Escuela Técnica Superior de Ingenieros Informáticos

Universidad Politécnica de Madrid

Visualization and interpretation in large Bayesian networks

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial

AUTHOR: Sergio Paniego Blanco
SUPERVISORS: María Concepción Bielza Lozoya
Pedro María Larrañaga Múgica

July 2019

Acknowledgment

I would like to express my gratitude to my supervisors, Concha Bielza and Pedro Larrañaga for the opportunity provided. Their guidance, their valuable and constructive suggestions during the development of this project need to be remarked too.

My sincere thanks also goes to Mario and Niko, fellow labmates at Computational Intelligence Group at Universidad Politécninca de Madrid, for their advice and assistance throughout the project. I would also like to extend my thanks to the rest of the people working at the laboratory.

I wish to thank my parents for the extraordinary effort conducted to give me the opportunity to receive a good education. Finally, a special thanks to my friends for their support and encouragement throughout my study.

Resumen

Dentro de la Inteligencia Artificial, el campo del aprendizaje automático es un subcampo que estudia desarrollar técnicas que permiten inducir conocimiento a partir de un conjunto de datos de entrada. En los últimos años, se ha visto un auge en este campo, debido en parte a la posibilidad de acceso a grandes cantidades de datos y al aumento en la capacidad de computación. Uno de los problemas más acuciantes dentro de este campo es la baja interpretabilidad de la mayoría de los modelos más utilizados, siendo estos sistemas típicamente considerados cajas negras, lo que vendría a decir que son sistemas a los que se les proporciona una entrada y dan una salida pero sin que se pueda llegar a concretar el razonamiento que hay detrás de dicha decisión. Uno de los modelos de aprendizaje automático que sí que permite la interpretabilidad del razonamiento resultante son las redes Bayesianas. En los últimos tiempos, se está viendo una tendencia investigadora que mira hacia la interpretabilidad de los modelos de aprendizaje automático, impulsado en gran medida por el hecho de que para que se puedan implementar estos modelos en sistemas en la vida real, es necesaria una interpretación y explicación del razonamineto que hay detrás para garantizar su buen funcionamiento, la confianza por parte de los usuarios en el sistema y el hallazgo de posibles responsabilidades en caso de fallo.

En el presente trabajo, se busca estudiar la visualización e interpretación de este modelo de aprendizaje automático para poder extraer conocimiento a través de los resultados devueltos por el mismo. Este estudio surge para poder aplicarse al proyecto *Neurosuites*, concretamente a su análisis de redes de regulación genética que relaciona genes según sus relaciones y permite analizar estas relaciones en diferentes circunstancias. Gracias a esta visualización, un experto en la materia podría utilizar la herramienta para, dado un caso de estudio donde se tengan las alteraciones genómicas de un paciente, extraer una predicción del impacto de éstas respecto a las enfermedades con las que están relacionadas. El problema de visualización en este caso de estudio, viene en parte por la complejidad de la red generada, con de decenas de miles de nodos y conexiones, de las que es muy complicado extraer conocimiento solamente con las técnicas que actualmente son el estado de la cuestión en visualización e interpretación de redes Bayesianas. El uso de estas grandes redes es posible gracias a los últimos avances en potencia computacional y especialmente en las GPUs.

Teniendo en cuenta esto, se ha tomado como punto base el estado de la cuestión y se ha buscado mejorar y adaptar las técnicas de visualización para poder extraer conocimiento, buscando que el estudio sea interesante no solamente para este caso concreto de aplicación sino para el estudio de la visualización e interpretación de modelos de aprendizaje automático, especialmente para redes Bayesianas.

Abstract

Within artificial intelligence (AI), machine learning is a sub-field that investigates the development of techniques that enable the induction of knowledge from a set of input data. In recent years, this field has experimented an impressive period of development, in part thanks to the access to big data sets and the increase in the computational power. One of the most pressing problems in this field is the low interpretability of the majority of the most used algorithms, usually considering these systems to be black boxes. A black box is a system that given some input data, it processes them and produces and output result but with the problem that there is no good understanding of how the system has reached that result or is untraceable. One of the machine learning algorithms that lets a user to interpret and visualize the reasoning followed to reach a conclusion is Bayesian networks. In recent times, a research trend is observed that investigates the interpretation and explanation of machine learning models, mainly driven by the fact that in order to make use of these models in real-world risk environments, it is necessary an interpretation and explanation of the reasoning behind a certain result so its proper functioning is guaranteed and possible responsibilities in case of failure are found.

In this project, the visualization and interpretation of this machine learning model are studied to be able to extract knowledge through the results that it returns. This investigation is created to be applied in *Neurosuites* project, specifically in its analysis of gene regulatory networks that connects genes based on their relationships and enables the investigation of how these relationships behave in different situations. Thanks to the visualization, a human expert could use the application to a given case study. The visualization problem in this case study, appears partly because of the complexity of the network generated by the genome data, creating tens of thousands of nodes and connections between them. Having such an enormous network, it is really complicated to extract knowledge using only the already known techniques that are today the state of the art in visualization and interpretation of Bayesian networks. The use of these huge networks is possible in part thanks to the latest developments in computational power, especially in GPUs.

Considering this situation, the starting point for this project is the current state of the art in Bayesian network visualization and interpretation. The work presented in the project implements improvements and adaptations of the visualization techniques in order to extract knowledge, trying to make this project's improvements useful not only for the case study but for the general investigation in visualization and interpretation in machine learning models and specially for Bayesian networks.

Contents

Acknowledgment **iii**

Resumen **v**

Abstract **vii**

1 Introduction and motivation **1**

1.1 Interpretable machine learning models and its rise in research 1

1.1.1 The black box problem 1

1.1.2 The emergence of Explainable Artificial Intelligence (XAI) 2

1.2 State of the art in visualization and interpretation in Bayesian networks and its application in large and complex networks 4

1.2.1 Seeking an improvement in the interpretability of complex networks 5

1.3 Machine learning application in neuroscience 5

1.4 Contributions 6

2 Visualization and interpretation in Machine Learning **9**

2.1 Foundation 9

2.2 Observed problem 10

2.3 Approaches in solving this problem 11

2.4 Bayesian networks as an interpretable model 14

3 Review on visualization and interpretation in Bayesian networks **17**

3.1 Foundation 17

3.1.1 D-separation 19

3.1.2 Parametrization 20

3.2 Inference in Bayesian networks 20

3.3 Learning Bayesian networks 20

3.3.1 Structure learning 21

3.3.2 Parameter learning 21

3.4 Bayesian networks applications 21

3.5 Visualization in Bayesian networks 22

3.5.1 Software 23

3.5.2 Problems 24

4 Visualization in Bayesian network modelling 27

4.1 Computation of the importance of nodes 27

4.1.1 Degree of importance of a node based on its number of neighbors 28

4.1.2 Betweenness centrality 28

4.2 Community detection 29

4.2.1 Louvain modularity 29

4.3 Node parameters visualization 30

4.3.1 Visualization of discrete probability distribution 30

4.3.2 Visualization of Gaussian probability distribution 31

4.4 Setting evidences and updating probability distributions 32

4.4.1 Individual evidence 33

4.4.2 Evidence on a set of nodes 33

4.5 Groups of nodes creation 33

4.6 Network structures comparison 34

4.6.1 Applied metrics 34

4.7 Change in probabilities in huge networks visualization 37

4.8 D-separation visualization 38

4.9 Probabilistic clustering visualization. Gaussian mixture 38

5 Application of the proposals in visualization and interpretation of Bayesian networks in *Neurosuites* 41

5.1 *Neurosuites* 41

5.1.1 Project architecture 41

5.1.2 Bayesian network functionality 43

5.2 Visualization in *Neurosuites* before this project 44

5.3 Gene regulatory network problem 45

5.4 Application to the visualization of huge networks and results 45

5.4.1 Nodes' importance in the network 46

5.4.2 Community detection 49

5.4.3 Parameter visualization 51

5.4.4 Evidence setting and probability updating 54

5.4.5 Group creation 56

5.4.6 Network structure comparison 58

5.4.7 Probability update visualization in huge networks 62

5.4.8 D-separation 64

5.4.9 Probability clustering visualization. Mixture of Gaussians . . . 65

6 Conclusions and future work 69

6.1 Conclusions 69

6.2 Discussion and future work 70

6.3 Personal remark 73

Bibliography 73

Appendix **81**

A Project management **83**

A.1 Timeline and methodology 83

A.2 Tools used and documentation 84

List of Figures

1.1 Scheme of a black box machine learning model work-flow. 2

1.2 Explainability vs. Prediction accuracy level of today's models of machine learning where it is highlighted that the level of precision is already high but the level of transparency is still small. Source: Gunning (2017) 3

2.1 Scheme of current machine learning answers and how the new models should address the problem. On today's models, the answer is not usually easily interpretable so explainable models are needed. Source: Gunning (2017). 11

2.2 Scheme of current and future state in learning techniques as regards performance versus explainability. Each dot represents a learning technique. Source: Gunning (2017) 12

3.1 Bayesian network example. The graphical representation has two elements, the direct acyclic graph (DAG) and the CPDs. Adapted from: P. Murphy (1998). 18

3.2 D-separation triples relation. Black nodes represent observed nodes and white ones are unobserved nodes. Source: Koller and Friedman (2009). 19

4.1 Scheme of steps followed in the Louvain modularity algorithm. Source: Blondel et al. (2008) 31

4.2 Example of the visualization of a discrete probability distribution. . . 31

4.3 Example of the visualization of a Gaussian probability distribution . 32

4.4 Confusion matrix. 35

4.5 Example of ROC and Precision-Recall curve. Source: Chicco (2017) . 37

4.6 Gaussian mixture example. Source: Mason (2012) 39

5.1 *Neurosuites* architecture. 42

5.2 New options added to the drop-down menu. 46

5.3 Importance degree algorithm with 200 neighbors as threshold and without including neighbors in the visualization. Five nodes have more than 200 neighbors, the bigger ones. 47

5.4 Importance degree algorithm with 200 neighbors as threshold and including neighbors in the visualization. The same five nodes are represented with a big size but the color now is the same. 48
5.5 Betweenness centrality algorithm visualization over a network. The size of the node represents its importance in the network. 49
5.6 Buttons added to the drop-down menu for the new options of highlighting important nodes and communities. 49
5.7 Louvain community detection algorithm visualization over a network. Nodes with the the same color represent a group. 50
5.8 Selection of groups created by the Louvain algorithm. 50
5.9 Visualization of a group created by the Louvain algorithm. In this case, group 3 is selected and its nodes are highlighted. 51
5.10 Discrete parameter visualization for a node in the network. 52
5.11 Discrete parameter visualization when setting an evidence. 52
5.12 Continuous parameter visualization for a node. In the right image an evidence is set with a value of three for the node. 53
5.13 Continuous parameter visualization for a node with the difference between prior and posterior probability distributions. 54
5.14 Nodes selection from all the available nodes. 55
5.15 Group options with ability of setting the evidence or clear it. 55
5.16 Evidence set by using the number of standard deviations from the mean. 55
5.17 Result of evidence setting over the nodes. The red nodes indicate those with evidence set. 56
5.18 Drop-down node selection menu where the nodes that are part of a group are selected to create a group. 56
5.19 Modal window where group name and color of the nodes can be selected when creating the group. 57
5.20 Group and category selection. Once the category is selected, the evidence can be set or cleared. 57
5.21 Example of visualization of a group. Group nodes and connections between them are displayed. 58
5.22 Structure 1 and 2 upload sections with the buttons to select the corresponding file or with an area to drag and drop them. The screen on the right appears when *Add additional structure* is clicked. 59
5.23 Small example of performance for visualizing two structures. First and second structures represent Structure 1 and Structure 2. In the picture on the right, black color represents Structure 1, blue Structure 2 and green represents an edge that is part of both structures. 59
5.24 Menu for selecting structure comparison. Four options are available: Both, Structure 1, Structure 2 and Common edges. 60
5.25 Selection of Structure 1 for the small example. The black edges shape Structure 1 and the white edge represents an edge that is not in the current featured structure but it is part of another one. 60

5.26 Visualization of 2 structures with a real gene regulatory network. . . 61
5.27 Table of metrics to compare the structures shown. The metrics for the two different structures are shown and options to export the data generated to a file are also available. The first image is the complete table and the second and third are a zoomed version. 62
5.28 Evidences effect in the node sorted by the Kullback–Leibler divergence. The node is represented by its name and the amount of change in the distribution computed using the Kullback–Leibler divergence. . 63
5.29 Drop-down menu for the selection of groups based on that KL-divergence after some evidences have been set. 63
5.30 Visualization menu with options for zooming on nodes depending on the amount of change they have experimented. Options for KL divergence, mean and standard deviation. 64
5.31 D-separation menu in *Neurosuites* 64
5.32 D-separation notification in *Neurosuites* that shows if the d-separation is confirmed and the nodes are conditionally independent. 65
5.33 Gaussian probability distributions for a certain node. 66
5.34 Gaussian mixture. 66
5.35 Gaussians mixture applied over four clusters. 67

A.1 Approximate timeline followed during the project 84
A.2 General *Neurosuites* view 85
A.3 Bayesian network visualization area in *Neurosuites* 86

List of Tables

2.1 Measure of Explanation Effectiveness for a machine learning model. Source: Gunning (2017). 13

4.1 Changes in visualization and interpretation over Bayesian network studied. 27

5.1 Functionalities of *Neurosuites'* Bayesian network visualization section. 44

Chapter 1

Introduction and motivation

1.1 Interpretable machine learning models and its rise in research

Machine learning is a field within AI. The main purpose of study in this field is the development of algorithms that allow machines to learn automatically from a bunch of input data, equipping them with the competence of performing predictions or making decisions without someone directly programming this behavior. Machine learning field is itself divided into a set of sub-fields like supervised classification or clustering as it can be reviewed in Larrañaga (2018). In recent years, there has been a boom in research and the actual commercial use of this area of knowledge, largely thanks to the increased computing capacity experienced and the availability of a bigger amount of data to work on. Real life uses of machine learning include market analysis (Chang and Tian (2015)), the recognition of celestial bodies in astrophysics (Zharkova and Jain (2007)), commercial recommendations based on user behavior (de Campos et al. (2006)), textual and voice recognition (Zweig (1998)) or biometric recognition (Ma and Guo (2014)), to name a few examples, although the number of fields where it is starting to be used is increasing.

The most common form of machine learning algorithm taxonomy divides them into supervised learning, unsupervised learning and reinforcement learning algorithms. The present work focuses on a series of cross-cutting problems to the whole spectrum of automatic learning algorithms, their visualization, interpretation and explanation. This problem is commonly known as the problem of *the black box (Rudin (2018))*.

1.1.1 The black box problem

One of the most important downsides of the nowadays most commonly used machine learning techniques is their lack of interpretation and explanation capabilities about why a certain answer is given. Figure 1.1 presents an scheme of how the work-flow of a black box machine learning model looks. On this kind of models, the model takes some input data, it does some computations over that data and outputs a result but this result does not come with an explanation of how the model has reached such a conclusion and it is usually really complicated to understand the reasoning behind the answer.

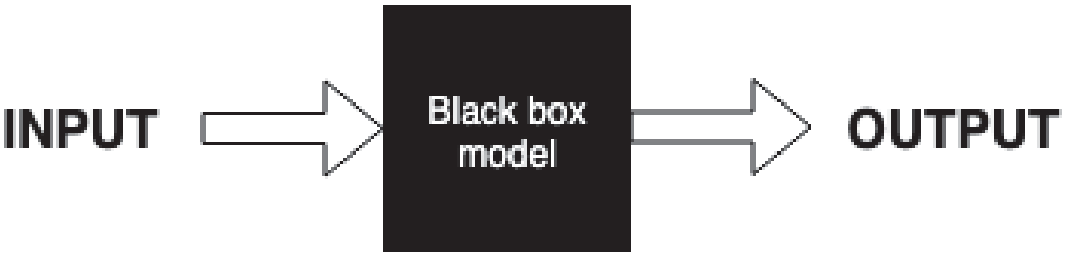


Figure 1.1: Scheme of a black box machine learning model work-flow.

Machine learning models can be classified by their level of interpretability into different groups. On a white box model, the explanation of how it works and reaches a conclusion is clear. Opposite to these models are black box models, where the capability of explaining the behavior of the model is low.

On this project, the application studied, *Neurosuites*, highlights this weakness of some of the most cutting-edge learning systems used today. Neurosuite is an online web application that provides a series of computational neuroscience tools. The fact that machine learning lacks interpretation and explanation capabilities makes it lose possible fields of application. Working in that direction is necessary in order to make AI usable in real world applications, otherwise it will be condemned to stay in laboratories or academia.

1.1.2 The emergence of Explainable Artificial Intelligence (XAI)

Lately, the trend in machine learning research has seen a boom in what is known as Explainable Artificial Intelligence (XAI) (Nott (2017)). This new trend of reaseach aims to makeAI systems more transparent, giving details of the reasoning behind the decision made by a specific system, improving the current techniques and/or generating new ones that are innerly interpretable.

XAI usually refers to the construction of models that can extract the knowledge from already created models, whereas interpretable AI focuses on creating new models that are explainable. Both terms are present in the project and the difference between them is not generally clear in literature.

In Figure 1.2, this trend can be seen graphically, where it is reflected that a high level of accuracy has been reached in some machine learning algorithms but as a down-side these models have usually the problem of being opaque to the view of a user who tries to understand the reasoning made. Learning techniques are divided in different groups that are interconnected based on their relationships. Neural nets contains deep learning. This group is connected to statistical models and graphical models. In statistical models, stochastic and-or-graphs (AOGs) and support-vector machines (SVMs) are found. This group is also connected to graphical models. In graphical models, three sub-groups are contained and interconnected, Bayesian belief nets, statistical relational learning (SRL), conditional random fields (CRFs), hierarchical Bayesian networks (HBNs), Markov logic networks (MLNs) and Markov models. Graphical models is connected to the two previously mentioned groups and to ensemble methods and decision trees. These two groups contain Random Forests.

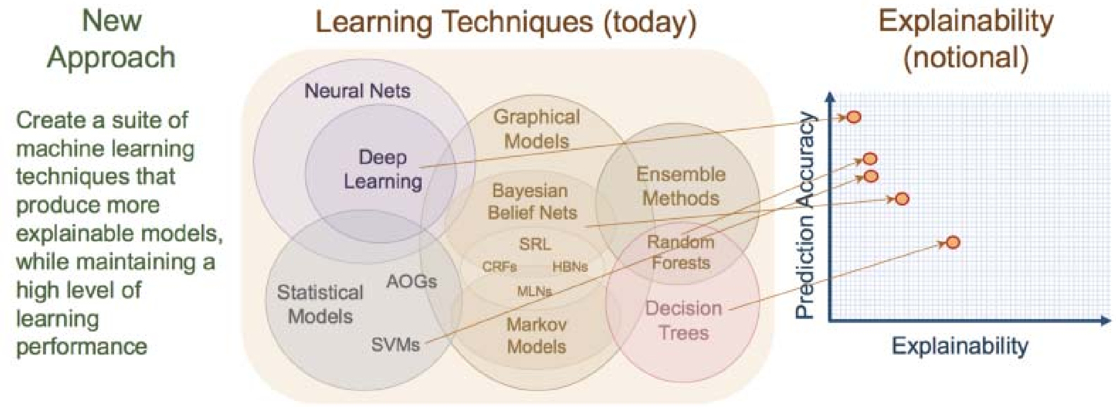


Figure 1.2: Explainability vs. Prediction accuracy level of today's models of machine learning where it is highlighted that the level of precision is already high but the level of transparency is still small. Source: Gunning (2017)

The rise of this line of research can be seen clearly by the amount of resources that are being used to study this field. As a clear example of this situation, the initiative of the Defense Advanced Research Projects Agency (DARPA) can be named. This agency is behind a plan (Gunning (2017)) that involves several research organizations and universities and whose goal is to develop new techniques that guarantee that the machine learning techniques produce outputs with a comprehensible explanation for a human.

The importance of this explanation lies in being able to design fair algorithms, identify possible biases on the system's training data set and ensure that the system operates according to what was sought in the beginning. The current trend followed in research is more focused on explaining deep learning systems as in Gilpin et al. (2018), where the authors seek to establish a base for what to consider a good interpretation and how to create it, reviewing existing literature and proposing a possible line of research that could serve as a guide. Another example would be Samek et al. (2017), where the reasons why an interpretable system of AI is necessary are exposed:

- Verification of the system. Check that the system is doing what the developer wants to do.
- Improvement of the system. If the system is transparent, it is easier to find weakness and correct them.

- Learning from the system. Possible patterns found by the system could be hidden to humans

- Compliance to legislation. Assignment of responsibility when the system makes a wrong decision.

There is another line of research contrary to this (Rudin (2019)), that shows the characteristics of other machine learning methods that are now receiving less attention and criticizes the position expressed in Gunning (2017), explaining that the idea of XAI presented is wrong. This topic is presented in depth in Chapter 2.

Throughout this work, this purpose of achieving an explainable and interpretable AI is followed, trying to give understandable answers to a non-expert user. For this purpose, Bayesian networks with their graphical modelling capabilities are used. These networks are composed of directed acyclic graphs (DAG). This type of model offers greater interpretability of the resulting output than other models thanks to its clear graphical model interface. The purpose of the project is to provide this type of model with a visualization that adds value, enabling a user to interpret the results obtained in a comprehensible manner for the case study.

1.2 State of the art in visualization and interpretation in Bayesian networks and its application in large and complex networks

Bayesian networks are a kind of machine learning algorithm. This algorithm models graphically probabilities over a set of variables. One of the points that is studied in this project is the review of literature regarding visualization and interpretation in Bayesian networks and the analysis of these proposals to study their application to the case study of this project. As studied in Chapter 3, where a complete review of the literature in visualization and interpretation in Bayesian networks is conducted, the majority of literature related to this topic presents ideas that are only suitable for small or medium-size networks. In this project, the networks that are used are enormous, with tens of thousands of nodes and edges, which usually makes the already known methods unsuitable for the study.

In recent years, there has been a growing interest in the study of huge and complex networks in form of graphs, also known as network science (National Research Council (2005)). This science studies networks like the ones developed from social networks interactions and some of the methods proposed, in order to improve the visualization and interpretation of Bayesian networks are related to this area. Thoughout this project, it is considered if any of the methods proposed in that area could be extrapolated to gene regulation networks with Bayesian networks, as they are the object of study in this project.

Another interesting point to be considered is that most studies and commercial applications available give more importance to the use of discrete variables, but in this project, the variables follow continuous Gaussian distributions. With the practical work presented throughout this document, an answer to the needs of updating the visualization and interpretation in Bayesian network is given, always keeping in mind that this visualization has to serve as an explanation to the output given by the network, as it is the goal pursued by XAI.

1.2.1 Seeking an improvement in the interpretability of complex networks

One of the objectives of this project is to show the importance of the machine learning techniques already present in the literature and that can serve as an answer to the need for interpretability that is necessary for this algorithms to be actually useful in a future society. Having an algorithm that provides a graphical view of the model as Bayesian networks, an advancement in this graphical power is seeked. The improvement in the computing capabilities available in part thanks to the emergence of GPUs makes this project suitable because tens of thousands of nodes and edges can be shown in a computer in real time without much effort and the viewing possibilities are open to a wider range of options.

1.3 Machine learning application in neuroscience

Neuroscience is a branch inside science that studies the nervous system. The case study presented in this project explores gene regulation networks, a field that is within neuroscience. The amount of data available from this field has experimented a high increase lately, and with the arise of greater computational power, machine learning techniques can be used to extract knowledge from this data.

In these networks, each node represents a human gene and the connections between them represent regulatory relationships. From the data collected in experiments, the relationships between a series of diseases and genes that might be related to them are extracted. Many different computational approaches are used in the study of gene regulatory networks, for example, Bayesian networks. Using Bayesian networks and these sets of data, studies can be carried out to clarify the level of influence of a set of genes based on their level of expression in the rest of the genetic regulatory network. The gene expression is the process where a specific part of the genome is read and the corresponding protein is synthesized.

In the case study of gene regulatory networks, the number of nodes is in a magnitude of tens of thousands of nodes, with each node representing a human gene and having a linear Gaussian distribution over the level of expression of the gene associated, representing the probability distribution. Having this enormous network, Bayesian networks can be really helpful because they are readily interpretable, having each node representing the expression associated with a node and the edges showing the conditional dependencies between them.

This type of research studies could be useful for example in pharmacology. Knowing the set of alterations in the genome that a certain medicine produces, it is possible to study the effects that it could cause in a human. This means that having the information about the intake of a certain medicine, it can be computed whether it would increase the risk of suffering from some type of specific disease related to the genome.

The present study aims to improve the visualization and interepretability of Bayesian networks in order to make this type of studies viable, giving the needed tools of visualization to the researchers.

1.4 Contributions

The main goal in this project is to conduct and study current trends on interpretability and visualization in AI, especially in Bayesian networks, and using this information, improve the visualization and interpretation of Bayesian networks with a special focus on gene regulatory networks. The structure of a gene regulatory network contains tens of thousands of nodes and edges, what makes this project complex and shows the need for techniques sometimes beyond the state of the art in visualization of Bayesian networks.

The following contributions have been added during the project:

- **Study of the rising trend of an interpretable and explainable AI through literature.** In recent years, a great development in machine learning techniques has been observed and the progress in this field covers headlines every week. This chapter has reviewed the literature related to XAI, explaining the problem with black box models and the new tendencies proposed with the critics around them. One of the key aspects that AI has to overcome to be able to become a determinant in risk fields, where today there is no room for its application, is the fact that it must provide explanations regarding the answers offered by the system. This point is developed in **Chapter 2**.

- **Review of the state of the art of visualization and interpretability in Bayesian networks.** In this section, a complete study of the current state of the art is conducted, going through the academic literature available regarding visualization and interpretation in Bayesian networks, as well as the available commercial applications. Apart from this, a brief review of key Bayesian network concepts needed for this project is also provided. This review covers **Chapter 3**.

- **Visualization in Bayesian networks implementing some improvements.** A series of improvements to the current state of the art in terms of visualization and interpretation in Bayesian networks are presented, especially considering the use of these improvements in large networks such as the one studied *Neurosuites*, the gene regulatory network, where these changes are implemented and their behavior is studied. **Chapter 4** of the document deals with these issues.

- **Application to the real-world scenario of gene regulatory networks implemented in *Neurosuites* and the study of gene regulatory networks.** Chapter 5 covers the application in a real-world scenario of the visualization and interpretation techniques introduced in the previous chapter. *Neurosuite* is an online web application that provides a series of computational neuroscience tools. One of the tools is a Bayesian networks application focused on the study of gene regulatory networks where the user has access to the typical Bayesian network functionality offered in the main software (upload a data-set, learn the structure and parameters, visualize the network, set evidences and update probabilities). The introduction of this software application and the actual development and results of the improvements made on visualization and interpretration in Bayesian networks is covered in **Chapter 5**.

Chapter 2

Visualization and interpretation in Machine Learning

This chapter's goal is to study the unsolved problem of visualization and interpretation in machine learning. An introduction to the problem is first given, then current state of the art and the paths that have been considered by the researchers are studied. Finally, Bayesian networks are introduced as a possible path to be followed in research for visualization and interpretation in machine learning and how this is addressed in this project.

2.1 Foundation

Explanation and interpretation aim to develop a fundamental and scientific understanding of the world, of why things happen. An entire field of study tries to understand these questions and give answers based on philosophy (Woodward (2017)). The question has long been studied and is still an active field of research.

Explanation in AI is usually focused on researchers' intuition instead of on principles that for example are studied in philosophy, psychology and cognitive science. These disciplines study how a person works with explanations, employing cognitive biases and social expectations in the process of explanation. Miller (2017) tries to link explanation in AI to the rest of the fields stated, from the foundation of what is an explanation to how an idea is developed.

Explanation in AI's purpose is to address how a certain answer/conclusion is reached and why this conclusion is convenient for a given answer. The development of AI has been focused since its beginning on decision-support systems that can give an opinion to an expert that can use it to corroborate a hypothesis instead of being intended to replace the expert. This statement means that the expert is indeed necessary in the process in order to extract the knowledge. An early example of this reality is Dannenberg et al. (1979), where computer decisions are used by medical students to evaluate their conclusions and update them based on what the computer has outputted. Another examples in medicine are Bleich (1972), where the explanation is textual, or MYCIN, presented in Buchanan and Shortliffe (1984), one of the most remarkable accomplishments in early AI steps.

2.2 Observed problem

Several concerns have raised on AI due to some failures in the development of the learning systems that have been featured in the main newspaper covers, with examples in the news about bias in the prediction of future criminals (Angwin et al. (2016)), bias in a recruiting tool (Dastin (2018)) or people incorrectly denied parole (Wexler (2017)).

These issues have also increased the interest in ethics concerning AI (Bostrom and Yudkowsky (2014)), which is now a trendy topic in research, where ideas like how an AI system should be developed in order to avoid biases against some collectives are covered. AI systems need to be developed in order to be understandable by the users, otherwise they will not trust them.

As the power of AI systems increases, the need to give users interpretation and visualization tools in order to understand the behavior of the system gains importance. This fact has been studied in investigations like Stubbs et al. (2007), where the researchers explain that as the level of autonomy of an AI system (in the study of a robot) increases, the concerns about why some action is taken also increase.

This growth in AI power is in part thanks to the availability of large data sets and the increase in computational power, but the model's explainability and visualization have not been developed at the same rate.

Black box models are how some machine learning algorithms are called, where the user cannot easily judge an answer given by the system because the process followed cannot be understood by a human. Guides to know how to face this problem have been developed (Guidotti et al. (2018)). This issue makes AI untrustworthy in certain situations. With these learning techniques, the model is created using some training data. Once the model is ready to be used, the user inputs some data and the model outputs some prediction, as shown in Figure 1.1. The problem here is that this model does not give any explanation on why this prediction was outputted.

In Figure 2.1 a scheme of this situation is shown. Today, the learning systems give no explanations to the user on the reasoning behind an answer, although they are highly accurate. In order to evolve in this field, explainable models are needed, as it is also shown in Figure 2.1, where the user is given an explanation of why an answer is given, so he/she can trust the system.

It is also a legal problem because there is also a need to understand how a certain system has been built in order to find responsibilities in a situation where the system does something illegal. This is studied in Kroll et al. (2016), where they try to update the legal standards to a present and future where machine learning systems make decisions.

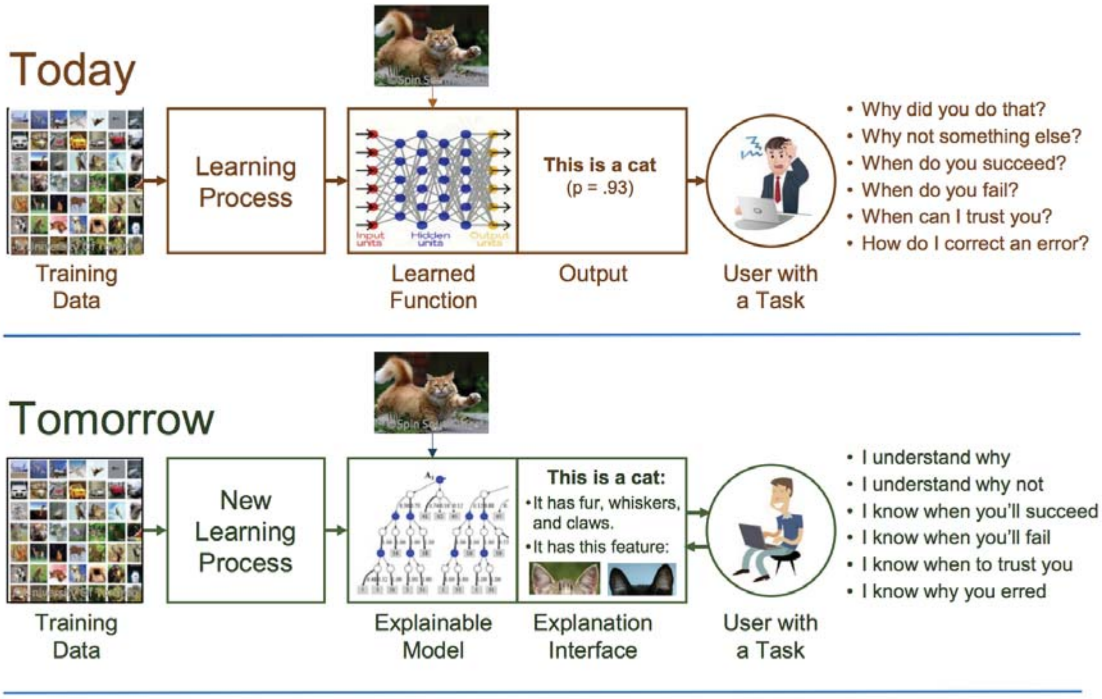


Figure 2.1: Scheme of current machine learning answers and how the new models should address the problem. On today's models, the answer is not usually easily interpretable so explainable models are needed. Source: Gunning (2017).

2.3 Approaches in solving this problem

The problem of explanation and interpretation in Artificial Intelligence, usually also called XAI, has already been tried to be solved. Some interesting approaches have been proposed and many conferences have already been organized on this topic, as for example AIMLAI-XKDD [1] (Advances in Interpretable Machine Learning and Artificial Intelligence & eXplainable Knowledge Discovery in Data Mining) which is defined by its organizers as a workshop with a focus on advancement of explainable, transparent, ethical and fair data mining, machine learning, and AI. This topic is also gaining attention in the press (Nott (2017)).

Interpretation in machine learning consists of the extraction of relevant knowledge from the relationships on a domain observed in a data set (Murdoch et al. (2019)). An interesting approach trying to solve the problem with interpretability of machine learning models is Kulesza et al. (2015), a proposal in debugging machine learning models where the user gives feedback to the learning system so there is an increase in participants' understanding.

Another interesting approach is LIME, presented in Ribeiro et al. (2016). LIME is an explanation technique that explains the predictions of any classifier by learning an interpretable model locally around the prediction.

[1] https://kdd.isti.cnr.it/xkdd2019/

The problem has also concerned governments, with for example the emergence of a program Gunning (2017) lead by the Defense Advanced Research Projects Agency (DARPA) that is focused on this problem. They are concerned by the need for a XAI because of the many use cases it has where there is risk involved and need of the users to trust these systems. What this program is trying to achieve is what is shown in Figure 2.2 and explained above, the development of explainable models that give the user a reasoning about the answer given by the learning system. It is also interesting to see the current state of the art, as shown in Figure 1.2. In this figure, the different learning systems are displayed in a graph. As the graph says, current learning techniques with the highest prediction accuracy are also the ones that provide the user with the least explainable systema. What this program and in general people working in this area are trying to achieve is shown in Figure 2.2. In red, today's systems are reflected and in green tomorrow's ones, where the learning performance remains equal or better and explainability experiments a boost.

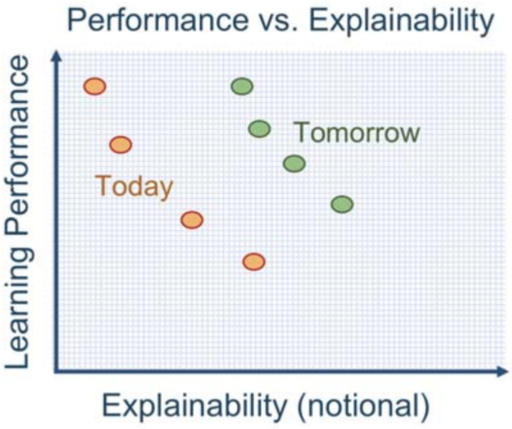


Figure 2.2: Scheme of current and future state in learning techniques as regards performance versus explainability. Each dot represents a learning technique. Source: Gunning (2017)

In order to achieve this goal, a *Measure of Explanation Effectiveness* is also proposed by the project conducted by DARPA, which can be used to determine if an explanation is good enough or needs to be improved (see Table 2.1).

A really interesting work on black box and everything concerning this topic is Rudin (2019). It presents a view of the problem completely different from the one viewed before in Gunning (2017) and discusses the current trend of developing a model to explain black boxes. The author has another interesting pieces of work that are highly related to this one like Rudin et al. (2018), where an analysis of the transparency of COMPAS (recidivism-risk scoring model used throughout the criminal justice system), a tool used in recidivism prediction. In this piece of work (Rudin (2019)), the author defines a black box in the following manner:

"A black box model is either a function that is too complicated for any human to comprehend, or a function that is proprietary; it is a model that is difficult to

User Satisfaction	**Mental Model**	**Task Performance**	**Trust Assesment**	**Correctability**
Clarity of the explanation	Understanding individual decisions	Does the explanation improve the user's decision, task performance?	Appropriate future use and trust	Identifying errors
Utility of the explanation	Understanding the overall model	Artificial decision tasks introduced to diagnose the user's understanding		Correcting errors
	Strength/weakness assessment			Continuous training
	'What will it do' prediction			
	'How do I intervene' prediction			

Table 2.1: Measure of Explanation Effectiveness for a machine learning model. Source: Gunning (2017).

troubleshoot. Deep learning models, for instance, tend to be black boxes because they are highly recursive."

The author also presents the key issues in explainable machine learning and interpretable machine learning, which are pretty interesting to be stated and discussed because they are opposed to some of the points presented in this review of the state of the art.

The key issues in explainable machine learning presented by the author are:

- It is a myth that there is necessarily a trade-off between accuracy and interpretability. It is usually said that nowadays a complicated model (black box) is needed for top predictive performance but it is not usually true since interpretable algorithms have a great performance too. The author also discusses Figure 2.2, saying that it does not show any quantitative measure but just what researchers believe.

- Explainable machine learning methods provide explanations that are not faithful to what the original model computes. The author explains that if the explainable model could provide a completely faithful explanation, it would not be needed because the model would be interpretable itself. Instead, the explainable model constructs an interpretation of the model but it cannot really be accurate.

- Explanations often do not make sense, or are incomplete. Saliency maps (image that shows each pixel's importance within the original image) are often considered to be a good explanation method but the author states that these maps only show where the black box is looking at but they do not tell how this focused area is being used.

- Black box models are often not compatible in situations where information outside the database needs to be combined with a risk assessment. If someone wants to calibrate the black model with some information outside the database, it is impossible to know how it may affect the model innerly.

In interpretable machine learning, the issues are:

- Corporations can make profits from the intellectual property afforded to a black box. Companies can make profits from making predictions with black boxes that they could not be making if the model was interpretable.
- Interpretable models can entail significant effort to construct, in terms of both computation and domain expertise.

Some research has focused on trying to undertake the increase of comprehensibility of the models, for example in Huysmans et al. (2011) or Rüping (2006). In Freitas (2014), a review of comprehensible models is conducted, evaluating the algorithms from an interpretability approach rather than focusing on predictive accuracy, which is the usual way of approaching this topic. It states the problem with the size of a Bayesian network when talking about interpretation. When they get bigger, the visualization and interpretation of the results start to be really difficult or even impossible. This is the main issue addressed in this project, as discussed further in the following chapters.

It is also interesting the idea that AI experts build models to be explainable just for themselves but not for a common user so they need other experts to come and work on this question in order to make the models explainable for people that are not experts in the area. This idea is discussed in Miller et al. (2017).

As stated in this section, there are different research tracks in this field of explainable and interpretable machine learning but it is a field that is gaining importance and where publications are constantly appearing. In the following section Bayesian networks are introduced as an interpretable machine learning model and it is the one studied over the rest of this project.

A review of XAI methods applied in deep learning is conducted in Samek et al. (2017) and a review of techniques for interpretable machine learning is Du et al. (2018), also focused deep learning but interesting in order to understand the trend in these fields on XAI.

2.4 Bayesian networks as an interpretable model

A Bayesian network (Pearl (1988)) is a probabilistic graphical machine learning model, shaped by a directed acyclic graph and the conditional probabilities involved.

The graphical component in Bayesian networks makes the model a pretty interesting approach in interpretable in machine learning.

This graphical model offers the possibility to be used for both prediction and explanation. Three types of explanations (Lacave (2002)) are studied in Bayesian networks: explanation of reasoning, explanation of model, and explanation of evidence.

A relevant review of explanation in Bayesian Networks is Yuan et al. (2014), where an in-depth review of explanation in AI is conduced and several of the previous references were extracted from this piece of work.

The explanation properties given by a graphical model like Bayesian networks make it a suitable model to be used in environments where the decisions' consequences are critical and a quality understanding of the reasoning behind the answer given by the model is needed (Andreassen et al. (1989); Shwe et al. (1992)). Many other machine learning models offer no explanation, as already mention, so they cannot be used in many real world problems.

The idea of this project is to boost the visualization and interpretation available in Bayesian networks, making this model an interesting approach in the future of AI thanks to its ability of explaining the answers given by an intelligent system.

Chapter 3

Review on visualization and interpretation in Bayesian networks

This chapter includes a review of key Bayesian network concepts and the current state of the art contributions in visualization and interpretation of Bayesian networks is conducted. Starting from the foundations of the concepts, then going through the problems observed that are important for this project and finally overviewing the different applications and software available. A deeper review on the topic example is Bielza and Larrañaga (2014), focused on the applications of Bayesian networks in neuroscience and Bielza and Larrañaga (2014), focused on discrete Bayesian network classifiers. In this project, the study is focused in Gaussian BNs.

3.1 Foundation

Bayesian networks (BNs) (Pearl (1988); Koller and Friedman (2009)) are a machine learning model that encode graphically probability distributions over a set of random variables. This model allows to work both with discrete and continuous variables. From a data set, a Bayesian network can be learned, both its parameters and structure. Not only this is possible, this model allows an expert to directly model the problem with his/her knowledge. Bayesian networks are structured as a directed acyclic graph (DAG), where the nodes represent the variables involved and the arcs model the probabilistic dependencies between them.

Each node in the network has a conditional probability table associated, expressing the probability of the variable at the node conditioned on its parents in the network. The joint probability distribution (JPD) p($X_1, ...X_n$) factorized according to the DAG provides a compact representation of every possible event in the network, defined by the combination of the values in the variables.

It is also important to remark that the connections between nodes do not represent causal relationships between them, but probability dependencies among variables. The dependencies between nodes are represented as conditional probability distributions (CPDs), quantifying the interaction between linked variables conditioned by its parents. In Figure 3.1, an example of a simple Bayesian network, adapted from P. Murphy (1998), can be observed, with the DAG and CPD of each node X_i given its parents $p(X_i|\mathbf{Pa}(X_i))$. This example represents the relationships between four variables. The relationships between the variables can be studied in the example, knowing for example that it has rained but the sprinkler is not working, the probability of wet grass can be extracted (0.9 wet grass, 0.1 dry for this particular example, as observed in the tables).

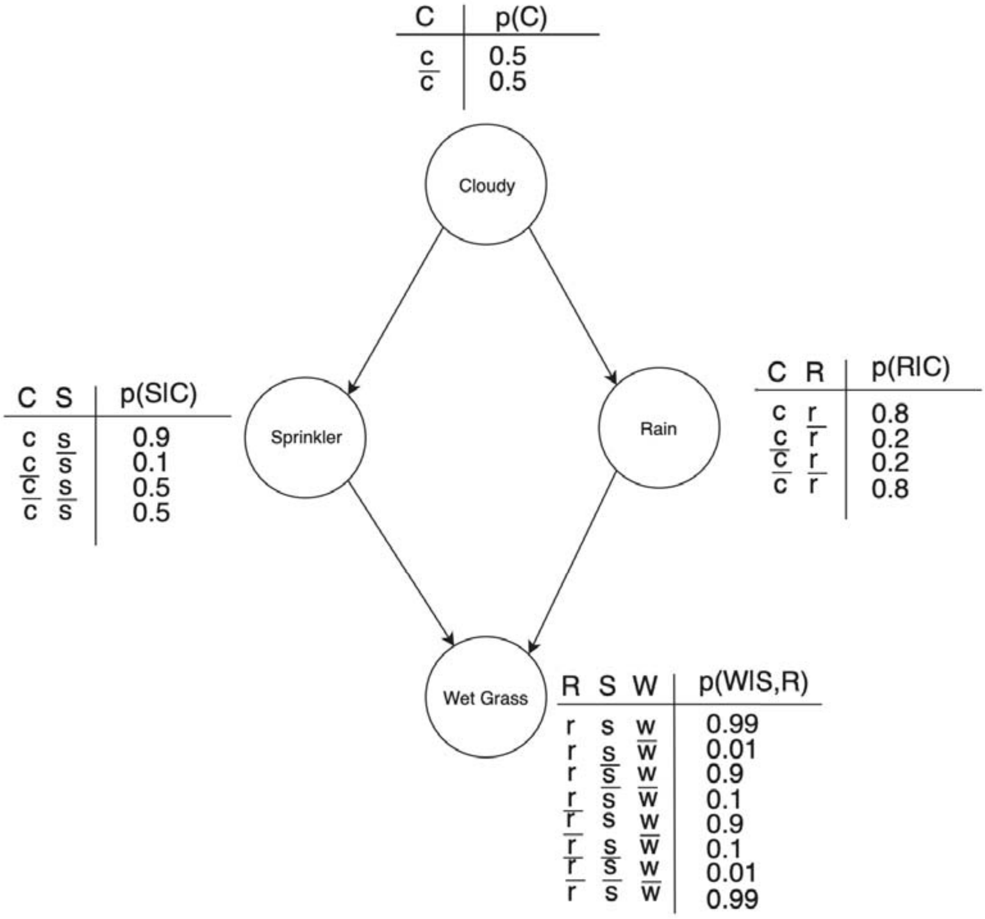


Figure 3.1: Bayesian network example. The graphical representation has two elements, the direct acyclic graph (DAG) and the CPDs. Adapted from: P. Murphy (1998).

Another interesting concept that is useful in Bayesian networks is the **Markov blanket** (Pearl (1988)) of a variable. When the Markov blanket of a random variable is observed, it represents all the information about that certain variable. For a certain node, the Markov blanket is defined as the set consisting of its parents, its children and the children's parents.

What the Markov Blanket says, is that all information about a random variable in a Bayesian network is contained within this set of nodes (parents, children, and parents of children). That is, if we observe ALL OF THESE variables, then our node is independent of all other nodes within the network.

3.1.1 D-separation

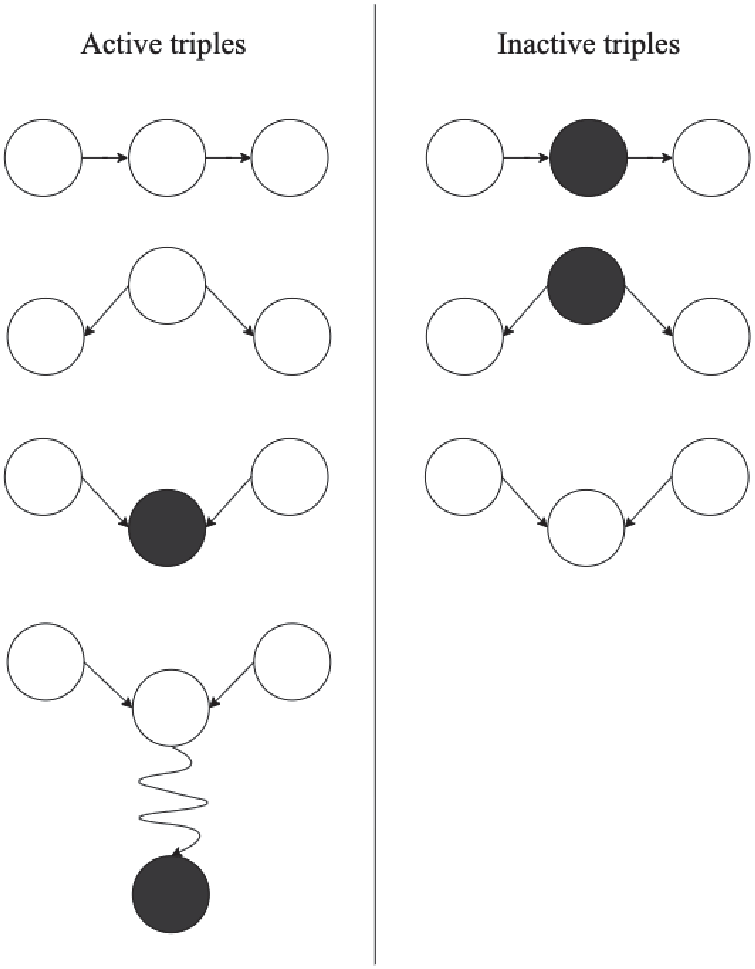


Figure 3.2: D-separation triples relation. Black nodes represent observed nodes and white ones are unobserved nodes. Source: Koller and Friedman (2009).

D-separation is a property from graph theory studied in Bayesian networks to find implied conditional probabilities. Given two groups of nodes **X** and **Y**, they are d-separated (out hence the conditional independency can be guaranteed) by a third set of nodes **Z** if none of the paths between them is composed by a full set of active triples, where a triple is every set of three contiguous nodes in a given path. The possible definitions of what an active or inactive can be are in Figure 3.2, where black nodes represent observed (presence of evidence) nodes and white ones are unobserved. the cases always analyze just three contiguous nodes, except for the case where two nodes converge in a third one and this last node is linked to an observed node, which is descendant. This case is shown in bottom-left part of the figure, see the last case in active triples relationships.

An algorithm to check D-separation is stated in Koller and Friedman (2009), called Reachable procedure, which has been implemented in this project as part of the improvements in visualization of Bayesian networks. This algorithm retrieves the group of nodes reachable from a given node that is the starting node, having the group of observed nodes in the network.

3.1.2 Parametrization

Parameters are one of the quantitative components of Bayesian networks. Depending on the type of data studied, two types of parameters can be found, discrete or continuous.

When all the nodes are discrete variables, they have conditional probability tables (CPT) attached. Each parameter in a CPT represents the probability of a certain state for the node given the values probability of the parent nodes.

Nodes can also be shaped by continuous variables, as it could be in gene regulatory networks. In this case, $\boldsymbol{X} = (X_1, ...X_n)$ is usually assumed to follow a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and each node X_i has associated a normal distribution with its own $\boldsymbol{\mu_i}$ and $\boldsymbol{\sigma_i}$. Any set of parameters can be expressed in terms of of the other set. The corresponding BNs are called Gaussian BNs.

3.2 Inference in Bayesian networks

Evidence is the name given to known knowledge inside the network; variables where their state is considered to be observed. Given the network, with its DAG and CPDs, and the evidences, the process of inference is based on calculating the *a posteriori* probabilities for the nodes given the evidences i.e $p(x_i|\mathbf{e})$ where $\mathbf{e}$ is the evidence and X_i any observed node.

Several methods for inference in Bayesian networks are available. This is not intended as a full review of the whole spectrum of methods, so the exacts methods are not studied. It is important to note that exact inference is known to be an NP-hard problem (Cooper (1990)), so many methods have been developed to approximate the probabilities instead of calculating the exact ones. In the case of Gaussian BNs, as the ones studied in this project, inference is easy since the conditional distributions are also Gaussian and both the mean and variance have closed formulas (Lauritzen and Jensen (2001); Cowell (2005)).

3.3 Learning Bayesian networks

Both the structure and parameters can be learned from data in a Bayesian network. In the following two subsections, the main approaches in structure learning and parameter learning are presented.

3.3.1 Structure learning

Two major approaches are considered in structure learning:

- Score-based methods. They are based on first defining a criterion to evaluate the performance of the network fitting the data and then searching over the space of possible networks trying to maximize the score. Examples of this method are K2 algorithm (Cooper and Herskovits (1992)) and the Chow-Liu tree (Chow and Liu (1968)).

- Constrained-based methods. They are based on testing from data the conditional independence among triples to then build the DAG representing all or a percentage close to all independence relations. A well-known algorithm for this method is the PC algorithm (Spirtes and Glymour (1991)).

3.3.2 Parameter learning

Two main approaches are also followed in parameter learning:

- Maximum likelihood estimation. The estimations for the parameters maximize the likelihood of the data. Problems with sparse data are known and can lead to some undefined conditional probabilities.

- Bayesian estimation. The prior distributions are usually chosen to be conjugate with respect to multinomial or Gaussian distributions.

An important problem when learning Bayesian networks is dealing with missing data, which is recurrent in real-life when working with real data sets and is approached using the expectation maximization (EM) algorithm (Dempster et al. (1977)).

3.4 Bayesian networks applications

Many applications of Bayesian networks are studied and used in industry and academia. The explanatory skills enjoyed by Bayesian networks make them a suitable approximation for problems where the decision risk is high. On the other side, other machine learning algorithms offer a high performance models but they cannot be applied to risk decisions due to their innability to explain the output results.

One example is medical diagnosis, where a doctor observes some symptoms in a patient that are used as input variables to the model and the goal of this model is to output the probability of a variety of diseases given the symptoms. Successful applications in medical diagnosis are Andreassen et al. (1989) and Shwe et al. (1992).

Gene regulatory networks are other area of application (Bielza and Larrañaga (2014)), as studied in this project. In this case, having as data the levels of expression of different genes present in humans, the network structure can be retrieved, which connects genes based in their regulatory relationships.

Another area of successful application of Bayesian networks is reliability (Langseth and Portinale (2007)). Many other examples can be found in Section 1.1, where the machine learning applications examples also used Bayesian networks.

3.5 Visualization in Bayesian networks

After the brief review on the foundations of Bayesian networks and their main components, this section presents the state of the art in visualization in Bayesian networks. This is necessary in order to understand the current limitations of the model's visualization and how to overcome them in this project.

One of the most recent and relevant works in visualization in Bayesian networks are Lacave (2002), Lacave and J. Díez (2001) and Lacave et al. (2007), where a detailed study on visualization and explanation of Bayesian networks is conducted. The explanation of the visualization is divided into three categories: the explanation of nodes, edges and global explanation.

In the explanation of nodes, interesting points to be noted are the function given to a certain node, which can be interpreted as the creation of different groups of interest and the importance factor of a node. This idea is updated in this project computing the importance of a node given some evidences using some metrics.

The explanation of edges graphically implies representing them with the amount of influence they transmit (which is already implemented in *Neurosuites*). For the explanation of the global network, it is also presented that it would be necessary to give the user the possibility of focusing on a certain part of the network. For this project, several approaches for this problem are proposed, such as the possibility of selecting nodes by their importance in the network or the possibility of creating groups to work with, for example by setting a common evidence for the group or understanding how some evidence has impacted the group.

More recent relevant work on visualization is Chiang et al. (2005), where some interesting points are studied. They present a system to create the Bayesian network layout, based on the idea pointed in Marriott et al. (2005); Marriott et al. (2011) that states the limitations of the layout's design, saying that they are usually really poor implementations and they are not well optimized. In this research, they implement the idea of graphical drawing based on the hierarchical approach by Battista et al. (1998), with three steps in creating the layout and also studying ideas like the layout design proposed in Sugiyama et al. (1981). This layout design idea has been studied in *Neurosuites*, where several layout implementations have been provided to the user and can be applied to the network on-demand, giving flexibility to the user's decision. In regards to visualizing nodes and edges, their proposal is not different from many others, with node sizes and colors depending on certain hidden properties (not specified in the paper) and edge thickness based on the relationship between the nodes. They also propose the idea of visualizing conditional probability tables as color-maps.

Another remarkable investigations in visual information in the graph using properties such as color, size and thickness of the nodes and nodes are Williams and Amant (2006) and Zapata-Rivera and Greer (1999). Wang and Druzdzel (2000) is focused on the visualization of CPTs, going through the many different available software tools and proposing a visualization based on bar charts, which is typical across all Bayesian networks' software tools when visualizating CPTs.

Visual causal semantics in an interesting area in the visualization of Bayesian networks, and in Kadaba and Irani (2007), several animations are presented to try to solve the visualization problem. This animations provided a visual explanation of the reasoning occurring in the network, but they are difficult to extrapolate to large scale visualizations as presented in the paper. Examples of the animations include animated movement in the edges to show the direction of the causality and animated increase of nodes to show importance. Verbal descriptions of the inference are another area already studied in Yap et al. (2008), which is useful in small-scale networks. An example of verbal description as presented in the paper presents how the reasoning is conducted considering the evidences and the probabilities used in the conclusion reached.

The major problems with visualization and interpretation in Bayesian networks are in large-scale networks. This problem has long been known and studied, like in Neil et al. (2000) with the idea of *building blocks* called *idioms* that combined can make a big network.

Another interesting research work is Cossalter et al. (2011), where the concept of CPT bubble anchor (visual connection between the nodes and the actual CPTs that are bubble shaped) is presented and the concept of CPT merging (comparisons between CPTs from different nodes). They also state the problem of visualizing large networks with huge amounts of nodes where they can not fit a computer screen. On visualization of evidence in Bayesian networks, Champion and Elkan (2017) is a really interesting piece of work, where ideas on visualization of the changes generated by an evidence and filtering of nodes to find relevant ones are given, drawing directly in the node how the evidences affect the probabilities.

For large network interactive exploration, Krishnan et al. (2013) introduces multi-focus and multi-window techniques that can be useful to focus on several areas of the network at the same time.

Some other researches are focused on interpretation of the network model induced directly from data, like in Dalgaard et al. (2004), but in the proposals stated in this project, Bayesian network layout and network model are considered to be already a problem studied in depth, so the focus is in another sides of Bayesian network study.

3.5.1 Software

Several commercial software applications are available, so a quick review of the key aspects presented by them as state of the art needs to be addressed in order to make sure where is the starting point for the real application where the improvements are deployed.

GeNIe[1] by BayesFusion is one of the most used Bayesian network software tools. They offer a graphical user interface to work with Bayesian networks that allows interactive model building and learning. Graphically, it shows the parameters for each node, with the probabilities for each state and the directed connections between the nodes. The nodes with evidence set have a different color than the rest. Another important feature provided is that the amount of evidence transmitted by a node to another is shown in the thickness of the edge between them. BayesiaLab[2] is another option when taking about Bayesian network software. They provide similar functionality. Visually, BayesiaLab offers interesting characteristics, like the possibility of assigning an image to a node in the network. It is more interesting visually, with a prettier user interface and features like the comparison between prior and posterior probabilities that helps understanding the effect of an evidence.

BayesServer[3] offers Bayesian network functionalies similar to the other approaches, like link strength or the creation of groups of nodes. There is a lack of tools to work with BNs when talking about huge networks, so this problem is common for the different approaches.

Elvira (Elvira Consortium (2002)) is a research Bayesian network software, where the ideas expressed in Lacave (2002) are implemented. Other examples included SAMIAM[4], developed in 2010 by the Automated Reasoning Group at UCLA and OpenMarkov[5], developed by UNED.

3.5.2 Problems

The problems observed in these available resources on visualization are the inability to manage large networks with tens of thousand nodes. They are focused on small examples with very few nodes and it is not possible to use the same available approaches when working with enormous networks.

New techniques are needed. The improvement in computational power experimented in the last few years is thanks in part to GPUs that make it possible to visualize tens of thousands nodes at the same time without any effort. This calls for an improvement in the state of the art in visualization in Bayesian networks to make it possible to extract knowledge from the network.

Now that technology allows the visualization of huge networks, the importance of the visualization is not only focused on showing all the nodes and edges in an application at the same time, but on giving the user tools to understand which parts of the networks are more important, how the evidences are spread across the whole network computing which nodes have been more affected, etc.

[1] https://www.bayesfusion.com/genie/

[2] https://www.bayesialab.com/

[3] https://www.bayesserver.com/

[4] http://reasoning.cs.ucla.edu/samiam/

[5] http://www.openmarkov.org/

This is the main goal of the improvements this project proposes and implements.

Chapter 4

Visualization in Bayesian network modelling

This chapter introduces a series of changes in visualization and interpretation over Bayesian networks (see Table 4.1) taking as a starting point the techniques of the state of the art in visualization presented in the previous chapter. These changes are presented on a generic basis, introducing them and studying the improvements that would be involved in the visualization and study of a generic Bayesian network.

Changes	**Observations**
Importance of nodes	Degree of importance and Betweenness centrality
Community detection	Louvain algorithm
Node parameter visualization	Discrete and Gaussian probability distribution visualization
Setting evidence and updating probabilities	Individual and group of nodes evidence
Group of nodes creation	
Network structure comparison	Different metrics are applied
Change in probabilities in large networks	
D-separation visualization	Different metrics are applied
Probabilistic clustering visualization	

Table 4.1: Changes in visualization and interpretation over Bayesian network studied.

4.1 Computation of the importance of nodes

One of the main singularities in the Bayesian networks studied throughout this project is their size. Being able to surpass 20000 nodes and with a very high number of connections, its complexity is enormous and the visualization becomes very complicated with the available techniques.

For this reason two methods of computing the importance of the nodes within the network are proposed. Once this importance is computed for the nodes, they are drawn with a size proportional to that importance so the user can understand easily which ones are more important. Apart from this, those nodes whose importance within the network is remarkable as calculated by the algorithms are coloured in different colors so they can be distinguished.

Two techniques are proposed to compute the importance of the nodes in the network. The first one calculates the degree of a node based on the number of neighbors and the second one computes the betweenness centrality, usually applied in network science to study big networks. The new techniques are presented below.

4.1.1 Degree of importance of a node based on its number of neighbors

This technique would be one of the simplest ones. The importance of a node is calculated based on the number of neighbors that are directly connected to the node. In Bayesian networks, the number of neighbors is given by the addition of the number of parents and the number of children of a particular node.

$$neighbors\_number(n) = parents\_number(n) + children\_number(n) \tag{4.1}$$

Having the number of neighbors of a given node, its importance can be calculated comparing it to the number of neighbors of the rest of the nodes in the network. Those nodes with more neighbors will weight more in the network. In order to graphically show this situation, the size of a node can be drawn considering the amount of neighbors, so the greater the number of neighbors the bigger the node is drawn.

4.1.2 Betweenness centrality

Betweenness centrality is a measure of centrality in graph theory that was first described in Freeman (1977). The measure of centrality in networks constructed as a graph is based on the shortest paths. Inside the network, for each pair of connected nodes, the shortest distance between them is computed. In the case where the edges are not weighted, the calculation is taken from the number of edges between the nodes. Thus, betweenness centrality g of node v is

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}. \tag{4.2}$$

The total number of shortest paths from node s to node t is defined by σ_{st} and $\sigma_{st}(v)$ is the number of those paths that pass though v. If the edges are weighted, the summation of weights of the edges is minimized,

$$s_i = \sum_{j=1}^{N} a_{ij} w_{ij}. \tag{4.3}$$

The adjacency between node i and j is a_{ij} and w_{ij} is the weight and N represents the number of nodes. A node with a high value of the measure will have more control over the network and therefore a change in this node will have a stronger effect in the rest of the network. For example, the setting of an evidence in a node with a high value of betweenness centrality will have a pronounced effect in the probabilities of the rest of the nodes.

4.2 Community detection

When studying large size Bayesian networks, another idea that emerges that could be also useful is the idea of detecting groups of interest within the network based in different criteria. This idea could help in the visualization and interpretation of Bayesian networks when the network is large because it is probably more useful to detect groups of interest than working with individual nodes. When talking about community detection algorithms, one of the most well-known and reputed algorithms is called Louvain modularity, that could be useful when working with really complex networks.

4.2.1 Louvain modularity

This community detection algorithm was first introduced in Blondel et al. (2008). The name of the algorithm is related to the fact that some of the authors worked at The Université Catholique de Louvain, in Belgium. The main purpose of this algorithm is the optimization of the modularity, being this concept a scalar measure that is in a range between -1 and 1 and that measures the density of connections inside a given community in comparison with the density of connections outside. The definition of the modularity measure for weighted edge graphs is calculated as

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{4.4}$$

where the value of Q is the value to be optimized, A_{ij} represents the weight between nodes i and j, k_i and k_j the sum of weights of the edges attached to nodes, $2m$ the sum of all the edge weights, δ a Kronecker delta function and c_i and c_j the nodes communities. The optimization of this value is calculated from the best node grouping. The algorithm follows an iterative process to make this optimization, divided into two steps that are shown in Figure 4.1.

At the first step, a different community is assigned for each node inside the network. After this process is completed, for each node i, the nodes j of i are considered and the evaluation in the gain of the modularity that a change in the community would imply is evaluated. Once this is calculated, the node joins the community with the biggest profit associated, what could even result in a node staying in the initial community after the calculation. This process is iteratively conducted until the optimization modularity measure cannot be improved any more. This is why the first step in the process is called Modularity Optimization.

One of the strengths of the algorithm is that the calculation to obtain the gain in the modularity for the change of the community for a certain node can be easily calculated as follows:

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{in}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]. \tag{4.5}$$

the sum of the weights' links in the community where i is moving into is $\sum_{in}$, the sum of all the weights of the links to nodes in the community i is moving into is $\sum_{tot}$, k_i is the weighted degree of i, $k_{i,in}$ is the sum of the weights of the links between i and other nodes in the community i is moving into and m is the sum of the weights of all links in the network.

The second step of the algorithm is called Community Aggregation. It is based in the construction of a new network whose nodes are the communities found in the previous step. To complete this step, the edges weight between the new nodes are given by the summation of the weights in the connections between nodes in the corresponding communities. Connections between nodes in the same community lead to auto-loops of this community in the new network. Once this second step is completed, a new iteration can begin from the first step.

4.3 Node parameters visualization

After the process of learning parameters in the Bayesian network, the probabilities at the nodes in the network are obtained. In some cases, these probabilities are discrete probability distributions and in other cases Gaussian continuous distributions *Neurosuites*, covered in chapter 5, gives more importance to Gaussian probability distributions. This is because many of the software tools and research papers studied focus their parameter visualizations on discrete probability distributions, so in this Gaussian case it has to be studied how to show the relevant information to the user and make it suitable with working with large networks. Therefore, two particular cases can be distinguished in this section: discrete probability distributions and Gaussian continuous distributions.

4.3.1 Visualization of discrete probability distribution

The first kind of visualization is already well studied in the literature and available software, so the general idea should be to follow the approaches already presented in other tools. In Figure 4.2, an example is presented. In this visualization there

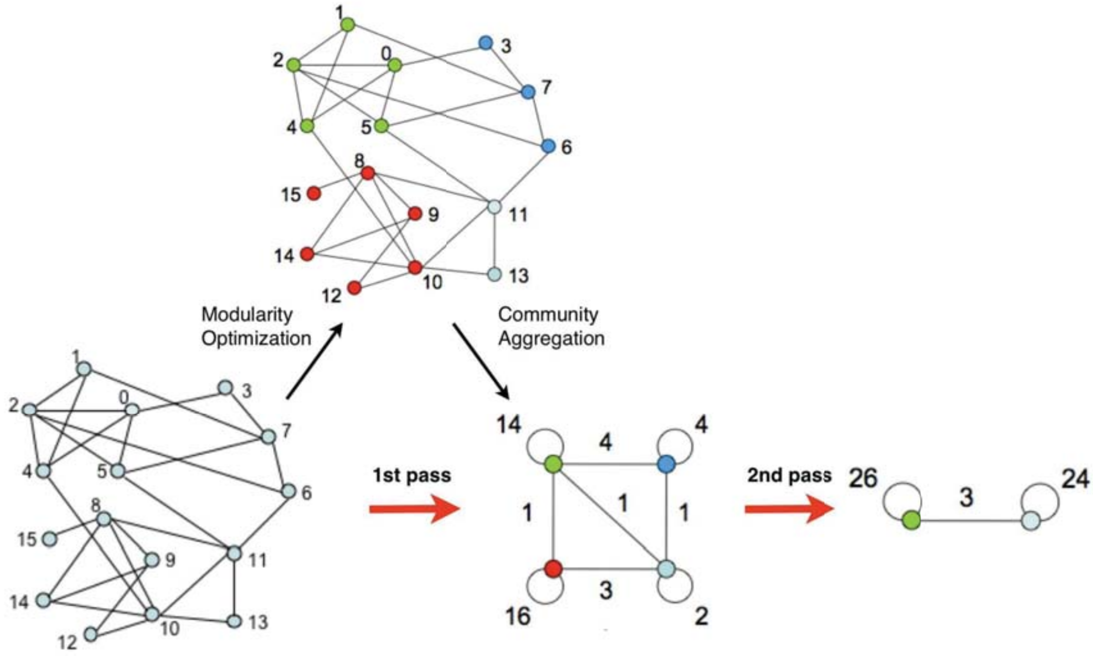


Figure 4.1: Scheme of steps followed in the Louvain modularity algorithm. Source: Blondel et al. (2008)

should be a top bar with the name of the node and then a bar for each of the possible states, with the name and current probability for that state.

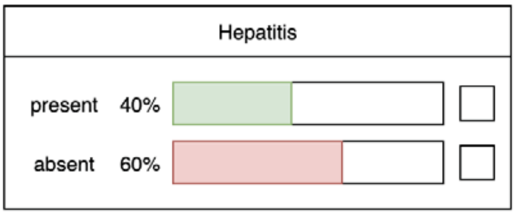


Figure 4.2: Example of the visualization of a discrete probability distribution.

Apart from that idea, it would be interesting to have a space to select the desired evidence. This is also shown in Figure 4.2, where on the right hand side of the box there is a space to select the evidence that is wanted.

4.3.2 Visualization of Gaussian probability distribution

The second type of visualization that is worth studying for this project is Gaussian probability distributions. In this case, the interest lies in showing the Gaussian density as such and in addition, it is also interesting to show the value of the mean and standard deviation for the marginal distribution (see Figure 4.3). These values are interesting when studying the changes suffered by the distribution once an evidence is introduced in one of the nodes of the network.

When setting the evidence for a certain node, the initial idea would be to make this process interactive so the user could click in the graphical tool where the evidence is wanted and that would set the evidence. Additionally, two buttons to set the clicked evidence or clear it would be interesting contributions.

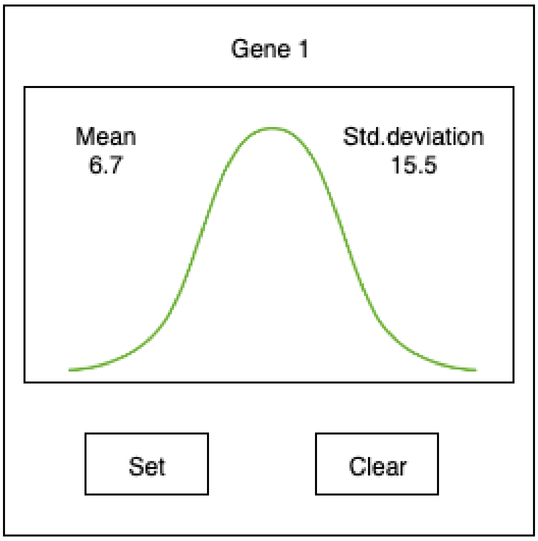


Figure 4.3: Example of the visualization of a Gaussian probability distribution

Additionally, it would be interesting showing how a certain evidence has affected the probability distribution. In this case, both distributions could be shown in the same plot and the difference between the prior and posterior probability distribution could be displayed.

4.4 Setting evidences and updating probability distributions

Taking into consideration the focus of the project on the visualization and interpretation of large Bayesian networks, the way an evidence is set in the network should be updated, to make it viable for a real world environment. The case study for which the software *Neurosuites* is initially intended is based on the use of gene regulatory networks, where it is interesting to study for example the effect of a series of changes in the evidences all at the same time. Considering this situation, it is interesting to seek two ways of working when setting an evidence in a large network: the first one would be setting an evidence as traditional software tool do, individually setting the nodes' evidence and the second one would be setting a common evidence for a group of nodes at the same time.

4.4.1 Individual evidence

This type of evidence is the usual one within the software available for the visualization of Bayesian networks. It is based on setting the evidence for each node in the network individually. This type of technique is necessary but the problem that it produces is that when the number of nodes and edges is huge (tens of thousands) it starts to lose value because it is really time consuming and not really productive. Additionally, in certain cases it would be interesting to have the possibility of setting the evidence for more than one node at the same time, probably not seeking a specific value for each node at the same time but a global value that could be a high value for all the nodes or a low value with respect a reference value.

4.4.2 Evidence on a set of nodes

This type of evidence setting appears from the requirement when working with large networks of allowing a user to easily set evidences. In this case, the sought purpose would be having a method to select a set of nodes from the network and set a common value for all of them. When the variables are Gaussian, this situation would be achieved setting the evidence as the number of standard deviations from the mean. For example, if a user sets the evidences to a value of 3, that would mean that for every node in the set (group) of nodes, the evidence would be set 3 standard deviations above the mean (to the right).

Using this feature, an anomalous high value for an evidence (positive number of standard deviations from the mean) or a low one (negative number of standard deviations from the mean) could be set for a group of nodes, which could be translated into a great impact (change in the probability distributions) on the rest of the nodes.

In the case of working with discrete parameters, this work-flow is more difficult to be implemented since in a real-world scenario it is more common that every node has different possible parameter states.

4.5 Groups of nodes creation

In the previous case, a need to work with sets of nodes when setting an evidence was found. In this section, the idea of creating groups of nodes is introduced and once again, this idea comes from the special needs created when working with large networks.

In this case, the idea of creating groups of nodes where you can label these groups to have access to them arises. The idea is to be able to select a set of nodes, give them a name that should be related to the interest these nodes have as a group and give the group a common color so they can be distinguished from the rest of the nodes in the network.

This feature could be used to group nodes that have a common aspect in the problem domain so a user can work over them easily. In the case study presented in this project of gene regulatory networks, this is seen in depth. This feature can be used in that situation to create groups of nodes that are genes representing, for example, the same area of the brain. To give this functionality a real utility, it is necessary to have the means to set the evidence to the group at the same time, as it is explained in the previous section, besides giving the user the chance to clear the evidence from all the group of nodes at once.

Another interesting idea regarding groups of nodes would be to compute the effect of an evidence on the set of nodes using some kind of metric that calculates this effect. This is explained further in Section 4.7.

In short, the idea of creating groups arises from the need in a large network of providing value to a user at the time of visualization and interpretation. The improvements in visualization and interpretation can be important developing this idea since this can lead to the addition of another techniques like setting an evidence to a group or computing the change in the probabilities in the node when some evidences are set.

4.6 Network structures comparison

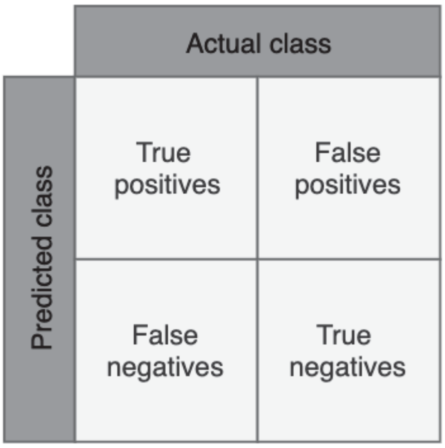
Considering the amount of structure learning algorithms in Bayesian networks, an interesting idea arises: giving the user a functionality to compare the output captured from different learning algorithms to visualize them and study the differences between the structures. Visually, the idea is to show the different Bayesian networks at the same time so a user can analyze their differences. In addition to the direct visualization of the changes in the edges (connections) in the networks, it is important to provide the user with a series of metrics that study the differences in the networks. It is also interesting to evaluate the performance of the learning algorithms, since the real structure of a network could be compared with a learned one and check how the algorithm performs.

4.6.1 Applied metrics

The different metrics used when comparing two Bayesian network structures are described below.

Binary evaluation is based on the confusion matrix. This matrix allows to discern the behavior of a certain algorithm. It is presented on a table format, with two rows and two columns. It represents the number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN) between the two evaluated structures, as it can be observed in Figure 4.4.

In this case, the evaluation is made on the edges that integrate the networks. The edges from the first structure are taken and compared to the edges in the second structure, creating as a result the confusion matrix. The actual class is the first one introducted to the system and the predicted is the second one. The meaning of the different values in the confusion matrix is the following:



	Actual class	
Predicted class	True positives	False positives
	False negatives	True negatives

Figure 4.4: Confusion matrix.

- True positives are the number of edges that are in the first structure and in the second structure as well.
- False negatives are counted when the second structure incorrectly has an edge that it should not have.
- True negatives are the number of edges that are not in the first structure and neither in the second one.
- False negatives are counted when the structure incorrectly doesn't have an edge that it should have.

From this confusion matrix previously defined, a series of different values can be obtained that are the metrics that would quantify the differences between the structures:

- **Accuracy**: It is the amount of correct answers with respect to the total amount of answers. It is defined as:

  $$ACC = \frac{TP + TN}{TP + TN + FP + FN}. \tag{4.6}$$

  The level of accuracy alone is not enough when the quality of the comparison wants to be computed, so other metrics are needed and are presented below.

- **Precision**: It calculates the proportion of correct identifications from the real correct number as:

  $$Precision = \frac{TP}{TP + FP}. \tag{4.7}$$

- **Recall**: Calculates the proportion of real positive results from the real class and it is computed as:

$$Recall = \frac{TP}{(TP + FN)}. \tag{4.8}$$

- **F-score**: It uses at the same time the concepts of Precision and Recall and it is calculated as:

$$F\text{-}score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}. \tag{4.9}$$

- **SHD**: Structural Hamming distance. Given two structures, this measure is the number of edges that would have to change in the second structure to become the first one.

- **MCC**: Matthews correlation coefficient, presented in Matthews (1975). It considers the true and false positives and is generally considered a balanced measure that can be used even if the classes have a different size. It retrieves a measure between -1 and +1, having +1 as the perfect prediction, 0 when a prediction is not better than random and -1 where there is a total disagreement between the prediction and the ground-truth. Chicco (2017) explains that this coefficient is more informative than other measurements that are directly obtained from the confusion matrix because it takes into consideration all the entries within the matrix. It can be calculated as follows:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP) * TN + FN)}}. \tag{4.10}$$

- **AUROC**: It refers to the Area Under the Receiver Operating Characteristic curve. This curve consists of a graphic representation of the sensitivity versus the specificity for a binary classification system, as the discrimination threshold is modified. In the case studied here, the edges are observed taking into account wheter or not they are equal to those of the real structure. The area under the ROC curve is the probability of categorizing a positive instance randomly chosen greater than a negative instance randomly chosen. The curve contains on its Y axis the proportion of true positives (Recall) and in the X axis the proportion of false positives. An example of a ROC curve can be observed graphically in Figure 4.5.

- **AUPRC**: It refers to the Area Under the Precision Recall Curve. In the Y axis the precision is positioned and in the X axis the recall values. With this metric and the previous one, grater values of the area below the curve are sought, meaning this situation that the comparison is good. In the case study in this project, it indicates that the structures have a great number of common edges.

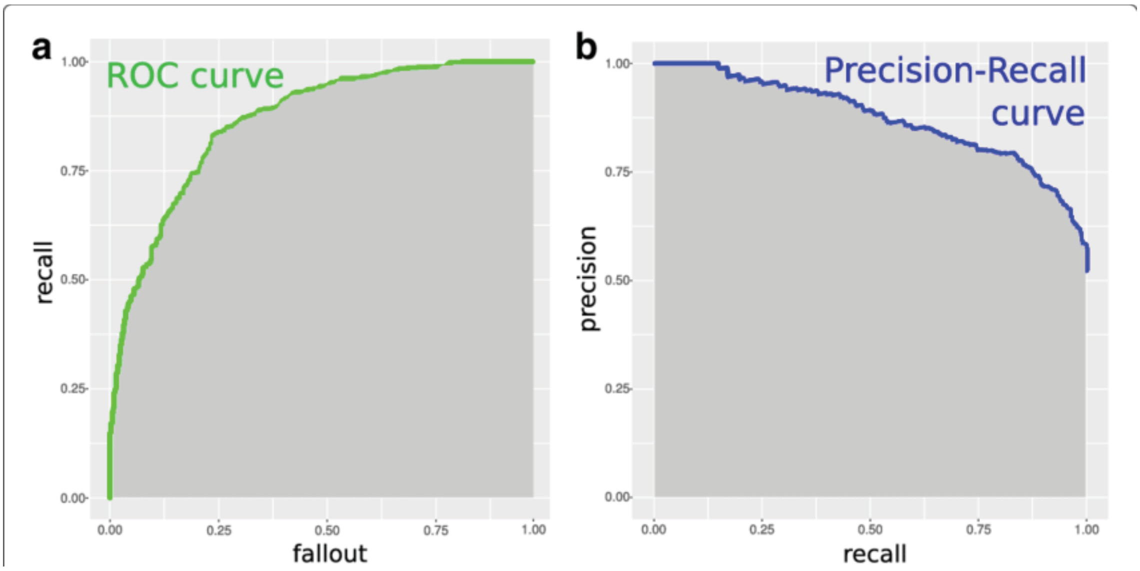


Figure 4.5: Example of ROC and Precision-Recall curve. Source: Chicco (2017)

4.7 Change in probabilities in huge networks visualization

The study of the large networks requires special functionalities to be able to extract knowledge in an easy manner from them. In this case, the focus is on the visualization in the modifications in probabilities for the nodes that shape the network. This means having the opportunity to compute the change experimented by the nodes from the initial state to the state where some evidences have been set. For instance, taking the probabilities that shape each of the nodes as Gaussian probability distributions, it would be interesting to show the amount of change experimented on these distributions from the initial state without evidences set to the situation where some evidences are set.

Bearing this in mind, both distributions are compared, the initial and the final one. Several options are available to compute the divergence between two distributions. For example the Hellinger distance (Ernst (1909)) or the Bhattacharyya distance (Bhattacharyya (1947)), which give a value between -1 and +1 to the difference between the distributions. Instead of using any of these methods, the Kullback-Leibler divergence is used (Kullback and Leibler (1951)). This measure is chosen in part because the value obtained is not bounded between -1 and +1; it may grow bigger. This is not itself an advantage, but it makes it easier for a user to understand visually changes if the values can grow bigger. Apart from this fact, this divergence is not symmetric, so the divergence of P and Q is different from the divergence of Q and P. The Kullback-Leibler divergence of two distributions P and

Q is computed as:

$$D_{KL}(P||Q) = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}, \tag{4.11}$$

where μ_i represents the mean and σ_i the standard deviation for the distribution i. This divergence measure can be applied both in situations where the target is on a specific node, computing how this node has been affected by the changes in the rest of the nodes, or for a group of nodes. The different between these situations is that the first one is an univariate computation and the computation for a group of nodes would imply a multivariate computation having several node distributions considered.

This measure can be additionally connected with the computation of the change in the mean and standard deviations. Thereby, for the case where the changes can be computed for a certain node, the changes in the mean and standard deviation can be observed. To calculate this change, a subtraction of the second measure from the first one is computed. These two metrics are only applicable when the changes are calculated for a specific node and they are not calculated for a group of nodes since each node has its own mean and standard deviation.

4.8 D-separation visualization

D-separation implies conditional independence of the nodes in the graph. The addition of this functionality would include two different interesting cases that are worth mentioning:

- **Check the dependency between three groups of nodes.** In this first case, three groups of nodes are considered in order to study the d-separation between two groups of nodes and the third group of nodes grouping the observed nodes. The d-separation is studied between the two groups of nodes given the third group where the observed ones are collected. With this procedure, a user can study if two groups of nodes are dependent on each other in a given situation.

- **Find the nodes that are dependent on a group of nodes.** In this second case, the idea is to give a group of nodes to study and for every node in this group retrieve the nodes that are dependent in the current situation in the network. This situation takes into account nodes with a certain evidence set.

4.9 Probabilistic clustering visualization. Gaussian mixture

Some visualization and interpretation techniques are extensible to another types of models and can be useful not only in the study of traditional Bayesian networks.

This visualization and interpretation techniques are useful in different scenarios. An example of this extension is probabilistic clustering visualization. Its visualization is quite related with the visualization of Bayesian networks presented and some of the changes introduced previously are also useful in its the study. In this situation, several clusters are shaped from the data and a joint visualization of the Gaussian probability distributions is interesting.

The different distributions that shape each node's probabilities are supposed to be drawn in a common plot. This information is interesting because it shows for a particular node, all the distributions for the clusters. Having all the distributions drawn, it is interesting to also indicate the mixture of these distributions. The Gaussian mixture (Robertson and Fryer (1969)) is the linear combination of normal distributions (see Figure 4.6) that can help understanding which areas are more important that others in the distribution. In order to create the mixture, a weight is needed for every distribution that is part of the mixture, showing the amount of importance of each distribution in the mixture.

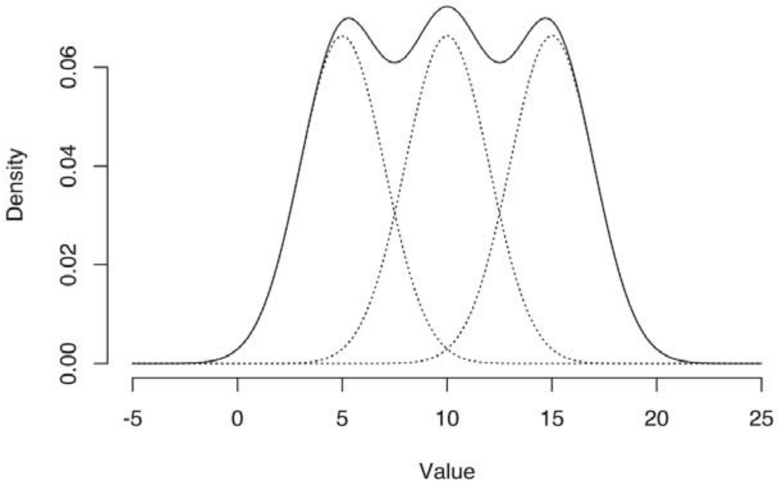


Figure 4.6: Gaussian mixture example. Source: Mason (2012)

The mixture distribution of K components is calculated as

$$p(\mathbf{x}) = \sum_{i=1}^{K} \phi_i \mathcal{N}(\mathbf{x}|\boldsymbol{\mu_i}, \boldsymbol{\sigma_i}), \tag{4.12}$$

where ϕ_i are the weights assigned to each distribution in the mixture. The sum of the weights must be 1,

$$\sum_{i=1}^{K} \phi_i = 1. \tag{4.13}$$

Chapter 5

Application of the proposals in visualization and interpretation of Bayesian networks in *Neurosuites*

This chapter aims to explain the applications of the proposals given in Chapter 4 to *Neurosuites*. A general view of what is *Neurosuites* and the problem it is trying to tackle is given, then the process of applying the previously explained methods is approached and the results of the improvements in visualization and interpretation are shown in order to prove the viability of them in the problem studied.

5.1 *Neurosuites*

Neurosuites[1] is a website that offers access to multiple neuroscience tools. This website has been developed at Computational Intelligence Group at Univesidad Politécnica de Madrid. It allows running neurocience tools that usually require an installed program in a computer.

5.1.1 Project architecture

A general picture of how *Neurosuites* architecture looks like is in Figure 5.1. This website has a **Django**[2] back-end, which is written in **Python 3**[3], where the requests from the front-end are received, computed and retrieved back to the front-end. The major computational load falls in the back-end, allowing a powerful computer to be used as server to handle the requests, leading to a better user experience. The front-end is mainly formed by **HTML pages**, **CSS style sheets** to style the pages and **JavaScript** to give computational power. The CSS part also includes **Bootstrap**[4] library in order to make it easier to implement a consistent style. JavaScript is

[1] Accessible at `https://neurosuites.com/`

[2] `https://www.djangoproject.com/`

[3] `https://www.python.org/`

[4] `https://getbootstrap.com/`

complemented by **jQuery**[5] library and several other libraries for the development and proper maintenance.

Apart from this general view of the architecture, *Neurosuites* uses several libraries which play a key role in the development of the project. In the Python back-end, **NetworkX**[6] is used to manage networks, **Plotly**[7] to create graph visualizations, **NumPy**[8] to manage big arrays and matrices, **SciPy**[9] for numerical computations, **sci-kit learn**[10] offers several machine learning tools and **Pandas**[11] to manipulate complex tables. In JavaScript, the most important library used for the project is **Sigma js**[12], which helps in the creation of the network visualization. This library makes use of **WebGL**, a JavaScript API that allows GPU-accelerated usage of physics and image processing. This functionality makes possible the visualization of tens of thousands of nodes and edges at the same time in real time with a smooth experience for the user.

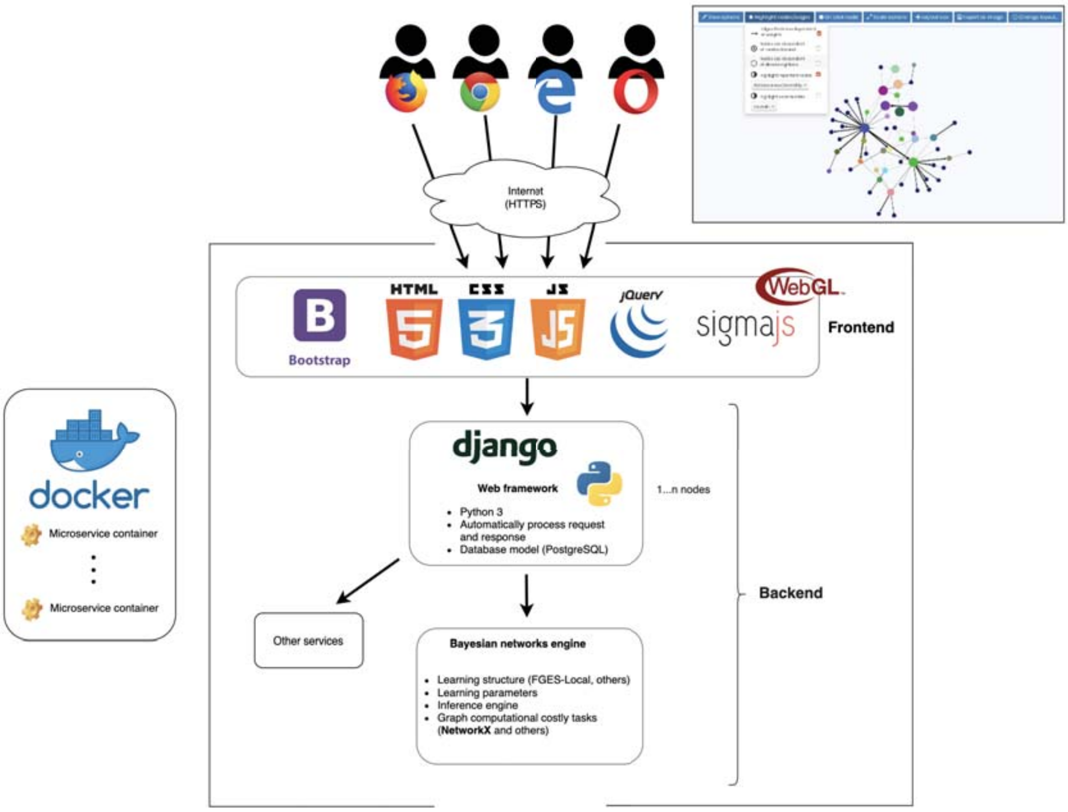


Figure 5.1: *Neurosuites* architecture.

[5] https://jquery.com/

[6] https://networkx.github.io/

[7] https://plot.ly/

[8] https://www.numpy.org/

[9] https://www.scipy.org/

[10] https://scikit-learn.org/

[11] https://pandas.pydata.org/

[12] http://sigmajs.org/

The architecture is built to enable multiple user interactions with the web at the same time. The project is maintained via a version control system, **Git**[13]. Git is used to keep track of the changes made to the website, without losing previous versions of the project, which could be useful if a problem arises. A version control system is also good when more than one developer is part of the project, giving tools for the team to make changes to the source code at the same time without losing any change in the meantime.

The web is hosted in a **Linux server** with 64GB RAM memory and a top range processor (i7-6800K CPU @ 3.40GHz 6 cores), giving enough power to be able to manage several requests from different users at the same time, computing large amounts of data, which is needed in this project. Inside this machine, a **Docker**[14] container is deployed, with a Linux image, several other images (microservices containers) and the *Neurosuites* project itself. Docker is a service that provides operating-system-level virtualization power and is composed of packages called Docker containers. The container is isolated from the rest of the system, with its own configuration files and libraries, avoiding conflicts in the computer.

Apart from the previously mentioned images, the Docker container includes some other interesting images like a **NGINX**[15] image, which gives a Web Server with some features like load balancing and a **PostgreSQL**[16] image, which is a relational database.

5.1.2 Bayesian network functionality

The focus of this project falls in the Bayesian network functionality offered by *Neurosuites*. It allows a user to learn a Bayesian network structure and parameters from a data set, and work over the learned network. Several learning algorithms are provided to the user and an option to upload directly the learned structure and/or parameters of the network is also available. The main goal of *Neurosuites* as far as this project is concerned, is to allow researchers to work over gene regulatory networks, studying them. Starting from a the data set with genome data, the idea is to create a Bayesian network that represents the genes as nodes, with a Gaussian probability distribution for every node's parameters. With this data, setting evidences to the nodes about the state of certain genes Neurosuites gives back the effect these changes produces in the network. This information can be used by researchers for example to analyze the effect of a certain drug in a human body, studying its possible counter effects.

[13] `https://git-scm.com/`

[14] `https://www.docker.com/`

[15] `https://www.nginx.com/`

[16] `https://www.postgresql.org/`

5.2 Visualization in *Neurosuites* before this project

This point is used to explain the visualization state of *Neurosuites'* Bayesian network section before the project and serves as a way to understand why the changes made are necessary for the tool. It is important to consider that *Neurosuites* is an application with already different tools available. In Table 5.1, the visualization buttons provided in *Neurosuites* are enumerated.

View options	**Highlight nodes/ edges**	**On click node**	**Scale options**	**Layout size**	**Export as image**	**Change layout**
Show/hide labels	Edges thickness dependent of weights	Show Markov blanket	Nodes size	Graph width		Dot
Full screen	Nodes size dependent of markov blanket	Show parents	Edge thickness	Graph height		ForceAtlas2 (client)
Reload graph	Nodes size dependent of direct neighbors	Show children				ForceAtlas2
Clean notifications		Show connections info				ForceAtlas2 (client)
		Show node parameters				Circular
						Grid

Table 5.1: Functionalities of *Neurosuites'* Bayesian network visualization section.

Apart from this, options to **zoom on a node** or **highlight a group of nodes** which should be given to the application as additional parameters are provided. The functionality provided by each button is the following:

- **View options**: general options for visualization of the network.
- **Highlight nodes/edges**: the first option (edges thickness dependent on weights) changes the size of the edges between nodes depending on the weights, which is useful to see which connections are more important. Other options are similar, making the node size dependent on a certain condition.
- **On click node**: different options of visualization that are applied when a node is clicked.
- **Scale options**: change the size of the nodes and edges. This option is useful when large networks are displayed, since having millions of edges or big nodes can make the visualization impossible.

- **Layout size**: change the width and height of the network.
- **Export as image**: export the network as an image
- **Change layout**: several layout options. Depending on the dataset and the purpose of the study, a certain layout can be more useful than others.

These elements are the visualization tools available in *Neurosuites*. In Section 5.4, the contributions made are explained. The available functionality was not enough to solve the problem of visualizing large networks, so several changes were required.

5.3 Gene regulatory network problem

Neurosuites' Bayesian networks functionality aims to be useful for solving a problem with gene regulation networks. This problem is the study of changes in human genes when certain states are observed in the genome. A possible situation could be, having a drug, studying the changes in the genes related to the drug. Several illnesses are known to be related to certain genes, so it could be studied if a certain drug could lead to develop a disease.

Data from tens of thousands of genes is collected from experiments and then it is introduced in the application. A Bayesian network structure and parameters can be learned from the data and with the network built, the study can be conducted. The interest of studying gene regulatory networks using Bayesian networks originates in the easy interpretation provided by them, representing each node the expression of a certain gene and the edges the regulatory relationships between the genes. The problem with the visualization of this network arises because of the huge amount of nodes and connections that have to be displayed, showing the need for new techniques of visualization and interpretation.

The problem is not only purely visual, but also about developing a tool with an interactive, manageable and easy to use interface, that allows a researcher to extract knowledge with ease. Developing new techniques of visualization is necessary in this situation, so the points explained in Section 5.4 are added to Neurouistes' functionality to make it viable.

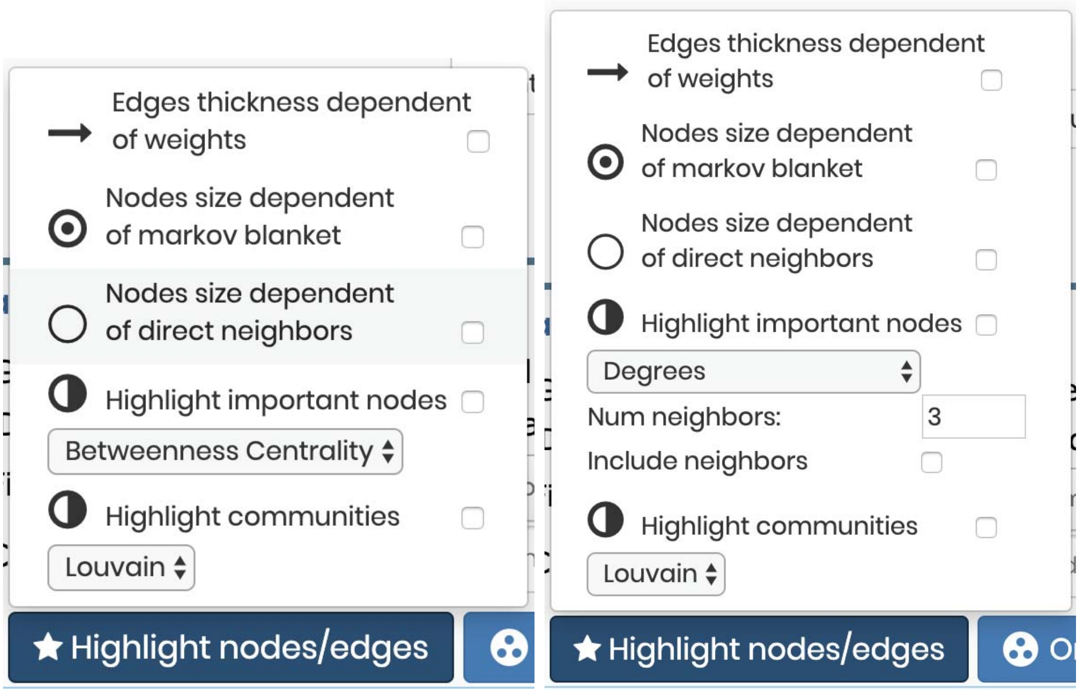
5.4 Application to the visualization of huge networks and results

In this section, the analysis of how the previously explained improvements in visualization are added to *Neurosuites* is conducted, introducing the changes made in the application and the results of each of them.

5.4.1 Nodes' importance in the network

For the purpose of computing the node importance in the network, two different approaches are studied and implemented. The first approach is the importance degree of a certain node taking into account the number of neighbors and the second approach is the implementation of betweenness centrality algorithm, which computes the importance of a certain node by studying the number of shortest paths that cross the node.

For the addition of this functionality, a new option in the drop-down menu entitled Highlight nodes/edges is appended, and also Highlight important nodes (see Figure 5.2) and giving an option to choose one of the previously commented algorithms.


Edges thickness dependent of weights □
Nodes size dependent of markov blanket □
Nodes size dependent of direct neighbors □
Highlight important nodes □
Betweenness Centrality
Highlight communities □
Louvain
Highlight nodes/edges
Edges thickness dependent of weights □
Nodes size dependent of markov blanket □
Nodes size dependent of direct neighbors □
Highlight important nodes □
Degrees
Include neighbors □
Highlight communities □
Louvain
Highlight nodes/edges
O

Num neighbors: 3

Figure 5.2: New options added to the drop-down menu.

For the first algorithm, the importance degree of the nodes, a threshold number of nodes is required, which would be the minimum number of neighbors that a node should have to make it important. Another option is the possibility of adding the neighbors of the node. In the case of having this option selected, the nodes' neighbors are also colored in the same color and when removing the selection, only the main node is colored, making it flexible.

In Figure 5.3, an example of the visualization of the importance degree algorithm is shown. The threshold is 200 neighbors without including neighbors. It can be seen that just five nodes, with a bigger size, have more than 200 neighbors, so these nodes can be shown as important in the network.

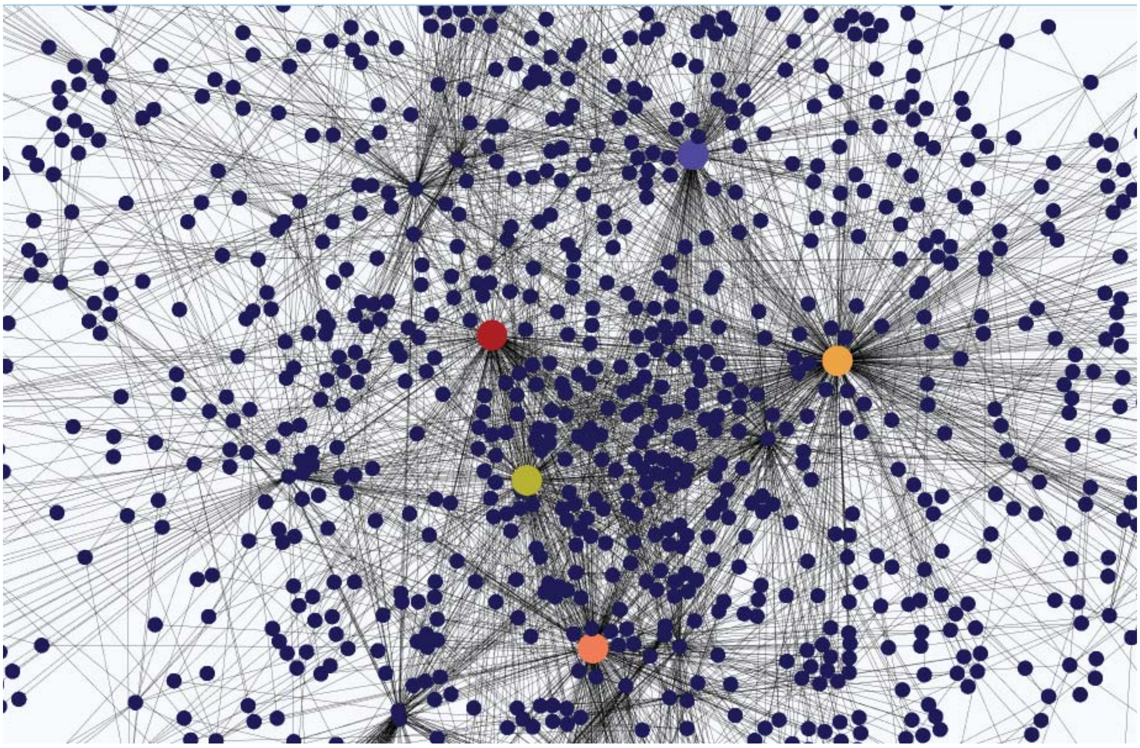


Figure 5.3: Importance degree algorithm with 200 neighbors as threshold and without including neighbors in the visualization. Five nodes have more than 200 neighbors, the bigger ones.

Figure 5.4 shows another example of the visualization of the importance degree algorithm. In this case, the threshold is the same as in the previous example, but including the neighbors. The same 5 nodes can be seen with a bigger size than the rest. The neighbors are coloured in the same color as the main nodes. They are all painted in the same colour because they have common neighbors, so some of the nodes are neighbors of more than one main node.

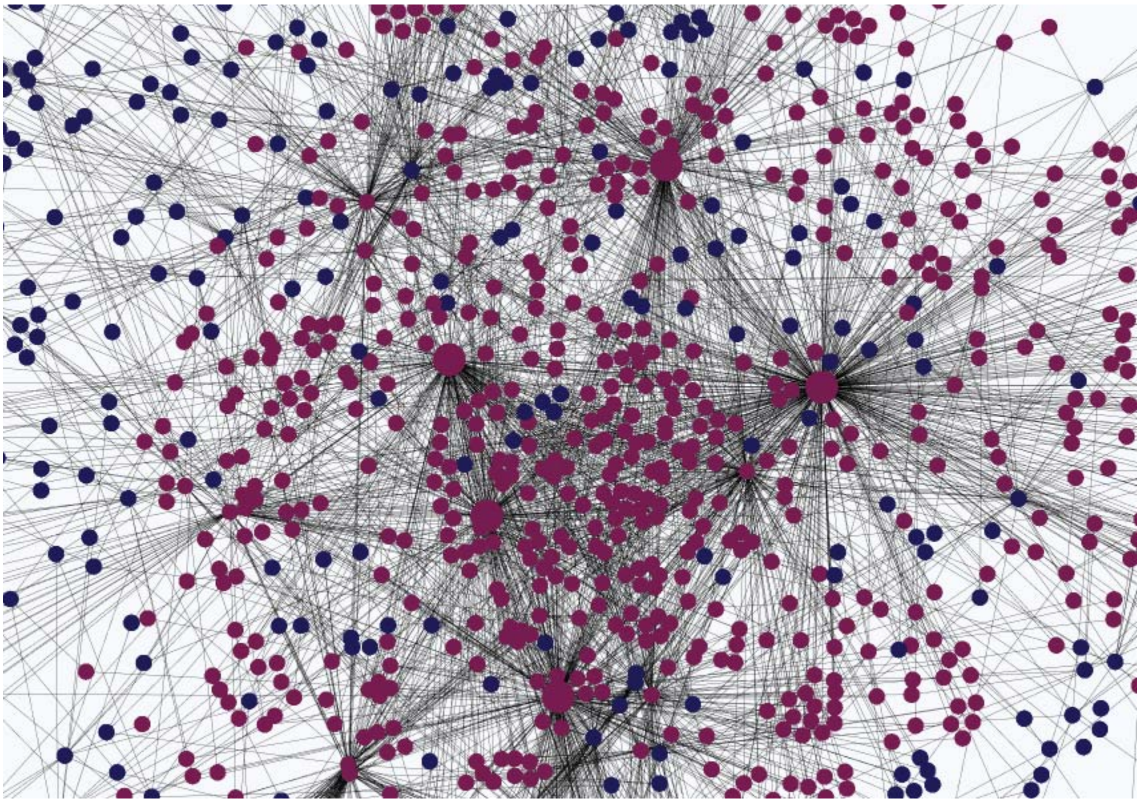


Figure 5.4: Importance degree algorithm with 200 neighbors as threshold and including neighbors in the visualization. The same five nodes are represented with a big size but the color now is the same.

In Figure 5.5, a visualization of the performance of betweenness centrality algorithm is shown. For a better interpretation of the results, the size of the nodes depends on the importance retrieved by the algorithm and the color means that the node betweenness centrality value is greater than 0. The algorithm that computes the Betweenness Centrality over the network is included in the NetworkX library, so that implementation is used by *Neurosuites*.

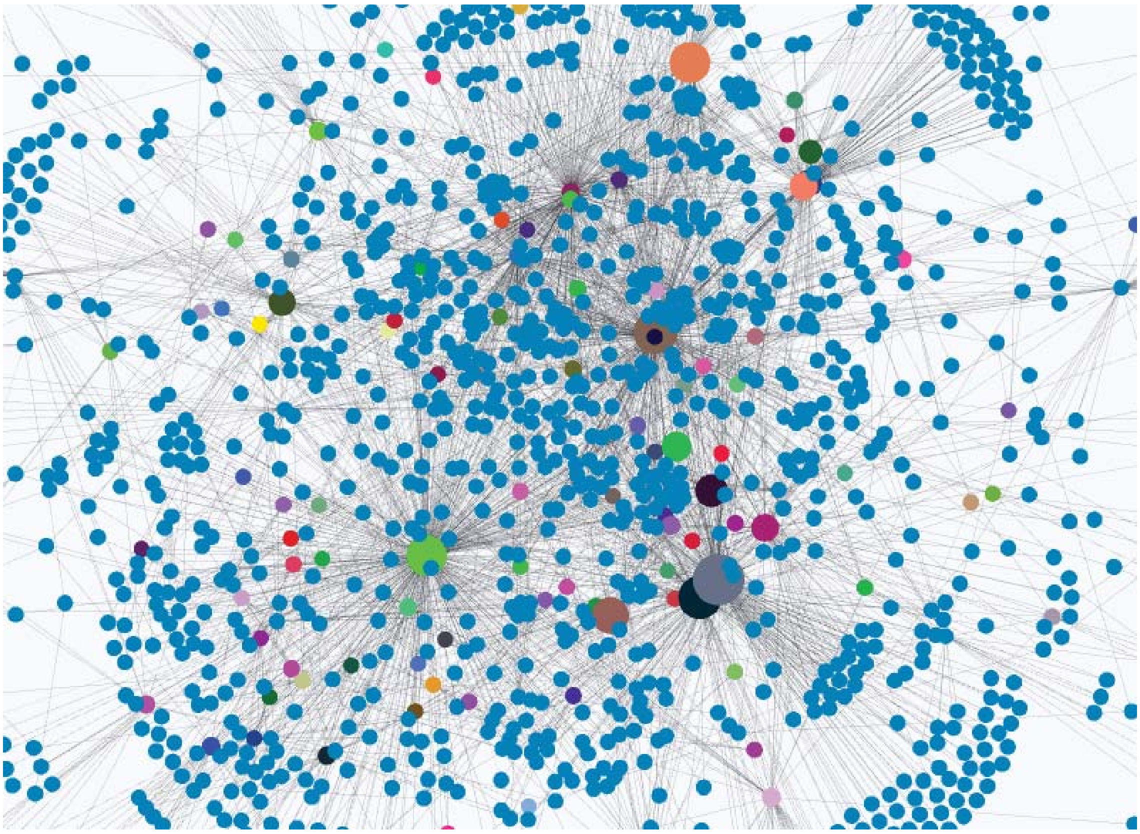
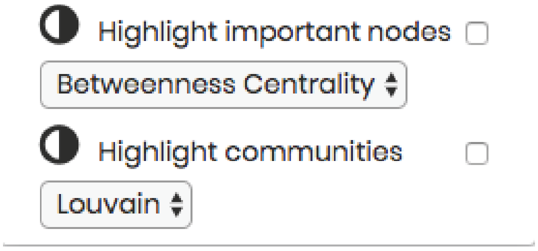


Figure 5.5: Betweenness centrality algorithm visualization over a network. The size of the node represents its importance in the network.

5.4.2 Community detection

Community detection is important when the number of nodes is high, because it makes it easier finding zones in the network that are related to each other. In this case, Louvain community detection algorithm was selected and implemented in *Neurosuites*. The algorithm is already implemented in NetworkX library, so it is straightforward implementing it.



Highlight important nodes □

Betweenness Centrality

Highlight communities □

Louvain

Figure 5.6: Buttons added to the drop-down menu for the new options of highlighting important nodes and communities.

To select the algorithm, a new option highlight communities to the drop-down menu (Figure 5.6). Once the option is selected, the communities are shown in the network, colouring each group of nodes with a different color, making the difference clear.

An example of the Louvain algorithm performance is shown in Figure 5.7, where each community is coloured in a different color. The size of the nodes in the community is also dependent on the importance of the node in the network.

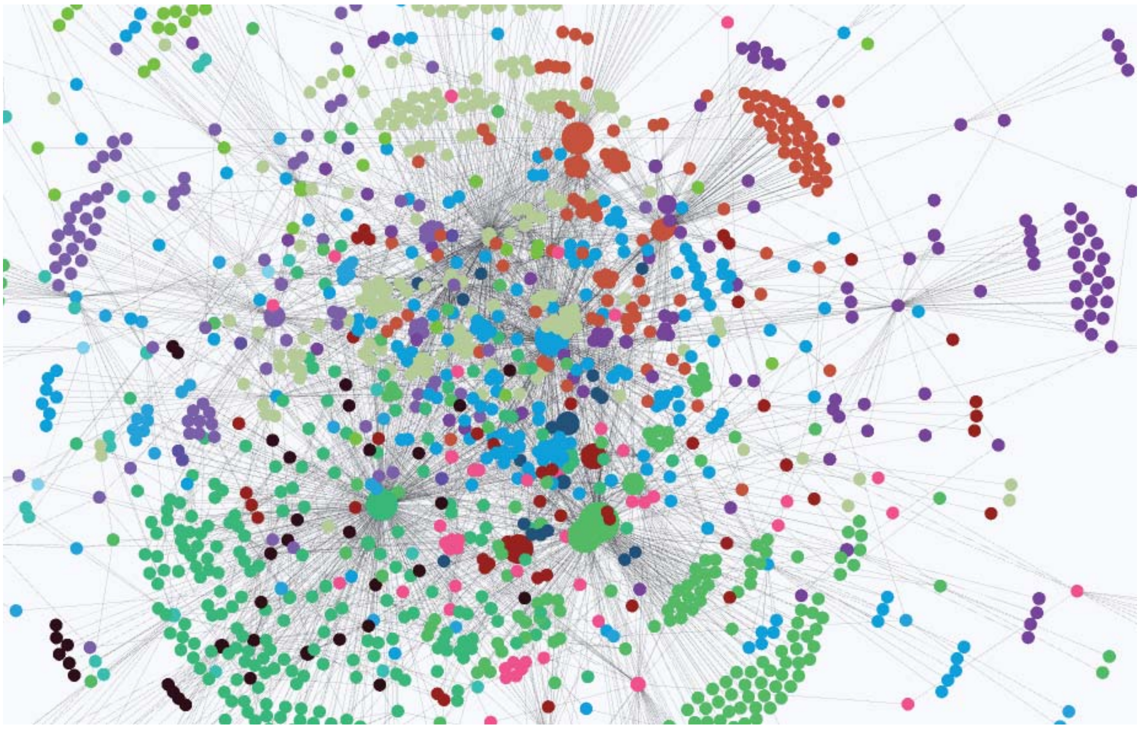


Figure 5.7: Louvain community detection algorithm visualization over a network. Nodes with the the same color represent a group.

The groups created by the Louvain algorithm can be easily visualized by selecting each of them, as Figure 5.8 shows. In Figure 5.9, group 3 is shown.

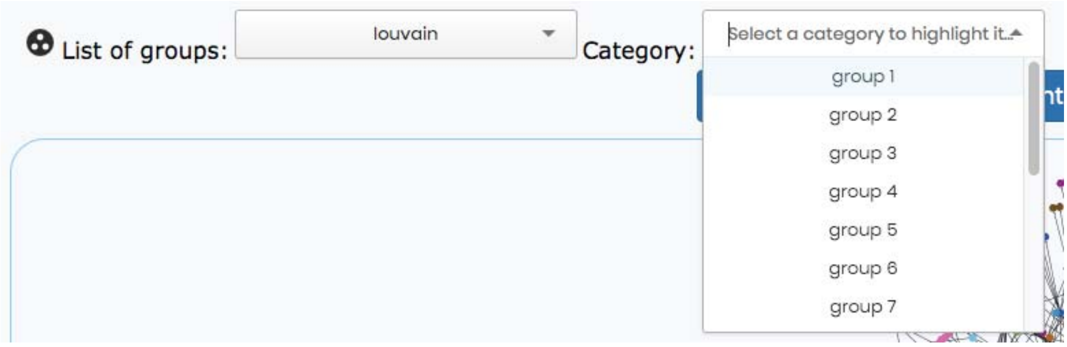


Figure 5.8: Selection of groups created by the Louvain algorithm.

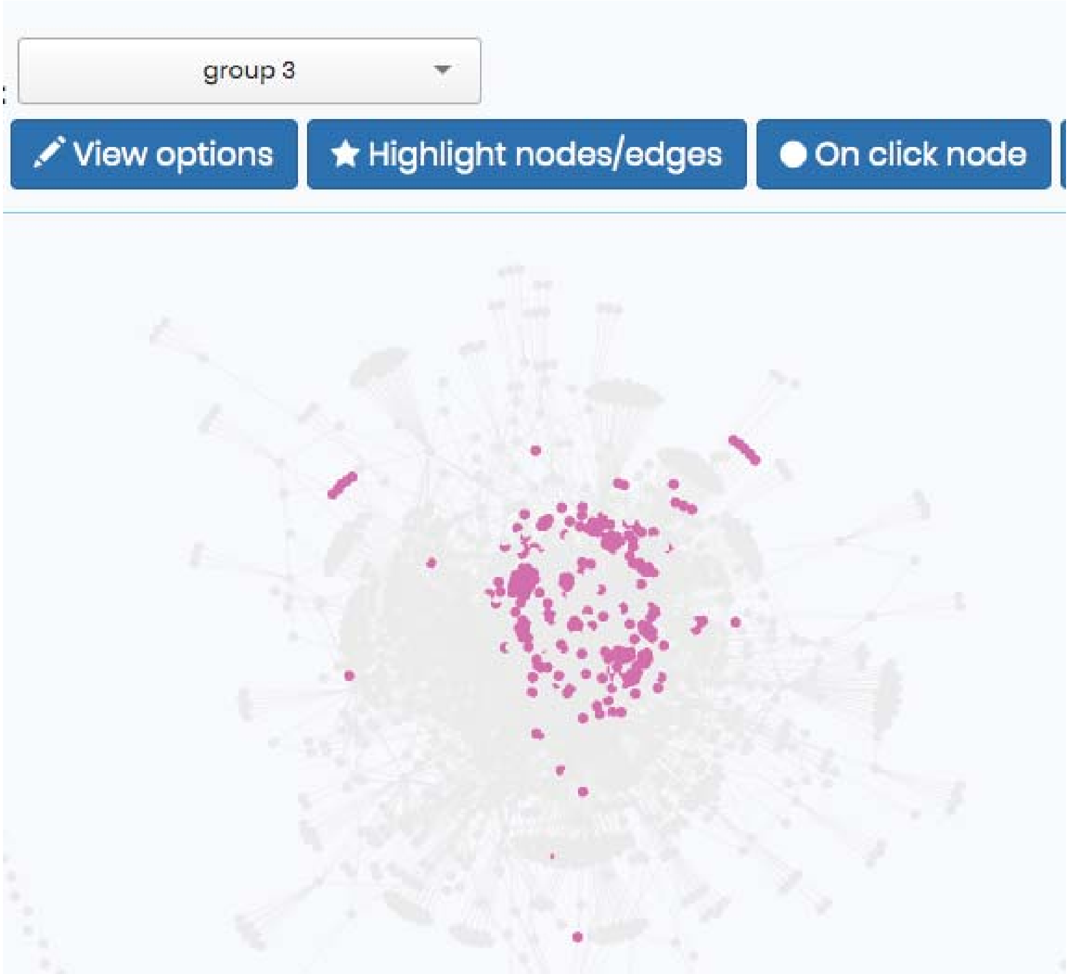


Figure 5.9: Visualization of a group created by the Louvain algorithm. In this case, group 3 is selected and its nodes are highlighted.

5.4.3 Parameter visualization

In parameter visualization, two cases are studied and implemented; the visualization of **discrete parameters** and **continuous parameters**. For both cases, the visualization is hidden unless requested by the user. To request the visualization of a certain node, the user clicks on the node and the parameters are shown. This implementation makes sense because of the probable huge size of the network, so the nodes are just represented as circles with their name while the user does not request the visualization of parameters.

For the first case, discrete parameters, the typical approach is followed, showing the different possible states with their corresponding marginal probabilities. To make the visualization smarter, a bar is displayed for each possible state with a size according to the probability of that certain state. Two buttons to set and clear the evidence are also provided. The result is shown in Figure 5.10, having in the first image the visualization of displaying the parameters and in the second one when the parameters are not displayed.

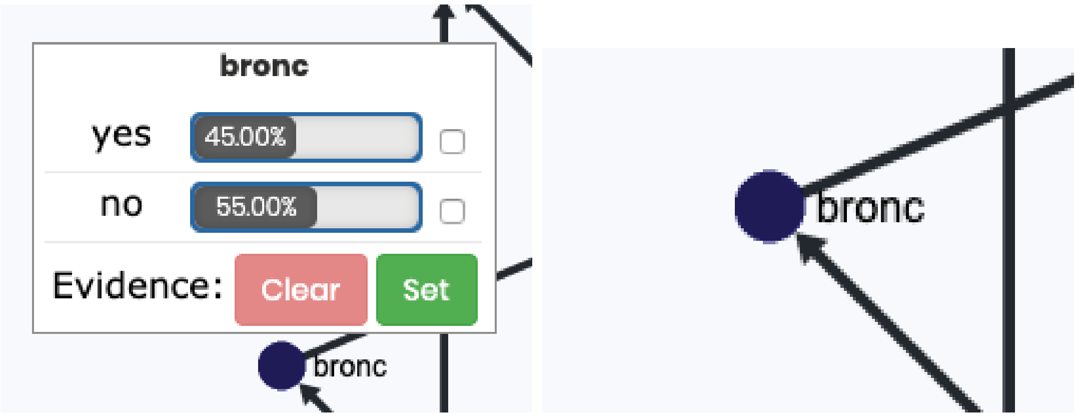


Figure 5.10: Discrete parameter visualization for a node in the network.

Figure 5.11 shows how the visualization changes when the evidence is set to a certain state, making the selected bar 100% and the other possible state 0%.

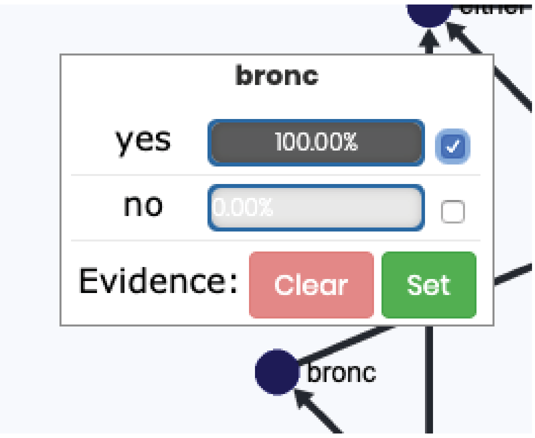
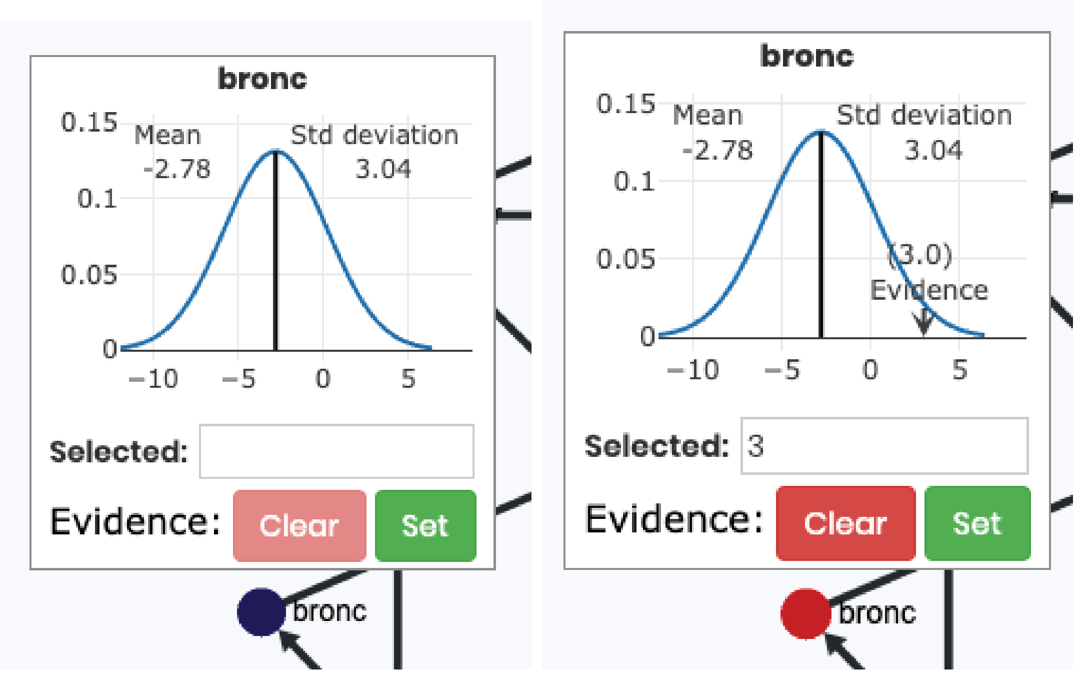


Figure 5.11: Discrete parameter visualization when setting an evidence.

The second case is the visualization of continuous parameters. In the case study presented for this project, the study of gene regulatory networks, this approach is more important than the discrete parameters, since the nodes in the network follow a Gaussian distribution for their parameters. When the node is clicked, a box is shown where the distribution is displayed, having the mean and standard deviation also drawn in the figure. This distribution is displayed thanks to the Plotly library, which makes it easy to display distributions. The data for the distributions are computed in the back-end (Python) and Plotly creates the graphical layout, which is then given to the front-end directly in HTML. This work-flow is interesting since the computational part is done by the back-end, so the front-end has less work to be done. In Figure 5.12 an example is displayed, with the left image showing the normal situation and the right image the situation when the evidence is set.



Selected:

Evidence:

Selected: 3

Evidence:

Figure 5.12: Continuous parameter visualization for a node. In the right image an evidence is set with a value of three for the node.

Setting an evidence can be made directly clicking on the point wanted in the distribution or by writing in a small box its value. Two buttons to set and clear the evidence are also provided. Every node with an evidence set changes its display color to be red, so they can be easily identified by a user.

The evidence transmits its effect to the reachable nodes. The effect transmitted is shown in the change in the probability distribution of those nodes. In *Neurosuites*, both probability distributions (prior and posterior distributions) are displayed in the same plot, so a user can interpret these results (see Figure 5.13). The posterior probability distribution, obtained after computing the effect of the evidence is always displayed in black, whereas the prior probability distribution is kept in blue as in the initial state.

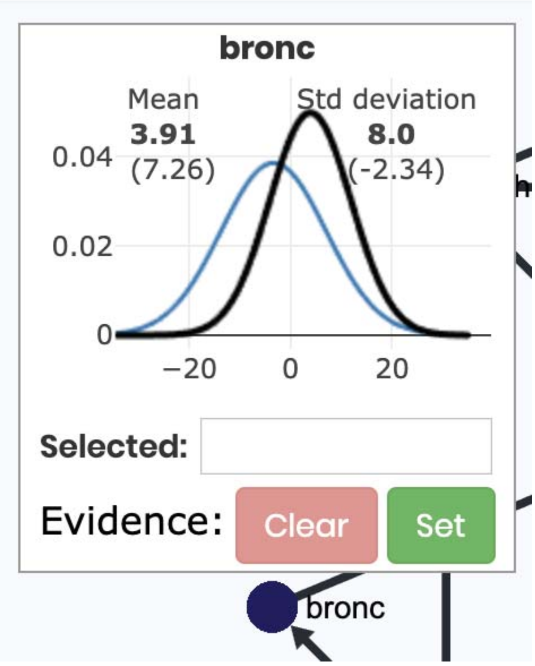


Figure 5.13: Continuous parameter visualization for a node with the difference between prior and posterior probability distributions.

5.4.4 Evidence setting and probability updating

Two different options are implemented to set and clear an evidence in the network, setting an individual evidence or setting a group of nodes evidence.

The first option is the most common and it is implemented in a similar way as in the rest of available software. The node evidence can be set/cleared in the parameter visualization, using the button provided to do so. This option is probably less interesting in the case of studying a large network, since setting evidences for nodes one by one can by unproductive. This work-flow visualization was displayed in Figures 5.10, 5.11 and 5.12.

The second option is probably more interesting and flexible, setting/clearing an evidence for a group of nodes. This option is more interesting because in a large network it is more probable that the user will want to know how some macro-change would affect the network as a whole, so that would be setting a group of nodes' evidence all at the same time. To do so, *Neurosuites* implements an input area where the nodes wanted are selected and then two buttons to set or clear the evidence (Figure 5.14). The same is done for the current group of nodes (Figure 5.15), so in the group of nodes' selection zone, two buttons for setting/clearing the evidences for the selected group are also provided.

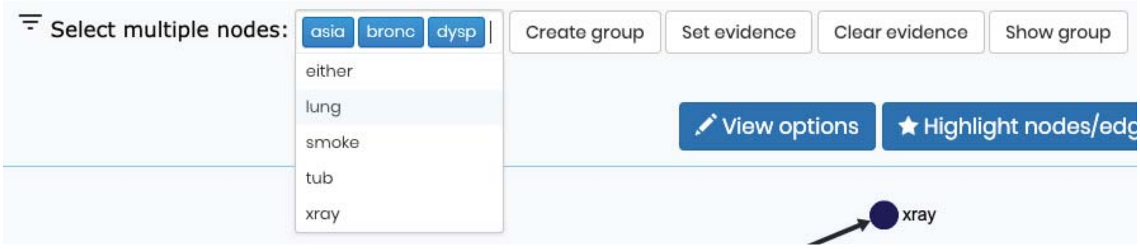

Select multiple nodes: asia bronc dysp

Figure 5.14: Nodes selection from all the available nodes.

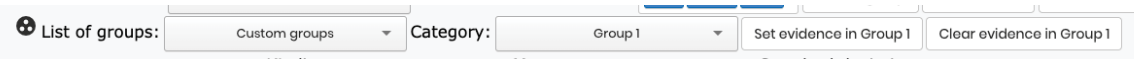
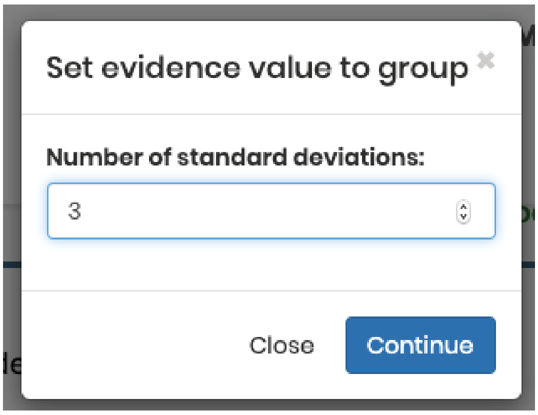

List of groups: Custom groups
Category: Group 1

Figure 5.15: Group options with ability of setting the evidence or clear it.

When the *Set evidence* button is pressed, a modal window is displayed (Figure 5.16), so the user can introduce the number for the evidence. As explained in Section 4.4.2, in this case, the evidence is set selecting the number of standard deviations from the mean. This option is more appealing for a group of nodes, since it allows setting a high or low state for all of them at the same time, which is probably more interesting for continuous distributions than setting a fixed number for all of the distributions. In this case, the focus is on a group of nodes with continuous parameters, since acting on a similar manner for discrete parameters is more complex, since every node parameters can be completely different. In Figure 5.17, the result of setting the evidence in the nodes is shown, with the red nodes indicating those with evidence set.


Set evidence value to group

Number of standard deviations: 3

Figure 5.16: Evidence set by using the number of standard deviations from the mean.

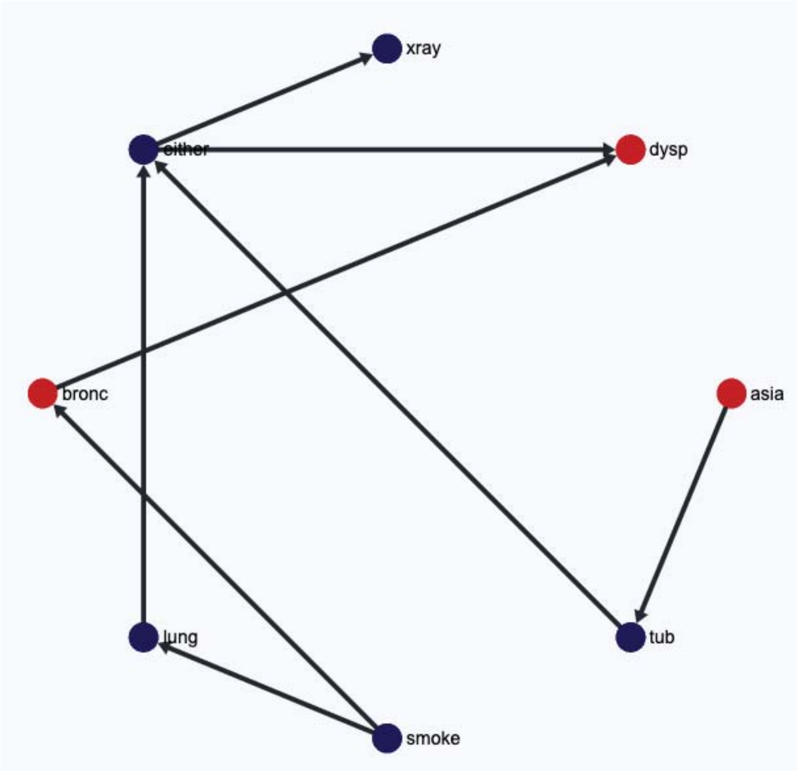


Figure 5.17: Result of evidence setting over the nodes. The red nodes indicate those with evidence set.

5.4.5 Group creation

Neurosuites allows the user to upload a file of additional parameters with information about group of nodes. Apart from this option, it is also interesting to provide the user with the possibility of creating new groups that can be useful for the study of the network. To do so, nodes are selected in an input area (see Figure 5.18). Then, clicking the *Create group* button, a modal window appears (see Figure 5.19). In this modal window, the user can give the group a special name to differentiate it from others and also select a color for the nodes color when displaying the groups.

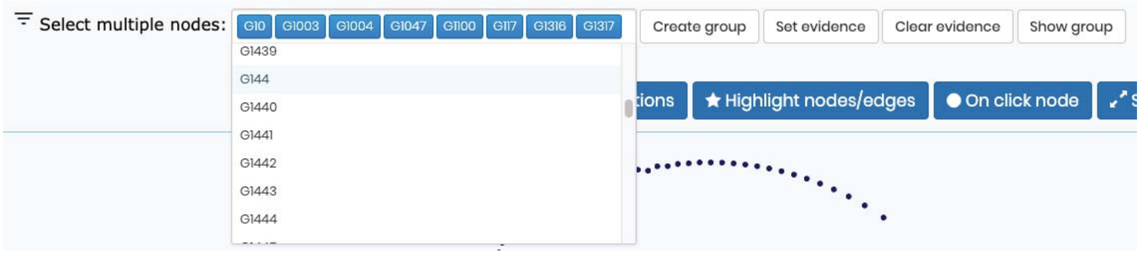
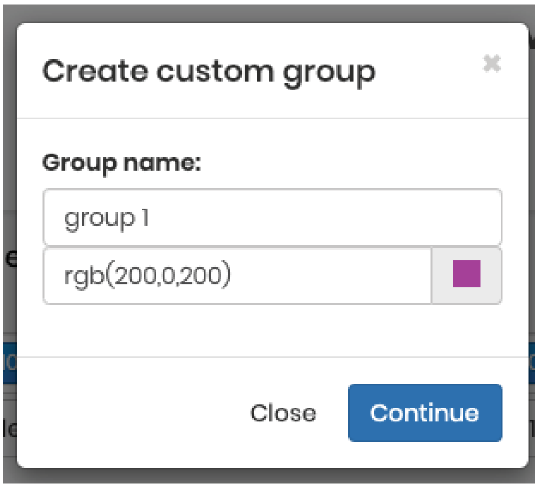


Figure 5.18: Drop-down node selection menu where the nodes that are part of a group are selected to create a group.



Create custom group

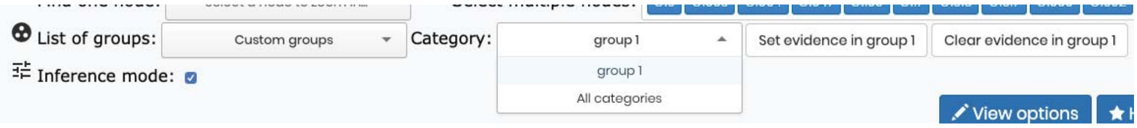
Group name:

group 1

rgb(200,0,200)

Figure 5.19: Modal window where group name and color of the nodes can be selected when creating the group.

Once this process is completed, a new group is added to the list of groups called *Custom groups*, including the different groups created by the user under the option *Category*. These groups are then processed as the rest of the groups, with the possibility of setting/clearing an evidence for a certain group of nodes (see Figure 5.20). When the group is selected from the list, their nodes are displayed in the visualization area (see Figure 5.21).



List of groups: Custom groups

Category: group 1

Inference mode: ☑

Figure 5.20: Group and category selection. Once the category is selected, the evidence can be set or cleared.

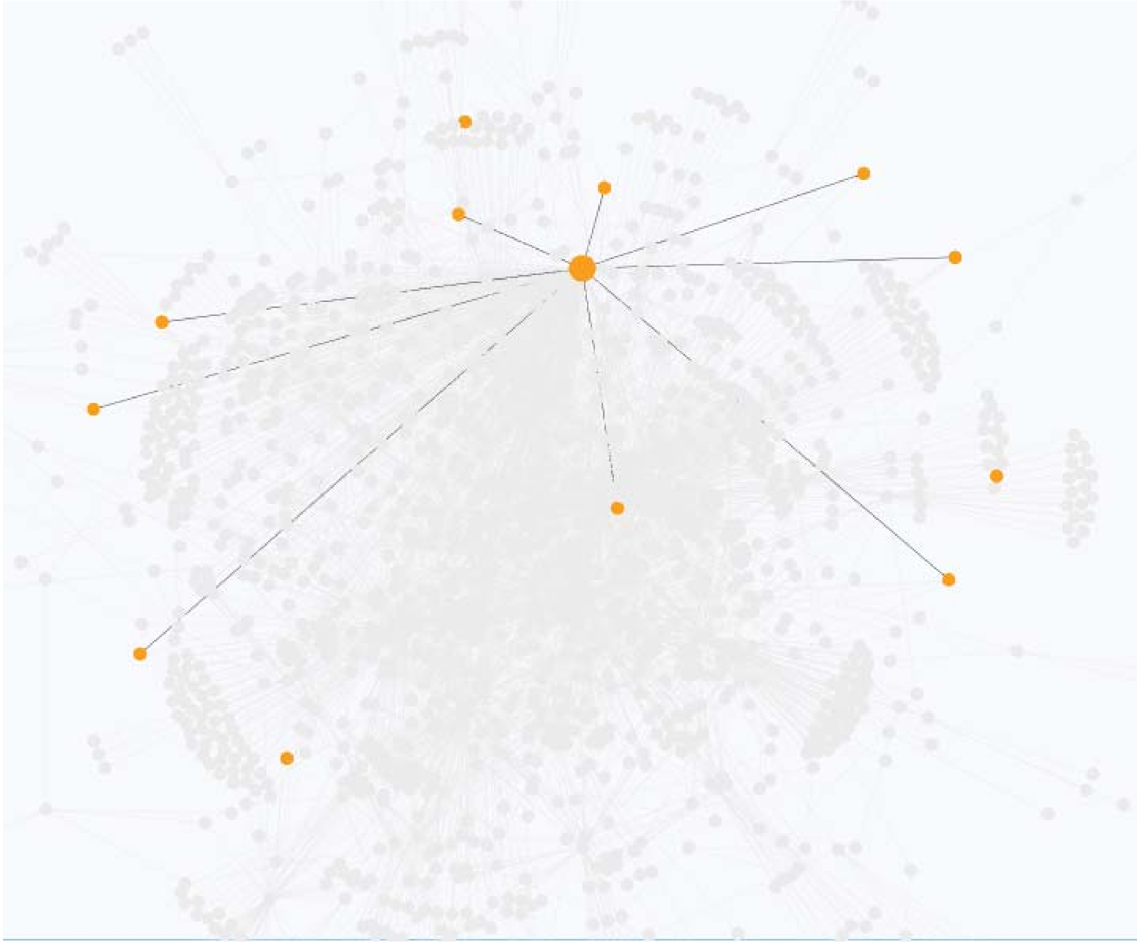


Figure 5.21: Example of visualization of a group. Group nodes and connections between them are displayed.

5.4.6 Network structure comparison

This functionality emerges from the requirement of visualizing the differences between structures, for example structures learned from different algorithms, to compare the performance of a certain algorithm in a particular case.

The structure uploading functionality was already part of the previous version of *Neurosuites*, so now a new button *Add additional structure* can be found next to the area to add the network structure as shown in Figure 5.22. When this button is pressed, a new area is displayed, where the user can add the second structure. Once this is finished, they are shown in the visualization section.

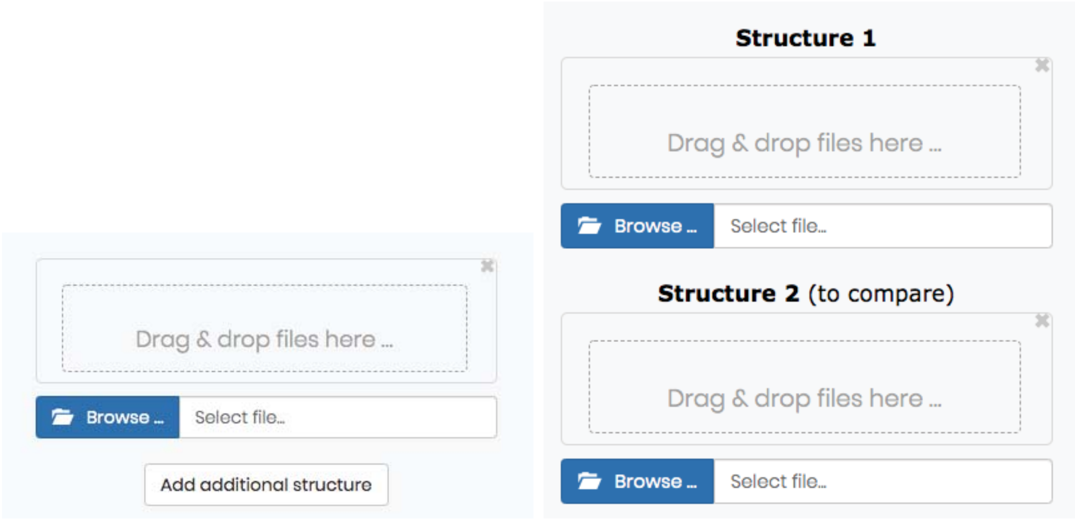


Figure 5.22: Structure 1 and 2 upload sections with the buttons to select the corresponding file or with an area to drag and drop them. The screen on the right appears when *Add additional structure* is clicked.

When both structures are displayed (see Figure 5.23), a new button *Select structure* is added (see Figure 5.24). This button allows four displays of the network: *Both* (all the edges at the same time), *Structure 1*, *Structure 2*and *Common edges* (only common edges between structures). When *Both* is selected, three cases are distinguished, edges that are part of both structures are displayed in a common color, edges that belong to Structure 1 are colored the same and similarly for edges in Structure 2 (see Figure 5.25).

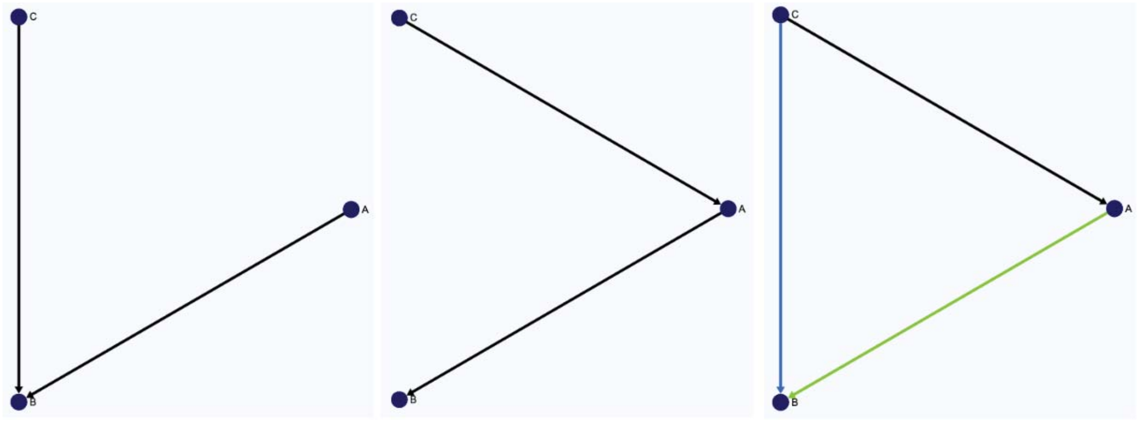


Figure 5.23: Small example of performance for visualizing two structures. First and second structures represent Structure 1 and Structure 2. In the picture on the right, black color represents Structure 1, blue Structure 2 and green represents an edge that is part of both structures.

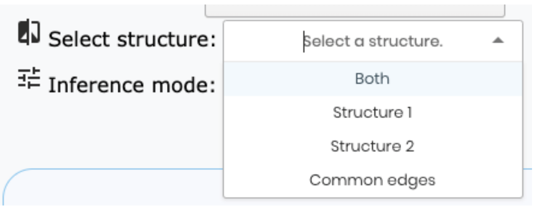


Figure 5.24: Menu for selecting structure comparison. Four options are available: Both, Structure 1, Structure 2 and Common edges.

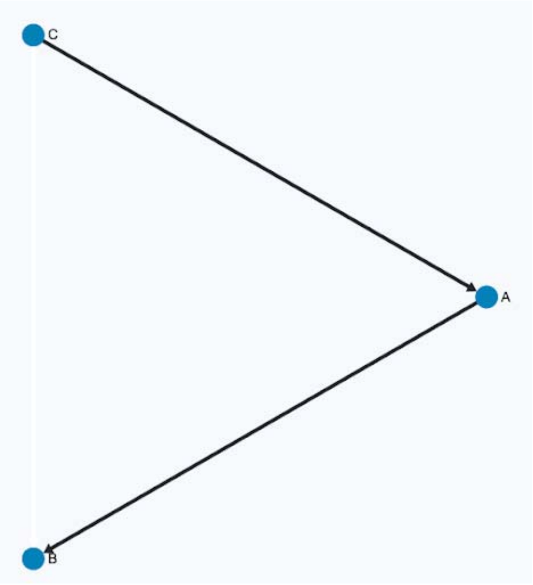


Figure 5.25: Selection of Structure 1 for the small example. The black edges shape Structure 1 and the white edge represents an edge that is not in the current featured structure but it is part of another one.

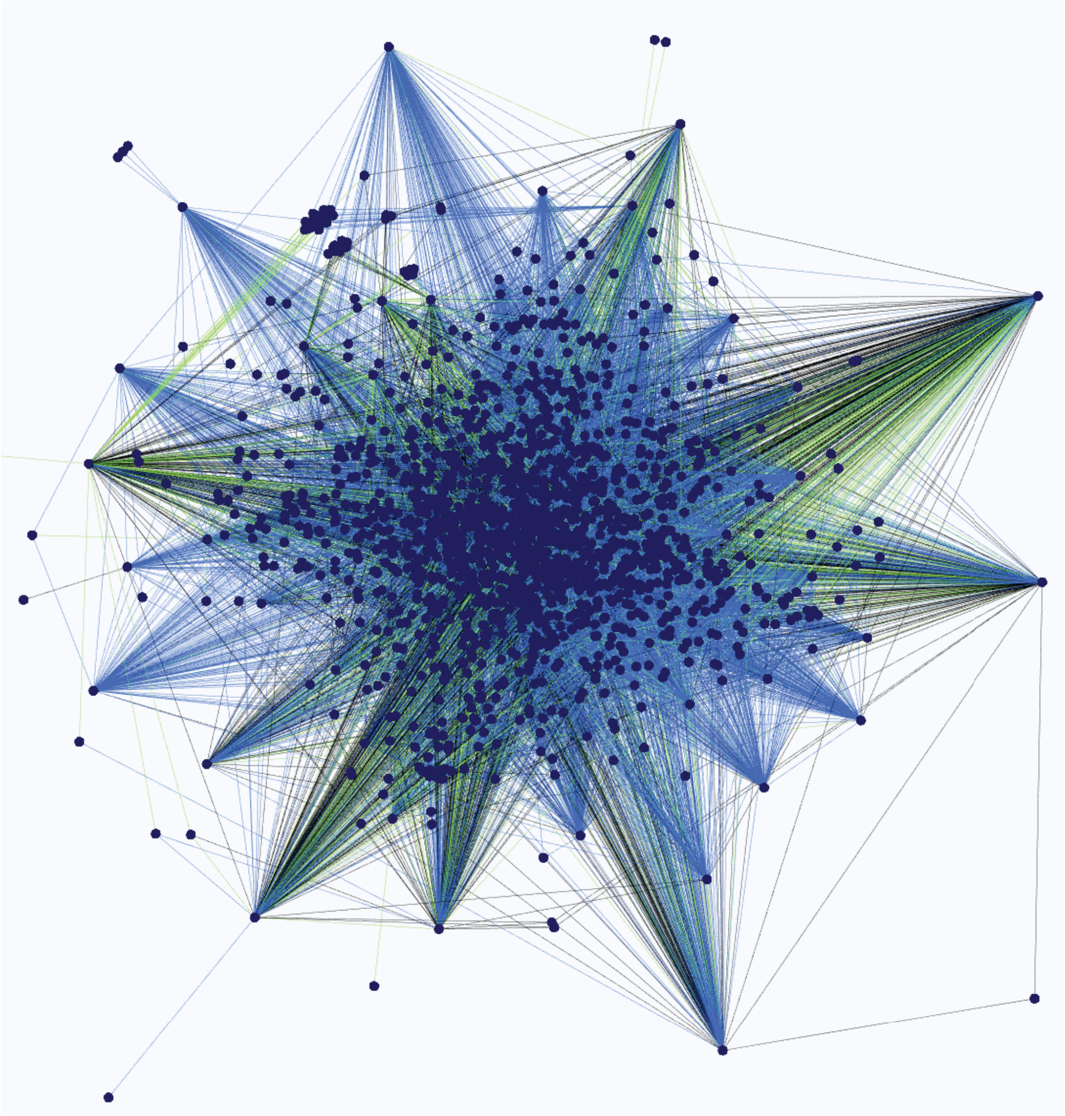
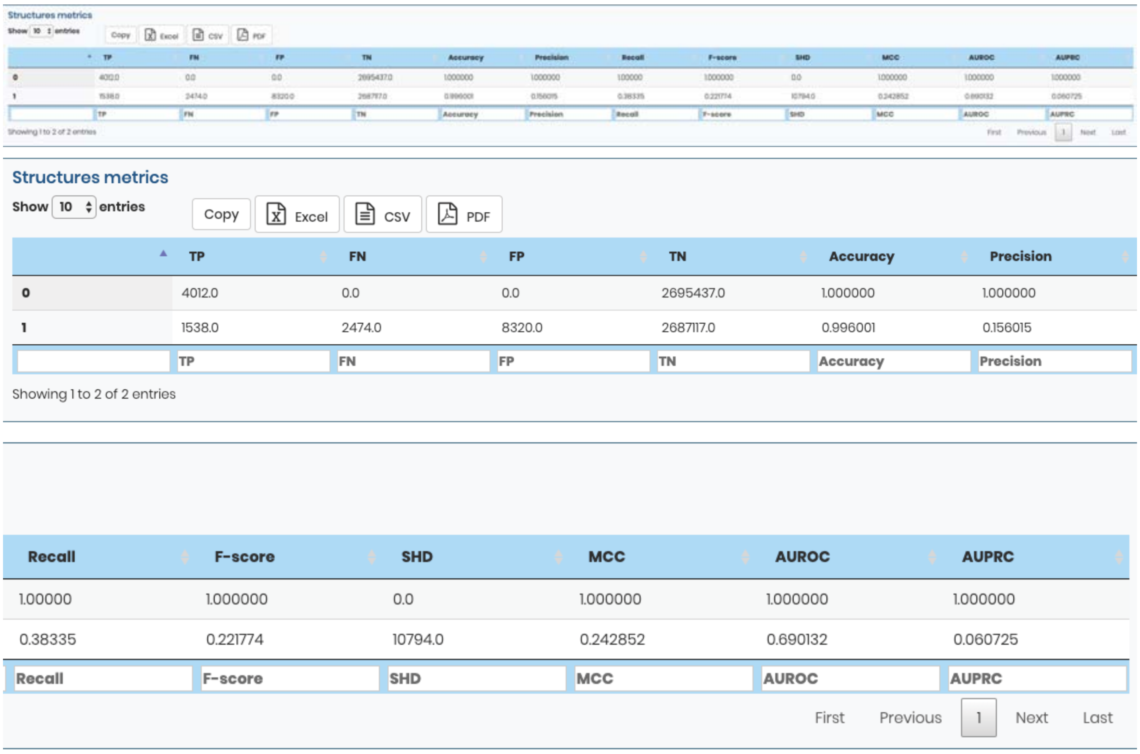


Figure 5.26: Visualization of 2 structures with a real gene regulatory network.

In Figure 5.26, a real example with a network with tens of thousands of nodes is presented. This is a real study comparison between a real gene regulatory network (Structure 1) and one learned from data (Structure 2). Edges that are part of Structure 1 are shown in blue. Edges that are only part of Structure 2 are shown in black and in green edges that are part of both structures are shown.

With this figure, it can be studed how the learning algorithm performed. It can be shown an area on the right with high concentration of green, which means that on that zone the algorithm performed as expected.

Furthermore, several metrics are computed over the structures to check their similarity. The metrics were explained in Section 4.6.1. They are shown as a table, with the title row showing the metrics and the two following rows comparing the structures. This functionality can be really useful as a way of understanding the differences between the structures and combined with the actual visualization of both structures it is a very appropriate way of studying them.



Structures metrics

	TP	FN	FP	TN	Accuracy	Precision	Recall	F-score	SHD	MCC	AUROC	AUPRC
0	4012.0	0.0	0.0	2695437.0	1.000000	1.000000	1.00000	1.000000	0.0	1.000000	1.000000	1.000000
1	1538.0	2474.0	8320.0	2687117.0	0.996001	0.156015	0.38335	0.221774	10794.0	0.242852	0.690132	0.060725

Showing 1 to 2 of 2 entries

Figure 5.27: Table of metrics to compare the structures shown. The metrics for the two different structures are shown and options to export the data generated to a file are also available. The first image is the complete table and the second and third are a zoomed version.

5.4.7 Probability update visualization in huge networks

One of the major problems in the study of large networks is visualizing and analyzing the changes made by the evidences that are added to the network. In order to understand the changes produced in the network, some metrics are introduced and studied to compute which area of the network is more affected by a certain change in the evidences. These metrics are, for the change produced in a certain node, the Kullback–Leibler divergence between the prior and the posterior probabilities and the difference in the mean and standard deviation, and for a group of nodes just the Kullback–Leibler divergence (for the multivariate probability distributions) in the group studied.

After setting the evidences, a new drop-down menu is displayed, where the nodes are shown with the amount of change for each metric, sorting the list of nodes according to this value. These buttons are displayed in Figure 5.28.

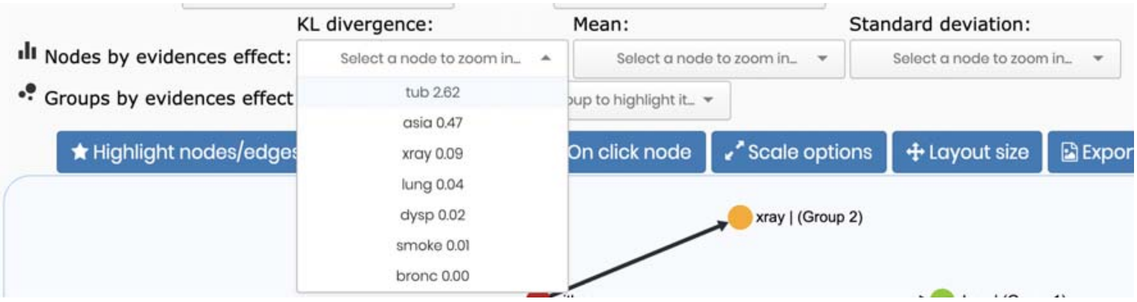


Figure 5.28: Evidences effect in the node sorted by the Kullback–Leibler divergence. The node is represented by its name and the amount of change in the distribution computed using the Kullback–Leibler divergence.

Similarly, for the list of groups, all of them are presented in a drop-down menu computing the Kullback-Leibler divergence, sorting them by the amount of change experimented. This can be seen in Figure 5.29.

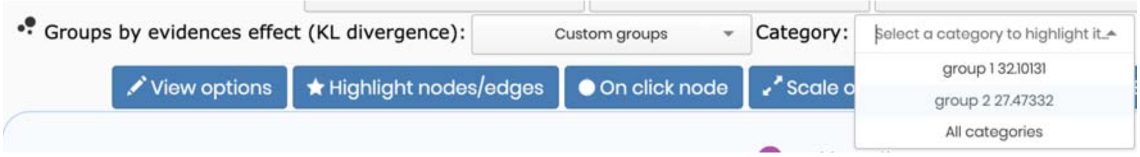


Figure 5.29: Drop-down menu for the selection of groups based on that KL-divergence after some evidences have been set.

Both new buttons intend to highlight the group or zoom on the selected node in case of node selection. A general view of how this functionality looks like in *Neurosuites* is in Figure 5.30.

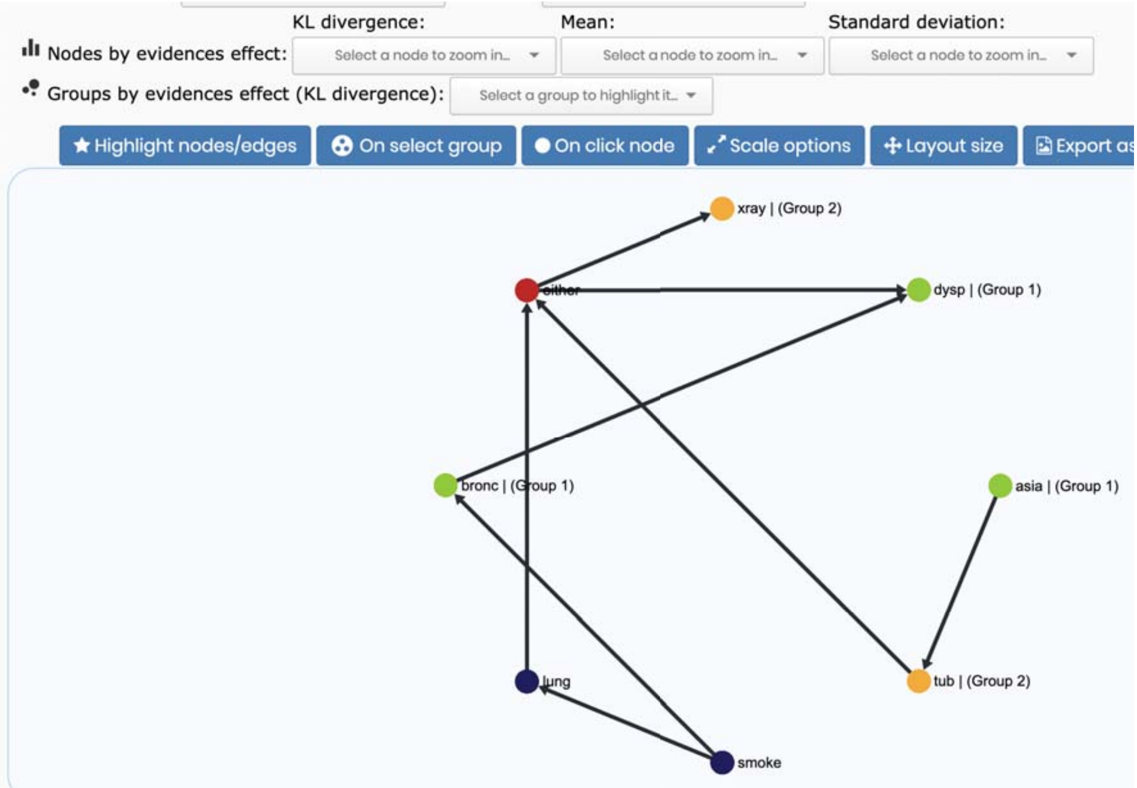
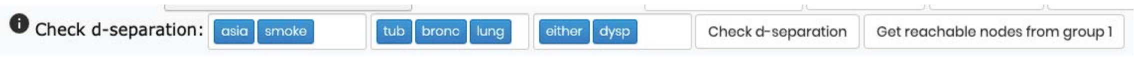


Figure 5.30: Visualization menu with options for zooming on nodes depending on the amount of change they have experimented. Options for KL divergence, mean and standard deviation.

5.4.8 D-separation

D-separation is useful to check the conditional independence between triplets of nodes. This property is included in *Neurosuites*. The interface generated is integrated by three different drop-down selectors where the user selects the nodes to study and a button to call the method that studies the D-separation between them (see Figure 5.31). Another button is also presented that retrieves the reachable nodes from the nodes in the first group of selected ones, the one on the left (group 1).

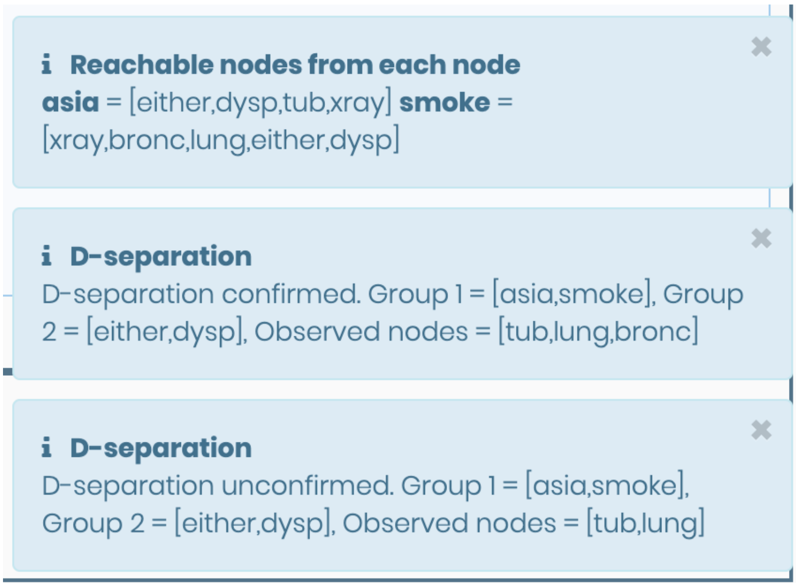


Check d-separation: asia smoke | tub bronc lung | either dysp

Figure 5.31: D-separation menu in *Neurosuites*

The implementation is based on the algorithm proposed by Koller and Friedman called *Reachable procedure* (see Section 3.1.1). This algorithm retrieves reachable nodes from a starting point to an end point given the observed nodes, studying if the paths are active or inactive. If the algorithm finds an active path while checking the graph, it does not include that node in the group of reachable nodes. Once the algorithm is finished, all the reachable nodes are found.

The result is computed and sent to the web, where a notification is created with the value obtained, as Figure 5.32 shows. In this figure, the notification for both methods is shown, the d-separation is checked and the list of reachable nodes for every node in the group 1 is shown.



i Reachable nodes from each node
asia = [either,dysp,tub,xray] **smoke** = [xray,bronc,lung,either,dysp]

i D-separation
D-separation confirmed. Group 1 = [asia,smoke], Group 2 = [either,dysp], Observed nodes = [tub,lung,bronc]

i D-separation
D-separation unconfirmed. Group 1 = [asia,smoke], Group 2 = [either,dysp], Observed nodes = [tub,lung]

Figure 5.32: D-separation notification in *Neurosuites* that shows if the d-separation is confirmed and the nodes are conditionally independent.

5.4.9 Probability clustering visualization. Mixture of Gaussians

Neurosuites Bayesian networks visualization techniques are extensible and useful not only for traditional Bayesian networks. Another models can be introduced to the software and the techniques developed can still be useful in their study. A particular example of this fact is probabilistic clustering visualization. In this case, the different clusters are uploaded to the software tool and the same previously explained techniques can be used in its study. Another interesting techniques have been also developed in *Neurosuites* to make the software tools extensible to the study of different models, but they are not covered in this project because its implementation was still in development when this project finished.

The only technique that is covered here is the visualization of the mixture of Gaussians. The already developed techniques allow the visualization of the different clusters distribution for a particular node. The new changes include the common visualization of all this Gaussian probability distributions for a given node (see Figure 5.33), where each cluster is drawn in a different color, with information about each mean and standard deviation.

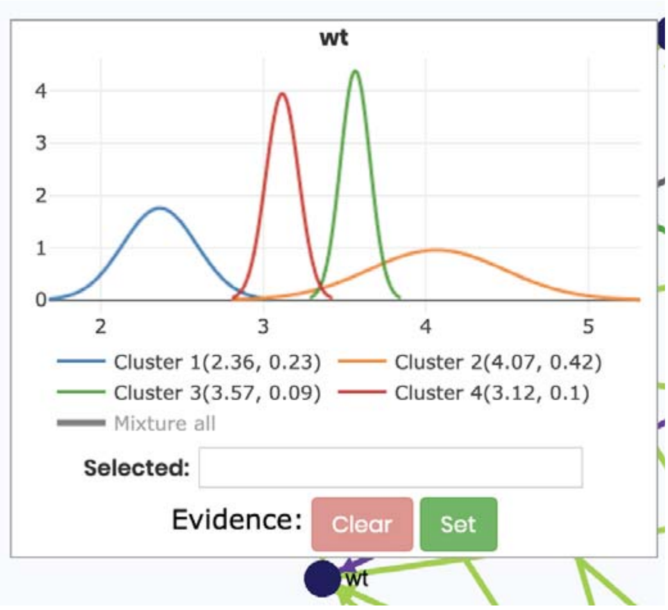


Figure 5.33: Gaussian probability distributions for a certain node.

Having this information, it is interesting to add the mixture of Gaussians of the distributions drawn (see Figure 5.34). It is computed from the clusters' distributions and the weights for each distribution is the number of instances of particular cluster divided by the total amount of instances. The weights summation is 1. Once this distribution is computed, it is drawn using a black line (see Figure 5.34), with thicker lines than the other lines.

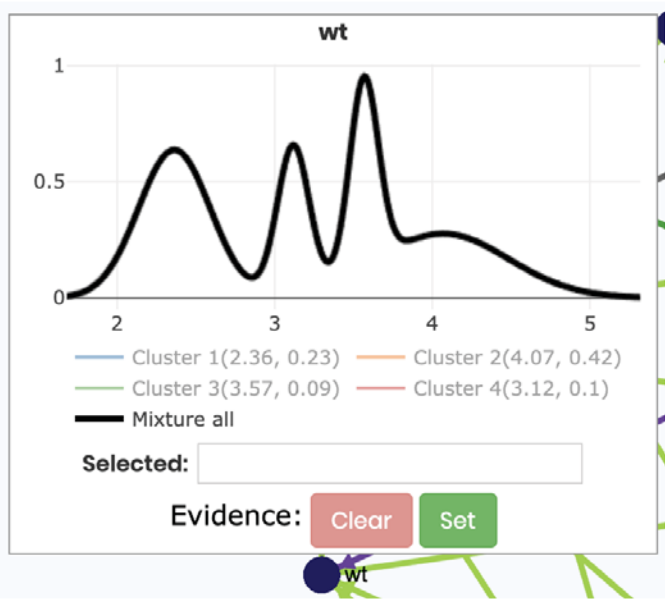
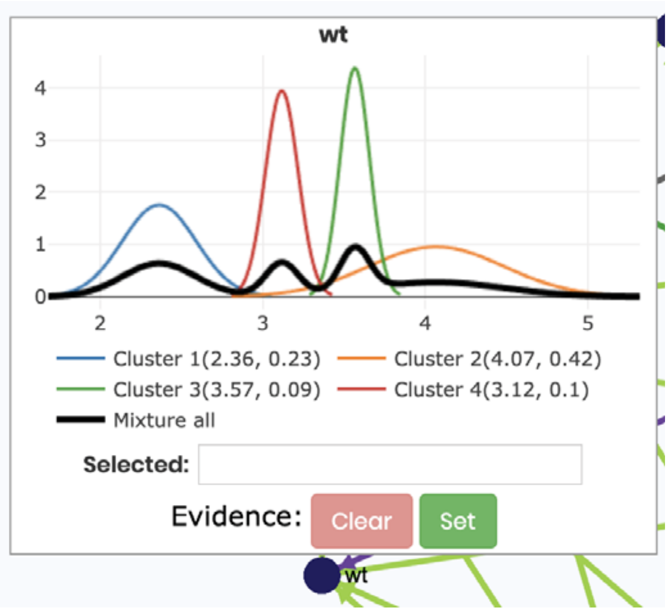


Figure 5.34: Gaussian mixture.

A common view of the distributions and the mixture of the Gaussians is also possible (see Figure 5.35) and can be useful to extract information about the clusters.



Selected:

Figure 5.35: Gaussians mixture applied over four clusters.

Chapter 6

Conclusions and future work

This last chapter presents the conclusions and results reached after the development of the project with a discussion about them and some possible future lines of work. A personal remark is also presented.

6.1 Conclusions

The main conclusions reached though the development of this project are the following (the first number represents the chapter):

- **Conclusion 1.1**: a review of the state of the art in visualization, interpretation and explanation in machine learning has been conducted starting from the foundation accompanied with a historical perspective, understanding that is a line of research that researcher are currently still studying.

- **Conclusion 1.2**: the problem observed in machine learning related to the black box models and the need for new techniques or the improvement of the current ones has been presented.

- **Conclusion 1.3**: a study of the two main tendencies in explanation and interpretation in machine learning has been discussed, presenting the two main lines of research. The first one creating models that can understand black boxes and the second one creating new algorithms from scratch that are inherently interpretable and explainable. As it has been studied, it is a problem still to be solved.

- **Conclusion 2.1**: The use of Bayesian networks as an interpretable model has been presented, making and alternative to the current discussion in machine learning which is focused on other algorithms, like deep learning. Bayesian networks are an acceptable model for this type of problem that wnats to be studied.

- **Conclusion 2.2**: A review of needed concepts about Bayesian networks has been studied, with a revision of the current state of the art in visualization in order to set the starting point from which to start working.

- **Conclusion 3.1**: a set of improvements in the visualization and interpretation of Bayesian network has been presented and studied in detail in order to understand how these changes improve the current visualization techniques. The need for new visualization techniques from different fields like network science has been exposed.

- **Conclusion 4.1**: *Neurosuites* has been introduced, with a focus on its Bayesian network functionality and the problem that it aims to solve. The power of the software tool has been presented.

- **Conclusion 4.2**: the actual implementation of the improvements in visualization of Bayesian networks has been studied and included in *Neurosuites* and the results have also been shown. It has been studied the difficulty of the problem to be solved and the improvements have been found useful in the study of large Bayesian networks.

6.2 Discussion and future work

In a first stage of the project, a review of the state of the art in visualization, interpretation and explanation in machine learning has been conducted (**Conclusion 1.1**). This part of the project focuses on the whole spectrum of machine learning alternatives and not only to Bayesian networks. Starting from a broad view of what is understood as a good explanation to examples in AI of how this problem is approached.

The current problem observed in machine learning has also been studied in (**Conclusion 1.2**), with an introduction to black boxes and the problem of explanation and interpretation. The problem is not really related with Bayesian networks itself but it has become trendy in machine learning in general because many of the models used and studied today have this problem in common. The improvement in visualization and intrepretation in Bayesian networks studied and implemented in this project aims to show the ability of this algorithm to easily be understood by a user thanks to the techniques implemented.

It has been studied in (**Conclusion 1.3**) that the main tendency in explanation and interpretation in AI aims to create models that can extract the explanation directly from the black box model or with another different technique but without creating new models that are not black boxes.

On the other hand, another tendency tries to create models that are innerly explainable and interpretable on the whole life cycle. This part of the research is focus on all the different techniques in machine learning and not only in one of them as is more common in the other tendency.

As it has been studied, this field has a wide spectrum of work to be done and is not yet a solved problem, so in upcoming years is expected that a lot of effort will be directed into this field. The study of this field is truly key in general in AI, because the results provided by a model have to be interpretable and explainable so a human can make use of them. If this problem is not solved, many interesting areas where machine learning and AI would be helpful are out of the scope of applicability. No AI application can be used in a risk area of application without a well studied intepretability and explainability of why the system is acting as it is doing.

The answer is not clear yet on what is the best direction to follow in the research of this area, but it is essential that this question is studied in order to avoid another AI winter (period where the research interest in the field is low mainly due to pessimism over the possible discoveries).

A review of the area of Bayesian network has been conducted (**Conclusion 2.1**), with special focus on its graphical part that is the part that has experimented an improvement during this project. A study of the current state of the art in the visualization and interpretation in Bayesian networks has retrieved some interesting points that are important to discuss (**Conclusion 2.2**).

In the first place, the majority of techniques proposed in the area of visualization and interpretation in Bayesian networks are directed to the applicability to really small structures with a few number of nodes. In the case studied during the project, the number of nodes is on a range of tens of thousand, so these techniques are usually not relevant in this scenario.

In the second place, a need for an upgrade was needed and is yet an interesting area where many different new approaches can be introduced. Almost none of the tool available was focused on a macro vision of the network. In this project this point has been studied and techniques related with the whole network have been introduced. Having such an enormous network and with the problem studied of gene regulation networks, it is more important to have a broad view of the network and to give more importance to study areas of the network rather than concrete nodes.

Considering this idea, many improvements have been projected and its viability has been studied in (**Conclusion 3.1**). These proposals have been introduced with the idea of improving the general area of application of visualization in Bayesian networks and with the ideas previously stated as points to be followed. In a future project, this area is yet to be studied and it is open to many different improvements. As a personal though, a macro vision of the network is what is more important to be improved, because the applications of the algorithm are now focused on bigger networks and this view is more interesting when the amount of nodes and connections is huge.

Neurosuites has been introduced and studied as stated in Conclusion 4.1. This tool is still in development and many different improvements could be introduced on it, so this project has just intended to improve it but there is still a long way for improvements. It is important to note that the emergence of more powerful computational components in recent years, with a really important remark to GPUs, has made projects like this with the introduction of more computationally expensive techniques possible. This is important to be note and is an indicator that on upcoming years this area could experiment a great development because of the new possibilities created.

Finally, the different improvements proposed have been introduced to *Neurosuites* (**Conclusion 4.2**). With the ideas previously considered, the tools to visualize and interpret the network have been added, so evidences can be set to a group of nodes all at the same time and the study of how these changes affects the network can also be studied having groups of nodes as object of study. Several changes have been added to the software and the results have shown an improvement in the interpretation and visualization section. The future of visualization and interpretation in Bayesian network is clearly on this area, because the networks are usually bigger than before, so new techniques to visualize and interpret a macro vision of the network are to be studied and developed.

A lot of improvements can be implemented to *Neurosuites* in the future. The different proposals made and implemented to the software tool have made this tool a lot more useful and have improved the visualization and interpretation of Bayesian networks in general but it is also important to note that the ideas implemented have been focused on the problem that was intended to address, the gene regulatory networks. This is important to mention because since the time line of the project is really narrow many ideas that could be helpful have not been studied nor implemented.

This opens a lot of possible future work that could be covered. It would be important for example to complete the visualization of the discrete variables part since is basic in all the different software tools studied but since the focus of this project has been on variables with a continuous probability distribution, that has been the main part of study. Another interesting improvement that is now open after the completion of the project is how to visualize the nodes reachable for every node in a group of nodes, functionality based in the d-separation. The proposal presented here is suitable for small networks but once they are huge a new approximation has to be taken. There is a problem in the visualization of the network when the user zooms in. When this situation appears and a node is moved, it sometimes disappears or moves on a strange manner. This problem has been addresses during this project and some improvements have been reached, but it is not completely solved and it could take a while to be completed because the causes behind this behavior are related to the libraries used, not directly with the project. Having a small window showing where the user is when zooming is another interesting improvement that could be implemented. The addition of multi-window capabilities like having two windows to work on differents areas of the network could also be interesting.

The area of visualization and interpretation in Bayesian networks has experimented a slow development in the actual software tools in recent years but the appearance of these components that make possible an enormous computational power open a new area of possible software applications and additionally a new sector of possible real-world applications. The graphical properties given by Bayesian network give an alternative against the so called black boxes and can be helpful in risk areas of development, as the one studied in the project of gene regulation networks.

6.3 Personal remark

As a final statement, I would like to express how this project has affected myself personally on an academic perspective. The idea of completing a project of this magnitude can be overwhelming on the first stages. It has been clearly challenging to study a problem on this depth, starting from the fact of searching and selecting relevant literature to study and understand the state of the art in the area, the problems observed and how are these problems addressed by other researchers.

The field studied is quite broad and many of the ideas in the literature are not directly related to what I wanted to develop but those ideas have helped in my understanding of the problem. For example, the ideas of XAI that are now trendy can be seen to have little relevance (they are not usually focused in Bayesian networks) in the software project that I was to develop but these ideas are relevant in order to understand what the problem really is and different approaches that have been studied.

Another important point for me has been the fact of making changes to a software tool already under development and production, where I had to understand the structure and the code already developed in order to change it. Not only this point was challenging, but also the case study of gene regulation networks, a new topic hard to understand. Prioritizing possible changes and improvements has taken into consideration where and how those gene regulatory networks are to be used.

This project has also helped me improve my knowledge of all the different ideas covered and has motivated myself to keep studying this passionate field which is full of interesting discoveries to be made.

Bibliography

S. Andreassen, F. V. Jensen, S. K. Andersen, B. Falck, U. Kjærulff, M. Woldbye, A.R. Sørensen, A. Rosenfalck, and F. Jensen. MUNIN: An expert EMG assistant. In *Computer-Aided Electromyography and Expert Systems*, pages 255–277. Pergamon Press, 1989. ISBN 0444811060.

J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`, May 2016. [Online; accessed June-2019].

G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, 1998. ISBN 0133016153.

A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, (35):99–109, 1947.

C. Bielza and P. Larrañaga. Discrete Bayesian network classifiers: A survey. *ACM Computing Surveys*, 47(1):Article 5, 2014.

C. Bielza and P. Larrañaga. Bayesian networks in neuroscience: A survey. *Frontiers in Computational Neuroscience*, 8:131, 2014. ISSN 1662-5188. doi: 10.3389/fncom.2014.00131.

H. L. Bleich. Computer-based consultation: Electrolyte and acid-base disorders. *The American Journal of Medicine*, 53(3):285 – 291, 1972. ISSN 0002-9343. doi: https://doi.org/10.1016/0002-9343(72)90170-2.

V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):10008, Oct 2008. doi: 10.1088/1742-5468/2008/10/P10008.

N. Bostrom and E. Yudkowsky. The Ethics of Artificial Intelligence. In *The Cambridge Handbook of Artificial Intelligence*, page 316–334. Cambridge University Press, 2014. doi: 10.1017/CBO9781139046855.020.

B. Buchanan and E. Shortliffe. *Rule-based Expert System – The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, 01 1984.

C. Champion and C. Elkan. Visualizing the consequences of evidence in Bayesian networks. *CoRR*, abs/1707.00791, 2017.

K. C. Chang and Z. Tian. Market analysis and trading strategies with Bayesian networks. In *Proceedings of the 18th International Conference on Information Fusion (Fusion)*, pages 1922–1929. IEEE, July 2015.

C. Chiang, P. Shaughnessy, G. Livingston, and G. Grinstein. Visualizing graphical probabilistic models. *techreport 2005-017, UML CS*, 2005.

D. Chicco. Ten quick tips for machine learning in computational biology. *BioData Mining*, 10(1), 2017. doi: 10.1186/s13040-017-0155-3.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, May 1968. ISSN 0018-9448. doi: 10.1109/TIT.1968.1054142.

G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2):393 – 405, 1990. ISSN 0004-3702. doi: https://doi.org/10.1016/0004-3702(90)90060-D.

G. F. Cooper and E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, Oct 1992. doi: 10.1007/BF00994110.

M. Cossalter, O. J. Mengshoel, and T. Selker. Visualizing and understanding large-scale Bayesian networks. In *Scalable Integration of Analytics and Visualization*, 2011.

R. G. Cowell. Local propagation in conditional gaussian Bayesian networks. *Journal of Machine Learning Research*, 6:1517–1550, December 2005. ISSN 1532-4435.

J. Dalgaard, J. Peña, and T. Kocka. Featuring multiple local optima to assist the user in the interpretation of induced Bayesian network models. In *Proceedings of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2004)*, pages 1683–1690, 2004.

A. L. Dannenberg, A. R. Shapiro, and J. Fries. Enhancement of clinical predictive ability by computer consultation. *Methods of information in medicine*, 18:10–4, 02 1979. doi: 10.1007/978-1-4613-8554-7_13.

J. Dastin. Amazon scraps secret AI recruiting tool that showed bias against women. `https://reut.rs/2Od9fPr`, Oct 2018. [Online; accessed June-2019].

L. M. de Campos, J. M. Fernández-Luna, and J. F. Huete. A Bayesian network approach to hybrid recommending systems. In *Proceedings of the 11th International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 2158–2165, 2006.

A. Dempster, N. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38, 01 1977.

M. Du, N. Liu, and X. Hu. Techniques for interpretable machine learning. *CoRR*, abs/1808.00033, 2018.

Elvira Consortium. Elvira: An environment for creating and using probabilistic graphical models. *Proceedings of the First European Workshop on Probabilistic Graphical Models*, pages 222–230, 01 2002.

E. Ernst. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1909(136), 1909. doi: 10.1515/crll.1909.136.210.

L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35, 1977. doi: 10.2307/3033543.

A. A. Freitas. Comprehensible classification models: A position paper. *SIGKDD Explorations Newsletter*, 15(1):1–10, March 2014. ISSN 1931-0145. doi: 10.1145/2594473.2594475.

L. H. Gilpin, D. Bau, B. Z. Yuan, A.a Bajwa, M. Specter, and L. Kagal. Explaining explanations: An overview of interpretability of machine learning. *arXiv e-prints*, art. arXiv:1806.00069, May 2018.

R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, and F. Giannotti. A survey of methods for explaining black box models. *CoRR*, abs/1802.01933, 2018.

D. Gunning. *Explainable Artificial Intelligence (XAI) Program Update.* DARPA, 2017.

J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, 51:141–154, 04 2011. doi: 10.1016/j.dss.2010.12.003.

N. R. Kadaba and P. P Irani. Visualizing causal semantics using animations. *IEEE Transactions on Visualization and Computer Grpahics*, 13(6):1254–1261, 2007.

D. Koller and N. Friedman. *Probabilistic Graphical Models Principles and Techniques.* The MIT Press, 2009.

P. Krishnan, O. Mengshoel, and T. Selker. Multi-focus and multi-window techniques for interactive network exploration. volume INC 8654, 02 2013. doi: 10.1117/12.2005659.

J. A. Kroll, J. R. Huey, S. Barocas, E. W. Felten, J. R. Reidenberg, D. G. Robinson, and Y. Harlan. Accountable algorithms. In *165 University of Pennsylvania Law Review*, 663. University of Pennsylvania Law School Review, 2016. URL `http://dl.acm.org/citation.cfm?id=647462.726278`.

T. Kulesza, M. Burnett, W-K. Wong, and S. Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 126–137. ACM, January 2015. doi: 10.1145/2678025.2701399. Stumpf, S. et al | ACM 2015.

S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. doi: 10.1214/aoms/1177729694.

C. Lacave. *Explicación de Redes Bayesianas Causales*. PhD thesis, Departamento de Inteligencia Artificial, Universidad Nacional de Educación a Distancia (UNED), 2002.

C. Lacave and F. J. Díez. A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, 17:107–127, 05 2001. doi: 10.1017/S026988890200019X.

C. Lacave, M. Luque, and F. J. Díez. Explanation of Bayesian networks and influence diagrams in Elvira. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 37:952–65, 09 2007. doi: 10.1109/TSMCB.2007.896018.

H. Langseth and L. Portinale. Bayesian networks in reliability. *Reliability Engineering and System Safety*, 92:92–108, 01 2007. doi: 10.1016/j.ress.2005.11.037.

P. Larrañaga. Bayesian networks in action. In *XVIII Conferencia de la Asociación Española para la Inteligencia Artificial*. AEIA, 2018.

S. L. Lauritzen and Frank Jensen. Stable local computation with conditional Gaussian distributions. *Statistics and Computing*, 11(2):191–203, Apr 2001.

Y. Ma and G. Guo. Support vector machines applications. In *Support Vector Machines Applications*, chapter 8, pages 269–299. Springer International Publishing, 2014.

K. Marriott, P. Moulder, L. R. Hope, and C. Twardy. Layout of Bayesian networks. volume 38, pages 97–106, 01 2005.

K. Marriott, P. Sbarski, T. van Gelder, D. Prager, and A. Bulka. Hi-trees and their layout. *IEEE Transactions on Visualization and Computer Graphics*, 17(3): 290–304, March 2011. ISSN 1077-2626. doi: 10.1109/TVCG.2010.45.

S. Mason. Gaussian-mixture-example. `https://commons.wikimedia.org/wiki/File:Gaussian-mixture-example.svg`, Jun 2012. URL `https://commons.wikimedia.org/wiki/File:Gaussian-mixture-example.svg`. [Online; accessed June, 2019].

B. W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta*, page 442–451, 1975.

T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *CoRR*, abs/1706.07269, 2017.

T. Miller, P. Howe, and L. Sonenberg. Explainable AI: Beware of inmates running the asylum or: How I learnt to stop worrying and love the social and behavioural sciences. *CoRR*, abs/1712.00547, 2017.

W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu. Interpretable machine learning: definitions, methods, and applications. *arXiv e-prints*, art. arXiv:1901.04592, Jan 2019.

National Research Council. *Network Science*. The National Academies Press, Washington, DC, 2005. ISBN 978-0-309-10026-7. doi: 10.17226/11516.

M. Neil, N. Fenton, and L. Nielson. Building large-scale Bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, September 2000. ISSN 0269-8889. doi: 10.1017/S0269888900003039.

G. Nott. 'Explainable Artificial Intelligence': Cracking open the black box of AI. `https://www.computerworld.com.au/article/617359/explainable-artificial-intelligence-cracking-open-black-box-ai/`, 2017. [Online; accessed June, 2019].

K. P. Murphy. *A Brief Introduction to Graphical Models and Bayesian Networks*. UC Berkeley, January 1998.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988. ISBN 0-934613-73-7.

M. T. Ribeiro, S. Singh, and C. Guestrin. "why should I trust you?": Explaining the Predictions of Any Classifier. *CoRR*, abs/1602.04938, 2016.

C. A. Robertson and J. G. Fryer. Some descriptive properties of normal mixtures. *Skandinavisk Aktuarietidskrift (Scandinavian Actuarial Journal)*, 1969:137–146, 01 1969. doi: 10.1080/03461238.1969.10404590.

C. Rudin. Please Stop Explaining Black Box Models for High Stakes Decisions. *arXiv e-prints*, art. arXiv:1811.10154, Nov 2018.

C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1: 206–215, May 2019.

C. Rudin, C. Wang, and B. Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv e-prints*, art. arXiv:1811.00731, Nov 2018.

S. Rüping. *Learning interpretable models.* PhD thesis, Universität Dortmund am Fachbereich Informatik, 2006.

W. Samek, T. Wiegand, and K. Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv e-prints*, art. arXiv:1708.08296, Aug 2017.

M. A. Shwe, D. Middleton, B.and Heckerman, M. Henrion, E. Horvitz, H. Lehmann, and G. F. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base: I. the probabilistic model and inference algorithms. *Methods of Information in Medicine*, 30, 02 1992. doi: 10.1055/s-0038-1634846.

P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, 1991. doi: 10.1177/089443939100900106.

K. Stubbs, P. Hinds, and D. Wettergreen. Autonomy and common ground in human-robot interaction: A field study. *IEEE Intelligent Systems*, 22:42–50, 04 2007. doi: 10.1109/MIS.2007.21.

K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical systems structure. *IEEE Transactions on Systems, Man and Cybernetics*, 11:109 – 125, 03 1981. doi: 10.1109/TSMC.1981.4308636.

H. Wang and M. J. Druzdzel. User interface tools for navigation in conditional probability tables and elicitation of probabilities in Bayesian networks. page 617–625, 2000.

R. Wexler. When a computer program keeps you in jail. `https://www.nytimes.com/2017/06/13/opinion/how-computers-are-harming-criminal-justice.html`, 2017. [Online; accessed June, 2019].

L. Williams and R. S. Amant. A visualization technique for Bayesian modeling. *11th International Conference on Intelligent User Interface, 2006*, page 263–278, 2006.

J Woodward. Scientific explanation. In *The Stanford Encyclopedia of Philosophy.* Metaphysics Research Lab, Stanford University, 2017.

G. Yap, A. Tan, and H. Pang. Explaining inferences in Bayesian networks. *Applied Intelligence*, 29:263–278, 12 2008. doi: 10.1007/s10489-007-0093-8.

C. Yuan, H. Lim, and T. Lu. Most relevant explanation in Bayesian networks. *CoRR*, abs/1401.3893, 2014.

J. Zapata-Rivera and J. E. Greer. *Visualizing and Inspecting Bayesian Belief Models.* IOS Press, 1999.

V. Zharkova and L.C. Jain. *Artificial Intelligence in Recognition and Classification of Astrophysical and Medical Images*. Studies in Computational Intelligence. Springer, 2007. ISBN 9783540475118. URL `https://books.google.es/books?id=ukjUQBe6s64C`.

G. G. Zweig. *Speech Recognition with Dynamic Bayesian Networks*. PhD thesis, University of California, Berkeley, 1998.

Appendix A

Project management

This appendix is intended to give a view of the timeline and methodology the project has followed and to show the tools used and a brief documentation on how to use the project effectively.

A.1 Timeline and methodology

The project has been divided into several phases with different objectives to complete. The amount of work has been at the beginning set to be spread over around four months of full-time work on the development of the proposals. The project started in January 2019 with the definition of goals to complete, and a review of literature and concepts related to the area.

From February to June 2019 the workload has been based on the development of the proposals for the project in the software application *Neurosuites*. The end of May and June have been dedicated to write the thesis document and complete the final proposals for the application.

After finishing writing the thesis, the thesis defense has been prepared and by the end of July, on 22nd, it will be defended. An overview of the timeline can be shown in Figure A.1. This timeline is simplified but it mainly reflects the reality of how the development has been done.

The methodology followed during this project has been iterative and incremental. The project is divided into several temporal block or iterations. Each iteration is executed and evaluated when completed to add it to the actual project. This methodology is common in software development and is useful in this situation. Every Friday, a meeting has been held with the advisors, where the work done has been discussed and new tasks have been proposed for the following week. With this methodology, every week the project has feedback to be analyzed and the direction of the project can be conducted based on it.

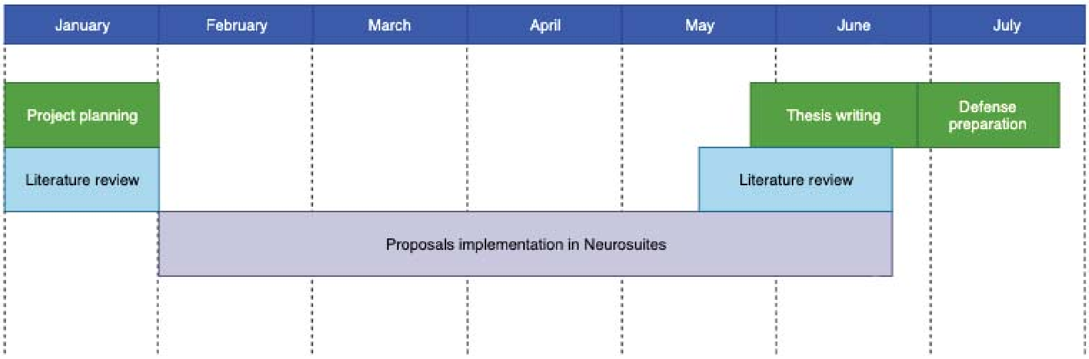


Figure A.1: Approximate timeline followed during the project

A.2 Tools used and documentation

During the course of this project, the following tools have been used:

- **Python3 and JavaScript**: main programming languages used on the development of the project. Python for the backend part of the application and JavaScript for the frontend.

- **Django**: Framework used in the backend.

- **Docker**: Virtualization software based on containers used in the project to deploy the web.

- **HTML, CSS, JQuery and Bootstrap**: Main frontend technologies used and libraries.

- **Sigma JS**: JavaScript library devoted to draw networks and used in the project to draw the Bayesian networks.

- **NetworkX, Plotly, NumPy, SciPy, sci-kit learn and Pandas**: Main python libraries used in the backend. NetworkX is focused on the study of graphs, Plotly in the visualization of plots, and the rest are common libraries in machine learning.

- **LaTeX**: Document preparation system used in the development of the thesis documentation.

- **Google Scholar**: Main search engine used in the search for literature.

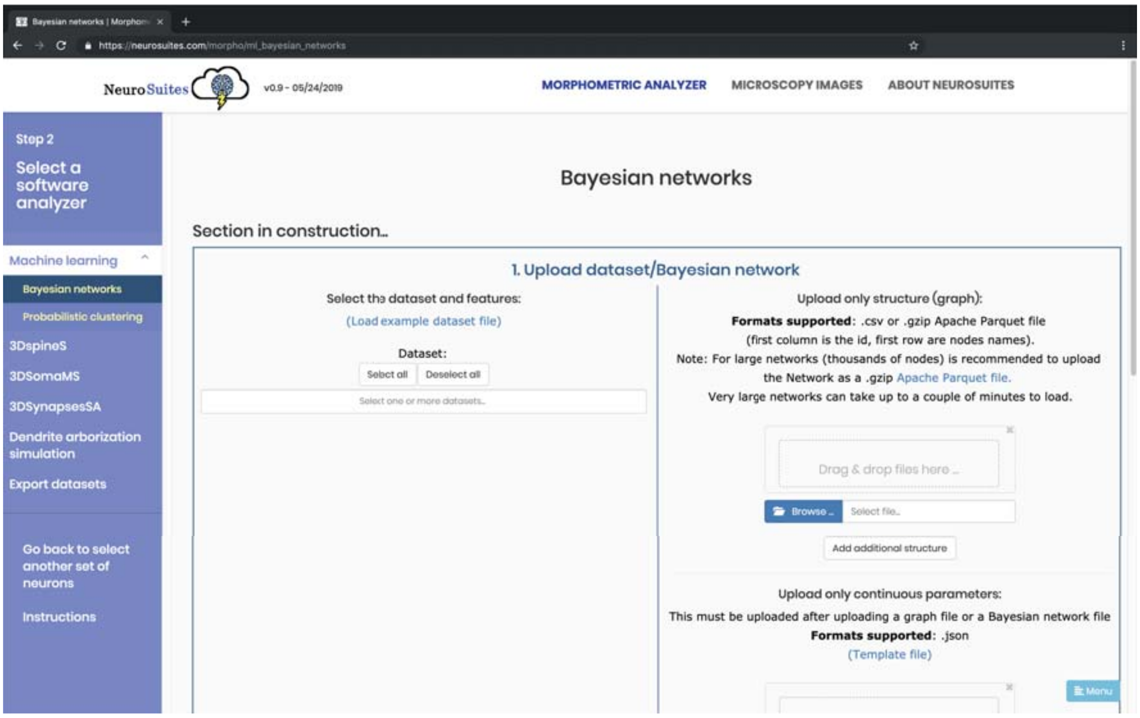
- **Git and Gitlab**: Git has been used as version-control system and a Gitlab repository is the platform where the project is stored.

- **PyCharm and Google Chrome**: The first one is the IDE (Integrated Development Environment) used to write the code for the application and the second one is the browser where the test of the functionality has mainly been conducted.

The project *Neurosuites* can be accessed at `https://neurosuites.com/`. The changes developed and the most interesting part of the project can be found at `https://neurosuites.com/morpho/ml_bayesian_networks` and `https://neurosuites.com/morpho/ml_probabilistic_clustering`.

In Figure A.2, a general view of how the *Neurosuites* website looks like in the Bayesian network section is shown. On the left hand side a menu to access the different sections is shown.

Figure A.3 shows the detail of how the visualization area for Bayesian network looks like in a full-screen screenshot.



Bayesian networks

Section in construction...

1. Upload dataset/Bayesian network

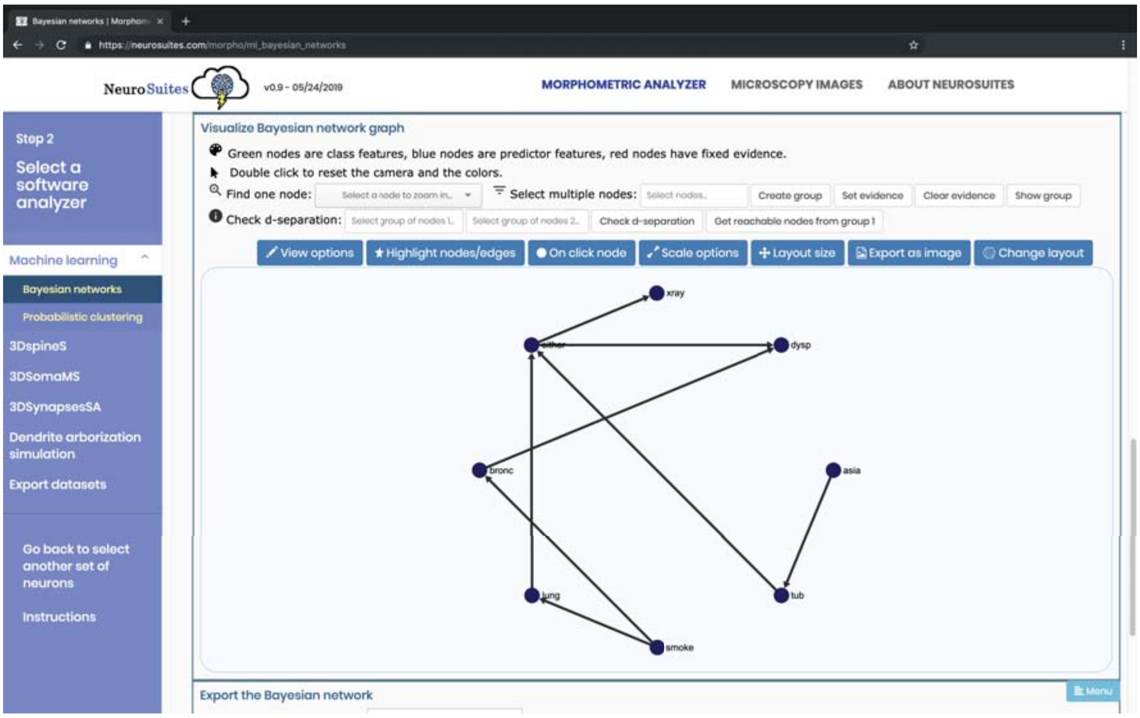
Select thə dataset and features:
(Load example dataset file)

Dataset:

Upload only structure (graph):
Formats supported: .csv or .gzip Apache Parquet file
(first column is the id, first row are nodes names).
Note: For large networks (thousands of nodes) is recommended to upload the Network as a .gzip Apache Parquet file.
Very large networks can take up to a couple of minutes to load.

Upload only continuous parameters:
This must be uploaded after uploading a graph file or a Bayesian network file
Formats supported: .json
(Template file)

Figure A.2: General *Neurosuites* view



Visualize Bayesian network graph

Green nodes are class features, blue nodes are predictor features, red nodes have fixed evidence.

Double click to reset the camera and the colors.

Figure A.3: Bayesian network visualization area in *Neurosuites*