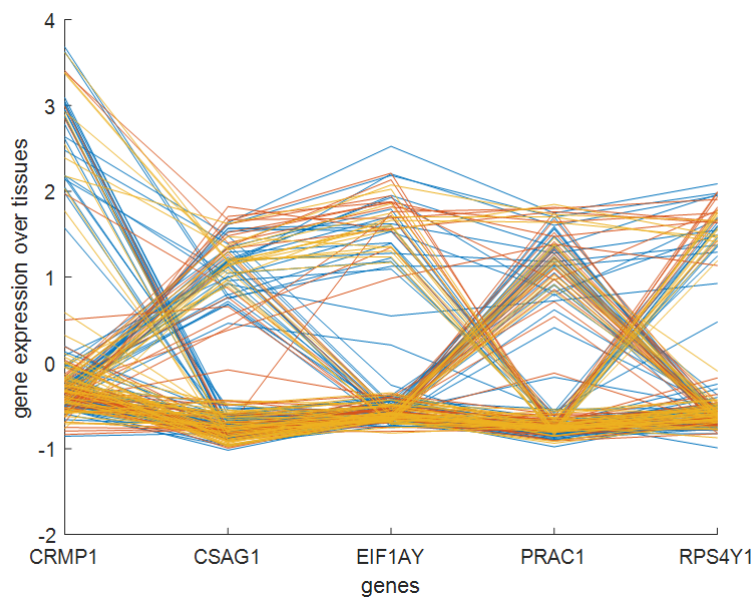


Figure 2.6: Parallel plot over the resulting gene selection.

Chapter 3

Supervised Manifold Analysis

As anticipated at the beginning of this part, in this chapter a supervised manifold analysis is performed. In addition, a simple model is built through a classic Multi Layer Perceptron (MLP): it correctly classifies unlabeled data according to the patient response to drug. Most of content exposed in this chapter was presented at the WIRN2017 conference [29].

3.1 Supervised Feature selection

As in the previous case, a manual feature selection based only on the parallel plot analysis is unfeasible due to the huge amount of features (15396) and to the difficulty to ascertain the colored bundle groupings. In order to circumvent this problem, a new algorithm for supervised feature selection, based on the Davies-Bouldin [30] clustering index, has been devised. Each gene has been evaluated in its capability of discriminating tissues with different response to drugs which is indeed the skill in grouping well-separated clusters, with high level of cohesion. Specifically, tissues have been divided, for each gene, into unidimensional clusters, exploiting only the associated label. Then, the resulting cluster quality is estimated. The Davies-Bouldin index is suitable for the case of study, because it considers both inter-cluster distances and intra-cluster distances, estimated according to the Euclidean distance. A quality threshold has been empirically selected in such a way that only the best genes are chosen. Hence, 19 genes have been retained as shown in fig 3.1. This way of selecting feature is unconventional because it does not calculate directly the correlation between genes and the response to drugs, but it still selects the genes that will be more useful and reliable for a classification model based on those genes only.

3.2 Sub-Manifold Analysis

A subspace composed of the genes selected by the feature selection has been extracted from the original dataset, creating a matrix composed by 203 tissues and only 19 genes. In order to have a better insight of the data extracted, a PCA analysis has been performed for this reduced database as in the previous section. The inspection of the PCA explained variance suggests the intrinsic dimensionality of the manifold to be 11. This could imply data stay on a 11-dimensional hyperplane. A further insight of the manifold has been

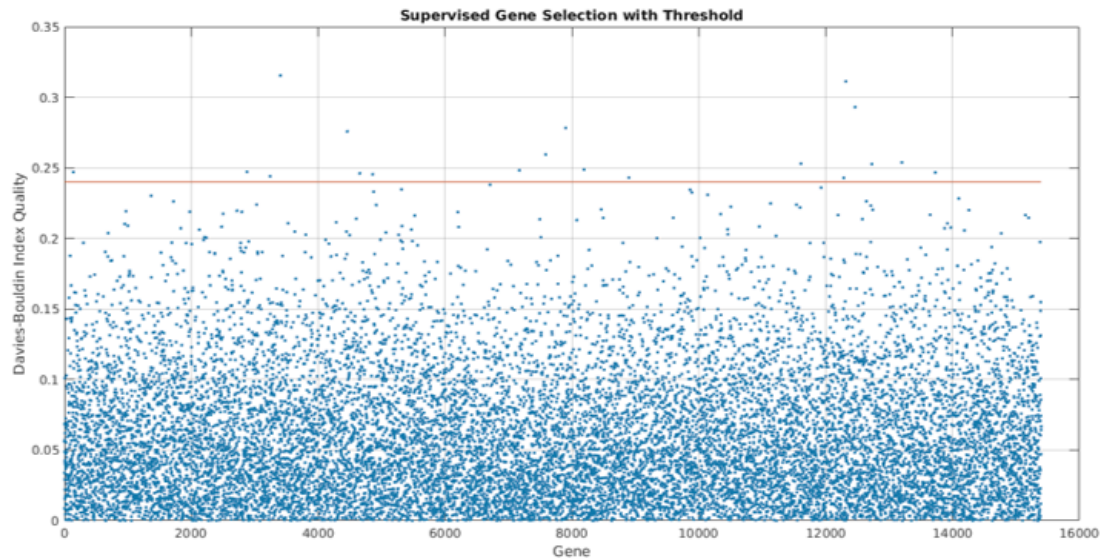


Figure 3.1: Cluster quality evaluation. The red line is the threshold.

obtained through a CCA. Here CCA performs a projection from a 19-dimensional to a 11-dimensional space. As shown by the dy-dx diagram (see fig. 3.2), most pairs stay around the bisector, just confirming the hypothesis of 19-dimensional hyperplane.

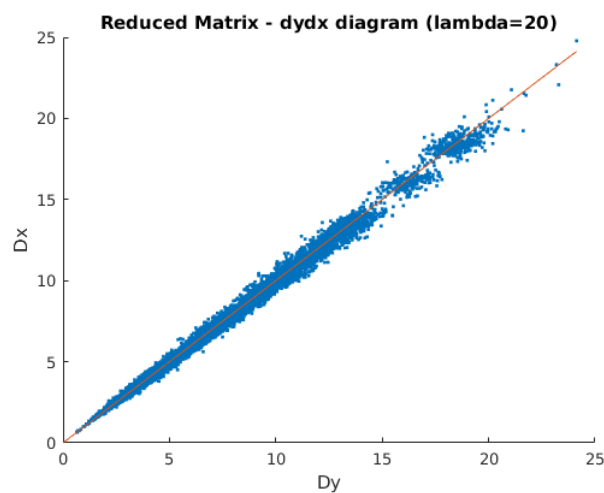


Figure 3.2: dy-dx diagram.

3.3 Lasso regression

Lasso regression [31] generally targets to improve the prediction accuracy and interpretability of the regression model by selecting only a subset of the available variables to use in the final model rather than using all of them. Lasso is able to achieve it by forcing the sum of the absolute value of the regression coefficients to be less than a fixed value, which forces certain coefficients to be set to zero, effectively choosing a simpler model that does not include those coefficients. It requires a regularization parameter λ which controls the trade-off between regression and constraint on the coefficients. Greater values of λ correspond to a lower number of variables inserted in the model. In the case of study, Lasso regression has been useful to confirm the intrinsic dimensionality of the reduced matrix previously established and, more importantly, to identify the 11 genes. This step has been possible because it is based on the linearity assumption of the reduced database, deduced from the previous manifold analysis. The response variable used previously cannot be exploited here, because it is discrete. Instead, the other variable associated to the tissues can be used (the tumor difference in size after 3 weeks). Fig. 3.3 shows gene coefficients that decrease until zero as λ increases. The value of λ suggested by Lasso (the dotted line in figure) is given by 0.05, because it is the one that guarantees the least Mean Square Error (MSE) as shown by Fig 3.4. It is important to notice that the number of nonzero LASSO coefficients still present at $\lambda = 0.05$, is 11. This result confirms the previous assumption about the fact that the reduced manifold is a 11-dimensional hyperplane.

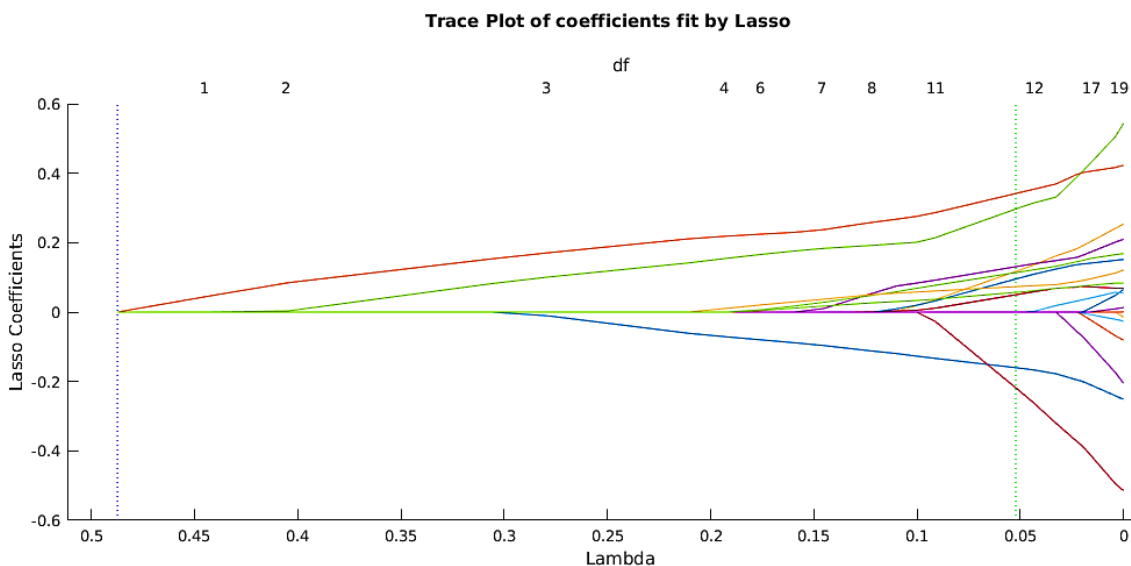


Figure 3.3: Lasso coefficients as a function of λ .

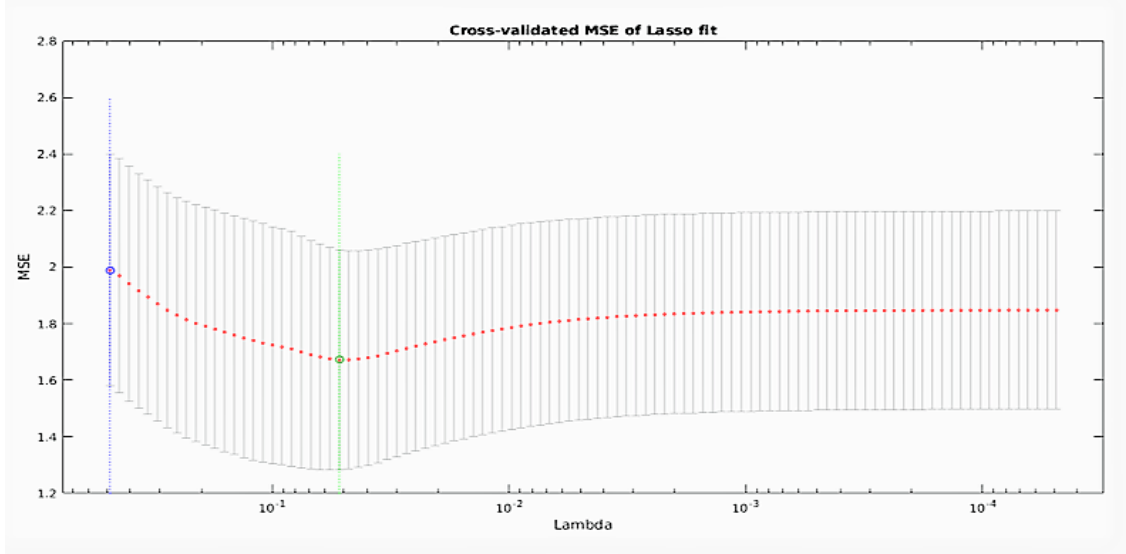


Figure 3.4: MSE of Lasso regression as a function of λ .

3.4 Biological Feedback

The 11 genes selected are the most meaningful for predicting the increment or decrement of the cancer volume size after three weeks. Their names are the following: LOC645233, FSCN1, ACSS2, SMAD9, MED1, TMEM118, LOC728505, SF3B4, LOC651316, SERPHIN1, GPR126. Some of these genes are already well known in the medical literature as correlated with cancer:

- FSCN1 is supposed to be related to cell motility [28];
- ACSS2 is supposed to be related to cancer cell survival [28];
- MED1 is supposed to be related to gene transcription [28]).

Correlation with cancer of the remaining genes has not been proved yet. Their presence in this work, however, suggests that they should be involved, at least in this particular context of the CRC response to drugs. In fact, it is important to observe that a gene expression may not be relevant for the presence of a tumor, but it may remain important for the survival of tumoral cells. Specifically, average expressions in patients of genes LOC645233 and ACSS2 seems to be in contrast with literature. Nevertheless, results regarding those genes have been published, since they are not an artifact of the analysis but they concern raw data. This analysis proposes a novel approach whose results may be considered as suggestions for further biological research.

3.5 Classification

At last, the expression of the selected 11 genes is used in order to train a classification model. Several models have been tested: the one that shows the best accuracy on the test set is

the Support Vector Machine (SVM, [32]) with an accuracy of 78%. The model is tested through the hold-out validation with 25% of data randomly put in the test set. A further attempt to improve the accuracy of the model has been done through the use of a Multilayer Perceptron (MLP, [32]). It is composed of 11 inputs, 20 hyperbolic tangent hidden neurons and 1 output whose activation function is the logistic sigmoid. It is equipped with the cross-entropy error function and the backpropagation learning algorithm is used in order to evaluate the error derivatives for the BFGS training. For the purpose of this analysis two target classes have been selected: the first (1) corresponding to tissues with a regressive or stable response and the second (0) for tissues where the disease has worsened. The robustness of the model is corroborated by both validation and test sets. The accuracy is shown in the Test Confusion Matrix (fig. 3.5) and is given by 80%. This result is not only important in itself, but can be considered as a figure of merit for the selected 11 genes: how accurate 11 genes over 15396 are in modeling the progression of the tumor.

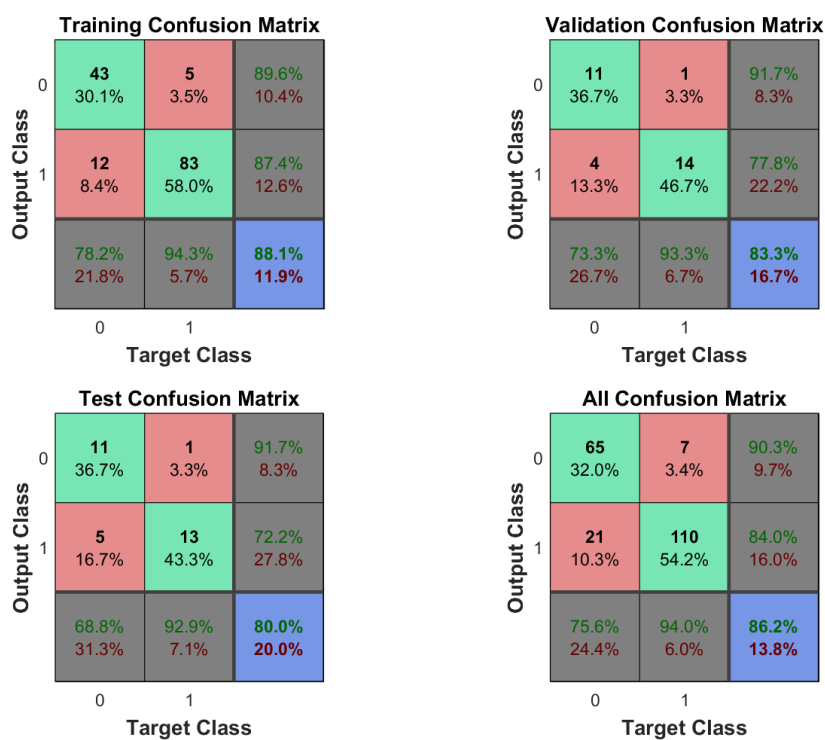


Figure 3.5: Confusion Matrices for the MLP classifier.

Part III

Advanced Data Analysis Through Neural Techniques

Chapter 1

Unsupervised Neural Techniques

1.1 Introduction

Following the unsupervised path previously introduced in chapter 2 of part II, further and deeper analysis have been conducted using several neural techniques. A further analysis in this direction has been required as class discovery is a key aspect in analyzing gene-expression data as reported in [33]

As already explained the high dimensional problem prevents a direct application of clustering algorithms in the feature space. Nonetheless, this is a common problem in gene expression analysis and in general while clustering high-dimensional data. As suggested in [34], this obstacle may be overcome through the use of Biclustering. This technique has been the key of the work: several advanced neural networks techniques performing clustering have been indeed tested in a biclustering framework. Eventually, a novel neural network, Growing-Hierarchical EXIN (GH-EXIN), devised ad hoc for the problem at hand has been successfully employed. GH-EXIN is based on a novel unsupervised neural approach called G-EXIN (Growing-EXIN [35]).

In section 1.2, the biclustering technique is explained; in the following one, section 1.3, two state-of-the-art techniques, used also in the context of biclustering, are reported; in chapter 2.1 the novel neural network, GH-EXIN, is introduced; a comparison between this technique and the previous ones is presented in section 2.2; results of the neural framework applied on the current dataset is reported in section 2.3; at last, an application of the neural framework on an external problem of face recognition is reported in section 2.4.

1.2 Biclustering

Biclustering was introduced in the 60's, but has been properly defined only by Cheng and Church in 2000 and is also known as two-way clustering or manifold (subspace) clustering, [36]. As previously introduced, this technique has been chosen as it allows to perform grouping on a reduced dataset, overcoming the high dimensional problem. Nonetheless, this is not its only quality.

Basically, clustering can be applied to either the rows or the columns of the data matrix, separately. Biclustering, instead, performs clustering in both dimensions simultaneously. In this work it is achieved by alternating both row and column clustering on projected data derived from the previous steps. Compared to clustering, biclustering has several advantages since it groups items based on a subset of the features so that it does not only perform grouping but also discovers the context (subspace) in which the groups are found. Furthermore, the projection of the biclusters into the features or the samples space allows to analyze the results as grouping of samples or features, respectively. In this work, biclusters are projected into the sample space in order to discover in which conditions - i.e. for which individuals - different genes coregulate.

In fact, common requirements in analyzing gene data are the grouping of genes according to their expression under multiple conditions (tissues) and the grouping of conditions based on the expression of a number of genes. These can be achieved by using clustering techniques. However, many activation patterns are common to a group of genes only under specific experimental conditions. Indeed, subsets of genes are coregulated and coexpressed only under certain experimental conditions, but behave almost independently under other conditions. Finding these local expression patterns is the goal of biclustering [34] and is the key for class discovery which in this specific case means uncover unknown genetic pathways.

Biclustering searches for biclusters with constant values, with constant values on rows or columns and with coherent values, respectively. It can be proved that the rank of the corresponding submatrices is less than or equal to three in the noiseless case. Hence, the numerical rank can be used as a figure of merit of the quality of the bicluster. The H_{cc} index, introduced by Cheng and Church [36], is used to control the quality of the bicluster as it also takes into account the noise in data. It is expressed as:

$$H_{cc} = \frac{\sum_i^{N_r} \sum_j^{N_c} r_{ij}^2}{N_r N_c} \quad (1.1)$$

where N_c represents the total number of columns of the matrix, N_r represents the total number of rows and $r_{i,j}$ is the residue, which is calculated as:

$$r_{ij} = a_{ij} - \frac{\sum_k^C a_{ik}}{C} - \frac{\sum_h^R a_{hj}}{R} + \frac{\sum_i^R \frac{\sum_j^C a_{ik}}{C}}{R} \quad (1.2)$$

The terms $a_{i,j}$ are the elements of the matrix (rows and columns represent faces and descriptors). C and R are the number of columns and of rows of the bicluster at hand, respectively. The second term is the average value of the i th row, the third term is the average value of the j th column, while the last one is the average value of the whole bicluster. This index decreases as the values in the bicluster tend to be constant, differing for a constant on the rows or a constant on the columns. It goes to zero for the trivial 1×1 bicluster. This fact implies additional controls on the biclusters in order to avoid this drawback.

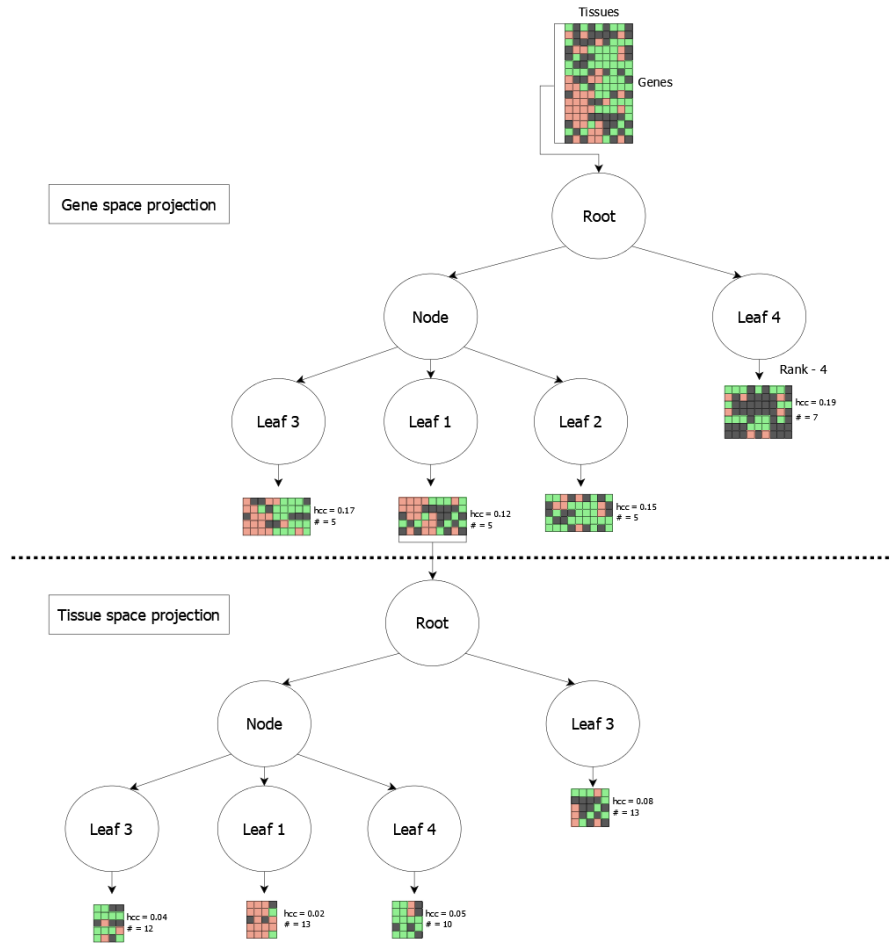


Figure 1.1: Neural Biclustering Framework

1.2.1 Neural Framework

In order to detect biclusters in the gene expression matrix, gene and tissue clustering are alternated. The preferable type of clustering in this case is a hierarchical divisive clustering: it easily allows to select the desired cluster resolution level at each iteration, by simply stopping the network when a certain height in the hierarchy has been reached. This is fundamental in biclustering because it allows to alternate the clustering in the two spaces several times until the best biclusters are found. As shown in fig 1.1, indeed, a first a hierarchical clustering is achieved on genes in the tissue space, because it is the lowest dimensional space (in order to avoid the curse of dimensionality, which cannot be avoided if working on the gene space). Then another hierarchical clustering is performed on the tissues in the space of the genes associated to the best leaves produced in the first step (reduced gene space, as an orthogonal projection from the original space). The best leaves of the second step reduce the tissue space for the genes selected after the first step. This corresponds to another orthogonal projection. Resuming, each cluster (leaf) decreases the

dimensionality of the problem for the subsequent clustering, whose leaves yield a further dimensionality reduction. Considering that clustering implies a feature selection, this can be viewed as an orthogonal projection of the vectors. Indeed, selecting only some components results in setting the other components to zero. Considering that the basis is canonical, it corresponds to an orthogonal projection into the reduced subspace (cluster). These two steps, which pseudocode is illustrated in Alg. 1, are repeated (alternated projections) until bicluster candidates are identified.

The growth of the tree is controlled by the index H_{cc} , by simply modifying the stop criteria of the chosen algorithm. However, as seen before, the index tends to zero as the cardinality of the leaves decreases. In order to avoid trivial biclusters, before each clustering step, a check on the minimum number of data in the leaf (Cmin) is performed both in the tissue space and in the gene space (additional check). The choice of the quality index depends on the goal of the analysis. Other indices can be added (e.g. an index about the shape of the cluster) or replace H_{cc} . However, this choice remains basically heuristic and is an open problem.

Algorithm 1 Biclustering Pseudocode

```

1: biclustering:
2: Clustering on genes
3: for all leaves do
4:   if leaf.cardinality  $\leq$  Cmin1 then
5:     skip leaf
6:   else
7:     Clustering on the tissues of the leaf (projection)
8:     for all leaves do
9:       if projectedLeaf.cardinality  $\leq$  Cmin2 then
10:        skip leaf
11:       else
12:         save projected leaf
13:         goto biclustering
14:       end if
15:     end for
16:   end if
17: end for
18: return

```

Resuming, the parameters needed by the biclustering algorithm are the minimum cardinalities of leaves for both dimensions i.e. both in the number of genes grouped and in the number of tissues. These parameters, together with the bicluster quality index (H_{cc}), control the search and require a deep analysis, which, however is out of the scope of the work.

1.3 Clustering State-of-the-art

1.3.1 GHNG

The Growing Hierarchical Neural Gas (GHNG, [37]) is a hierarchical self-organizing neural model, which learns a tree of Growing Neural Gas (GNGs) where each subgraph (i.e. each GNG) is the child of a processing unit (a.k.a. neuron) of the upper level. At each hierarchical level a GNG network is created by using the Voronoi set of the father neuron. Initially, a GNG network is composed of two neurons joined by a connection. At first, each one is initialized to a random sample from the Voronoi set of the father unit. When a new sample x_t is presented to the GNG network, the algorithm finds the nearest neuron w_q and the second nearest one w_s to the sample, and it increments the age of all the edges departing from w_q . The error variable associated to the winner e_q is incremented with the squared Euclidean distance between the winner itself w_q and the new sample x_t :

$$e_q(t+1) = e_q(t) + \|w_q(t) - x_t\|^2 \quad (1.3)$$

Then, all the direct topological neighbors of w_q (i.e. neurons joined to w_q with an edge) are updated with step size ϵ_n , and w_q itself is updated with step size ϵ_b :

$$w_i(t+1) = (1 - \epsilon)w_i(t) + \epsilon x_t \quad (1.4)$$

If the winner w_q and the second winner w_s are connected with an edge, the age of this edge is set to zero; otherwise, if they are not joined, an edge is created linking the two neurons. Finally, all the edges older than age a_{max} are removed, as well as all the isolated neurons (if any). At this point, if the current time step t is a multiple of a user-dependent parameter λ , then a backup copy of the GNG network is saved. Then the algorithm selects the neuron having the largest error w_r and, among its neighbors, the one having the largest error w_z . A new neuron w_k is then created halfway between w_r and w_z , decreasing the quantization error of the GNG graph. Otherwise, if the time step t satisfies the relationship:

$$\text{mod}(t, 2\lambda) = \left\lfloor \frac{3}{2}\lambda \right\rfloor \quad (1.5)$$

a check is done in order to evaluate if the growth process has resulted in an improvement of the quantization error. Given the mean quantization of the last backup copy MSE_{old} and of the current GNG network MSE_{new} , the backup graph is restored if:

$$\frac{MQE_{old} - MQE_{new}}{MQE_{old}} < \tau \quad (1.6)$$

where $\tau \in [0,1]$ is a user-dependent parameter. If the above relationship is satisfied, then the graph enters the convergence phase. Thus, for high values of τ the quantization error improvement must be significant in order to continue the growth phase. Finally, the error variables are decreased by multiplying them by a user-dependent constant d . If the maximum number of time steps is reached the algorithm stops. Otherwise another sample is presented to the GNG graph. When the learning process ends, then the Voronoi set of each neuron is used to train another GNG network recursively (see Fig. 1.2). The vertical growth in a branch of the hierarchy stops when the deepest GNG enters the convergence

phase having only two neurons. This leaf node is pruned because it is too small to represent any relevant distribution.

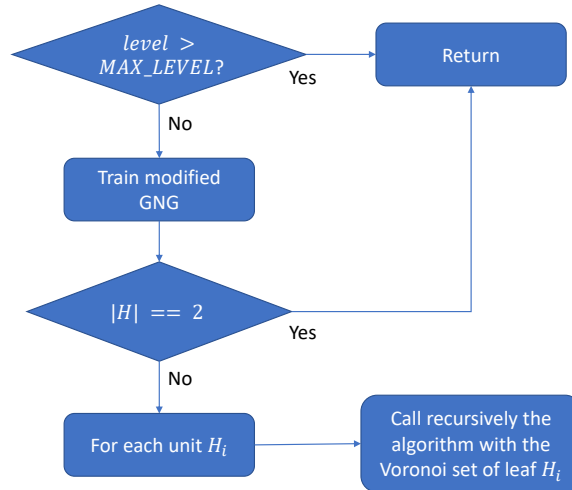


Figure 1.2: GHNG flowchart.

1.3.2 DGSOT

The Dinamically Growing Self-Organizing Tree (DGSOT) [38] is a self-organizing neural network. Similarly to all clustering algorithms presented in this work, it builds a hierarchical divisive tree.

It is an enhanced version of the Self Organizing Tree Algorithm (SOTA) [39]: basically, DGSOT adds to each vertical growth performed in SOTA also a horizontal growth which allows to better determine cluster partitioning at each level. SOTA, in fact, builds a binary tree: each cluster, if partitioned, is only split into two parts, but this is a very limiting approximation.

The type of tree built by DGSOT is slightly different from the one built by GHNG: in the latter, each node in the tree seems to represent the training of an associated neural network (GNG), which is composed of many neurons. In this case, instead, each node effectively represents a neuron. In the following, hence, the terms "neuron" and "node" will be used alternatively, as, at least in this context, they are interchangeable.

The tree initialization consists in the assignments of all data to the root node and in its positioning as the centroid of the dataset.

Then, while there exists at least a leaf whose heterogeneity is higher than a threshold T_R , a vertical growth is performed on this leaf. The heterogeneity of a node is defined as the average distance of the data to the neuron reference vector. Two descendent nodes of the current node are created and their reference vectors are initialized with father node's reference vector.

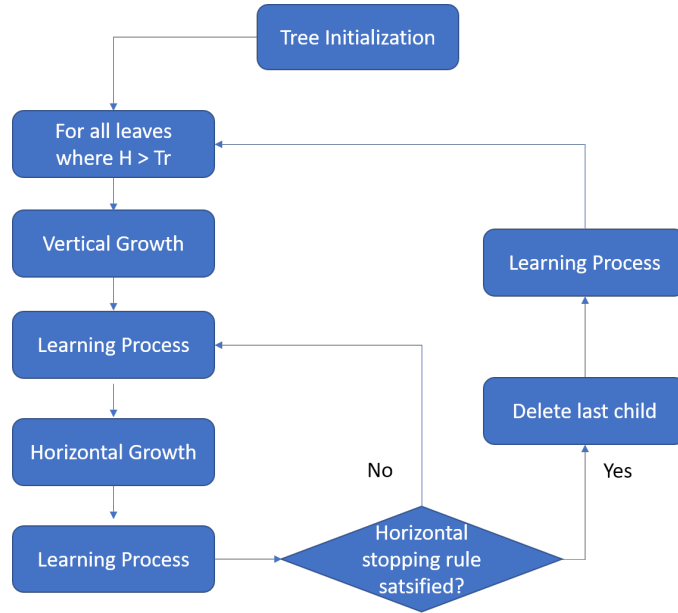


Figure 1.3: DGSOT flowchart

A learning process, in general, is the presentation of all data (epoch) for a few times to neurons for them to learn. The learning, basically, consists in the assignment of data to winner neurons and in the following reference vector adaptation. In this particular case, data previously assigned to the father node, are presented through the K -level up Distribution (KLD) mechanism. It selects potential winner neurons among all the leaves belonging to the sub-tree starting from the K -ancestor node, the ancestor node K -level above father node, where K is a parameter of the network. This mechanism allows improperly clustered data in early hierarchical levels to be re-evaluated during later at lower layers. The winner, as commonly done, is chosen as the nearest node. Weight adaptation, instead, occurs not only for the winner but also for its neighbours, in a *winner-take-most* strategy. Neighborhood is defined as the union siblings and father node. For both winner node and neighbor nodes, reference vectors are updated according to the following function:

$$\Delta w_i = \phi(t) \cdot (x - w_i) \quad (1.7)$$

where w_i is the reference vector of the winner node, x is the datum and $\phi(t)$ is defined as:

$$\phi(t) = \alpha \cdot \eta(t) \quad (1.8)$$

where α is a user-dependent parameter which differs according to node taken into consideration, and, in general, is close to 1 for the winner, smaller for the siblings and close to 0 for the father node; $\eta(t)$, at last, is a function of the time t which represent the number of times a neuron is selected as winner node: a good choice may be $\eta(t) = 1/t$

The learning process is repeated until the relative heterogeneity of all child nodes compared to the previous epoch is less than a user-dependent parameter T_E .

After the learning process a horizontal growth is performed. This kind of growth consists in the addition of a node to the current group of child nodes. It is always followed by a

further learning process in which the current group of nodes finds the correct quantization associated to the current number of neurons. These two steps are repeated until the current number of neurons is found. After the learning process, in fact, a cluster validation test is performed in order to check the quantization error of the current neural network. In order to do that, DGSOT calculates the Cluster Separation (CS) as:

$$CS = \frac{E_{min}}{E_{max}} \quad (1.9)$$

Where E_{max} is the maximum distance between two of the current neurons, while E_{min} is the minimum distance. In case CS is above a given threshold T_c , user-dependent parameter, the process is repeated - i.e. a horizontal growth is performed again followed by a learning process. Otherwise, the last child is deleted and previous configuration is restored through another learning process.

Chapter 2

GH-EXIN

2.1 A Novel Neural Technique: GH-EXIN

2.1.1 Self-Organizing and Data Driven Approaches

The proposed approach builds a divisive hierarchical tree in an incremental and self-organized way, as shown in fig. 2.1. Indeed, it is a top-down technique which divides data at deeper and deeper levels. It is data driven (self-organization), in the sense that the final tree is automatically estimated. Also, it does not require a predefined number of units and levels (incremental with pruning phase). The resulting tree is neither binary nor balanced, because of its dependence on data. Both the GH-EXIN, GHNG and DGSOT algorithms follow these criteria.

2.1.2 Basic Neural Units

The basic neural unit is intended as the neural network chosen for the clustering of the input data. They are composed of units called neurons, which are represented by weight (a.k.a. reference) vectors. As an abuse of language, the terms neuron and weight vector are used with the same meaning. They do not have any fixed topology (the induced topology is generated in the linking phase). All these methods use a basic neural network for the processing of each leaf. GH-EXIN is based on the stationary variant of G-EXIN, say sG-EXIN. GHNG uses a variant of GNG, while DGSOT, deriving from SOTA, exploits as basic neural network an enhanced version of SOM.

Neuron Creation

Online Learning The basic neural units are incremental, i.e. they have a variable number of neurons (driven by data). It is, in general, achieved by the mechanism of neuron creation and pruning. With regard to the former, a neuron is added (a new weight vector is created) if the existing neurons are not able to sufficiently describe the incoming data. Several choices can be found in the literature for this novelty detection. Both GHNG and DGSOT handle neuron creation by considering the batches of the whole set of samples, at the end of an epoch (DGSOT) or each λ (user-dependent parameter) iterations (GHNG).

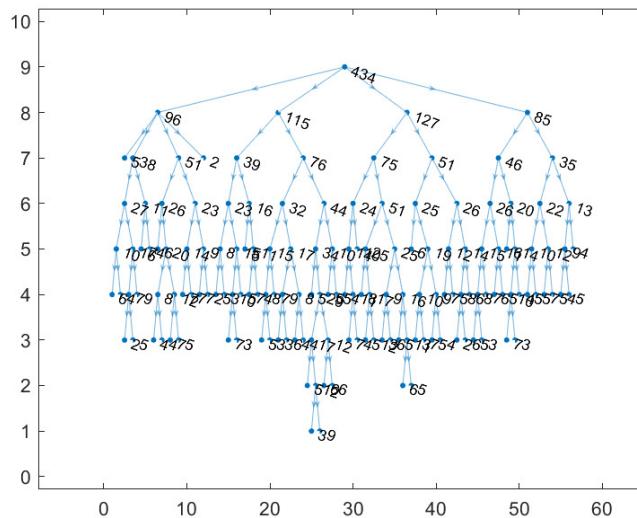
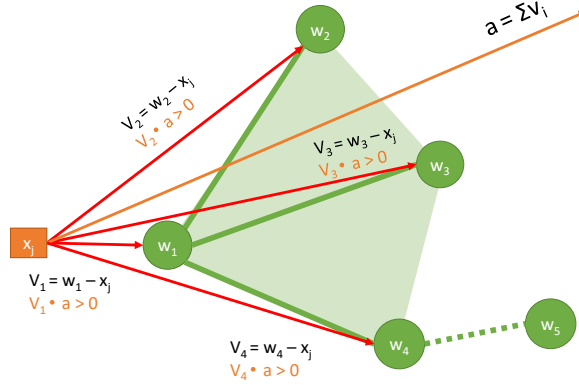


Figure 2.1: GH-EXIN tree neural structure: the number associated to each node in the tree represents its Voronoi cardinality

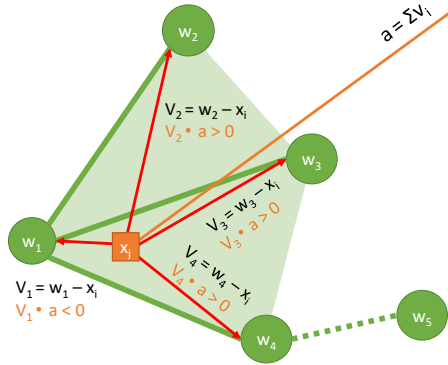
On the contrary, GH-EXIN, by deriving from a clustering algorithm designed for non-stationary data, decides datum by datum whether to create a neuron or not (novelty test). Additionally, in order to take this decision, GH-EXIN automatically takes into account also previously presented data by considering network topology. This is a more efficient approach, as GH-EXIN, by taking decisions during the epoch, may converge in fewer epochs than GHNG and DGSOT.

Semi-Isotropic Quantization The decision regarding neuron creation is generally based on the quantization error of the network. Several approaches can be found in literature for the definition of the quantization. They are mainly isotropic, in the sense that each neuron represents locally the input space by means of a hypersphere centered at its weight vector. The corresponding radius is named neuron threshold. An exhaustive description can be found in [40] Both GHNG and DGSOT keep adding neurons as far as the quantization error (a.k.a. *cluster separation* in [38]) does not fall below a user-dependent threshold 1.9, 1.3. However, GHNG computes the quantization error from data, by considering the average distance between the reference vectors and their Voronoi sets, while DGSOT from network topology, by considering the rate between the minimum and the maximum distance among neurons. Similarly to DGSOT, GH-EXIN takes into account the network topology. However, differently from DGSOT, it considers both the extent (isotropic criterion) and the shape (anisotropic criterion) of the neuron neighborhood (determined by the linking phase). The isotropic criterion is based on the average of the Euclidean distances between the winner neuron (w_γ) and its N neighbors (w_i):

$$T_\gamma = \frac{1}{N} \sum_{i=1}^N \|w_\gamma - w_i\|^2 \tag{2.1}$$



(a) If all the dot products between v_i (vectors from the new datum x_j to the neuron w_i) and a (vector addition of v_i) have equal sign, then the new datum x_j is outside the convex hull of the winner (w_1).



(b) If the dot products between v_i (vectors from the new datum x_j to the neuron w_i) and a (vector addition of v_i) have not equal sign, then the new datum x_j is inside the convex hull of the winner (w_1).

Figure 2.2: A new point $x_j \in \mathbb{R}^2$ is presented to the sG-EXIN neural network composed of four connected neurons.

In case the new datum is farther than T_r from the winner, a new neuron is created on the datum. However, this isotropic approach does not take into account the true shape of the neighborhood. For instance, given a data and the corresponding winner, consider the hyperplane in the input space passing through the winner weight vector and normal to the vector joining the data and the neuron. Assume that all winner neighbors are placed in the same half-space (which obviously does not contain the data). If an isotropic threshold is used, also a portion of the input space in the other half-plane is considered

for explaining the input. This inconsistency becomes worse if there are very far neighbors. At this aim, the novelty detection also requires an additional anisotropic check, explained in the following subsection. This approach represents one of the major novelties, as both GHNG and DGSOT make use of isotropic thresholds.

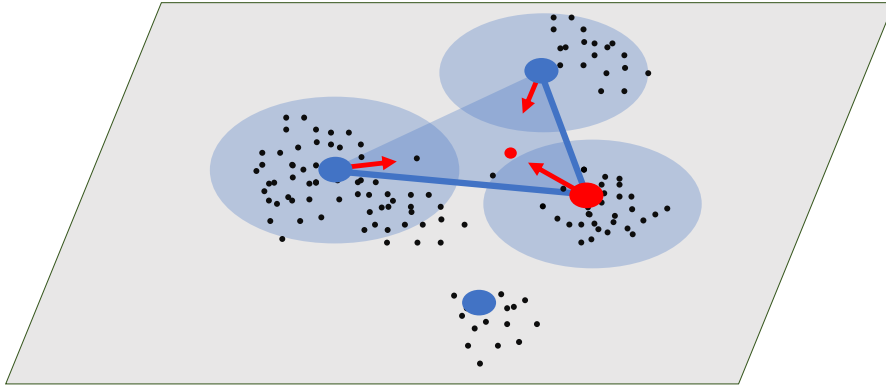


Figure 2.3: The semi-isotropic quantization exploits both thresholds (isotropic criterion) and convex-hull (anisotropic criterion) providing a hybrid quantization. When a new datum (small red dot) is presented, the winner (big red dot) and its neighbors (blue dots) move towards it. The blue unconnected neuron (lonely neuron) below does not move as it is not part of the winner neighborhood.

Convex-Hull sG-EXIN creates a new neuron only if winner neuron γ is not able to correctly represent the new datum x_j . If this is the case, x_j is considered novel w.r.t. to neuron γ . This novelty test is satisfied when: the distance d between γ and x_j is greater than the local threshold T_γ and, contemporary, x_j is outside the region which represents the neighborhood of γ . This portion of space, say NG_γ (neighborhood of γ), is modelled by the convex hull (bounded convex polytope) of the weight vector of neuron γ and the weights of its topological neighbors (i.e. those connected through edges). This idea is shown in Fig. 2.2 in two-dimensional space. Here, the green segments represent the links from neuron w_1 . The w_1 convex hull is represented by a grey shaded area; only the green neurons belong to it. As an example, the green dotted segment connects a w_1 topological neuron (w_4) and one of its neighbors (w_5); hence, this neuron is not a w_1 topological neuron and, so, it does not belong to NG_1 , but only to NG_4 .

However, if data lie in a D -dimensional space, the convex-hull technique can be exploited only in case γ has $D + 1$ neighbors. In fact, with less than $D + 1$ neighbors, the region represented by the convex-hull has measure zero, hence no data can lie within it. In such a case, the novelty test is based only on the isotropic criterion. In order to check whether a datum lie within the convex-hull a simple test is performed. First, the sum of the difference

vectors δ_i between x_j and NG_γ neurons is computed:

$$\Psi = \sum \delta_i \quad (2.2)$$

If all the scalar products between δ_i and Ψ have the same sign (null products are ignored), then x_j is outside the polytope (see Fig. 2.2). Otherwise, x_j is inside the polytope.

Lonely Neuron A neuron with no edges is named *lonely neuron* (Fig. 2.3). Since DGSOT does not make use of edges, all its neurons are lonely. On the contrary, both GHNG and GH-EXIN exploit the concept of lonely neurons to determine nodes to be pruned. In both algorithms, a neuron may become lonely in case all its edges are pruned. However, in GHNG neurons already have connections when created. In GH-EXIN, instead, new neurons born lonely. Connections may or may not be generated only during next iterations.

Soft-Competitive Learning

The weight computation (training) is based on the Soft Competitive Learning (SCL) [41] (Fig. 2.3) paradigm, which requires a winner-take-most strategy. The closest neuron to the new datum is named (first) winner. The set of potential winners differs according to the clustering algorithm. In both GHNG and GH-EXIN, all neurons belonging to the same basic neural unit are competitive in the learning phase, which means that, for each presentation of data from a training set, all weight vectors are ranked according to their distance (in general Euclidean). In DGSOT all neurons belonging to the sub-tree below the K -ancestor of the current node are considered. At each iteration, both the winner and its neighbors change their weights but in different ways. The winner w_γ and its direct topological neighbors w_i are moved towards x_j by fractions $\alpha_\gamma(t)$ and $\alpha_i(t)$ (learning rates), respectively, of the vector connecting the weight vectors to the datum:

$$\Delta_w = \alpha(t) \cdot (w - x_j) \rightarrow w = w + \Delta_w \quad (2.3)$$

Where $\alpha(t) = \alpha_0 \cdot 1/t$, and α_0 is a user dependent parameter, higher for the winner and smaller for the neighbours, and t is number of times a neuron wins (conscience) The differences among the three algorithms consists on the neighborhood determination. DGSOT considers all neurons belonging to the same basic neural unit as neighbors, plus the father neuron. On the contrary, the neighborhood of both GHNG and GH-EXIN is composed of all neurons connected through an edge.

Edge Creation and Network Topology

An edge is a connection placed between two neurons. Edges are exploited in order the determine the topology (neighbors) of a network, this is achieved by the Competitive Hebbian Learning (CHL) rule [41]. CHL is used for creating the neuron connections: each time a neuron wins, an edge is created, linking it to the second nearest neuron, if it does not exist yet (Fig. 2.4). If there was an edge, its age is set to zero. As previously introduced, DGSOT does not exploit edges, while GHNG and GH-EXIN do. Besides, both algorithms use the same aging procedure. They increment the age of all links emanating from the winner by one. However, in case a link age is greater than the age_{max} scalar parameter, it is eliminated (pruned).

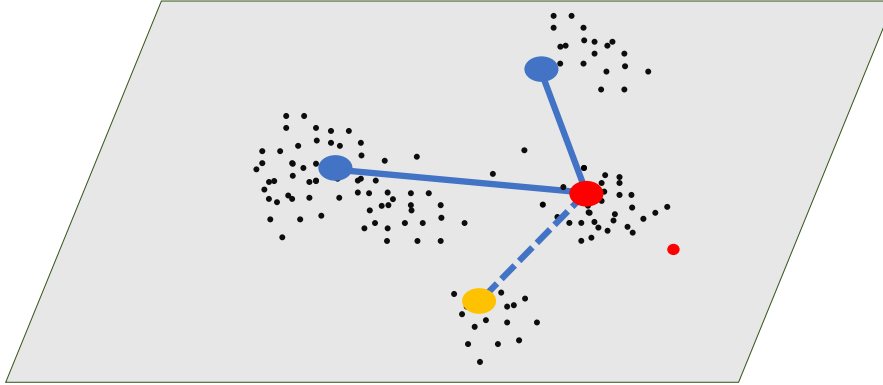


Figure 2.4: When a new datum (small red dot) is presented, the winner (big red dot) and the second winner (big yellow dot) are joined through an edge (blue dotted line), if the edge did not exist before.

Pruning

Pruning is the process through which nodes can be removed if redundant. DGSOT does not exploit any pruning technique, in the sense that the redundancy is only checked each time a neuron is added, but old neurons cannot be removed. Vice versa, GHNG and GH-EXIN remove all lonely neurons. GH-EXIN checks for lonely neurons at the end of each epoch, while GHNG checks at each iteration. Besides, GHNG may prune an entire set of neurons if GNG enters convergence phase. In this case, all the neurons created before last convergence check are pruned.

2.1.3 Tree Building

Hierarchical divisive clustering algorithms build a tree starting from a root node. Through vertical and horizontal growths successive splits are determined. For each father neuron a neural network is trained on its corresponding Voronoi set, i.e. the set of data represented by the father neuron. Sons are the neurons of the associated basic neural unit and determine a subdivision of the father Voronoi set. For each leaf the procedure is repeated.

Root Leaf

The root of the tree is the node of the hierarchy. Both DGSOT and GH-EXIN associate the whole data set to a fictitious neuron (a.k.a. root node). The first basic neural unit is then trained on the Voronoi set of this fictitious unit, i.e. on the whole data set. All nodes created by the first basic neural unit are sons of the root node. Conversely, GHNG does

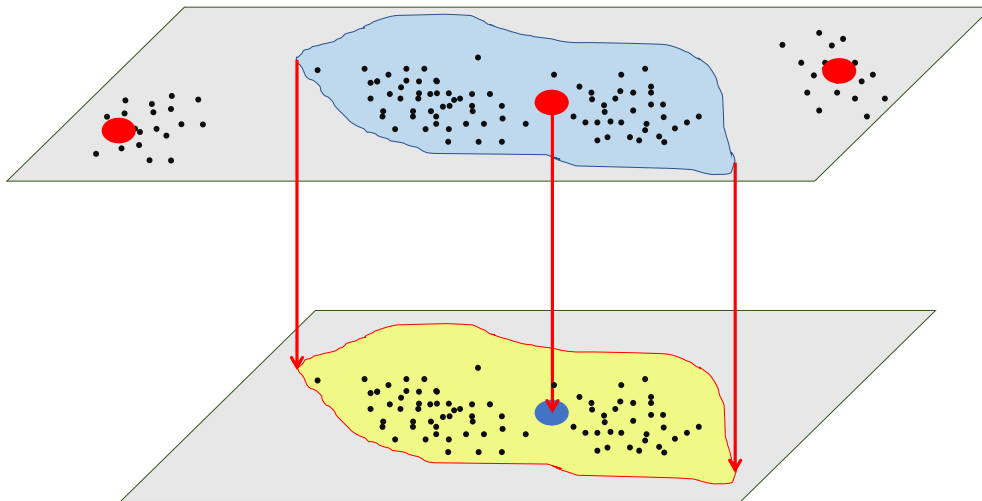


Figure 2.5: The leaf node, represented by the red dot inside the light blue region, has not met the stop criteria. Therefore, a new hierarchical level is generated. In the child level, the content of the Voronoi set of the father node will be further analyzed.

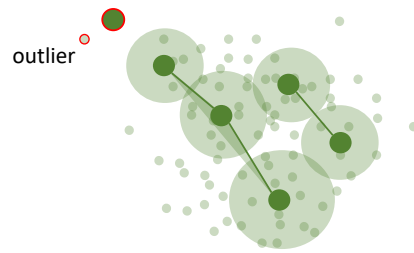
not have any fictitious father, all nodes created at the first layer are orphans. It could be argued that GHNG builds a forest other than a single tree structure.

Vertical Growth

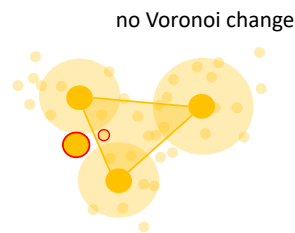
Vertical growth is the process in which a deeper layer is added to a leaf node of the hierarchy (Fig. 2.5). In all considered algorithms, it always implies the creation of a *seed*, i.e. a pair of neurons which represents the starting structure of a new basic neural unit. This kind of growth is exploited as long as a higher resolution is needed. In order to understand whether a vertical growth is necessary, all considered algorithms check if the quantization error of the basic neural unit is below a user-defined threshold. DGSOT checks data heterogeneity as the average distance of the data to the neuron reference vector. GHNG, instead, checks whether the number of levels in the hierarchy exceeds a user-dependent parameter MAX_LEVEL or the last basic neural unit inside a branch has not created any additional neuron to the initial seed. Finally, GH-EXIN contemporary checks both data heterogeneity through the H_{cc} index and data cardinality, i.e. the size of the leaf Voronoi set.

Data Reallocation

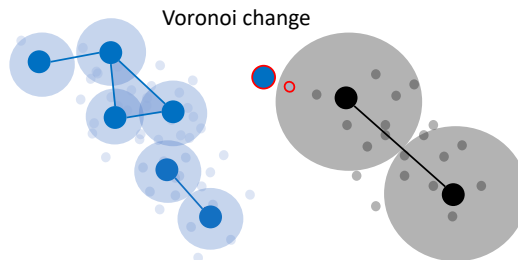
Data reallocation is the mechanism by which data associated to pruned nodes (orphan data) are possibly reassigned to other neurons. This mechanism is a novelty introduced in GH-EXIN (see Fig. 2.5 for a visual interpretation). In fact, DGSOT, by not pruning nodes, does not need to reassign orphan data. GHNG does not reallocate orphan data, despite lonely neurons are removed. In GH-EXIN, instead, all orphan data are labelled as



(a) The orphan datum cannot be explained by the green leaf node, as the hypersphere of the closest neuron does not reach it.



(b) The new datum is inside the convex-hull (or inside a hypersphere) of the current network component. Therefore, it will be assigned to the Voronoi set of the closest yellow neuron.



(c) The new datum (previously belonging to the blue leaf node) is now closer to the black network. Therefore, it will be assigned to the Voronoi set of the closest neuron, i.e. the black one on top-left).

Figure 2.6: In the above figures different colors represent different leaf nodes. Small dots represent data while big dots represent neurons. The shaded areas correspond to regions explained by the neural network. Big red circles mark lonely neurons, while small red circles represent data belonging to them. As lonely neurons are pruned, GH-EXIN tries to reallocate these data or to mark them as outliers.

potential outliers, at the end of each epoch. For each potential outlier, GH-EXIN looks for a new winner among all leaf nodes. Nonetheless, in case the winner belongs to the same basic neural unit of the pruned node and datum is outside its hypersphere, datum is

definitely marked as outlier and is not reassigned.

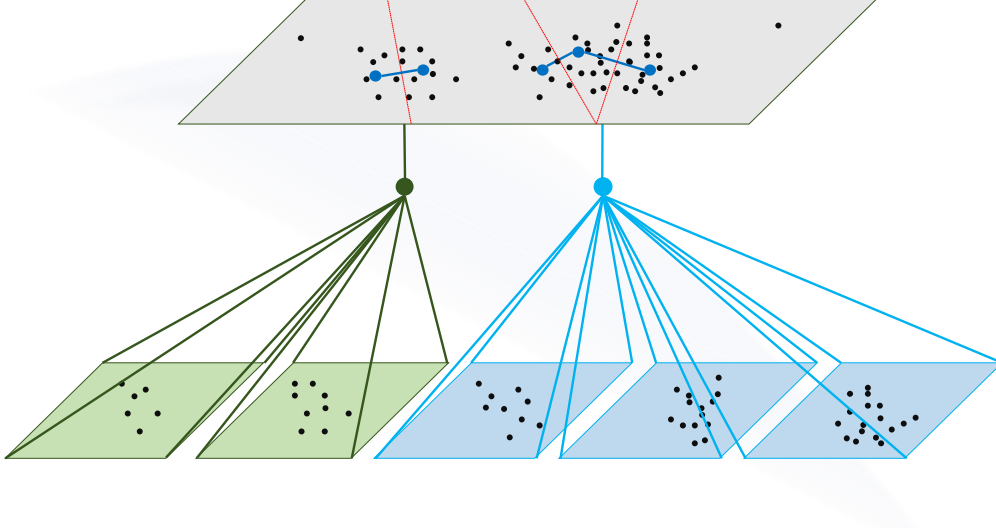


Figure 2.7: The Voronoi regions of each neuron are represented with dotted red lines. The learning process in the father node has generated five neurons, which normally results in five leaf nodes. However, the network topology (through the analysis of the connected components) suggests that data are separated in two distinct clusters. Therefore, in order to take into account this finding, two abstract neurons are placed in the centroid of each cluster, and the corresponding nodes are inserted in the middle of the hierarchy.

Topology Abstraction Check

Another remarkable novelty introduced by GH-EXIN consists in a simultaneous vertical and horizontal growth (Fig. 2.7). In fact, at the end of the training process,

Horizontal Growth

Horizontal growth refers to the addition of further neurons by the basic neural unit to the initial seed. This characteristic allows to build more complex hierarchical structures other than binary. This process is performed by all algorithms through the respective neuron creation mechanism previously described. The resulting graph of the basic neural unit is analyzed by searching for connected components. If more than one connected component is detected, the algorithm tries to extract an abstract representation of data. Hence, each connected component, representing a cluster of data, is associated with a novel abstract neuron. The reference vectors of abstract neurons are placed in the centroids of the respective clusters. The tree structure is modified by inserting abstract neurons between the leaf nodes and the father node, resulting in a simultaneous vertical and horizontal growth.

2.1.4 The GH-EXIN Algorithm

Recalling fig. 2.1, for each father node, a neural network is trained on its corresponding Voronoi set (set of data represented by the father neuron). The Voronoi cardinality is shown in fig. 2.1 next to each vertex. The sons are the neurons of the associated neural network and determine a subdivision of the father Voronoi set. For each leaf, the procedure is repeated until either the H_{cc} index of the leaf has fallen below $H_{cc}max$ (user-dependent parameter) or the cardinality of the leaf is less the min_{card} . The initial structure of the neural network is a seed, i.e. a pair of neurons, which are linked by an edge, whose age is set to zero.

For each epoch the basic iteration starts at the presentation of a new data, say x_i . All neurons are ranked according to the Euclidean distances between x_i and their weights. The neuron with the shortest distance is the winner w_1 . In case the datum result to be novel - i.e. both outside of the convex polytope and of the hypersphere of radius T_r of the winner (novelty test) - a new neuron n_{new} is created (left branch of fig. 2.8). The initial weight vectors and neuron thresholds T_r are given by heuristics: n_{new} is place on x_i and its threshold is set equal to w_1 threshold. No edge is created at this time: n_{new} is labelled as *lonely neuron*. This label will be removed the first time an edge will be placed between n_{new} and another neuron.

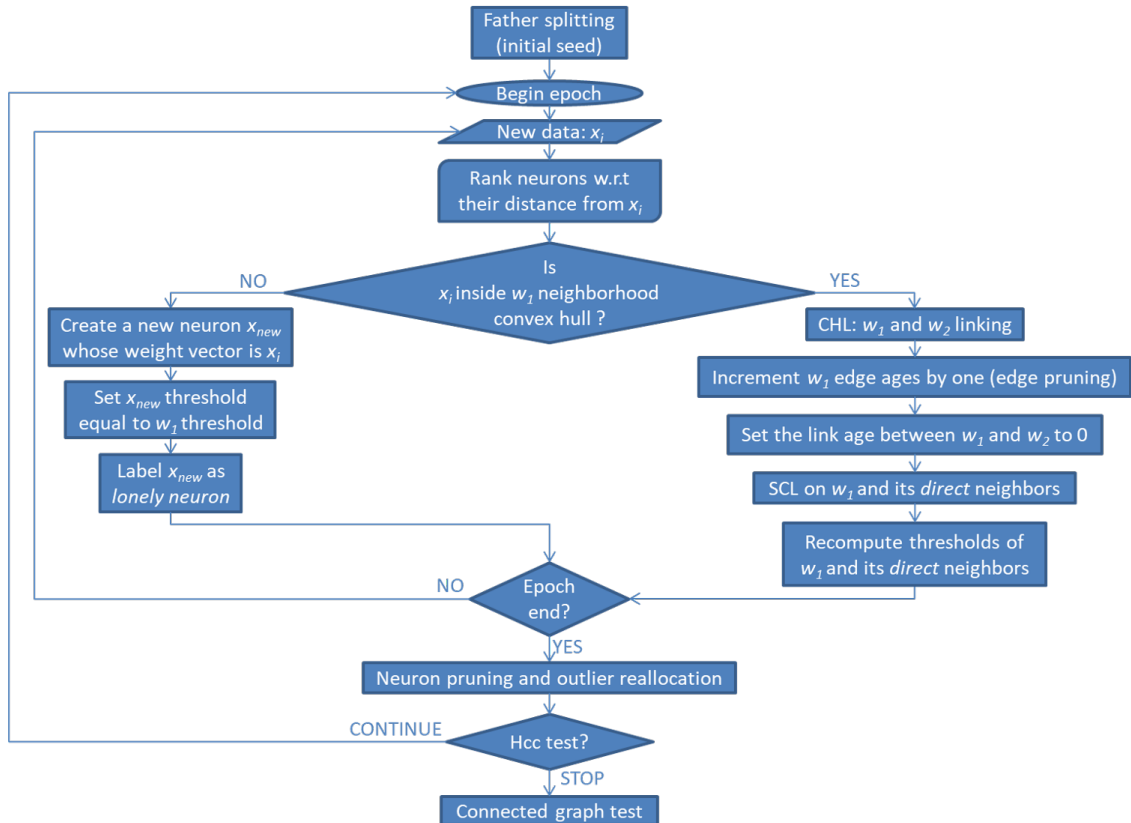


Figure 2.8: GH-EXIN flowchart

Otherwise, in case datum is not novel, first winner w_1 and second winner w_2 are linked by an edge according to *CHL* rule, if it does not exist yet. If there was already an edge, its age is set to zero. Anyhow, the age of all other links emanating from the winner is incremented by one; if a link age is greater than the *agemax* scalar parameter, it is eliminated (pruning). Reference vectors of w_1 and its direct neighbors are updated according to the aforementioned equation 2.3. Thresholds of the winner and of its neighbors are recomputed as their position has been modified. This process is repeated for all data. At the end of each epoch, if a neuron remains unconnected (no neighbors), it is pruned, but the associated data are analyzed and possibly reassigned.

Each leaf neural network is controlled by H_{ccperc} because GH-EXIN is searching for biclusters (it is estimated by using the data of each Voronoi set). In particular, the training epochs are stopped when the estimated value of this parameter falls below a percentage of the value for the father leaf.

This technique builds a vertical growth of the tree. The horizontal growth is generated by the neurons of each network. However, a simultaneous vertical and horizontal growth is possible. At the end of a training, the graphs created by the neuron edges are checked. If connected subgraphs are detected a further both vertical (a layer is added) and horizontal (as many neurons are inserted as the number of subgraphs found) growth is performed.

GH-EXIN has been developed in MATLAB. The code is freely available at [42].

2.1.5 User-Dependent Parameters

Resuming, the GH-EXIN neural network requires some user dependent parameters:

- the two learning rates constants, $\alpha_{\gamma 0}$ and α_{i0} , used to update reference vectors of the winner and its neighbours;
- the scalar *agemax* used for edge pruning: it has to be lowered if more edges (and neurons) have to be pruned, indirectly controlling the leaf cardinality;
- the biclustering quality indices, i.e. H_{ccperc} the percentage of H_{cc} , used to determine when to stop training epochs, and its maximum value H_{ccmax} , which indicates when a compact bicluster has been found ;
- the minimum cardinality of leaves, used to avoid single point clusters, meaningless in the context of bclustering.

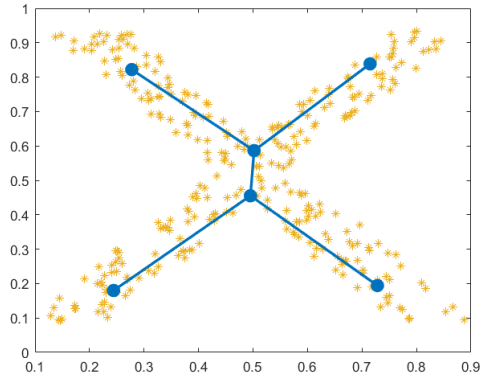
2.2 Benchmark Comparison

In this section synthesis experiments are shown and discussed. These experiments concern the comparison among GH-EXIN and other two hierarchical and self-organizing neural networks. The first one is Growing Hierarchical Neural Gas (GHNG) which is the hierarchical version of Growing Neural Gas. The second one is Dynamically Growing Self-Organizing Tree (DGSOT) which derives from SOTA. These three neural networks are tested on five datasets having different characteristics. The data of the first four datasets are randomly drawn from:

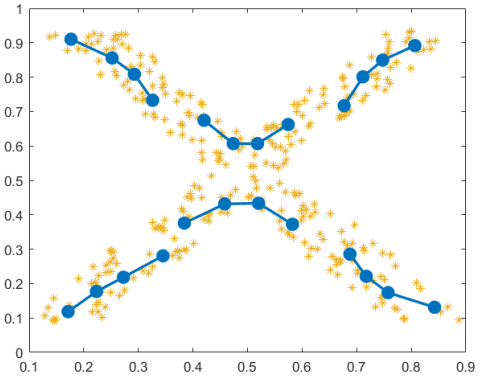
- A planar X-shape manifold
- A three-dimensional bimodal distribution a.k.a. twin-peaks
- A planar square-shape manifold having a beta distribution
- A planar spiral-shape distribution

The fifth dataset consists on the RGB triplet colors of a baboon image. This dataset selection has been done so that the networks can be evaluated in addressing different tasks. In particular, the X-shape dataset is a simple starting point to evaluate manifold learning; indeed, it is a two-dimensional manifold in a two-dimensional space. It is easy to learn and evaluate visually. The twin-peaks and the baboon datasets are slightly more complex both to learn and to evaluate (at least visually). The twin-peaks is an artificial example of a regular function, while the baboon dataset is closer to a real application of color quantization. The last two experiments are even harder. The square dataset tests the networks in a non-uniform distribution setting while the spiral one tests the ability in learning a one-dimensional manifold in a space having a higher dimensionality. In all the following experiments the neural networks' setup is chosen so that both the structure and the number of neurons of each layer are approximately the same for all the networks.

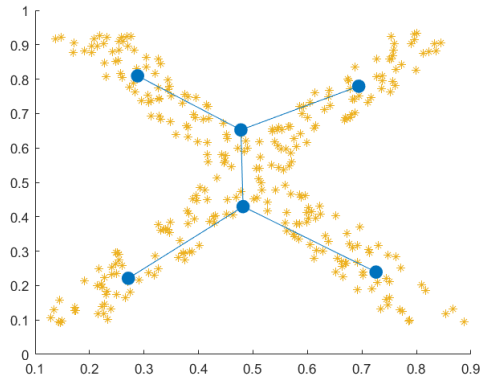
Figure 2.9 shows the first two hierarchical layers of the three neural networks on the X-shape distribution. At a glance, the three algorithms have learnt approximately well the manifold. To be fussy, GH-EXIN appears to be more elegant and efficient in the second layer, while DGSOT seems a little bit under stress.



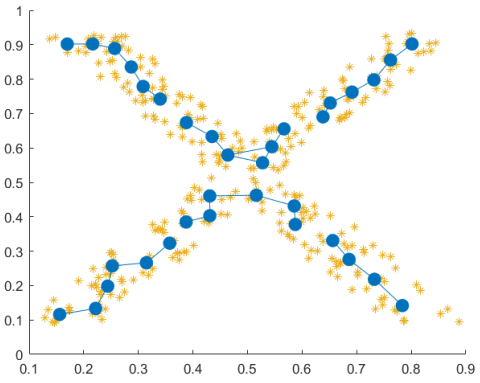
(a) GH-EXIN first layer



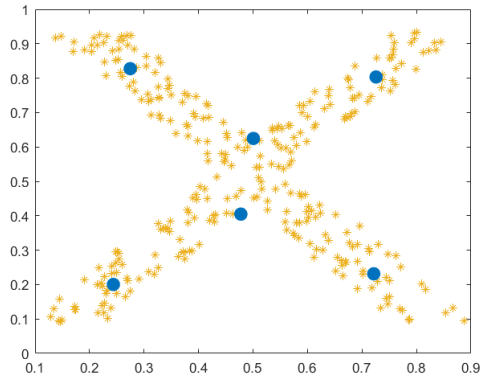
(b) GH-EXIN second layer



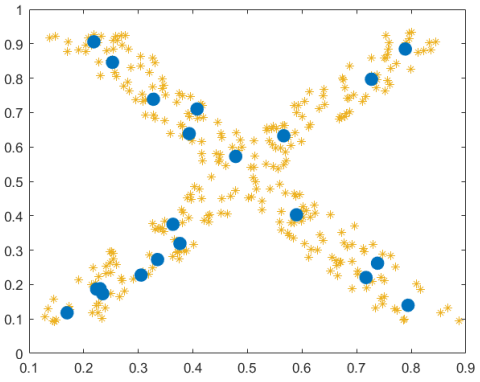
(c) GHNG first layer



(d) GHNG second layer



(e) DGSOT first layer



(f) DGSOT second layer

Figure 2.9: First (**left**) and second (**right**) layers of GH-EXIN, GHNG and DGSOT on the X-shape distribution.

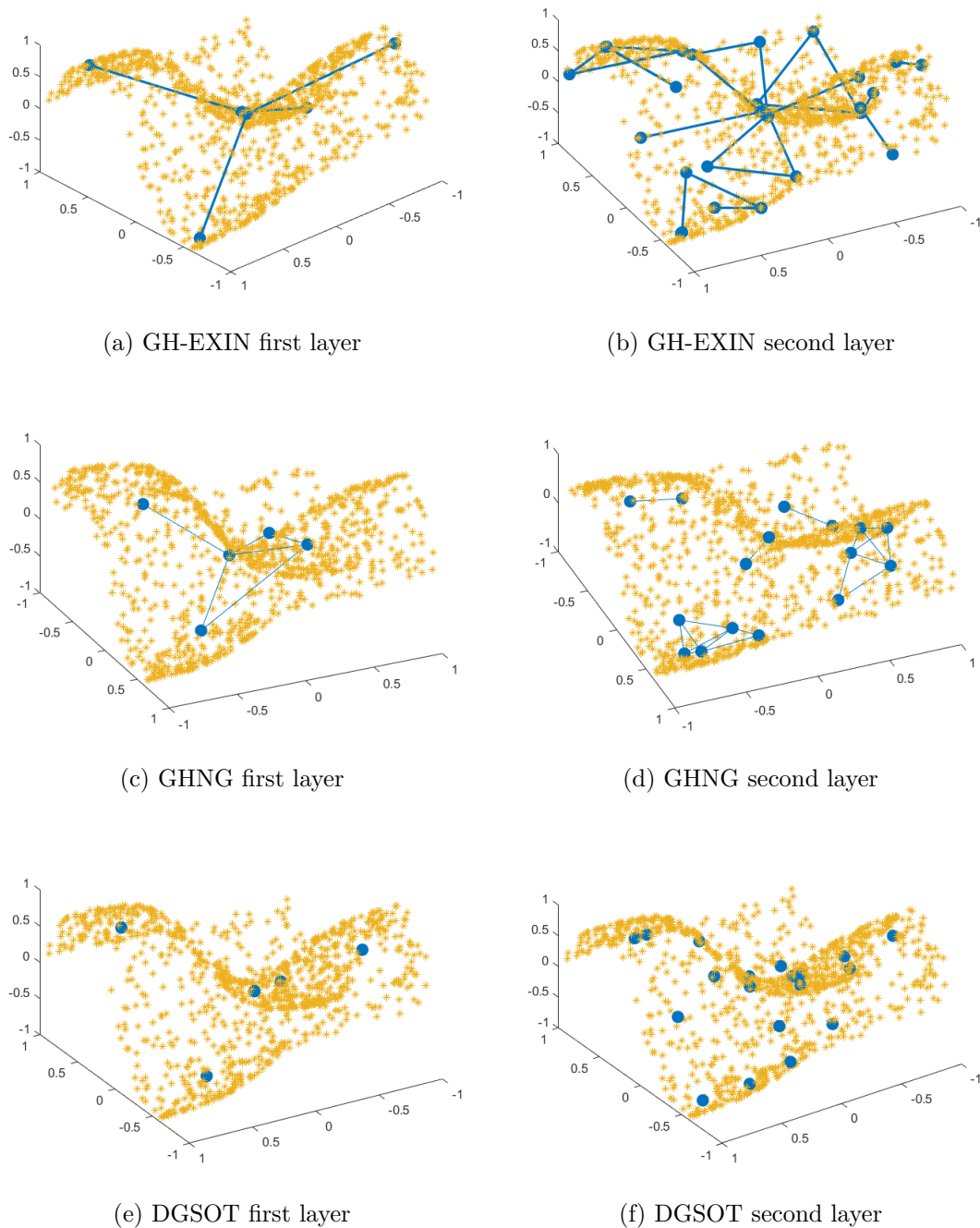
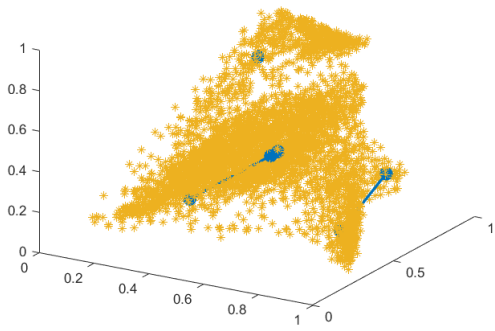
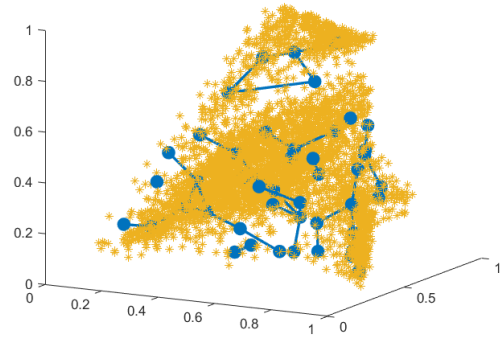


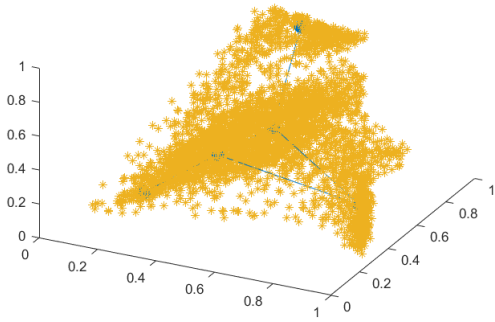
Figure 2.10: First (**left**) and second (**right**) layers of GH-EXIN, GHNG and DGSOT on the twin-peaks dataset.



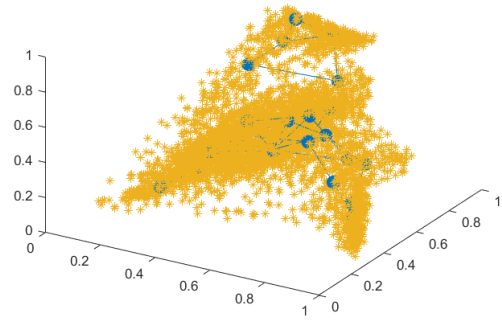
(a) GH-EXIN first layer



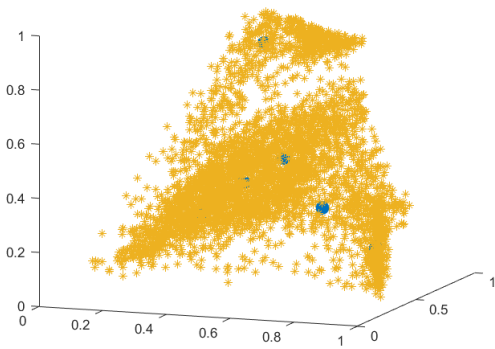
(b) GH-EXIN second layer



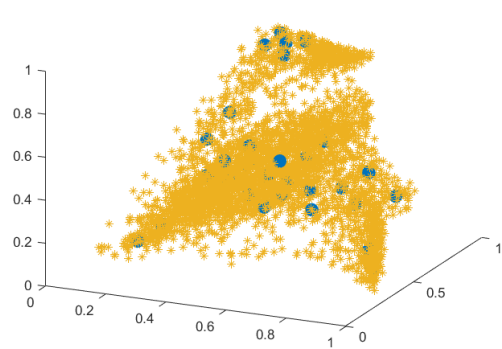
(c) GHNG first layer



(d) GHNG second layer



(e) DGSOT first layer



(f) DGSOT second layer

Figure 2.11: First (**left**) and second (**right**) layers of GH-EXIN, GHNG and DGSOT on the baboon dataset.

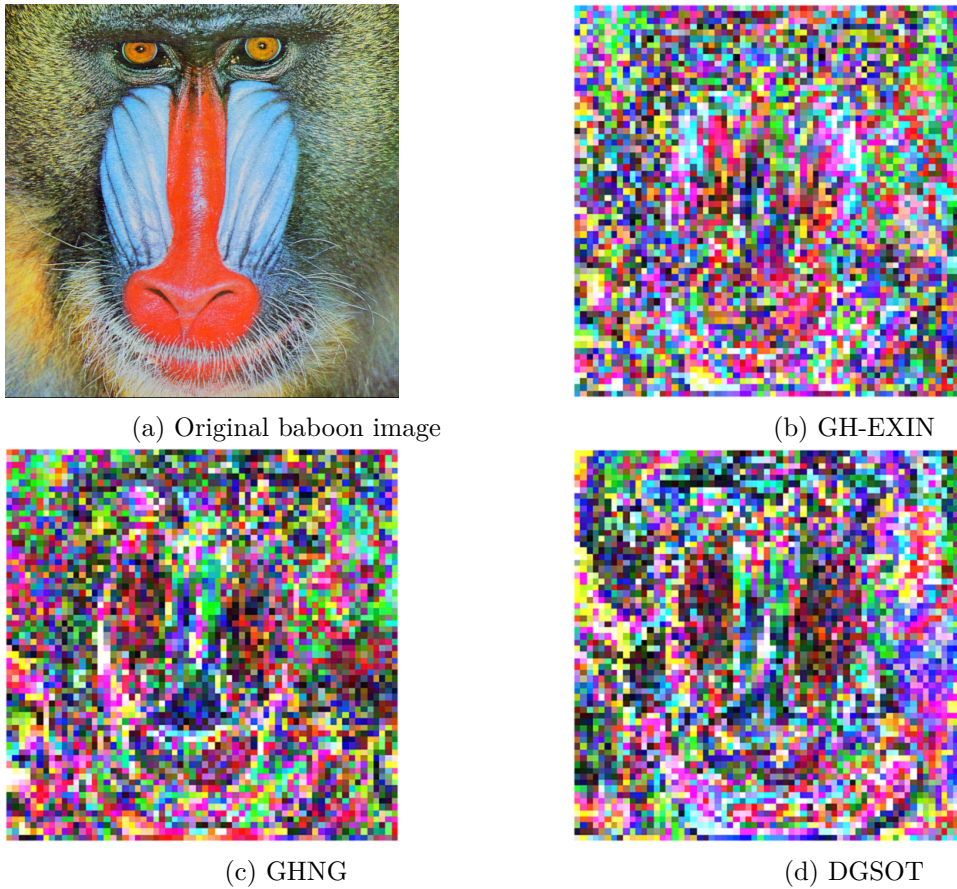
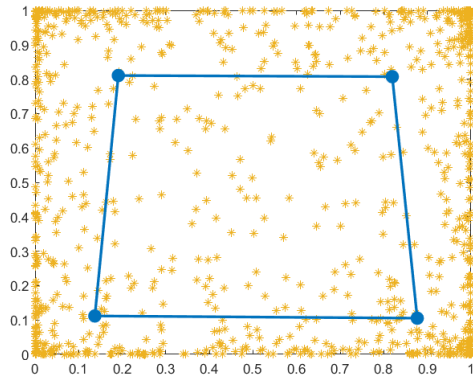


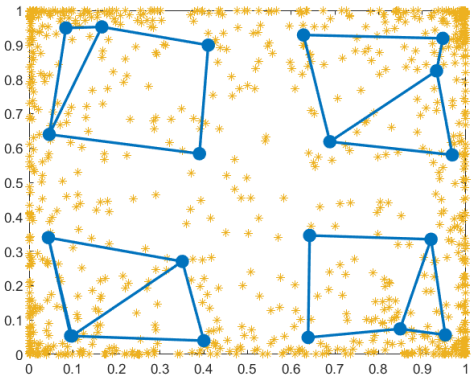
Figure 2.12: Difference between the original baboon image and the

Figure 2.10 shows the first two hierarchical layers of the three neural networks on the twin-peaks dataset. Here it is more difficult to evaluate the self-organizing abilities of the networks visually. Anyway, all the three networks learn the basic structure of the manifold in the first layer: the minima, the maxima and the saddle point. However, the edge distribution of GH-EXIN is more symmetric (as it should be) than the GHNG distribution which links only one of the two maxima to the minima. The DGSOT neural networks does not have edges by design, so it cannot show if the manifold is connected or not.

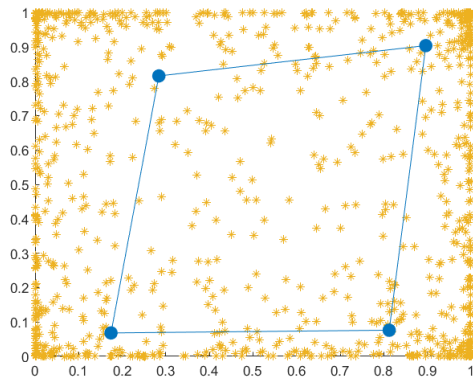
Figure 2.11 shows the first two hierarchical layers of the three neural networks on the baboon dataset. In this case the network evaluation is even harder because of the huge amount of data and their distribution. However, even in this case, the number of GHNG edges is greater than the number of GH-EXIN ones. Since the figures are not easy interpretable, it is worth showing the difference between the original baboon image and the quantized image. The darker the difference-image the better the color quantization. Figure 2.12 exhibits the difference-image by using the three neural networks. They seem to have similar performance, but DGSOT and GHNG use a few more neurons with respect to GH-EXIN.



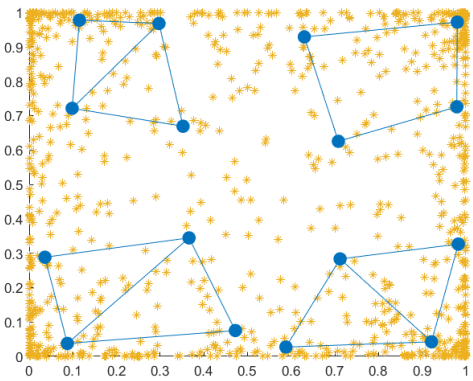
(a) GH-EXIN first layer



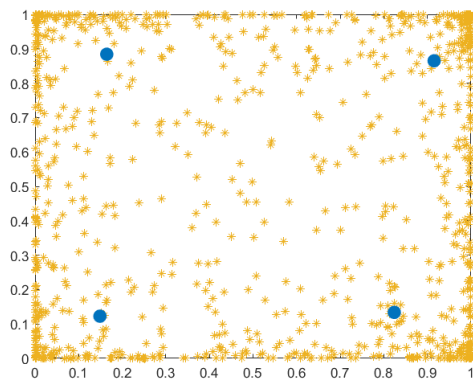
(b) GH-EXIN second layer



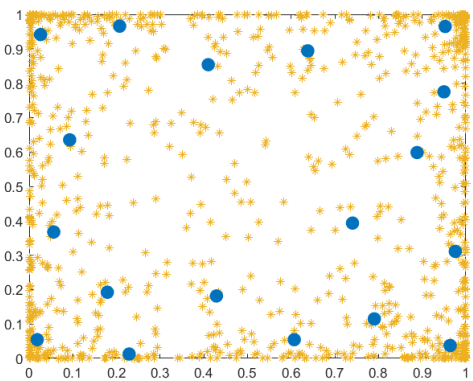
(c) GHNG first layer



(d) GHNG second layer

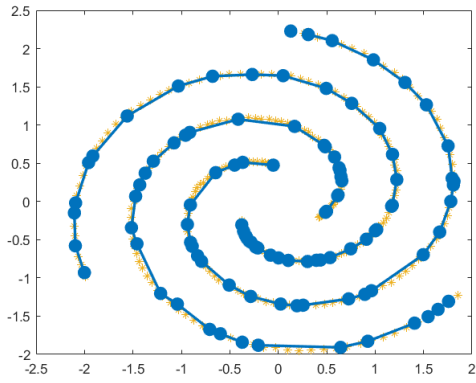


(e) DGSOT first layer

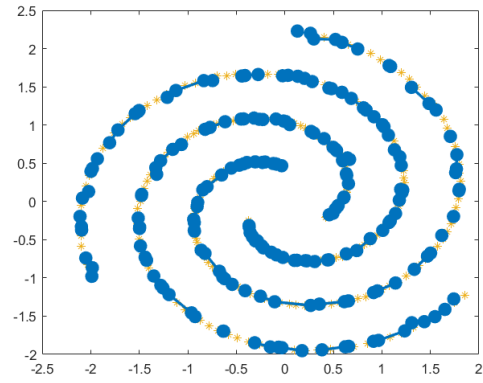


(f) DGSOT second layer

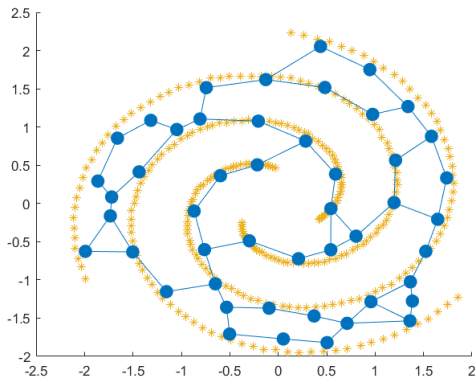
Figure 2.13: First (**left**) and second (**right**) layers of GH-EXIN, GHNG and DGSOT on the square distribution.



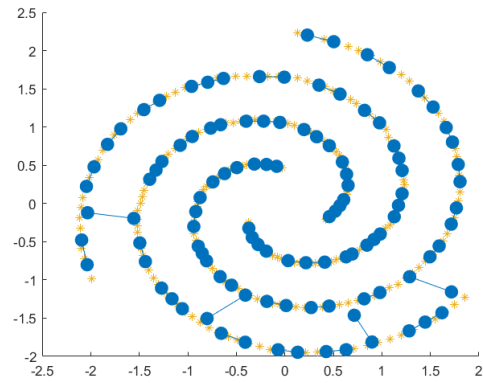
(a) GH-EXIN first layer



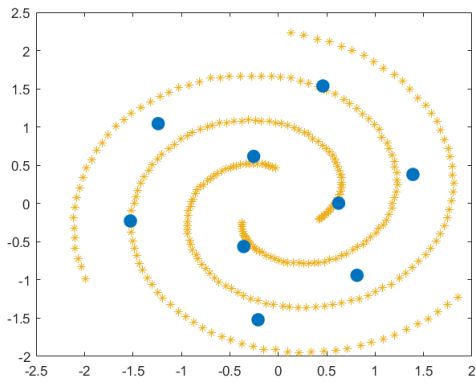
(b) GH-EXIN second layer



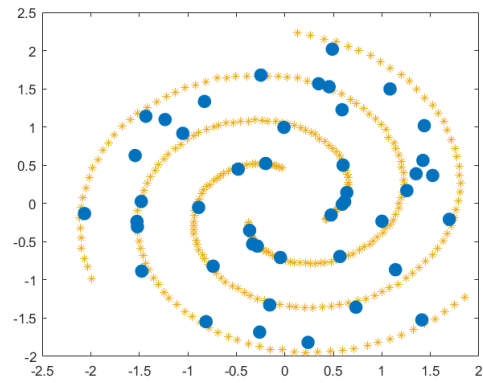
(c) GHNG first layer



(d) GHNG second layer



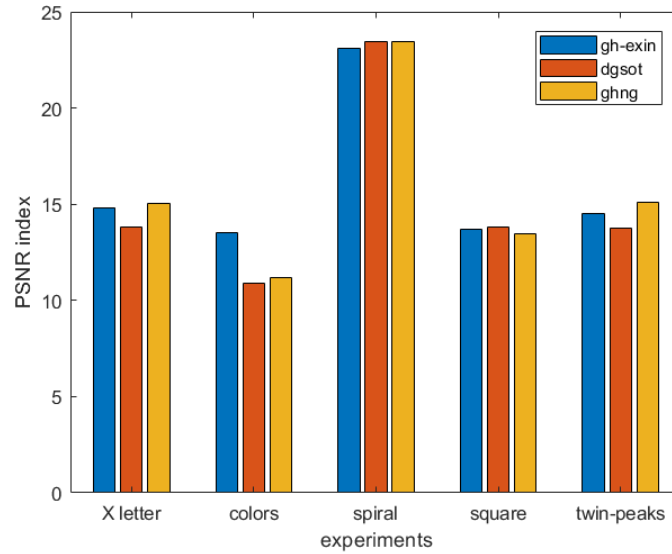
(e) DGSOT first layer



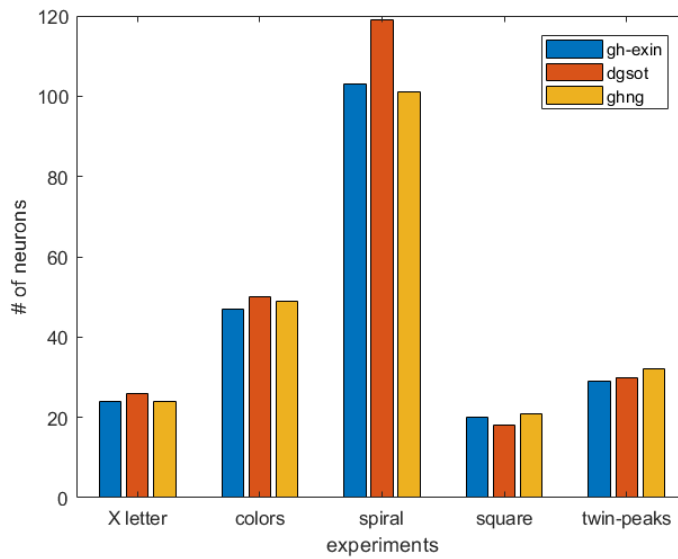
(f) DGSOT second layer

Figure 2.14: First (**left**) and second (**right**) layers of GH-EXIN, GHNG and DGSOT on the spiral distribution.

Figure 2.13 shows the first two hierarchical layers of the three neural networks on the square dataset. Here all the networks behave almost the same. However, in the second



(a) PSNR



(b) Number of neurons

Figure 2.15: Number of neurons and PSNR of each neural network on each dataset.

layer the DGSOT neurons are much more attracted by the borders than in the other two algorithms. Overall, in different ways, all the three networks perform well on this task.

Figure 2.14 shows the first two hierarchical layers of the three neural networks on the spiral dataset. Only for this task, if the number of neighbors of the winner neuron is greater than or equal to the space dimension, then GH-EXIN does not use the average threshold check but it uses only the convex hull check (refer to section X for further details). By

using this change, GH-EXIN creates more neurons in the first layer following the manifold more accurately. Remarkably, unlike GHNG, GH-EXIN does not create edges between different arms of the spiral. On the other hand, DGSOT really has trouble in generating more neurons in the first layers, failing in a satisfying manifold understanding. Only at the third level this network seems to improve the quality of its organization.

Since neural network comparison is not trivial, a quantitative evaluation method is needed. For this purpose, the peak-signal to noise ratio (PSNR) index is used as in Palomo et al. GHNG paper. The PSNR index is defined as follows:

$$PSNR = 10 \log_{10} \left(\frac{MAX_l^2}{MSE} \right) \quad (2.4)$$

where MAX_l^2 is the squared Euclidean norm of the vector which joins the two most distant points in the input distribution support. The mean squared error (MSE) is given by:

$$MSE = \frac{1}{M} \sum_{i=1}^M \|w_i - \hat{x}_i\|^2 \quad (2.5)$$

where M is the number of the input samples, x_i is the i -th input sample and w_i is the winning neuron corresponding to x_i . Fig. 2.15(a) shows the PSNR index of the three neural networks on the five input distributions.

Fig. 2.15(b), instead, shows the number of neurons on the leaf nodes generated by each network on the five input distributions. Observing the two images, it is possible to make some quantitative comparisons. Firstly, from a general point of view, the number of leaf nodes and the PSNR is approximately the same for all the three networks. However, there are some exceptions. The first one is the number of neurons of the DGSOT network for the spiral distribution. In this experiment DGSOT needs a significant extra number of neurons (and one more layer! It needs three layers instead of two like GHNG and GH-EXIN) to reach the same performance of GH-EXIN and GHNG in terms of PSNR. The second relevant exception concerns the color quantization task. In this setting GH-EXIN has less leaf neurons than GHNG and DGSOT but it has by far the best PSNR performance. Unfortunately, the PSNR is a good way to have a quantitative evaluation for the neuron positions, but it does not tell anything about the quality of the connections. Observing the above network images, especially the two-dimensional ones, it seems that GH-EXIN is much more elegant in connecting neurons, providing a better manifold representation.

2.3 Gene Analysis

2.3.1 Neural Biclustering with GH-EXIN

In the previous sections, we presented the GH-EXIN neural network and we compare its pros and cons with state-of-the-art hierarchical neural techniques. In this section we exploit GH-EXIN in order to analyze CRC microarray data. The previously introduced high-dimensional issue in handling microarray data may be bypassed by unsupervised techniques thanks to biclustering. Indeed, by considering genes as samples and cellular tissues as features, the dimensionality of the input space is dramatically reduced. Successively, each batch of genes is further analyzed by clustering cellular tissues in a lower dimensional space. We will refer to *neural biclustering* as the successive exploitation of GH-EXIN in biclustering microarray data. Each time GH-EXIN is exploited, a tree is built, either in the gene or in the tissue space, for the gene clustering in the higher-level leaves. The validity of the leaves is tested and possibly GH-EXIN is called again in the corresponding projected space of each leaf. This procedure is recursively repeated until the cardinality of the tissues or the cardinality of the genes of a leaf is under the minimum threshold. At this point the leaf is saved and the algorithm continues by processing the other leaves. The order in which leaves are processed depend on their ranking, based on their Hcc value. Low values of Hcc associated to an acceptable cardinality do not imply a final bicluster has been detected, above all for the presence of high noise in data. An additional analysis is required, which depends on several considerations.

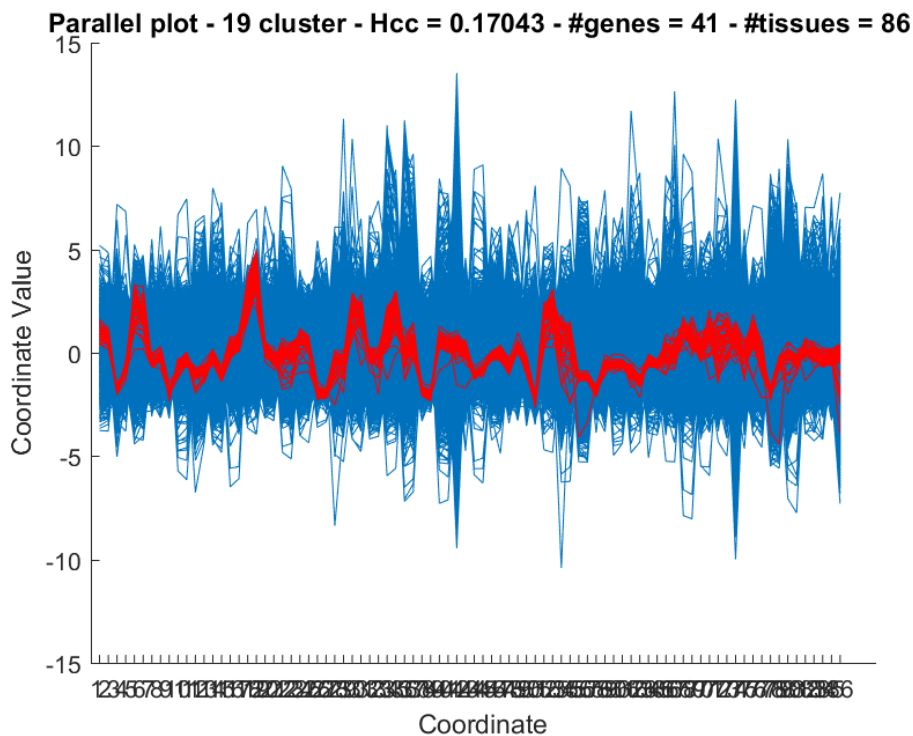


Figure 2.16: Parallel coordinates of a cluster of gene

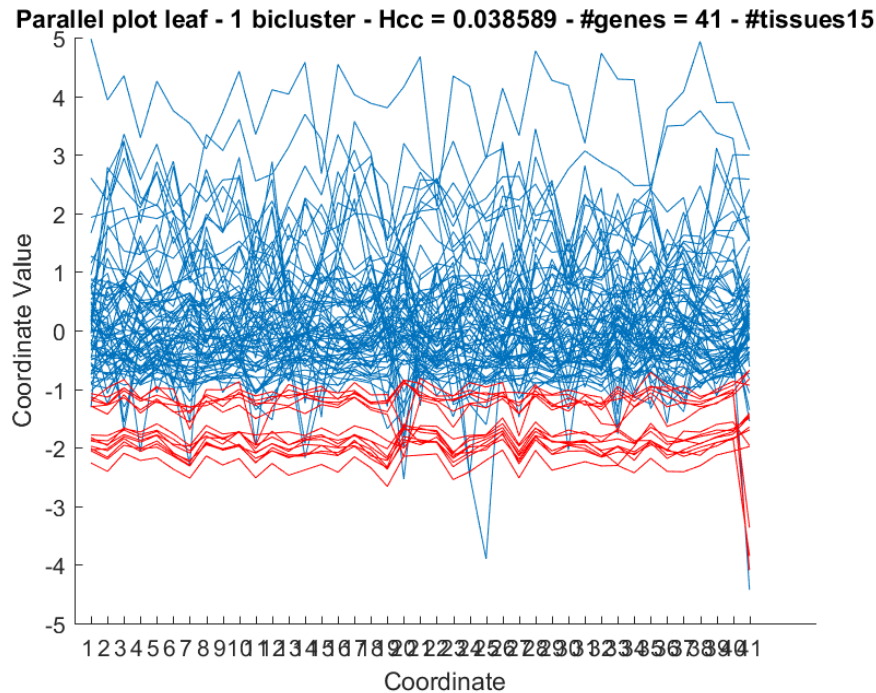


Figure 2.17: Parallel coordinates of a bicluster.

Here, the final leaves are studied from two different points of view: parallel coordinates and singular value decomposition (SVD), for the analysis of the numerical rank of the submatrices associated to the biclusters.

2.3.2 Validating Techniques

Fig. 2.16 shows this kind of plot by visualizing genes as samples (colored polylines) and murine tissues as features (parallel vertical axes) on a leaf of GH-EXIN in the gene space, whose characteristics are shown in the top line of the figure. Blue polylines represent all genes available in the dataset, while red polylines stand for genes collected in the 19th gene cluster. The red grouping of polylines show coherency, which confirms the quality of gene clustering. A similar validation analysis is used after the GH-EXIN clustering in the tissue space which is run after projecting the Voronoi set of the 19th gene leaf (cluster).

Figs. 2.17 and 2.18 show two parallel coordinate plots in which vertical axes (here visualized as the corresponding abscissas in the coordinate axis) represent the 41 genes belonging to cluster 19, while polylines stand for murine tissues. In particular, blue polylines represent all the tissues and red ones the tissues grouped in the bicluster. The difference between the two images consists in a different setup of a parameter of the algorithm, $Cmin2$, which regulates the maximum number of tissues accepted in a bicluster. In the first case a higher value of the parameter is set, in order to find a bigger bicluster. However, both pictures show an excellent bicluster coherency revealing the goodness of GH-EXIN as a tool for biclustering. The biclusters shown in both figures are coherent additive values

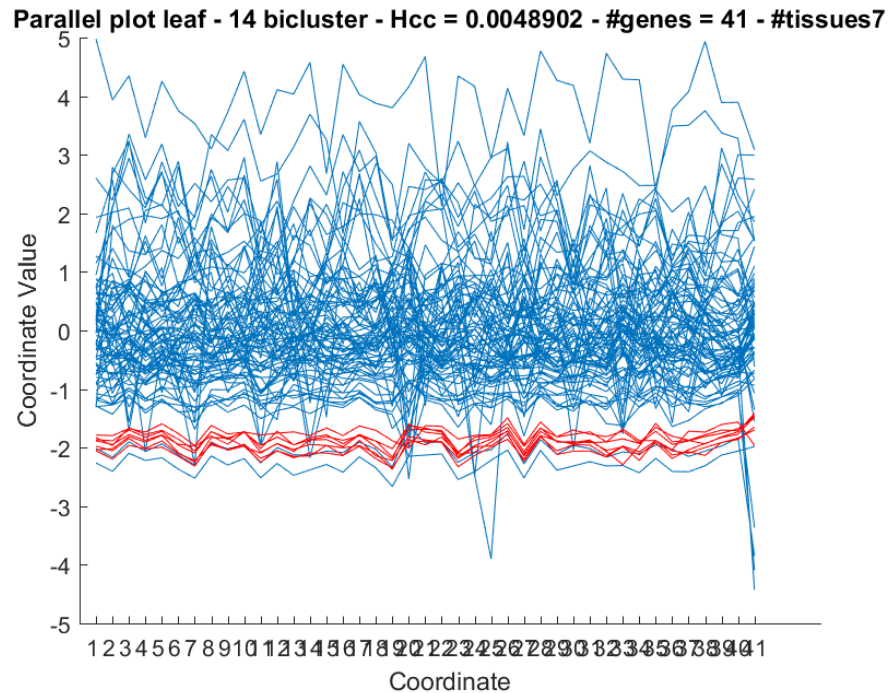


Figure 2.18: Parallel coordinates of a smaller bicluster.

biclusters in which the values vary both according to the rows (the axes in this case) and according to the columns (the polylines). This can be inferred from the pictures, because a difference is present between two gene expressions on different polylines but along the same axes, but this difference remains stable along the polyline. The same is also valid between two gene expressions on different axes of the same polylines.

This visualization tool can be considered as a first validation of the quality of the leaves. A second validation can be performed by analyzing the singular values of the resulting bicluster matrices. According to the theory, in case of noiseless data, biclusters with constant values and with constant values on rows or columns have rank one, while biclusters with coherent values have rank three. The difficulty raises in case of noise, because there are no more zero singular values. Indeed, the size of the last values increases with the level of noise. It then becomes a problem in numerical rank estimation. The SVD of the matrix of the bicluster in fig. 2.17 has the first two singular values (42.5 and 4.5) well separated from the other ones (the third one is equal to 1.5), considering also that the matrix has been scaled in the preprocessing stage). This result represents the sum of two biclusters of rank one, certainly, considering the associated parallel plot, two constant row biclusters. Indeed, fig. 2.17 shows two clusters (coherent polylines, whose thickness depends on noise level). Hence, it can be deduced that a further clustering (and projection) is needed in order to have a single bicluster. Instead, the SVD of the matrix of the bicluster in fig. 2.18 (see fig. 2.19) has only the first singular value (32.5) well separated from the remaining ones (the second one is equal to 1.1). As also confirmed in fig. 2.18, it represents a constant row bicluster. This result does not require a further analysis.

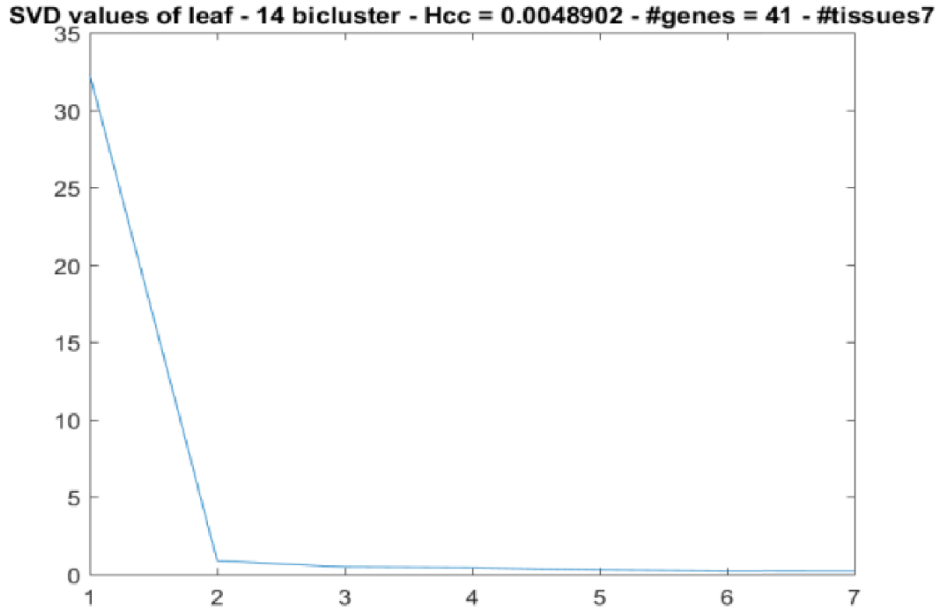


Figure 2.19: Singular values for leaf 14.

| Leaf node | Cardinality | | Hcc |
|-----------|---------------------|----------------------|-------|
| | <i>tissue class</i> | <i>genes tissues</i> | |
| 1 | 54 | 8 | 0.027 |
| 1 | 58 | 6 | 0.059 |
| 2 | 8 | 8 | 0.027 |
| 2 | 8 | 6 | 0.029 |
| 2 | 71 | 5 | 0.031 |
| 2 | 5 | 5 | 0.039 |
| 3 | 7 | 8 | 0.025 |
| 3 | 7 | 13 | 0.025 |
| 3 | 19 | 6 | 0.026 |
| 3 | 19 | 5 | 0.026 |
| 3 | 41 | 13 | 0.028 |
| 3 | 19 | 8 | 0.029 |

Table 2.1: Best leaves in terms of biclustering quality (H_{cc} index).

2.3.3 Experimental Findings

In order to better analyze genetic expressions common for different patients, the dataset has been divided into three parts (classes). This division follows the murine tissues response to anti-cancer drugs. At the end, three datasets have been derived, one for the mice which

started recovering after three weeks of treatments, a second one for the mice which had a stable situation and at last one also for the case in which drugs had no effect and the cancer kept growing. This type of division has been maintained also in the summary Table 2.1, where it has been reported the information about the cardinality of the biclusters, both in the tissue and in the gene space and the value of the Hcc index. In the table there are only the best biclusters for each class, ranked according to the class and the Hcc index. As last step, as a biological feedback, the scientific relevance of the selected genes has been taken in account. Among all the biclusters found, the one that grouped the most interesting genes in the cancer field has been the one that also had the lowest Hcc index value. Indeed, the 7 genes present in the bicluster are the following:

- "CSAG1", "CSAG3", "CSAG3A", which belong to the same CSAG family. These genes are well known in literature as associated with chondrosarcomas, but they are also present in normal tissues. Furthermore, CSAG3 and CSAG3A are gene coding the "Chondrosarcoma-associated gene 2/3 protein" which is a "drug-resistance related protein, its expression is associated with the chemotherapy resistant and neoplastic phenotype. May also be linked to the malignant phenotype" [43].
- "MAGEA2", "MAGEA3", "MAGEA12", "MAGEA6", which belong to the same MAGEA family. These genes are melanoma antigens which "Reduce p53/TP53 transactivation function" and also "Represses p73/TP73 activity" [44]. Both p53 and p73 are tumor suppressor proteins which regulate cell cycle and induct apoptosis.

The relevant issue is that these gene families are not only important by themselves, but this analysis suggests that, at least in the observed condition, they may also coregulate each other. It is also important to notice that this bicluster phenomenon has been observed within the tissues belonging to the third class, the one where tissues unable to respond to drugs are present.

2.4 An Application to Face Recognition

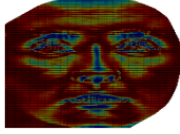
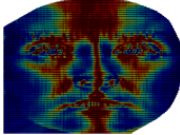
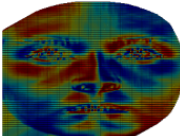
In order to further test the capabilities of analysis of both the neural biclustering framework and of the GH-EXIN algorithm, they have been also applied on a completely different context: the 3D Face Recognition. Differently from what previously done, biclustering has not been used to understand which face descriptors correlate. Instead, biclusters have been searched, here, to prove the discriminative capabilities of face descriptors. The analysis described in the following have been exposed at the 2018 WIRN conference [45]

Introduction to the 3D Face Recognition

3D face recognition has been deeply investigated in the last decades due to the large number of applications in both security and safety domains, even in real-time scenarios. The third dimension improves accuracy and avoids problems like lighting and make-up variations. In addition, it allows the adoption of geometrical features to study and describe the facial surface.

The dataset at hand is also composed by 7 novel geometrical descriptors which rely on shape and curvedness index and the coefficients of the first fundamental forms. These descriptors have been presented in [46]. The face model is a "mean face" evaluated with a 100 neutral training faces from the Bosphorus database [47]. Formulas and facial mappings of the novel descriptors are shown in Table 1.

In addition to these novel features, other descriptors, presented in [48], have been used including Euclidean and geodesic distances between landmarks, the nose volume, and the shape index [49]. Overall, a set of 11 feature types was generated. All geometrical descriptors, i.e. those reported in Table 1 and the shape index, are adopted in this work in the form of histograms.

| descriptor formula | facial map |
|--|--|
| $E_{den2} = \frac{E}{1 + Z_x^2 + Z_y^2}$ |  |
| $G_{den2} = \frac{G}{1 + Z_x^2 + Z_y^2}$ |  |
| $S_{fond1} = -\frac{2}{\pi} \arctan \frac{E + F + G}{E + G - F}$ |  |

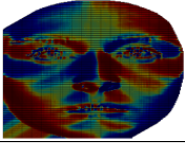
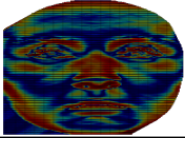
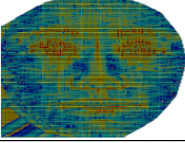
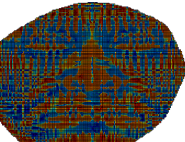
| | |
|---------------------------------|--|
| $\arctan F$ |  |
| $\arctan G$ |  |
| $\log C$ |  |
| $S_{pers} = S[Z + (Z - Z_F M)]$ |  |

Table 2.2: Formulas of new descriptors and respective point-by-point mappings on a Bosphorus facial depth map. Z_x and Z_y are derivatives with respect of x and y , respectively, of the facial surface Z . Complete formulas of E , F , G are given in [46]; C is the curvedness index theorized by Koenderink and vanDoorn [49].

Database Creation and Goals

The global set of features for each face is given by 171 features: 12-bins histograms of 6 novel geometrical descriptor (72 features); 7-bins histograms of the shape index and the personal shape index (14); Euclidean distances between landmarks (62); geodesic distances (22); nose volume (1). The faces in this dataset belong to 62 subjects of the Bosphorus database [47], chosen in such a way to create a different dataset, composed of 7 facial expressions each (Ekman’s basic emotions [50]), meaning 434 faces overall.

The methodology proposed in this work investigates the capabilities of features to discriminate between different individuals, so that the inter-person variability (between subjects) is maximized and intra-person variability (between different expressions of the same subjects) is minimized. Selecting discriminating features for subject identification and recognition is desirable and has been recently addressed in the 3D context [51]. Specific methodologies have been developed to improve stability, interpretability, and predictability [52], but the advances in the 3D domain are still underway.

Data Analysis

This study proposes a novel feature selection technique. It is based on an original self-organizing neural network, the Growing Hierarchical EXIN algorithm (GH-EXIN, [53]),

which builds a hierarchical tree in a divisive way, by adapting its architecture to data. This clustering technique is integrated in a biclustering framework in which face and descriptor (feature) clusterings are alternated and controlled by the quality of the bicluster. In our case, biclustering allows to get meaningful insights of what are the most interesting features by analyzing the subspaces composed of the clustered faces.

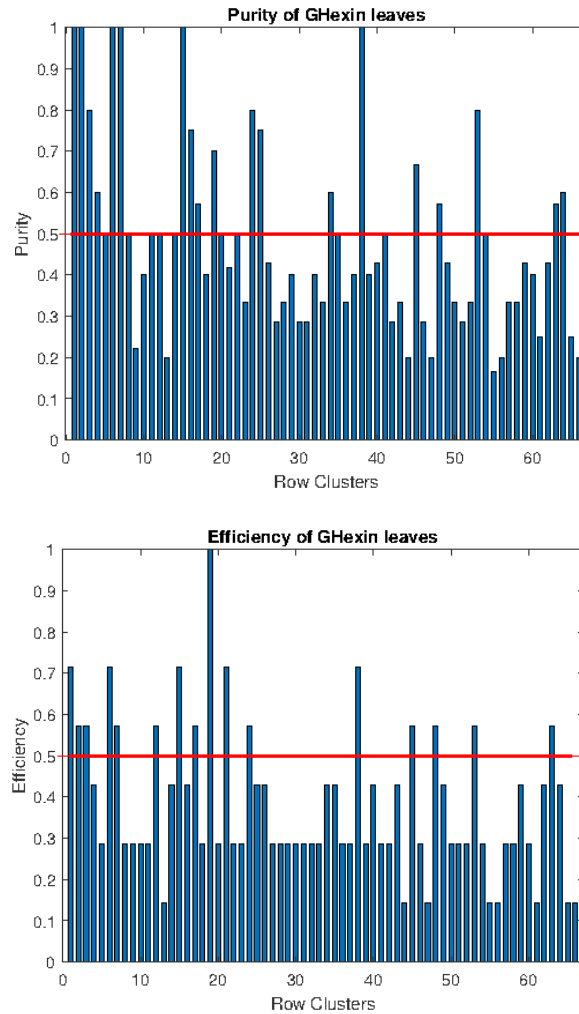


Figure 2.20: Purity and efficiency histograms after the first GH-EXIN

Analysis of the Database

In this work, biclustering is used for estimating the descriptor subspaces in which clusters of faces (mostly same person) are detected. These subspaces are meaningful for 3D detection. It can be argued that their intersection is the core for a correct classification. Here, this intersection is related to the number of times (frequency) a feature is found in each bicluster. At this aim, a two-step approach is proposed.

Firstly, a hierarchical clustering of the samples (faces) is performed by using GH-EXIN.

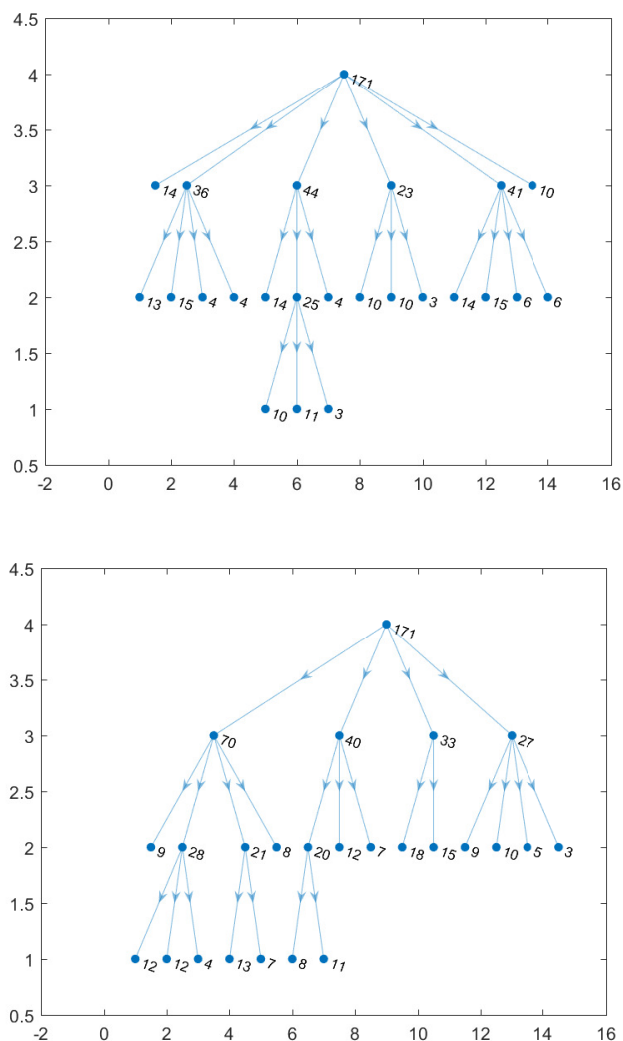


Figure 2.21: GH-EXIN trees of the second step of the biclustering (descriptor clustering)

Then, some leaves are selected according to the criteria of efficiency and purity. They are useful indexes in clustering analysis and give an estimate about the classification accuracy of the algorithm. They both compute the number of elements in a cluster belonging to the same class. While purity compares it with the cardinality of the cluster, efficiency compares it with the cardinality of the class in the whole dataset. A common issue using purity and efficiency separately is that they may select clusters composed of too few or too many elements, respectively. The use of both indexes at the same time avoids a further check on the cardinality of the clusters. This approach allows the selection of those clusters whose Voronoi set contains faces belonging mostly to the same person. As shown in fig. 2.20, only leaves whose efficiency and purity indexes are above 0.5 (red line) are chosen.

The Voronoi sets of the neurons of the best leaves have been found by using all the

descriptors of the faces. In order to find the most meaningful features, the role of samples and descriptors is reversed and, for each leaf, a hierarchical tree is created by GH-EXIN. Fig. 2.21 shows two of these neural structures.

The goal of the second step is the clustering of descriptors for each selected leaf (mostly one person, proportional to its purity). The resulting biclusters identify the best subspace for the faces of the leaf. Indeed, they are identified by lower values of H_{CC} index (an upper threshold of 0.1 is adopted).

The descriptors shown in fig. 2.22 are those that are found at least 6 times in the 12 selected leaves (the best ones of the first clustering). These descriptors have, therefore, the highest discriminative power for their capability in assigning similar values to faces of the same person, which implies to distinguish persons.

Among all descriptors, this study revealed the discriminative capability of LogC. Its bins have been selected until 11 times over 12. As shown in table 2.2, the corresponding figure clearly reveals the intrinsic capability of the descriptor to display the most important trait of a person. As previously said in the introduction, this descriptor has been recently introduced in [46] and further research in the future will be done in order to analyze its importance. Bins of AtanF and AtanG are also selected many times. AtanF is a descriptor capable to highlight all the critical points of the face, like the nasion, and to take into account face asymmetries. AtanG, instead, is used to show the curvedness of the face. In table 2.2, the blue, yellow and red color represent the negative curvedness, the flat surface and the positive curvedness, respectively. The remaining selected geometrical descriptors are only a few Euclidean distances. Nevertheless, they are among the most important in literature, like the distance between the pronasal landmark and the inner eyebrow landmark.

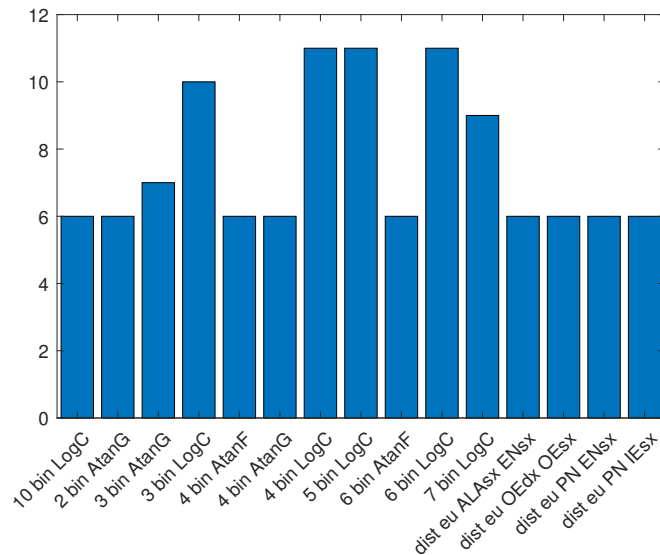


Figure 2.22: Histograms of most discriminative descriptors

Conclusion

This part of the work faces the problem of the discriminative power of the 3D face descriptors in an original way. Indeed, instead of taking into account the classification results, a neural network is created for solving a biclustering problem, by exploiting the purity and efficiency results for the choice of the meaningful leaves. In this sense, it can be stated that the power of the descriptors is integrated in the neural architecture. The consequent approach of selecting the most frequent features in the biclusters results in the assessment of the importance of the novel curvedness descriptors and only of a very few Euclidean distances, in accordance with the analysis in [48], where the low intrinsic dimensionality of the corresponding manifold is determined.

Future works in this field will deal with the analysis of the manifold of the novel descriptor features, and their impact in a new 3D face neural classifier.

Chapter 3

Supervised Neural Techniques

3.1 Introduction

In the following chapters, the supervised path is enriched with advanced and deeper analyzes. In the first chapter a powerful combination of evolutionary algorithm and neural techniques are exploited in order to enhance class prediction models in the context of cancer prognosis. Although powerful and accurate when tested on unseen samples, the neuroevolved model is opaque i.e. it is difficult to understand and interpret from a human point of view. This issue motivates section 3.3, where a very simple neural-based model has been used to investigate the underlying phenomenon more transparently. The analysis of both sections were presented in two different works at the 2018 WIRN conference [54] [55].

3.2 Neuroevolution

Evolutionary Algorithms (EA) are powerful metaheuristic procedures able to explore efficiently the search space of complex (NP-hard or NP-complete) problems finding good approximate solutions. The hyper-parameter optimization of a neural network is a complex problem because there are no polynomial-time algorithms able to solve it. In the following, we exploit EA in order to find satisfying approximate solutions to address this problem.

One of the most widely spread EA is the Genetic Algorithm (GA) [56] [57]. GAs are metaheuristics inspired by natural selection processes. Broadly speaking, GAs involve the evolution of a population of candidate solutions towards better ones. A predefined fitness function evaluates the individual goodness.

3.2.1 Mathematical Model of the Shallow Neural Network

The neural network architecture we used in the following experiments is known as *Adaline* [58] [59]. It has 20023 inputs corresponding to the input features (genes) and one output neuron equipped with a linear output function. The network does not have hidden layers. During forward propagation, the network computes the dot product between the weight vector w and the i^{th} sample $x^{(i)}$ plus the bias b . This corresponds to a weighted sum of

the inputs with bias correction (as in a linear regression model):

$$z^{(i)} = w^T x^{(i)} + b \quad (3.1)$$

$$\hat{y}^{(i)} = f(z^{(i)}) = z^{(i)} \quad (3.2)$$

where w is the weight vector, b the bias, f the activation function and $\hat{y}^{(i)}$ the network output.

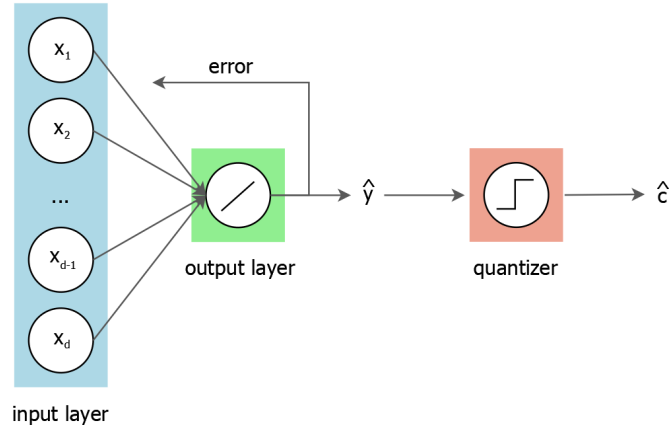


Figure 3.1: Shallow neural network architecture.

Objective Function

The squared error function evaluates the performance of the algorithm on an individual sample:

$$\mathcal{L}(\hat{y}^{(i)}, y^{(i)}) = (y^{(i)} - \hat{y}^{(i)})^2 \quad (3.3)$$

where $y^{(i)}$ is 1 if the i^{th} sample belongs to class 1 and 0 if it belongs to class 0. In order to evaluate the global performance of the classifier, we use a cost function with L2 regularization of the weights. L2 regularization is a technique that applies to objective functions in ill-posed optimization problems [60] [61]. In our case, the proposed neural model is ill-posed, since the solution is not unique and it changes continuously according to initial conditions and randomness in the cross-validation procedure. Appending a term to the cost function that penalizes large weights leads to a reduction of the search space, and the problem becomes less sensitive to initial conditions:

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \|w\|_w^2 \quad (3.4)$$

where λ is the regularization parameter and $\|w\|_w^2$ is the L2 norm of the weight vector. For big values of λ the regularization is stronger, increasing the penalization related to weights. As a result, the weights which are not useful for the purpose of minimizing the MSE (i.e. the first part of the objective function) are shrunk towards zero. On the contrary, for low

values of λ , the regularization effect is weaker¹. In order to provide a quantitative measure of the network performance, we transform the regression outcomes into class labels by using a Heaviside step function:

$$\hat{c}^{(i)} = \frac{d}{d\hat{y}} \max\{0, \hat{y}^{(i)}\} \quad (3.5)$$

and we compute the accuracy as if it were a classification task.

Parameter Optimization

Since the cost function measures the errors in the current predictions, the problem of the learning process is equivalent to the minimization of the cost function. Whereas the training samples are fixed, the cost function depends only on the network's parameters (weights and bias). So, the cost function minimization is equivalent to the optimization of the network parameters. For the following analyses, we use the Adaptive momentum estimation optimizer (Adam). Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments [62]. It is a variant of the classical gradient descent algorithm, designed to combine the advantages of two popular methods: AdaGrad and RMSProp. According to [62] Adam's advantages are that its step-sizes are approximately bounded by the learning rate, it does not require a stationary objective, it works with sparse gradients, and it naturally performs a form of step size annealing. In the context of feed-forward neural networks, the objective function to be minimized is the cost function $J_t(\theta)$, where t denotes the t^{th} epoch and θ is a label for w and b . The authors identify with g_t the gradient, i.e. the vector of partial derivatives of J_t , w.r.t w and b evaluated at epoch t (3.6). This estimate is then used to update two exponential moving averages of the gradient (m_t , (3.7)) and the squared gradient (v_t , (3.8)). The two hyper-parameters $\beta_1, \beta_2 \in [0, 1)$ control the exponential decay rates of these moving averages. High values for β_1, β_2 reduce the time-window size of the moving averages, resulting in low inertial effects and greater oscillations. On the contrary, low values of β_1, β_2 increase the time-window size, providing a stronger smoothing effect. The first moving average m_t is an estimate of the 1st order moment (the mean) of the gradient. The second one instead is an estimate of the 2nd order moment (the uncentered variance) of the gradient. Since these moving averages are initialized as vectors of zeros, the moment estimates are biased towards zero during the initial time-steps (especially when the decay rates are small, i.e. the β s are close to 1). This issue can be alleviated by the bias correction shown in (3.9) and (3.10). The ratio of the two moving averages corresponds to a standardization of the first order moment of the gradient. The network parameters are finally updated by using the classical formula of gradient descent in (3.11). The term ϵ (typically 10^{-8}) ensures that the denominator is always non-zero, avoiding numerical

¹The described shallow neural network model is equivalent to a linear regression model with an L2 regularization of the parameters also known as Ridge Regression [61].

issues.

$$g_t = \nabla_{\theta} J_t(\theta_{t-1}) \tag{3.6}$$

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{3.7}$$

$$v_t = \beta_2 \cdot m_{t-1} + (1 - \beta_2) \cdot g_t \odot g_t \tag{3.8}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{3.9}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{3.10}$$

$$\theta = \theta - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{3.11}$$

The initial conditions are: $m_0 = 0$, $v_0 = 0$ and $t = 0$. Typical values for β s are $\beta_1 \approx 0.9$ and $\beta_2 = 0.999$. Overall, Adam is a very efficient algorithm, requiring very few computations and memory space, which is crucial in our case, given the size of the data set.

3.2.2 Individuals

As previously outlined, the objective of this work is the optimization of a neural network model. In particular, the Adaline model presented in the previous section can be optimized tuning its hyper-parameters. Since each set of hyper-parameters uniquely identifies a neural network, then each neural network can be represented by its hyper-parameters. For this reason, an ordered list of hyper-parameters is an efficient representation of an Adaline model. Having assigned a value to each hyper-parameter from its domain, then the list is called candidate solution or individual. The set of optimized hyper-parameters is composed of:

- the learning rate α ;
- the learning decay rate r ;
- the number of epochs T ;
- the regularization parameter λ .

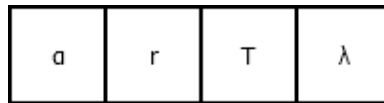


Figure 3.2: The graphical representation of an individual. It is represented as an ordered list of “genetic material”. Each “gene” stands for a neural network hyper-parameter.

3.2.3 Generator

The generator is an EA method devoted to the initialization of new individuals. One of the most common generators is a random generator. In this case each hyper-parameter is sampled by a uniform distribution within a user defined range. The ranges chosen are:

- $\alpha \in [10^{-1}, 10^{-7}]$
- $r \in [10^{-1}, 10^{-7}]$
- $T \in [10, 300]$
- $\lambda \in [10^{-1}, 10^{-8}]$

The uniform distribution guarantees that the initial individuals are sufficiently different from each other. This biodiversity will help the EA search since there is more genetic material available for exchanges.

3.2.4 Evaluator

In order to optimize individuals generation by generation, the next population should be better than the previous one. Therefore, after the generation process, each individual is evaluated in order to estimate the goodness of its genetic material. To do this, an Adaline is built for each candidate solution, i.e. it is set up by using the corresponding hyper-parameters. The learning process is validated through a 10-fold cross validation. At the end of the training process, the average validation accuracy is considered as an estimate of the individual fitness.

3.2.5 Selector

After the evaluation process, the GA selects a random set of individuals from the population and selects a subset of them through a fitness-based criterion. It ranks the randomly selected individuals in ascending order according to their fitness and it picks out some of the best ones. These small sets of individuals are then used to produce the next generation.

3.2.6 Variator

In order to modify and (hopefully) improve the current population, the selected solutions should be slightly modified. At first, they go through a crossover (or recombination) process in which pairs of individuals mix their genetic material to produce two new child solutions.

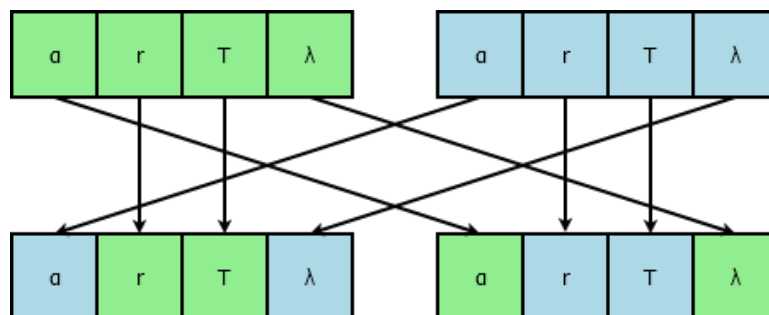


Figure 3.3: During the crossover process, two individuals mix their genetic material in order to produce two child solutions.

Secondly, the child solutions randomly mutate one of their components (hyperparameters).

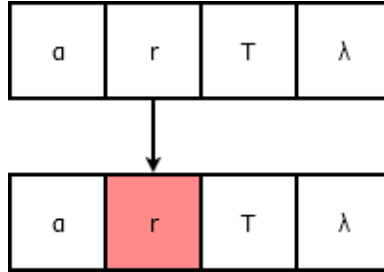


Figure 3.4: The figure shows the mutation process. The individual randomly mutates one of its “genes”.

3.2.7 Replacer

After having generated new candidate solutions, the replacer method selects the best half of old population individuals and the best half of offspring in order to select the candidate solution of the next generation.

3.2.8 Terminator

For the purpose of this work, the entire process is repeated for a certain amount of generations exploring the hyper-parameter space and (hopefully) providing better and better configurations.

Algorithm 2 Genetic Algorithm

- 1: **Input:** DNA microarray and cancer growth targets
 - 2: Generate random individuals
 - 3: **for each** generation **do**
 - 4: **for each** individual **do**
 - 5: Evaluate the individual through cross-validation
 - 6: **end for**
 - 7: Select next generation parents
 - 8: Breed random parents
 - 9: Mutate child individuals
 - 10: Replace old individuals with the new generated ones
 - 11: **go to** next generation
 - 12: **end for**
 - 13: **Output:** best individual
-

3.2.9 Comparison of the Results

The GA set up includes at least the choice of the population size, the maximum amount of generations and the mutation rate. In this work the following choices are made:

- population size: 100
- number of generations: 100
- mutation rate: 20%

In the end the GA provides the list of the candidate solutions of the last (and hopefully the best) generation. The best individual is picked up and the corresponding neural network is evaluated by using an unseen test set. The experiment is repeated 10 times in a 10-fold cross-validation setting. The outcomes are then compared with state-of-the-arts algorithms [63] in Fig. 3.5.

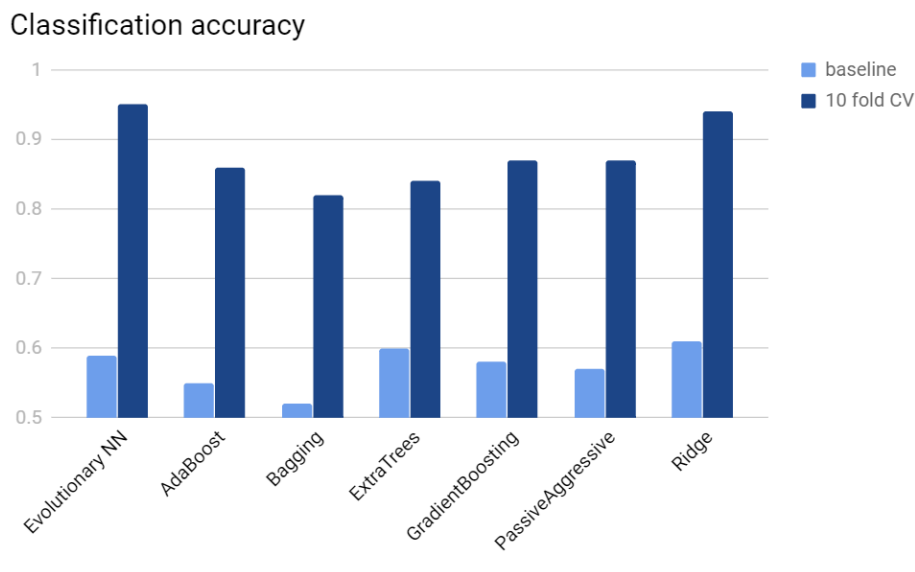


Figure 3.5: The figure shows the accuracy over a 10-fold cross-validation of several classifiers. The baseline accuracy refers to the performance of the classifier for random generated labels.

The proposed algorithm outperformed state-of-the-art techniques. However, linear based techniques, such as Ridge, show great performances, very close to the evolutionary approach. Nonetheless, although powerful and accurate, these algorithms are excessively complex (they are composed of thousands of parameters), resulting in opaque models, difficult to analyze and interpret from a human point of view. This consideration lead us towards a simpler and more transparent approach described in the following section.

3.3 Transparent Neural Based Model for Feature Selection

In this section, we propose a supervised feature selection process based on the recurrent exploitation of the Adaline model described in the previous section. In order to assess the goodness of the proposed approach, we perform a series of cross-validated training of the Adaline model. In particular, at each iteration the neural network is trained 30 times, each of which using a 10-fold cross validation with random folds. The neural network hyperparameters are heuristically fixed to:

$$\lambda = \frac{1}{\#\text{samples}} \quad (3.12)$$

$$\alpha = \frac{1}{\lambda + L + 1} \quad (3.13)$$

according to [64] [65], where L is the maximum sum of the squares over all samples. Since the objective function to minimize contains the L2 norm of the weights (see equation 3.4), weights who are not fundamental for the classification task are shrunk towards zero by the optimizer [61]. Fig. 3.6 and 3.7 show respectively the histogram and the notched box plot of the weights after the training process in the first iteration. It is important to notice that most of the weights are set to zero or are very close to zero. This means that their contribution to the weighted sum in equation (3.1) is almost negligible. Exploiting this result, for each fold we take note of the input features (i.e. the genes) which correspond to weights having an absolute value w_j after a training process:

$$|w_j| > 2\sigma_w \quad (3.14)$$

where σ_w is the variance of the weight distribution (see fig. 3.7). At the end of the 30 iterations, we found that some input features are chosen more frequently than others.

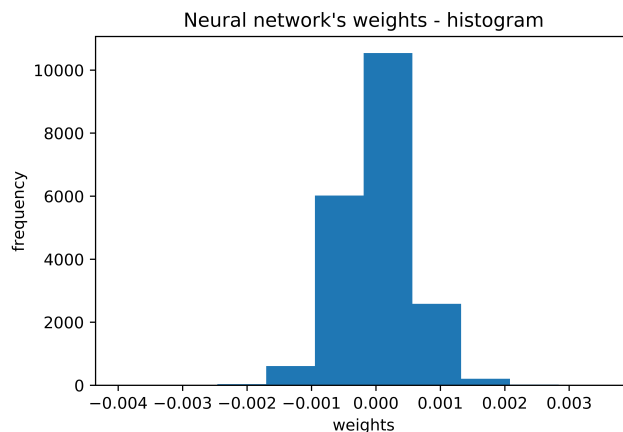


Figure 3.6: An example of histogram of the neural network weights after the first training iteration.

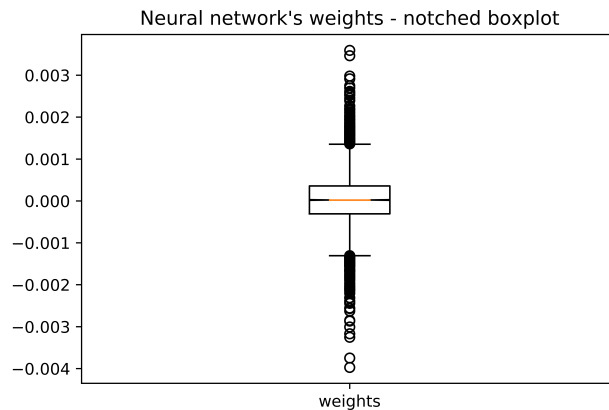


Figure 3.7: An example of notched box plot of the neural network weights after the first training iteration.

Biologically speaking, this result suggests that the information contained in the DNA-microarray related to these genes may be relevant in understanding the cancer resistance to drugs. In order to investigate more deeply the biological phenomenon, we repeat the same experiments, modifying the database by keeping only the most frequently selected features, i.e. those which are selected at least half of the times after the 300 training processes. So, iteration by iteration we gradually reduce the number of input features used to train the neural network. Fig. 3.8 shows for each iteration the number of features used to train the neural network and the corresponding 10-fold cross validation accuracy. The blue bars correspond to the feature selection technique described above, while the violet ones to the ANOVA-F statistic method². Notice that, initially, by using the original data set, the cross-validation accuracy is around 0.7. Such result may have two main explanations. First, the classes are not perfectly balanced, since 66% of samples belong to class 0. Secondly, the high dimensionality of the data may generate a slight overfitting. However, by reducing the input features using the method previously, the cross-validation accuracy raises above 0.9, decreasing progressively as the number of features are further diminished.

It is important to notice that the neural network used as classifier is linear, i.e. geometrically speaking it delimits the input space with a hyperplane in order to classify data. This means that the proposed approach provides better results if the underlying phenomenon represented by the input data set is also linear. The results in fig. 3.8 show how the shallow neural network classifier delivers better results in the 737-dimensional space identified by the proposed feature extraction technique, than in the original 20023-dimensional data set. This may suggest that the underlying biological phenomenon at the DNA-microarray level is more linear in the reduced space than in the original one. Practically speaking, a linear problem is much easier to understand and tackle because the superposition principle

²The corresponding standard deviation is always in the order of few percentage decimals and it is not directly displayed since it is not relevant for the purpose of the discussion. However, you can reproduce the experiment by using our code if you need more precision.

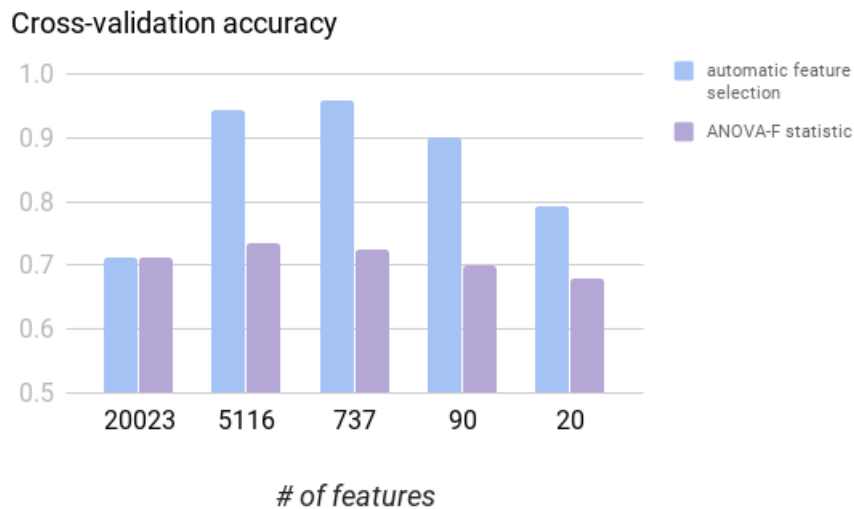


Figure 3.8: Histogram displaying the 10-fold cross validation accuracy at each iteration of the experiment.

holds i.e. the net response caused by two or more stimuli is the sum of the responses that would have been caused by each stimulus individually. Therefore, from a biological point of view, these results may suggest that the above experiment generates sub-spaces of the input features where the cancer resistance to treatments can be studied more easily. In particular, in the 737-dimensional space the biological phenomenon is easier in the sense that it is more linear than in the original space; while in the 90 or 20-dimensional spaces it is easier because, while the classification accuracy decreases, the limited number of genes involved can be more thoroughly analyzed by human experts.

Part IV

Conclusion and Future Developments

Chapter 1

Critical Analysis

In this work we presented a summary of the analyzes performed on the largest CRC xenograft data set available in the academic world. On one hand, these analyzes are relevant because the outcomes could be used to instruct further biological and clinical research. On the other hand, the issues encountered in approaching high-dimensional data has led to the development of novel techniques, which may be exploited in different fields other than biology.

The major machine learning novelty is represented by the creation of GH-EXIN, a new neural-based technique for hierarchical clustering. The comparison with DGOST and GHNG techniques shows how GH-EXIN is typically more efficient, as it reaches similar performances in terms of peak-signal to noise ratio (PSNR) by using fewer neurons. Moreover, qualitative evaluation of the resulting topology shows how GH-EXIN is much more elegant in connecting neurons, providing superior manifold representations. Finally, the restricted number of user-dependent parameters makes the tuning process of GH-EXIN very easy.

The application of the biclustering framework integrated with GH-EXIN on the biological dataset revealed some interesting gene correlation patterns. These results have been submitted to the attention of IRCC doctors, who are currently analyzing them for possible scientific implications.

Other minor novelties have been introduced within both the unsupervised path and the supervised one. They always enhanced effectiveness levels compared to current state-of-art techniques.

The results above are promising and highlight the potential for future work. From the point of view of the biological advances, the outcomes of both the unsupervised and the supervised path are promising yet opaque: while the models can be used effectively, the results are difficult to interpret from a human point of view. As for the unsupervised direction of work, the GH-EXIN neural network resulted to be effective and easy to use as aforementioned; results provided by the biclustering framework, instead, are rather difficult to interpret without a statistical knowledge, since biclusters need several additional tools to be correctly evaluated. As for the supervised direction, the major advances with respect to previous analyses are achieved by exploiting the shallow neural network model. Indeed, the simplicity of such model makes it easier to handle and interpret. On limiting the number of features used, however, the accuracy drops significantly.

Moreover, major limitations of our work directly derive from the analyzed data. On the one hand, the analyzed data represent an estimate of the amount of times each gene is transcribed in a tumor xenograft. However, gene replication does not always result in protein generation. Indeed, this kind of data may not represent cell behaviour correctly. On the other hand, the restricted amount of samples was the most serious issue, since machine learning reliability is directly related to the amount of data provided.

Hence, within the biological domain, future developments will involve the use of up-to-date data (e.g. representing proteins instead of gene expressions) and the integration with other sources of data, such as image samples. Besides, from a machine learning point of view, models easier to interpret may be developed in order to provide more reliable and human-understandable outcomes. Future research in this field will consist in devising new algorithms overcoming the intrinsic weaknesses of machine learning, above all understandability. To this purpose, novel algorithms which integrate classic symbolic artificial intelligence with machine learning techniques seem to be very promising. Further details are given in the next chapter. Both authors will carry on these researches during the doctorate.

Chapter 2

Future Works: Towards Artificial General Intelligence

Strictly concerning machine learning, future works will consist first of an analysis of current state of art of Machine Learning (ML), in order to clearly understand whether it is possible to enrich current techniques capabilities.

A second part of the future researches, instead, will consist in devising new AI algorithms that may go towards an Artificial General Intelligence (AGI). This part will be considered either in case machine learning issues result to be unsolvable, or not, as it seems to be promising per se. A good starting point "may be to integrate deep learning, which excels at perceptual classification, with symbolic systems, which excel at inference and abstraction. One might think such a potential merger on analogy to the brain; perceptual input systems, like primary sensory cortex, seem to do something like what deep learning does, but there are other areas, like Broca's area and prefrontal cortex, that seem to operate at much higher level of abstraction" [16]

Interestingly, symbolic systems and Machine Learning in computer science somehow correspond in philosophy to deductive and inductive reasoning respectively. In fact, deductive reasoning is a process that tries to reach a certain conclusion by applying general rules, narrowing the space of possible conclusions until only one is left. Classic logic and expert systems are strongly based on these principles. Inductive reasoning, instead, is the process in which starting from observations a possible conclusion is derived. The conclusion anyway cannot be considered as undoubtedly truth. All the ML techniques are based on these assumptions.

Summing up, a long-standing controversy exists in literature regarding the role of induction and deduction in reasoning. Nevertheless, none would state that human reasoning is based on only one of them and neither we should suppose it in AI: combining several approaches is the only way to reach AGI.

At the same time, future researches will also study developmental psychology to understand how reasoning processes take place in human brain in order to get useful hints about what are the best ways of replicating them. It is also possible that completely new paradigm, with little in common with existing ones, can be derived from these observations as happened with neural networks.

Bibliography

- [1] National Cancer Institute. What is cancer? <https://www.cancer.gov/about-cancer/understanding/what-is-cancer>, Feb 2015. Accessed on 2018-10-06.
- [2] Alfred G. Knudson. Mutation and cancer: Statistical study of retinoblastoma. *Proc Natl Acad Sci U S A*, 68(4):820–823, Apr 1971. 5279523[pmid].
- [3] J. S. de Bono and Alan Ashworth. Translating cancer research into targeted therapeutics. *Nature*, 467:543 EP –, Sep 2010. Perspective.
- [4] Y. Komeda, H. Handa, T. Watanabe, T. Nomura, M. Kitahashi, T. Sakurai, A. Okamoto, T. Minami, M. Kono, T. Arizumi, M. Takenaka, S. Hagiwara, S. Matsui, N. Nishida, H. Kashida, and M. Kudo. Computer-aided diagnosis based on convolutional neural network system for colorectal polyp classification: Preliminary experience. *Oncology*, 93(suppl 1)(Suppl. 1):30–34, 2017.
- [5] Shelley McGuire. World cancer report 2014. geneva, switzerland: World health organization, international agency for research on cancer, who press, 2015. *Adv Nutr*, 7(2):418–419, Mar 2016. 012211[PII].
- [6] Manuel Hidalgo, Frederic Amant, Andrew V. Biankin, Eva Budinská, Annette T. Byrne, Carlos Caldas, Robert B. Clarke, Steven de Jong, Jos Jonkers, Gunhild Mari Mælandsmo, Sergio Roman-Roman, Joan Seoane, Livio Trusolino, and Alberto Villanueva. Patient derived xenograft models: An emerging platform for translational cancer research. *Cancer Discov*, 4(9):998–1013, Sep 2014. 25185190[pmid].
- [7] Illumina. Beadarray microarray technology. <https://emea.illumina.com/science/technology/beadarray-technology.html>, July 2017. Accessed on 2018-10-08.
- [8] Gartner. Top trends in the gartner hype cycle for emerging technologies. www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/, July 2017. Accessed on 2018-10-08.
- [9] Marvin Minsky and Seymour Papert. *Perceptrons: an introduction to computational geometry*. 1969.
- [10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS’12, pages 1097–1105, USA, 2012. Curran Associates Inc.
- [11] WHO: World Health Organization. Cardiovascular diseases (cvds). http://www.who.int/cardiovascular_diseases/en/, July 2017. Accessed on 2018-08-08.
- [12] WHO: World Health Organization. Cancer. <http://www.who.int/cancer/en/>, July 2017. Accessed on 2018-08-08.

- [13] WHO: World Health Organization. Road traffic deaths. www.who.int/gho/road_safety/mortality, July 2017. Accessed on 2018-08-08.
- [14] Andrew Ng. What artificial intelligence can and can't do right now. <https://hbr.org/2016/11/what-artificial-intelligence-can-and-cant-do-right-now>, November 2016. Accessed on 2018-10-10.
- [15] Francois Chollet. *Deep Learning with Python*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2017.
- [16] Gary Marcus. Deep learning: A critical appraisal. *CoRR*, abs/1801.00631, 2018.
- [17] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dhharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529 EP –, Feb 2015.
- [18] Ken Kanksy, Tom Silver, David A. Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, D. Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. *CoRR*, abs/1706.04317, 2017.
- [19] Barbiero P., Bertotti A., Ciravegna G., Cirrincione G., Pasero E., and Piccolo E. Unsupervised gene identification in colorectal cancer. In *Quantifying and Processing Biomedical and Behavioral Signals*. Springer International Publishing, 2018.
- [20] Sokal R. R. and Michener C. D. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 28:1409–1438, 1958.
- [21] Sokal R. R. and Michener C. D. The comparison of dendrograms by objective methods. *Taxon*, 1962.
- [22] Ward Joe H. JR. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 1963.
- [23] Calinski T. and Harabasz J. A dendrite method for cluster analysis. *Communications in Statistics*, 1974.
- [24] Wegman Edward J. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 1990.
- [25] Jolliffe I. T. *Principal Component Analysis*. Springer Series in Statistics, 2002.
- [26] Demartines P. and Héroult J. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 1997.
- [27] Gower J. C. and Hand D. J. *Biplots*. Chapman and Hall, 1996.
- [28] Usa national center for biotechnology information.
- [29] Barbiero P., Bertotti A., Ciravegna G., Cirrincione G., Pasero E., and Piccolo E. Supervised gene identification in colorectal cancer. In *Quantifying and Processing Biomedical and Behavioral Signals*. Springer International Publishing, 2018.
- [30] Davies D. L. and Bouldin D. W. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1979.
- [31] Tibishirani R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 1996.
- [32] Bishop C. M. *Pattern Recognition and Machine Learning*. Springer, 2016.
- [33] James J Chen. Key aspects of analyzing microarray gene-expression data. *Pharmacogenomics*, 8(5):473–482, 2007. PMID: 17465711.

- [34] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, Jan 2004.
- [35] Cirrincione G. Ciravegna G. Pasero E. Randazzo, V. Inonstationary topological learning with bridges and convex polytopes: the g-exin neural network. In *IEEE - Proceedings of IJCNN 2018 International Joint Conference on Neural Networks*, 2018.
- [36] Yizong Cheng and George M. Church. Biclustering of expression data. *Proceedings. International Conference on Intelligent Systems for Molecular Biology*, 8:93–103, 2000.
- [37] Esteban J Palomo and Ezequiel López-rubio. The Growing Hierarchical Neural Gas Self-Organizing Neural Network. pages 1–10, 2016.
- [38] Latifur Khan and Feng Luo. Hierarchical clustering for complex data. *International Journal on Artificial Intelligence Tools*, 14:791–810, 2005.
- [39] Joaquin Dopazo and José María Carazo. Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree. *Journal of molecular evolution*, 44(2):226–233, 1997.
- [40] Mohamed-Rafik Bouguelia, Yolande Belaïd, and Abdel Belaïd. Online unsupervised neural-gas learning method for infinite data streams. In *Pattern Recognition Applications and Methods*, pages 57–70. Springer, 2015.
- [41] Giansalvo Cirrincione, Vincenzo Randazzo, and Eros Pasero. The growing curvilinear component analysis (gccca) neural network. *Neural Networks*, 103:108–117, 2018.
- [42] Gabriele Ciravegna and Pietro Barbiero. Gh-exin (version 1.0.1). https://bitbucket.org/machine_learning_research/ghexin/src/master/, 2018.
- [43] <http://www.genecards.org/cgi-bin/carddisp.pl?gene=CSAG3>. Accessed: 2017-11-20.
- [44] <http://www.genecards.org/cgi-bin/carddisp.pl?gene=MAGEA2>. Accessed: 2017-11-20.
- [45] Ciravegna G.*, Cirrincione G., Marcolin F., Barbiero P., Dagnes N., and Piccolo E. Assessing discriminating capability of geometrical descriptors for 3d face recognition by using the gh-exin neural network. In *Italian Workshop on Neural Networks (WIRN 2018)*, 06 2018.
- [46] Federica Marcolin and Enrico Vezzetti. Novel descriptors for geometrical 3d face analysis. *Multimedia Tools Appl.*, 76(12):13805–13834, June 2017.
- [47] Arman Savran, Neşe Alyüz, Hamdi Dibeklioglu, Oya Çeliktutan, Berk Gökberk, Bülent Sankur, and Lale Akarun. Biometrics and identity management. chapter Bosphorus Database for 3D Face Analysis, pages 47–56. Springer-Verlag, Berlin, Heidelberg, 2008.
- [48] Spada S. Vezzetti E. Cirrincione G., Marcolin F. Intelligent quality assessment of geometrical features for 3d face recognition. In Morabito F. Pasero E. Esposito A., Faundez-Zanuy M., editor, *Neural Advances in Processing Nonlinear Dynamic Signals*, pages 153–164. Springer, 2017.
- [49] Jan J. Koenderink and Andrea J. van Doorn. Surface shape and curvature scales. *Image Vision Comput.*, 10(8):557–565, October 1992.
- [50] Paul Ekman and Dacher Keltner. Universal facial expressions of emotion. *California mental health research digest*, 8(4):151–158, 1970.
- [51] Goksel Gunlu and Hasan S Bilge. Feature extraction and discriminating feature selection for 3d face recognition. In *Computer and Information Sciences, 2009. ISCIS*

2009. *24th International Symposium on*, pages 44–49. IEEE, 2009.
- [52] Yiyu Guo, Jinsheng Ji, Hong Huo, Tao Fang, and Deren Li. Sip-fs: a novel feature selection for data representation. *EURASIP Journal on Image and Video Processing*, 2018(1):14, 2018.
- [53] Barbiero P., Bertotti A., Ciravegna G., Cirrincione G., and Piccolo E. Neural biclustering in gene expression analysis. In *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, 11 2017.
- [54] Barbiero P., Bertotti A., Ciravegna G., Cirrincione G., Piccolo E., and Tonda A. Dna microarray classification: Evolutionary optimization of neural network hyperparameters. In *Italian Workshop on Neural Networks (WIRN 2018)*, 06 2018.
- [55] Barbiero P., Bertotti A., Ciravegna G., Cirrincione G., Piccolo E., and Tonda A. Understanding cancer phenomenon at gene-expression level by using a shallow neural network chain. In *Italian Workshop on Neural Networks (WIRN 2018)*, 06 2018.
- [56] Zbigniew Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs, 1996.
- [57] Aaron Garrett. inspyred (version 1.0.1) inspired intelligence. <https://github.com/aarongarrett/inspyred>, 2012.
- [58] Widrow B. and Lehr M. A. Artificial neural networks of the perceptron, madaline, and backpropagation family. *Neurobionics*, 1993.
- [59] François Chollet et al. Keras. <https://keras.io>, 2015.
- [60] Ng A. Y. Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *International Conference on Machine Learning*, 2004.
- [61] Hastie T., Tibshirani R., and Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics, 2009.
- [62] Kingma D. P. and Ba J. Adam: A method for stochastic optimization. *International Conference for Learning Representations*. 2017.
- [63] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [64] Schmidt M., Le Roux N., and Bach F. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 2013.
- [65] Defazio A., Bach F., and Lacoste-Julien S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in Neural Information Processing Systems*, 2014.