Pontificia Universidad JAVERIANA Cali

ÓMICA

DOCTORATE IN ENGINEERING AND APPLIED SCIENCES

Faculty of Engineering and Sciences

Hierarchical multi-label classification methods for gene function prediction

Miguel Angel Romero Gonzalez

Supervisors:
Prof. Camilo Rocha
Prof. Jorge Finke

Dissertation presented in partial fulfillment of the requirements for the degree of Doctorate in Engineering and Applied Sciences

July 2022

Hierarchical multi-label classification methods for gene function prediction

Miguel Angel ROMERO GONZALEZ

Examination committee:
Prof. Andrés Jaramillo Botero, chair
Prof. Camilo Rocha, supervisor
Prof. Jorge Finke, supervisor
Prof. Mauricio Quimbaya
Prof. Jesús Gómez-Gardeñes
(Universidad de Zaragoza, España)
Prof. Tanner Slagel
(NASA Langley Research Center, USA)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctorate in Engineering and Applied Sciences

July 2022

Acknowledgements

This achievement is possible thanks to the support, advice, and friendship of many people.

It has been an honor to work with Professor Camilo Rocha for all these years, starting from the competitive programming group (ECIGMA), through my bachelor's thesis, and the research project that lead to this dissertation. It has not been easy to keep up with his pace. I am grateful for his guidance, knowledge, patience, and support during this process. I deeply admire his dedication for teaching and research. His ways to encourage people to try harder is amazing.

I am especially indebted to Professor Jorge Finke for his support, ideas, and guidance in every stage of the Doctorate. His passion for learning and researching are inspiring. Our conversations about work and life through these years have been very rewarding. Certainly Camilo Rocha and Jorge Finke have been key in my work and role models for me.

I would like to express my gratitude to Pontificia Universidad Javeriana, Cali and the ÓMICAS project for funding my training and research. I would also like to thank the institutions that funded the ÓMICAS project within the Colombian Scientific Ecosystem: the World Bank, the Colombian Ministry of Science, Technology and Innovation, the Colombian Ministry of Education, the Colombian Ministry of Industry and Tourism, and ICETEX.

I am grateful to Celine Vens and her research group at KU Leuven Campus KULAK for having me as an intern. It has been an honor to work and learn from her and her research group, Robbe D'hondt, Klest Dedja, Michela Venturini, and Fateme Nateghi. I especially thank Felipe Kenji Nakano for his support from my first day in Belgium; we shared many days at the office, many stories, and meany beers. I am grateful for his contributions to this work.

I thank my friends, collaborators, and office mates for their contributions

in this work and their companion in this journey. I especially thank Sergio Ramírez, Fabián Suárez, Oscar García, Jennifer Rodríguez, Alejandro Sierra, Camila Riccio, Jenny Gallo, David Jiménez, Nicolas López, Juan Campos, Alicia Rosales, Carlos Pinzón, Adriana Manrique, Erika Gutiérrez, Hernán Carvajal, Oscar Ramírez, and Chrystian Sosa. There are too many stories that we shared over these years and I hope there will be more to come.

Finally, I thank my family, my parents Ligia and Benjamin, my siblings Fernando and Paola, my brother-in-law Diego, my nephew Mathias, my niece Ana Maria, and my parents-in-law Pilar and Elmer for their support, love, and trust. I deeply thank my wife Manuela Triviño for her love, patience, and encouragement in this journey. She has been my support in many different moments and I dedicate this work to her.

Abstract

This dissertation studies the problem of predicting gene functions from a computational approach. The goal of this problem is to predict associations between genes and functions, where genes can be associated to multiple biological functions and functions have a hierarchical organization. Four machine learning methods are developed focusing on different aspects of the problem, which has been modeled as a classification task: (a) considering hierarchical relations between functions to produce consistent predictions; (b) creating new data representations to built predictive models; (c) exploiting paths of functions in the hierarchy to detect missing annotations of genes; and (d) integrating information available for multiple organisms into the classification task. The main contributions of this work include novel methods that (i) overcome the limitations of the combinatorial gene function prediction problem; (ii) can be used to effectively identify associations between genes and functions of different organisms, including those that do not have enough data available to train predictive models; and (iii) help to narrow down the search space for *in vivo* experiments. These methods have been tested in efforts to predict gene functions in rice and maize, but have been formulated more generally and are applicable to any multi-label classification problem where the classes are organized into a hierarchy.

List of Abbreviations

CV Cross validation.

GCN Gene co-expression network.

HBN Hierarchical binomial-neighborhood.

HMC Hierarchical multi-label classification.

lcl Local classifier per level.

lcn Local classifier per node.

lcpn Local classifier per parent node.

PPI Protein-protein interaction network.

ROC Receiver operating characteristic.

SMOTE Synthetic minority over-sampling technique.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Identifying association between genes and functions is key to gain insight into how genomes work. A better understanding help us to design strategies to tackle problems associated to, e.g., environmental stresses or diseases. Biologically, this task generally depends on strategies that combine alignment-based information with *in vivo* experimentation [71]. These experiments require a significant amount of time and effort mainly due to the combinatorial nature of the problem; Genes can be associated to multiple functions and functions can be associated to multiple genes in a many-to-many relationship. Computational (*in silico*) approaches have emerged to address this challenges thanks to the amount of data increasingly available from high-throughput sequencing technologies [61, 47]. Nevertheless, biological processes in which many genes are involved remain largely unknown, which limits the use of computational approaches. For these reasons, the problem of efficiently associating genes with functions remains an open challenge.

Functions of genes or gene products, as defined by Gene Ontology (GO) [23] or the Kyoto Encyclopedia of Genes and Genomes (KEGG) [32], have different levels of specificity (i.e., some functions are more general and other more specific). Levels of specificity define hierarchical relations between functions that can be represented as a hierarchy, where higher levels identify general functions (i.e., the top of the hierarchy are the most general functions) and lower levels specific ones (i.e., the leaves of the hierarchy are the most specific functions). As an example, consider the GO hierarchy of biological processes illustrated in Figure 1.1. The hierarchy contains nine biological processes to which a gene may be associated, including *response to stimulus*, *detection of external stimulus*, and *detection of light stimulus*. According to this hierarchy: (i) the function

response to stimulus is the most general, (ii) the function *detection of light stimulus* is the most specific, (iii) the function *detection of external stimulus* is parent of *detection of light stimulus*, and (iv) all functions are ancestors of *detection of light stimulus* (except the function itself).

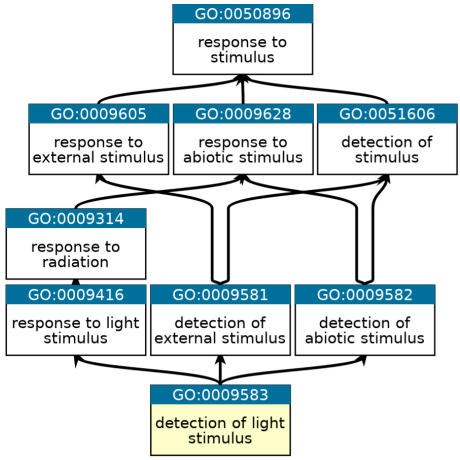


Figure 1.1: Example of the GO hierarchy for the *detection of light stimulus* biological process[1]. Functions in the GO hierarchy have an unique identifier that starts with "GO" followed by a seven-digits number.

Hierarchical relations between functions are an important characteristic of the gene function prediction problem. If a gene is associated to a function, then it must also be associated to all of its ancestors (i.e., more general functions). For example, in Figure 1.1, if a gene is associate to *detection of light stimulus*, then it must be associated to the other eight functions in the hierarchy. Likewise, if a gene is not associated to a function, then it cannot be associated to any of its descendants (which are more specific functions). This constraint is referred to, in machine learning, as the *hierarchy constraint* [85] or, in biology, as the *true-path rule* [82]. Overlooking such constraint can lead to wrong conclusions.

There can also be different types of relations between genes, such as gene co-expression. Genes are reported to co-express whenever they are simultaneously active under specific conditions. Co-expression relations between genes suggest that they may be associated to one or more common biological processes, i.e., share the same functions or are related to the same regulatory pathways [97, 77, 22]. Therefore, characterizing these co-expression relations through gene co-expression network (GCNs) may assist in identifying unknown functional annotations (gene-function associations) in a genome [56, 84, 83]. GCNs are

[1]Taken from QuickGO, https://www.ebi.ac.uk/QuickGO.

generally represented as undirected and weighted graphs, where vertices denote genes and weighted edges indicate the strength of the co-expression relation between two genes.

1.1 The Approach

This dissertation studies the problem of predicting gene functions from a computational perspective, i.e., using *in silico* approaches. Based on machine learning techniques and gene co-expression data, this dissertation aims at overcoming the limitations of the gene function prediction problem (i.e., its combinatorial complexity and shortage of information available for some organisms) by: (i) developing efficient methods that take into account the hierarchical relations between functions, (ii) helping to increase (or complete) functional information available, and (iii) reducing the search space for the *in vivo* experimentation required to verify functional annotations.

The gene function prediction problem is modeled as a classification task where genes can be associated to a set of biological functions and functions have a hierarchical organization. To this aim, hierarchical multi-label classification (HMC) is used to address the task of structured output prediction, i.e., the classes are organized into a hierarchy and instances may belong to multiple classes [85, 78]. Formally, this problem can be explained as follows. Given a set of genes V, a set of biological functions A, and an annotation function $\phi : V \rightarrow 2^A$, where each gene is associated with the collection of biological functions to which it is known to be related (e.g., verified through *in vivo* experiments). The goal is to use the information represented by ϕ, together with additional information of the genes (e.g., gene co-expression data), to predict a function $\psi : V \rightarrow 2^A$ that satisfies the hierarchy constraint (i.e., complies with the hierarchical relations between functions and extends ϕ). Additions identified by ψ are suggestions that need to be verified through *in vivo* experiments. The technical goal of this dissertation is to develop machine learning methods to build predictors for the function ψ.

1.2 Main Contributions

The overall goal of this dissertation is to develop novel machine learning methods to address the problem of predicting gene functions (also called gene functional annotation) based on gene co-expression data. The main contributions include the adaptation and development of new machine learning methods that can

be used to efficiently identify associations between genes and functions of different organisms, including those organisms that do not have enough data available to train predictive models. This dissertation aims to impact the areas of *bioinformatics* (since the problem of predicting gene functions is an open challenge that relies on computational and machine learning techniques) and *machine learning* (since new methods are developed and made publicly available for multiple a applications).

The methods developed in this dissertation tackle the following aspects of the gene function prediction problem:

- **Hierarchical multi-label classification:** The first step is to explore classification methods that take into account the structure of gene functions. Classes are, in general, considered independently in classification tasks, whereas functions of gene (or gene products) are known to be hierarchically structured. A hierarchical multi-label classification method that considers the hierarchical relationships between functions is required to avoid inconsistent predictions and improve the overall performance of predicting gene functions.

- **Feature extraction:** Extracting new features from gene co-expression and functional data that can be used to train classifiers and improve the performance of predicting gene functions. Clustering techniques can be used to identify groups of genes in the gene co-expression network that present similar expression patterns and are mostly associated to the same functions. New features are built based on these groups of genes and can be used to train predictive models that consider the hierarchical structure of functions.

- **Detecting missing annotations:** Redefining the problem of predicting gene functions as a problem of detecting missing annotations that allows us to identify associations at deeper levels of the hierarchy. Datasets of functional annotations are mostly highly imbalanced (i.e., deeper functions in the hierarchy have less annotations and get lower probabilities than the ones at the top). Hence, the task of detecting missing annotations focuses on identifying a set of associations between genes and functions instead of predicting all associations at once. The functional information available for certain organisms can be gradually increased.

- **Transfer learning:** Integrating the functional information of multiple organisms into a predictive model that allows to analyze organisms with a possible shortage of data. Based on the relations that genes of different organisms might have, it is possible to take advantage of the

knowledge available for multiple organism by combining hierarchical multi-label classification and transfer learning. The integration of additional information should enable the use of predictive models on organisms whose functional information is absent or poorly available for training a predictor.

Although these aspects can be addressed incrementally, here they are explored separately. The implementations associated to the proposed methods are delivered via public repositories and are licensed as open access code. The links to the repositories containing both implementation and data are available at the end of each chapter.

Additionally, it is important to note that the proposed methods have been formulated to be general and applicable to any multi-label classification problem with hierarchically structured classes, though they have been applied to the gene function prediction problem throughout this dissertation. A thorough presentation of related state of the art is included as part of each chapter.

1.3 Chapter Outline

The reminder of this dissertation is organized as follows:

- Chapter 2 provides notions, definitions, and intuitions required for the rest of the chapters, including hierarchical multi-label classification, gene co-expression networks, and transfer learning.
- Chapter 3 presents a classification method for hierarchical relations between functions. The method is applied to predict gene functions on rice.
- Chapter 4 introduces a method to extract new features combining GCNs and functional information of genes. These features are used to train multiple hierarchical multi-label classification models. The method is evaluated on maize.
- Chapter 5 introduces a method to detect missing annotations in hierarchical multi-label classification problems. The method is used to identify missing functional annotations of rice genes.
- Chapter 6 introduces a transfer learning method to predict gene functions by combining the information available for multiple organisms. The predictive performance of the method is showcased for rice using arabidopsis, maize, and soybean as sources of information.

- Chapter 7 presents concluding remarks and open research directions.

Chapter 2

Preliminaries

This dissertation builds on different concepts from biology, graph theory, and machine learning to address the problem of predicting gene functions. This chapter presents some definitions, notions and intuitions to read the document, including, gene-co-expression networks, spectral clustering, hierarchical multi-label classification, SHapley Additive exPlanations (SHAP), transfer learning, and domain adaptation. Based on these definitions, the problem of predicting gene functions is later redefined in Chapters 5 and 6.

2.1 Networks

A *network* or *graph* is a formal framework to specify relationships between interconnected entities (e.g., a social network where users are interconnected by friendship or follow relationships) [19].

Definition 1. *A* graph *is a pair* $G = (V, E)$ *where* V *is the set on entities, called* nodes *or* vertices*, and* E *is the set of connections between entities, called* edges*.*

A *directed graph* is a graph in which the edges have orientation, i.e., the relationship between nodes has a direction (e.g., Twitter users follow other users). On the contrary, if the edges of a graph do not have orientation, it is called an *undirected graph* (e.g., Facebook users have friendships with other users). In addition, edges can also have a value or *weight* that might represent the strength, cost, or length of the connections between nodes. A *weighted*

graph is a graph in which edges have weight (e.g., a graph of cities connected by roads of different lengths).

2.2 Spectral Clustering

Spectral clustering is a method with foundations in algebraic graph theory [29]. The aim of applying spectral clustering on a network is to identify groups of vertices sharing a (parametric) notion of similarity [92, 63]. This notion of similarity is often measured using distance or centrality metrics.

Given a graph G, the spectral clustering decomposition of G can be represented by the equation $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{L}$ is the Laplacian, $\mathbf{D}$ is the degree (i.e., a diagonal matrix with the number of edges incident to each node), and $\mathbf{A}$ the adjacency matrices of G. Spectral clustering selects the n eigenvectors associated to the n smallest nonzero eigenvalues of $\mathbf{L}$ (each vertex of the graph gets a coordinate in $\mathbb{R}^n$). The resulting collection of eigenvectors serves as input to a clustering algorithm (e.g., k-means) that groups the nodes of G in n clusters.

2.3 Gene Co-expression Network

A gene co-expression network (GCN) is represented as an undirected and weighted graph where each vertex represents a gene and each edge the level of co-expression between two genes.

Definition 2. *Let V be a set of genes, E a set of edges that connect pairs of genes, and $w : E \to \mathbb{R}_{\geq 0}$ a weight function. A* (weighted) gene co-expression network *is a weighted graph $G = (V, E, w)$.*

The set of genes V in a co-expression network is particular to the genome under study. The correlation of expression profiles between each pair of genes is often measured using the Pearson correlation coefficient. Every pair of genes is assigned and ranked according to a relationship measure, and a threshold is used as a cut-off value to determine E. The weight function w denotes the strength of the co-expression between each pair of genes in V. For example, in the ATTED-II database (version r21c), the level of co-expression between any pair of genes is measured as a z-score expressed as a function of the co-expression index LS (Logit Score) [54, 55].

In an *annotated* gene co-expression network, each gene is associated with the collection of biological functions to which it is related (e.g., through *in vivo* experiments).

Definition 3. *Let A be a set of biological functions. An* annotated gene co-expression network *is a gene co-expression network $G = (V, E, w)$ complemented with an annotation function $\phi : V \to 2^A$.*

2.4 Hierarchical Multi-label Classification

Classification problems are generally categorized into three different types *binary* classification refers to the task of predicting a single class (e.g., if a song is written in English or not) [33]. *Multi-class* classification refers to the case where the prediction problem consists of a single class, which is an option from a set of mutually exclusive classes (e.g., the language in which a song is written) [49]. *Multi-label* classification refers to the case where the prediction problem consists of a subset of classes (e.g., if a song has overlapping genres) [89].

Although the aforementioned classification types are frequently used, they do not consider hierarchical relations between classes. For such scenarios, hierarchical multi-label classification (HMC), an extension of multi-label classification, has emerged to address the task of structured output prediction: the input classes are organized into a hierarchy and an instance may belong to multiple classes [85]. In many problems, such as gene function prediction, classes inherently satisfy these conditions [42].

A classification problem is considered hierarchical if and only if its class hierarchy is a strict partial order (i.e., a *strict poset*) [78]. A strict poset over a finite set of classes C defines a binary relation $\prec$ on C that is asymmetric, anti-reflexive, and transitive. For instance, the hierarchy of biological processes can be defined over a strict poset [23].

Definition 4 ([85])**.** *Let I be an instance space and $(C, \leq_h)$ a class hierarchy, where C is a set of classes and $\leq_h$ is a strict partial order. The objective is to find a function $\psi : I \to 2^C$ such that for every $x \in I$ and $c \in C$:*

$$c \in \psi(x) \implies \forall c' \leq_h c : c' \in \psi(x),$$

i.e., ψ complies with the hierarchy constraint.

Note that $b \leq_h a$, for a and b in C means that the class a is the parent of class b. A class hierarchy can be represented as a directed acyclic graph (DAG) consisting of a set of classes C and a set of relations R between the classes, in

such a way that if $b \leq_h a$, then the pair $(a, b) \in R$. That is, the class hierarchy can be equivalently denoted in terms of the strict poset as $(C, \leq_h)$ or in terms of the DAG as (C, R).

The authors in [78] identify two types of approaches to explore the hierarchical structure. *Local* (or top-down) classifiers refer to partially predicting the classes in the hierarchy from top to bottom by taking into account the predictions of parent classes. *Global* classifiers refer to a single classifier that considers the entire hierarchy at once.

Figure 2.1 illustrates four HMC methods used in this dissertation: *Local classifier per node* (*lcn*) that consists of training one binary classifier for each class in the hierarchy except to the root node. The *local classifier per parent node* (*lcpn*) that consists of training a multi-label classifier for each parent node in the hierarchy to distinguish between its child classes. The *local classifier per level* (*lcl*) that consists of training one multi-label classifier for each level of the class hierarchy except for the root. The *global classifier* that consists of building a single multi-label classifier taking into account the hierarchy as a whole during a single run. The global classifier can assign classes at potentially every level of the hierarchy to an instance.

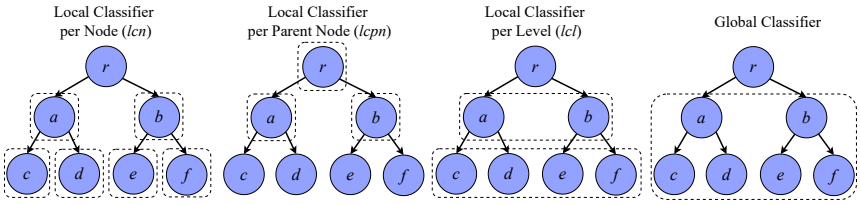


Figure 2.1: Example of global and local methods for hierarchical multi-label classification. Given a hierarchy of classes (r, a, b, c, d, e, and f), the dashed boxes show the number of classifiers required for each method. Note that the *lcn*, *lcpn*, *lcl*, and global classifiers require 6, 4, 3, and 1 predictors, respectively.

Classifiers that overlook the class relationships, by predicting only the leaf classes in the hierarchy (or predicting each class independently), may lead to *inconsistent predictions*, which refers to an scenario where a node is inferred to have a particular class, but the outcome of the classifier fails to infer the node's association to all of its ancestor classes in the hierarchy. In other words, an inconsistent prediction is reached when the prediction does not satisfy the this constraint for some class. Satisfying ancestral constraints is referred to as the *hierarchy constraint* in HMC [85] or the *true-path rule* in GO [82, 3].

2.5 SHAP Feature Contribution

The performance of classification algorithms depend on the features used to train a particular predictor. The SHapley Additive exPlanation (SHAP) is a framework that computes importance values for each feature in a dataset by using concepts from game theory [45, 46]. SHAP assigns Shapely values to explain which features in the model are the most important for prediction by calculating the changes in the prediction when features are conditioned. Given a predictor and a training set, SHAP computes a matrix with the same dimensions of the predictor's output containing the Shapely values for each instance and class. For example, in a binary classification problem where there are two classes, positive and negative, and a training set of n instances, the output of SHAP is a matrix of dimension $n \times 2$. In multi-label classification problems, the output is a matrix of dimension $n \times 2$ for each class, since classes are not mutually exclusive and the outcome is either positive or negative for each class.

2.6 Transfer Learning and Domain Adaptation

The concept of domains is needed before presenting the definitions of transfer learning and domain adaptation. A domain is an abstraction of a learning task described by its input and output (with their corresponding density functions) [39, 98].

Definition 5. *A* domain $\mathcal{D}$ *is a tuple* $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}, p\}$ *where* $\mathcal{X}$ *is the input or feature space,* $\mathcal{Y}$ *is the output or label space, and* p *is the associated probability density function.*

The feature space is a subset of the D-dimensional space $\mathbb{R}^D$, sometimes referred to as *feature vectors* or *instance set*, and the label space corresponds to the classes. The probability density function comprises the marginal distribution $p(x)$, the joint distribution $p(x, y)$, and the conditional distribution $p(x|y)$ for every $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Machine learning problems generally deal with a single domain of interest assuming that there is enough labeled data to train a classifier. Transfer learning extends this idea by using one (or more) additional domains as a source of additional information. This is particularly useful in cases in which labeled data is not enough or available for training. The domain of interest is called *target domain* and is denoted as $\mathcal{D}_\mathcal{T} = \{\mathcal{X}_\mathcal{T}, \mathcal{Y}_\mathcal{T}, p_\mathcal{T}\}$, and the domain used as source of additional information is called *source domain* and is denoted as

$\mathcal{D}_\mathcal{S} = \{\mathcal{X}_\mathcal{S}, \mathcal{Y}_\mathcal{S}, p_\mathcal{S}\}$. Domain-specific functions are marked with the subscript $\mathcal{S}$ or $\mathcal{T}$.

Definition 6. *Given a target domain $\mathcal{D}_\mathcal{T}$ and m source domains $\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_m}$, transfer learning aims to improve the performance on $\mathcal{D}_\mathcal{T}$ using the knowledge in $\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_m}$, where the target and source domains are pairwise distinct in any of their parts.*

The target and source domains are freely allowed to differ in sample space, label space, probability density function or all [57, 87, 39, 98]. On the contrary, domain adaptation is a particular case of transfer learning where the sample and label spaces remain unchanged between the target and source domains, and only the probability density function change.

Definition 7. *Given a target domain $\mathcal{D}_\mathcal{T}$ and m source domains $\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_m}$,* domain adaptation *aims to improve the performance on $\mathcal{D}_\mathcal{T}$ using the knowledge in $\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_m}$, where the sample and labels spaces of the target and source domains coincide, but their probability density functions are pairwise distinct.*

Chapter 3

A Local Supervised Learning Approach to Hierarchical Multi-label Classification in Networks

This chapter was previously published as:
Romero, M., Finke, J., and Rocha, C. A top-down supervised learning approach to hierarchical multi-label classification in networks. Applied Network Science 7, 8 (2022).

Parts of it were initially presented in:
Romero, M., Finke, J., Quimbaya, M., and Rocha, C. (2020). In-silico gene annotation prediction using the co-expression network structure. Complex Networks and Their Applications VIII. Studies in Computational Intelligence, vol 882. Springer.

Chapter Summary

Node classification is the task of inferring or predicting missing node attributes from information available for other nodes in a network. This chapter presents a general prediction method to hierarchical multi-label classification (HMC), where the attributes to be inferred are organized in a hierarchy (i.e., can be specified as a strict poset or a DAG). It is based on a local classification approach that addresses hierarchical multi-label classification with supervised learning by building one local classifier per class. The proposed method is applied to the prediction of gene functions for *Oryza sativa Japonica*, a variety of rice. It is compared to hierarchical binomial-neighborhood, a probabilistic method, by evaluating both methods in terms of prediction performance and computational cost.

Node classification refers to the task of predicting an attribute for a set of nodes based on the information of other nodes in the network [6]. A node classification problem takes as input a network consisting of nodes and connections (relations) between them, and an attribute that describes a (categorical) property of the nodes (i.e., values of the node attribute can be seen as classes). From a machine learning perspective, the nodes refer to instances and the node attribute determines the type of classification task to be addressed (e.g., binary, multi-class, or multi-label). The problem of predicting gene functions can be addressed as a node classification problem where the genes are connected through co-expression relations in a gene co-expression network (GCN) and the node attribute refers to gene functions. However, there are hierarchical dependencies between functions that must be considered and cannot be handled by binary, multi-class, or multi-label approaches.

This chapter introduces a local classification method that addresses hierarchical multi-label classification (HMC) using supervised learning. Given a network $G = (V, E)$, a node attribute consisting of a set of classes, an assignment of classes to nodes, and a class hierarchy specified as a directed acyclic graph $H = (A, R)$, the hierarchical multi-label classification problem is addressed by building a binary classifier for each class (i.e., local classifier per node). Classifiers are built iteratively from the roots of the hierarchy to the leaves. The method uses a correction mechanism to guarantee that the hierarchy constraint (or true-path) rule is satisfied by the classifier's outcome; it is enforced by computing cumulative probabilities along the paths of classes in the input hierarchy.

The method is showcased with a study on the prediction of gene functions for *Oryza sativa Japonica*, a variety of rice, and compared to the probabilistic HBN method [30]. We evaluate both methods in terms of prediction performance (using the true positive and true negative rates) and their computational cost (using the execution time). In the case study, the method takes as inputs: a gene co-expression network of rice and the hierarchical structure of biological processes defined in [23]. The goal is to infer gene functions from 15 sub-hierarchies grouping 1,938 biological processes associated to 19,663 genes.

The remainder of the chapter is organized as follows. Section 3.1 reviews related work. Section 3.2 introduces the method for node classification where classes have a hierarchical organization. Section 3.3 presents the results of applying the proposed method to *Oryza sativa Japonica*. Finally, Section 3.4 draws some concluding remarks and future research directions.

3.1 Related Work

Several studies have applied both local and global approaches across different domains [30, 20, 7, 60]. The authors in [30] propose a local method, called Hierarchical Binomial-Neighborhood (HBN), to predict protein functions in yeast *Saccharomyces cerevisiae*. It is shown by the authors that the hierarchical structure of functions can be exploited to avoid inconsistent predictions and, at the same time, outperform methods based on independent class prediction. However, the authors point out that the main limitation of their method is the high computational effort required for assigning probability weights to every protein-function pair.

The authors in [60] introduce a local method based on chained path evaluation, which uses a classifier to train each non-leaf class (i.e., each class with at least one descendant) in the hierarchy. Information on ancestral relations is included in the classifier by adding an extra feature with the prediction of parents of each class. As in [30], the computational cost of the method grows exponentially as a function of the number of paths in the hierarchy. The use of global approaches is, in general, also limited by their high computational demands. In [20], for example, the authors present a global method that addresses hierarchical multi-label classification based on predictive clustering trees. The computational cost of the method is directly proportional to the size of the hierarchy.

Other studies address the node classification problem and obtain state-of-the-art performance for different case studies (see, e.g., [1, 12, 27, 34, 47, 88]). However, they do not take into account dependencies between classes (hierarchical or not), for they focus on multi-class instead of multi-label problems. For this reason, such developments can not be compared directly to assess hierarchical multi-label classification.

3.2 Hierarchical Multi-Label Classification

This section presents a local classification approach in the form of a supervised learning method for hierarchical multi-label classification.

The input of the method consist of several elements: a graph $G = (V, E)$ specifying an undirected network with nodes V and edges E, a directed acyclic graph $H = (C, R)$, with vertices C and edges R disjoint from V and E, respectively, representing the hierarchy of classes, and a function $\phi : V \mapsto 2^C$ with a partial assignment of classes to nodes in the network. For $v \in V$, the set $\phi(v) \subseteq C$ is the collection of classes initially associated to v. It is assumed

that ϕ satisfies the true-path rule for the hierarchy H, meaning that if a node v satisfies $c \in \phi(v)$ for a class $c \in C$, then $\phi(v)$ must contain all the ancestors of c in H. The goal of the method is to build a function $\phi' : V \mapsto 2^C$ extending ϕ with new assignments of nodes in V to classes in C. Figure 3.1A depicts an example of the input of the method where the nodes of the network G are labeled with classes a-e and the hierarchy of classes H is a DAG. According to the true-path rule, nodes labeled with class e are also related to classes a, b, and c. The objective is to predict new associations between nodes and classes for either nodes with or without labels.

The rest of this section describes the main steps behind the construction of the supervised learning method.

3.2.1 Hierarchy Normalization

Hierarchies are represented as directed acyclic graphs where, in general, nodes can have any (finite) number of parents. Since the method presented in this work assumes that every node has at most one parent, a topological traversal algorithm for directed graphs (see, e.g., [37]) is used to transform H into a tree, when required. In this way, the resulting method can take as input any hierarchy.

The algorithm uses the structure of H (not its tree version) and its distribution of classes. Given an ancestral relation $b \leq_h a$ (i.e., class a is parent of class b), a weight $w(a, b)$ for such a relation is defined as the ratio between the number of nodes associated to b (i.e., the size of the set $\phi^{-1}(b)$) and the number of nodes associated to a (i.e., the size of the set $\phi^{-1}(a)$). Since all nodes associated to b must be associated to a, then by definition each weight $w(a, b)$ is in the range $[0, 1]$. For any node b with $n \geq 1$ parents $a_1, \ldots, a_n$ in H (i.e., $b \leq_h a_i$, for $1 \leq i \leq n$), the parent of b in the resulting tree is the node a_j maximizing $w(a_j, b)$ among all the a_i's. Ties are broken arbitrarily. This process can be effectively computed in time and space $O(|C| + |R|)$, namely, in resources linear in the size of H. Such a process, based on a topological-sorting traversal, is described in Algorithm 3.1. Finally, note that the topological sorting of the vertices of H in Algorithm 3.1, can be exploited to compute the value of function $w(\_, \_)$ by dynamic programming in space $\Theta(|C|)$. More precisely, a function $\rho : C \to \mathbb{N}$ assigning to each class b_i its number of descendants $\rho(B_i)$ in H can be computed from the direct descendants of b_i, which are processed before b_i in the topological sorting of C.

Algorithm 3.2.1: Topological-sorting based traversal for hierarchy normalization

```
1 input: H
```

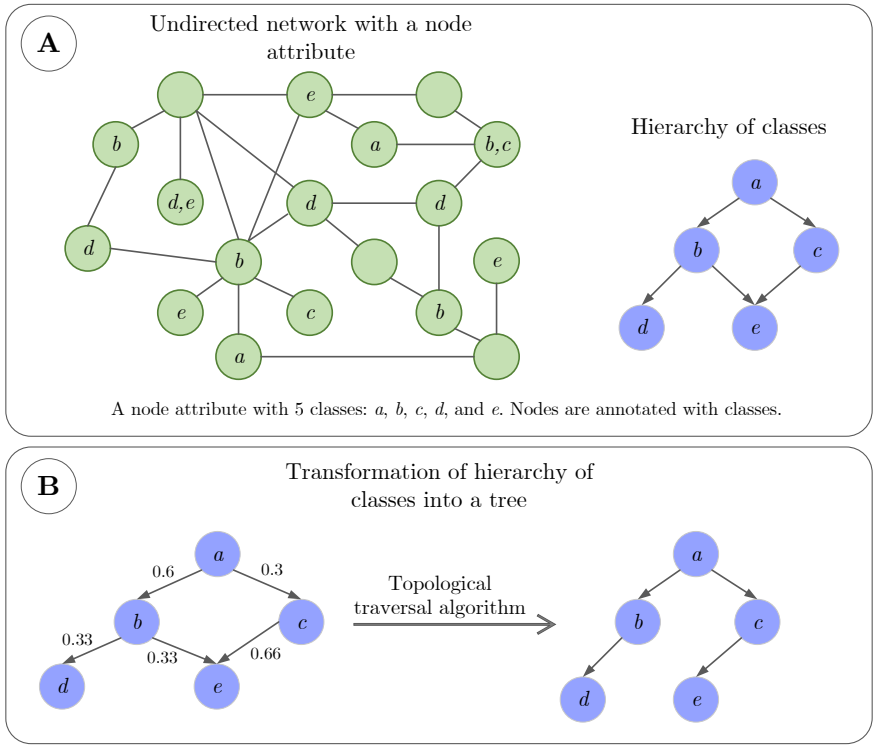


Figure 3.1: **A.** The classification method gets as input a network with a node attribute and a set of known association between nodes and classes, as well as the hierarchy of ancestral relations represented as a DAG. Note that there are more nodes associated to class b than c. **B.** The DAG representation of the hierarchy is transformed into a tree using a topological traversal algorithm, based on the distribution of the classes in the network. Since classes b and c are ancestors of e, the ratio of nodes associated to e and c is higher that the ratio for e, and b ($w(c, e) > w(b, e)$), the algorithm removes edge (b, e) and returns a tree.

```
output: T = (C, R_T)

set R_T = {}
compute a topological sorting n_1,...,b_n of C (leaves first)
foreach class b_i with 1 ≤ i ≤ n
  identify all parents a_1,...,a_m of b_i in H
  foreach parent a_j of b_i with 1 ≤ j ≤ m
    compute the weight w(a_j, b_i)
  end
  identify a_k with w(a_k, b_i) ≤ w(a_j, b_i) for 1 ≤ j ≤ m
  extend R_T with (a_k, b_i)
end
return the tree T
```

As an example, consider the hierarchy depicted in Figure 3.1B. Note that class e has more than one parent: there exists an ancestral relation from b to e and from c to e (i.e, $b \leq_h e$ and $c \leq_h e$, respectively). By the true-path rule, nodes associated to class d are also associated to class b, and the ones associated to class e are associated to both b and c. Since there are 4 nodes associated to e, 4 to d, 4 to b, and 2 to c, the weight $w(b, e)$ is 0.33 and the weight $w(c, e)$ is 0.66. Therefore, the topological-sorting traversal will remove edge (b, e).

In the rest of this chapter, it is assumed that hierarchy H is indeed a tree.

3.2.2 The Method

Given a network G and a hierarchy tree T, the method is built in a process comprising three stages. Figure 3.2 depicts the general approach.

Stage 0: data pre-processing. In this stage, topological features of G and T, and hierarchical information in T are prepared and combined for supervised learning.

Classes that are too specific (or too general) are ignored in the prediction to avoid overfitting and learning bias. In the case study presented in Section 3.3, a class is defined as too specific (or too general) if it is associated to less than 5 (or more than 300 genes) [30]. As a result, the input hierarchy T can be split into several sub-trees, each one representing a sub-hierarchy $T' = (C', R'_T)$ with $C' \subseteq C$ and $R'_T \subseteq R_T$, over which the method is applied independently. That is, a sub-hierarchy T' is a subset of the classes and ancestral relations in T. As a matter of fact, this situation arises in the case study presented in Section 3.3. Furthermore, each sub-hierarchy T' is associated to the subgraph

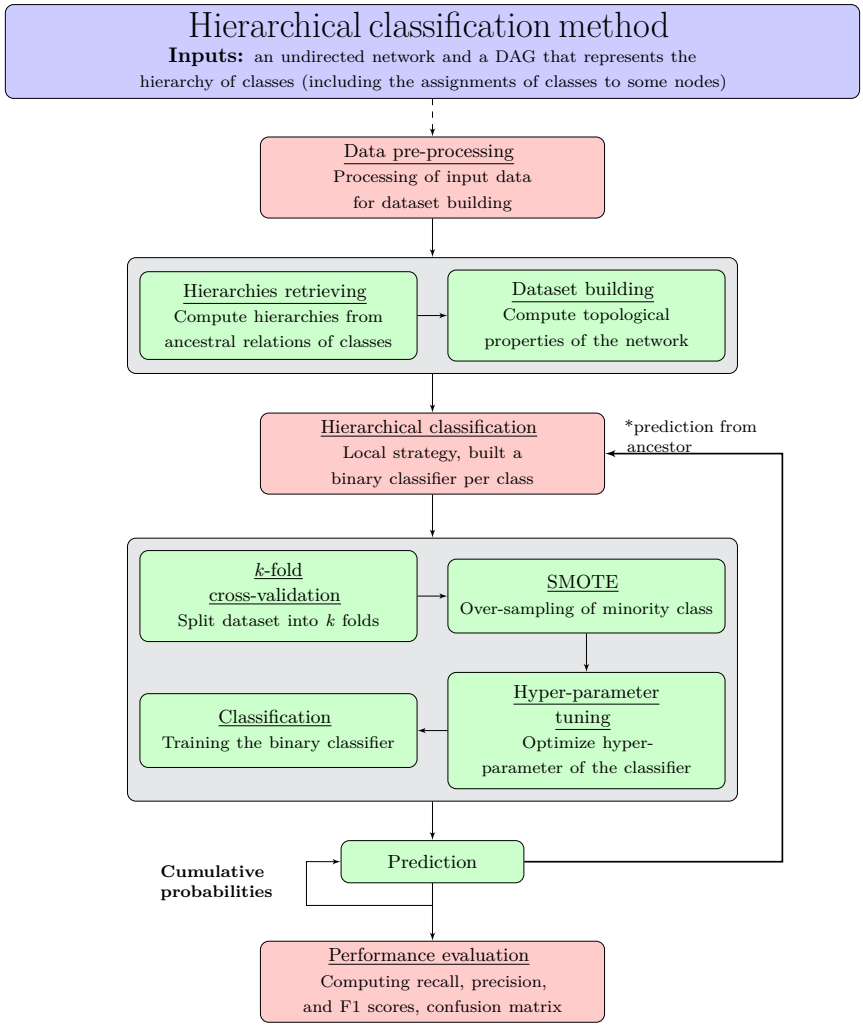


Figure 3.2: Framework of the hierarchical multi-label classification method. The method is split into three stages: data pre-processing, class prediction and performance evaluation. The method is independently applied for every resulting sub-hierarchy H'. Ancestral relations between classes are included in the method as features with the prediction of ancestors and are represented by the upward arrow in the prediction stage. In addition, a correction mechanism for inconsistencies is included by means of cumulative probabilities, which are computed according to the path of classes in the sub-hierarchy. If the probability of association between a node and a class is close to zero, then the cumulative probability of the association between the same node and the descendant classes will be close to zero as well.

of G consisting of all nodes labeled with the root class of T', that is, each sub-hierarchy T' is related to a different subgraph $G' = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E$. In this way, sub-hierarchies are considered independent problems with smaller inputs (*à la* divide and conquer).

For each sub-hierarchy T' in T, datasets are built based on two types of topological properties, namely, hand-crafted features and node embeddings. For the first type, properties of nodes V' such as degree, average neighbor degree, centrality, and eccentricity are computed. Additionally, for each class c in T', two features are computed to represent the probability of a node being associated to c and its parent in H based on the information of the neighborhood. For c and its parent, a node and its neighbors, and the associations between the neighbors and both classes, these new features represent the ratio between the number of neighbors associated to each class and the total number of neighbors. For the second type of properties, node embeddings (i.e., continuous representation that capture the characteristics of the nodes in G') are computed using node2vec [26].

Stage 1: Hierarchical classification. This stage comprises a local method combining different supervised machine learning techniques/tools. It builds prediction classifiers for each sub-hierarchy T' independently. The method uses stratified k-fold cross-validation, the Synthetic Minority Over-sampling Technique (SMOTE) [10], hyper-parameter tuning [5], and a binary classifier, (e.g., XGBoost [13] or graph convolutional networks [34]). These techniques are combined sequentially in a pipeline, which is used iteratively from the root to the leaves of each sub-hierarchy T'. Note that, since the local approaches builds a different classifier for each class in the sub-hierarchy, the method can be used for multi-class and multi-label problems. As a result, nodes can be independently associated to multiple classes.

The combination of the above-mentioned techniques/tools makes up the core of the method. Each technique has a different objective. Stratified k-fold aims to overcome overfitting by randomly selecting independent k subsets of the dataset where the distribution of the labels is similar for all folds. In this method, 5 folds are used for cross validation, that is the train-test ratio is 80/20. Over-sampling aims to overcome learning bias handling imbalanced datasets for underrepresented classes. SMOTE synthesizes new examples of the minority class from the existing ones. Hyper-parameter tuning aims to improve the performance of the prediction by optimizing parameters of the classifier such as, e.g., learning rate, number of estimators, and maximum depth of trees.

Two types of classifiers were used; namely, the XGBoost [13] gradient boosting decision trees and graph convolutional networks [34]. XGBoost was chosen for

interpretability [21, 70] and graph convolutional networks for state-of-the-art performance. In general, other binary classifier can be used in this stage. The typical parameter values used for XGBoost classifiers are: *gbtree* booster, area under Precision-Recall (*aucpr*) evaluation metric, learning rate (*eta*) of 0.05, maximum tree depth (*max_depth*) of 6, subsample ratio (*subsample*) of 0.9, and minimum sum of instance weight in a child (*min_child_weight*) of 3. For the graph convolutional networks, the implementation by [18] was used with the following parameters: 16 layers of 16 units each, RelU activation function, dropout rate of 50%, learning rate of 0.01, and binary cross-entropy loss function.

Classifiers for each class in T' are built independently, so that there is no relation between their predictions. Including information from the ancestors of a class c into its classifier is not enough to avoid inconsistent predictions. For this reason, a correction mechanism is included in this stage. Since ensuring the true-path rule is key in the proposed method, this stage computes cumulative probabilities along the paths of classes in T'. Namely, the probability of association between a node v and c is directly related to the predicted probabilities of the node being associated to all ancestors of c. Intuitively, the principle is as follows: if the probability of association of c to v is close to zero, then the probability of association for all descendant classes of c to v will be close to zero as well. The main consequence of enforcing the principle is that the classification computed from the cumulative probability satisfies the true-path rule and removes the inconsistencies in the prediction.

Stage 2: Performance evaluation. This stage comprises the evaluation of the metrics used for measuring the prediction performance of the classifiers. Performance evaluation focuses on recall (true positive rate) and precision scores. It also evaluates the precision-recall curve instead of the accuracy, loss, or ROC curves. This is mainly because datasets are often imbalanced (w.r.t. the positive class in a binary classification), thus both positive and negative classes of the binary classifier need to be analyzed separately. Recall and precision scores are computed from the predicted cumulative probabilities as a function of the *optimum threshold*, which is defined as the threshold that maximizes the F1 score from the precision-recall curve for the cumulative probabilities.

3.3 Gene Function Prediction

This section presents a case study on the prediction of gene functions (i.e., biological processes in which genes are involved) for the *Oryza sativa Japonica* rice variety. The outcome of applying the method proposed in Section 3.2 are

described. The results are compared to the probabilistic method proposed in [30].

3.3.1 Predicting Gene Functions in *Oryza sativa Japonica*

The goal of this case study is to predict gene functions, that is, the biological processes in which some genes are involved. We address the problem by using the method proposed in Section 3.2 on the GNC of *Oryza sativa Japonica* [54] and a hierarchy of biological processes for this organism [73]. The computational experiments were carried out on a cluster with 5 nodes, each one with 64GB of memory, and a AMD Opteron™ Processor 6376 with 64 CPU cores.

Formally, $H = (A, R)$ is a DAG which represents the hierarchical organization of biological processes, where R represents the ancestral relations between functions. Genes are associated to one or more biological processes through a function $\phi : V \to 2^A$, where A denotes the set of all biological processes. The predictive method combines the existing set of labels in ϕ, topological properties of G, and the hierarchical information of H to obtain a new labeling function ϕ' using the hierarchical multi-label classification approach. As a result, the function ϕ' contains suggestions of previously unidentified associations between genes and functions, which are guaranteed to satisfy the true-path rule.

The set of known associations between genes and functions contains 19,663 rice genes, 550,813 co-expression relations, 3,743 biological processes, 220,598 assignments of functions to genes, and 7,185 ancestral relations of functions (all biological processes belong to the same hierarchy). To avoid overfitting and learning bias, only those functions associated to more than 4 and at most 300 genes are considered [30]. Under this criterion, 1,938 functions (52%) are used for prediction. As a result, the function hierarchy breaks down into 27 sub-hierarchies, from which 12 correspond to isolated functions or small sub-hierarchies (fewer than 7 functions). The 15 remaining sub-hierarchies are described in Table 3.1, sorted in ascending order in terms of the number of functions A' and the number of genes associated with each function.

The prediction performance of the proposed method is compared with the HBN method in [30]. The HBN method uses a local approach that integrates relational data of protein-protein interaction network (PPI) with the hierarchical data of biological processes with the objective of predicting protein functions. For this case study, HBN is adapted to the problem of predicting gene functions based on GCNs. To predict the probability of a gene g being associated to function a, the local neighborhood information of g in the GCN and the ancestors of a in the hierarchy are considered. The HBN method computes the probability of gene g being associated to function a obeying the true-path rule.

	Root	Func	Genes	Description
1	GO:0040007	10	108	growth
2	GO:0002376	11	131	immune system process
3	GO:0051704	22	144	multi-organism process
4	GO:0044419	37	777	interspecies interaction between organisms
5	GO:0044085	50	377	cellular component biogenesis
6	GO:0000003	72	648	reproduction
7	GO:0006796	118	1,270	phosphate-containing compound metabolic process
8	GO:0032501	118	1,043	multicellular organismal process
9	GO:0032502	149	1,063	developmental process
10	GO:0016043	140	661	cellular component organization
11	GO:0051179	164	1,350	localization
12	GO:0050896	261	3,319	response to stimulus
13	GO:0065007	485	2,224	biological regulation
14	GO:0008152	775	5,862	metabolic process
15	GO:0009987	925	5,900	cellular process

Table 3.1: Sub-hierarchies generated for the gene co-expression network of *Oryza sativa Japonica*

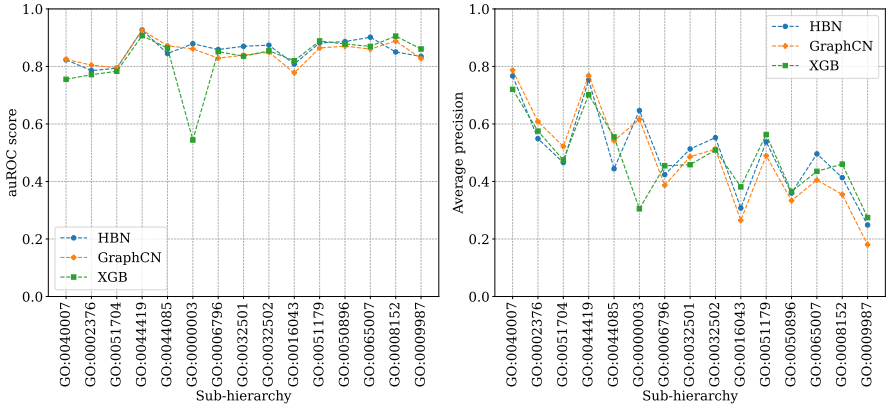


Figure 3.3: Prediction performance of the hierarchical multi-label classification method with XGBoost (XGB) and graph convolutional network (GraphCN) classifiers, and the probabilistic method (HBN) for the 15 sub-hierarchies of *Oriza sativa Japonica*. Performance is measured with area under the ROC curve and the average precision score.

Figures 3.3, 3.4, 3.5, and 3.6 show the mean performance for the proposed method and HBN between multiple experiments, in which each experiment represents the mean performance between the k folds used for cross-validation. In all of them, the variation (error bar or standard deviation) is not included because it is negligible and can add visual noise to the plots. Figure 3.3 illustrates the performance of the proposed method using XGBoost (XGB) and graph convolutional network (GraphCN) classifiers and HBN measured with the area under the ROC curve and the average precision score. Note that their performance seem to be similar in most sub-hierarchies and it is not possible to conclude which one performs best from Figure 3.3. However, since only biological processes associated to more than 4 and less than 300 genes are considered (less than 2% of the genes in the GCN), datasets generated for the filtered biological processes are highly imbalanced. For this reason, the area under the ROC curve is not suitable for the case study (this measured is biased for the over-represented class in the classification task), and the analysis should focus on other metrics such as recall and F1 score instead.

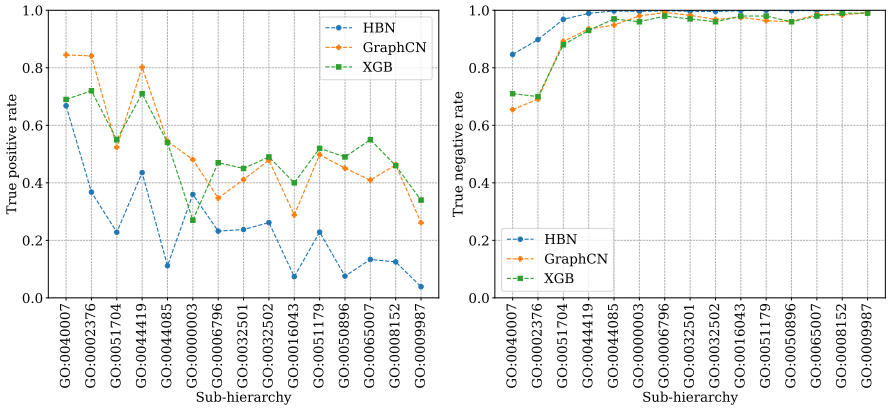


Figure 3.4: True positive rate (or recall) and true negative rate of the hierarchical multi-label classification method with XGBoost (XGB) and graph convolutional network (GraphCN) classifiers, and HBN for the 15 sub-hierarchies generated for *Oryza sativa Japonica*.

An outstanding difference between the performance of the proposed method and HBN is observed when the confusion matrices are analyzed. Figure 3.4 shows the true positive rate (or the measure of recall) and the true negative rate for the 15 sub-hierarchies. Note that the true positive rate of the proposed method is higher than HBN for most of the sub-hierarchies, whereas the true negative rate of HBN is higher for all sub-hierarchies. However, HBN is biased

for the negative class because the probability predicted by HBN for most of the associations between genes and functions is close to zero. As the datasets are highly imbalanced, the performance in terms of the positive class are key to determine which method is adequate. Recall that a dataset is said to be *imbalanced* for binary classification if one of the classes is under-represented in relation to the other one (i.e., the number of instances related to one class is much higher than the number of instances related to the other).

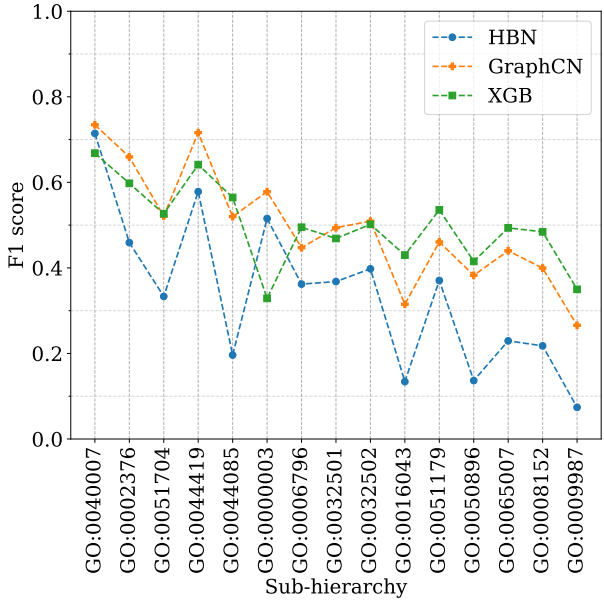


Figure 3.5: F1 score of the hierarchical multi-label classification method with XGBoost (XGB) and graph convolutional network (GraphCN) classifiers, and the HBN method for the 15 sub-hierarchies generated for *Oryza sativa Japonica*.

Based on the true positive rate metrics in Figure 3.4, the proposed method outperforms HBN in the identification of the (positive) associations between genes and functions. The performance varies between XGBoost and graph convolutional networks, but both classifiers have better overall performance than HBN. The results suggest that graph convolutional networks are better for small, while XGBoost for larger sub-hierarchies. Even though the true negative rate of the HBN method is close to 1 for all sub-hierarchies (as illustrated on Figure 3.4) the performance of the proposed method outperforms HBN in terms of the average of both recall and precision (i.e., F1 score). Figure 3.5 presents the F1 score of the proposed method and HBN for all 15 sub-hierarchies. In this

case study, there is no observable correlation between the size/depth/span of a hierarchy and the prediction performance according to the experiments. This is coherent with the overall computational complexity of the algorithms. On the other hand, there is no experimental evidence to suggest that some degree of correlation exists between the number of labeled nodes and the prediction performance.

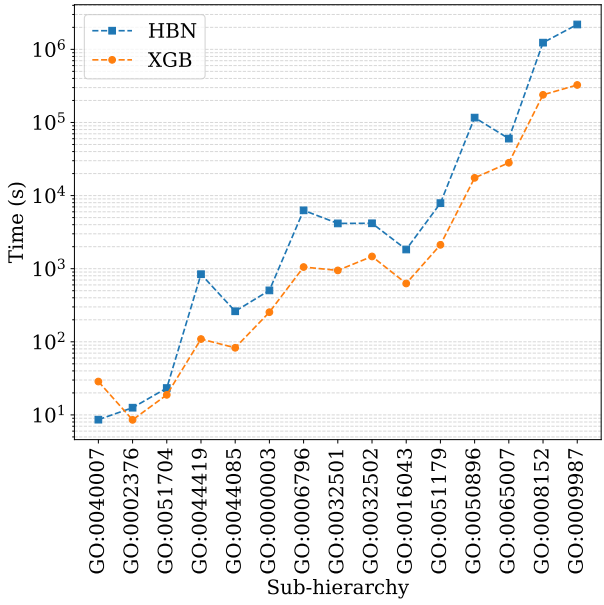


Figure 3.6: Execution time of the hierarchical multi-label classification method with XGBoost (XGB) classifier and HBN for the prediction of all 15 sub-hierarchies. The execution time is measured in seconds and plotted on a logarithmic scale.

Finally, the execution time of the proposed method and HBN is illustrated in Figure 3.6. The execution time for the graph convolutional network classifier is not included because the experiments were executed on CPUs rather than GPU. It is known that neural networks run much faster on GPUs; thus, it would not be fair to make a comparison with the available data. Note that the execution time is measured in seconds and plotted on a logarithmic scale. Except for the smallest sub-hierarchy (GO:0040007), the execution time of the proposed method, using XGBoost classifier, is smaller than HBN. On average, the execution time of the HBN method is approximately 4 times larger than the proposed one.

3.4 Concluding Remarks

By combining different techniques from machine learning and hierarchical multi-label classification, this chapter introduces an method to address the node classification problem for scenarios in which nodes can have attributes obeying a hierarchical organization. Taking into account hierarchical dependencies is shown to be a key aspect for obtaining more consistent predictions that satisfy the true-path rule.

We presented a baseline comparison between the proposed method (using two different classification techniques, namely, gradient boosting decision trees and graph convolutional networks) and the HBN method introduced in [30]. Both methods are applied to the problem of predicting gene function on the variety of rice *Oryza sativa Japonica*. The proposed HMC method outperforms HBN in two aspects. First, using topological information of the network is a key feature to obtain the overall best performance of the prediction. The true positive rates of the proposed method are significantly higher than HBN, whereas the true negative rates yield similar values (close to one). This result suggests that the proposed method can lead to good prediction of associations between genes and functions in *Oryza sativa Japonica*, as well as potentially in other organisms.

For scenarios in which the classes of the hierarchy are under-represented, i.e., datasets are imbalanced, it is important to focus on metrics that are not biased by the imbalanced dataset. Such metrics include the true positive rate (or the measure of recall), the true negative rate, and the F1-score. Other widely-used metrics, like the area under ROC curve and the measure of average precision, are misleading for evaluating the performance of a classifier under such conditions.

Second, the execution time of the proposed method for the XGBoost classifier is, on average, four times faster than that of HBN. The reduction in computational cost of the proposed local method can be attributed to the fact that it predicts the probability of associations between a class and every node of the network at the same time. Also, the efficient computation of the DAG into a tree helps in making the proposed method relevant to analyze larger networks and hierarchies.

Finally, although the performance of the proposed method is promising, it requires to gather sufficient information from node classes, which in some cases is incomplete or unavailable. For example, information about gene functions is limited for many genes and gene products. For some organisms there is no such information available at all. The shortage of information may lead to over-fitting or learning bias in the method, and consequently to misleading conclusions.

Availability of Data and Materials The datasets analyzed for the current study are publicly available from different sources. Gene co-expression data of *Oryza sativa Japonica* is available at ATTED-II (version r17c) [54], and functional data of rice genes is available at [73] and [41].

The data collected, cleaned, and processed from the above sources as used in the case study can be requested to the authors. The proposed method was implemented in Python 3 and is publicly available in [64].

Chapter 4

Feature Extraction with Spectral Clustering for Gene Function Prediction using Hierarchical Multi-label Classification

This chapter was previously published as:
Romero, M., Ramírez, O., Finke, J., and Rocha, C. Feature extraction with spectral clustering for gene function prediction using hierarchical multi-label classification. Applied Network Science 7, 28 (2022).

Parts of it were initially presented in:
Romero, M., Ramírez, Ó., Finke, J., and Rocha, C. (2022). Supervised gene function prediction using spectral clustering on gene co-expression networks. Complex Networks and Their Applications X. Studies in Computational Intelligence, vol 1016. Springer.

Chapter Summary

Gene function prediction addresses the problem of predicting unknown associations between gene and functions (e.g., biological processes) of a specific organism. Despite recent advances, the cost and time demanded by annotation procedures that rely largely on *in vivo* biological experiments remain prohibitively high. This chapter presents a novel *in silico* method for the annotation problem that combines cluster analysis and hierarchical multi-label classification (HMC). The method uses spectral clustering to extract new features from the gene co-expression network (GCN) and enrich the prediction task. HMC is used to build multiple estimators that consider the hierarchical structure of gene functions. The proposed method is applied to a case study on *Zea mays*. The results illustrate how *in silico* method are key to reduce the time and costs of gene annotation. More specifically, they highlight the importance of: (i) building new features that represent the structure of gene relationships in GCNs to annotate genes; and (ii) taking into account the structure of biological processes to obtain consistent predictions.

Connections to Previous Chapter

Two additional local and a global HMC methods are explored for the problem of predicting gene functions. In particular, we apply the same method to generate GO sub-hierarchies of biological processes from Chapter 3 to maize, based on the same input data, namely, gene co-expression networks, functional information, and the GO hierarchy of biological processes. Furthermore, an affinity graph that combines the GCN and functional information is introduced. Finally, a novel method to extract features using clustering techniques is presented. It is shown that extracting new features is key to improve the predictive performance. The main result suggests that using an affinity graph leads to higher performance than predicting gene functions based on GCN alone.

Extracting features from biological data is key to training machine learning methods that address biological problems. Depending on the type of data available, different techniques can be used for extracting features. For example, given a DNA sequence of an organism, embeddings can be used to transform the string of nucleotide bases to a numerical representation. When gene co-expression data is characterized as a GCN, the adjacency matrix (or a transformation, such as the Laplacian), topological properties of the nodes or node embeddings are some of the features that can be extracted from the GCN.

This chapter presents a feature extraction method for *in silico* annotation of genes. It follows a network-based approximation that uses cluster analysis and hierarchical multi-label classification (HMC) for building a predictor that assigns functions to genes satisfying the true-path rule. Cluster analysis plays the role of enriching the information available for predicting gene-function associations and extracting new features that represent structural properties of the GCN. Co-expression relations are used to identify gene clusters that ultimately help in associating functions to genes (i.e., guilt by association, see [59]). It has been shown that new features built from the GCN and associations between genes and functions with the spectral clustering algorithm are key to improve the prediction performance in the gene annotation problem [68]. The results in [68] show that using other features associated to structural properties of the GCN and gene functional information lead to lower performance.

The extracted features are filtered (using SHAP) based on their impact in the prediction task. HMC is used to predict gene-function associations taking into account the relations between biological functions. The proposed method illustrates how the performance of gene annotation is improved by combining: (i) new information extracted from the GCN; and (ii) classification methods that consider the relation between gene functions.

The method is applied to a case study on *Zea mays*, one of the most dominant and productive crops worldwide. *Zea mays* serves a variety of purposes, including animal feed and derivatives for human consumption and ethanol [96]. The co-expression information is imported from the ATTED-II database [54]. The resulting GCN, modeled as a weighted graph, comprises 26,131 vertices (i.e., genes) and 44,621,533 edges. The functional information (i.e., known gene-function associations) is taken from DAVID Bioinformatics Resources [28]. It contains a total of 255,865 annotations of biological processes for maize, i.e., pathways to which a gene contributes. The results highlight the importance of extracting features that represent structural properties of the GCN and the hierarchical structure of biological processes with HMC to improve prediction performance. Ultimately, the results provide experimental (*in silico*) evidence that the proposed method is a viable and promising approximation to gene function prediction.

This chapter is a extended version of [68], which:

- Addresses the gene function prediction as a hierarchical multi-label classification problem by considering the structure of gene functions (as in [23], ancestral relationships are represented as a DAG).
- Analyzes a larger functional database for the case study of maize. The new dataset consists of 255,865 associations between genes and functions, and 7,021 relations between functions. The number of genes associated to at least one function almost doubled from 5,361 to 10,049.
- Concludes that the ancestral relations between functions and the features extracted from the GCN improve the prediction performance when it is addressed as a hierarchical multi-label classification problem.

The remainder of the chapter is organized as follows. Section 4.1 reviews related work. Section 4.2 introduces the method to extract features from the gene co-expression network using cluster analysis. The proposed method to predict gene functions, based on hierarchical multi-label classification is presented in Section 4.3. Section 4.4 presents the case study for the *Zea mays* species. Finally, Section 4.5 draws some concluding remarks and future research directions.

4.1 Related Work

Zhou et al. [96] present an method to predict functions of maize proteins using graph convolutional networks. In particular, an amino acid sequence of proteins and the GO hierarchy is used to predict functions of proteins with a deep graph convolutional network. Their results show that their method is a powerful tool to integrate amino acid data and the GO structure to accurately annotate proteins. Similarly, the work in [17] aims to predict the phenotypes and functions associated to maize genes using: (i) hierarchical clustering based on datasets of transcriptome (set of molecules produced in transcription) and metabolome (set of metabolites found within an organism); and (ii) GO enrichment analyses. Their results show that profiling individual plants is a promising experimental design for narrowing down the lab-field gap.

Gligorijević et al. [24] propose a network fusion method based on multimodal deep autoencoders to extract high-level features of proteins from multiple interaction networks. This method relies on a deep learning technique that captures relevant protein features from different complex, non-linear interaction networks. Their results show that extracting new features from biological

networks is key to annotate gene with functions. The work in [95] is also closely related. They present Gene Ontology hierarchy preserving hashing, a gene function prediction method that retains the hierarchical order between GO terms. It uses a hierarchy preserving hashing technique based on the taxonomic similarity between terms to capture the GO hierarchy. Hashing functions are used to compress the gene-term association matrix, where the semantic similarity between genes is used to predict the functions of the genes. Their results show that the method preserves the GO hierarchy and improves prediction performance.

The authors in [14] present a Python-based toolkit for generating numerical feature representation schemes from protein sequences. It integrates algorithms for feature clustering, selection, and dimensionality reduction to facilitate training, analysis, and benchmarking of machine-learning models. In a related study, Mu et al. [51] show that feature extraction of protein sequences is helpful for predicting protein functions and interactions. They introduce feature extraction based on graphical and statistical features, a novel feature extraction method for protein sequences that combines graphical and statistical features. Their results show that similarity analysis of protein sequences has applications in the study of gene annotation, gene function prediction, identification and construction of gene families, and gene discovery.

4.2 Clustering-based Feature Extraction

An method for extracting features from the GCN using a clustering algorithm and Gene Ontology term enrichment is presented. It combines information from the GCN and the associations between genes and functions to create features capturing topological properties of a GCN.

The inputs of the method consists of several elements: a GCN, denoted by $G = (V, E, w)$, a set of (biological) functions A, an annotation function $\phi : V \to 2^A$, and a set $K = \{k_0, \ldots, k_{m-1}\}$ for sampling the number of clusters. The annotation function ϕ must satisfy true-path rule for the GO hierarchy [3, 82]. That is, if a gene is associated to a function, then it must also be associated to every ancestor of the function in the hierarchy, and if a gene is not associated to a function, then it must not be associated to any of its descendants.

The outputs are two feature matrices J_G and J_F, of dimension $|V| \times |A||K| \to [0, 1]$, specifying the likelihood of the genes V to be associated to the functions in A when the graph is decomposed in m clusters. Matrices J_G and J_F correspond to the GCN (that is the graph G) and an affinity graph defined the next subsection.

The feature extraction method consists of three stages, which are depicted in Figure 4.1. First, an affinity graph F with information in ϕ is created from G. Second, the spectral clustering algorithm is applied to both G and (its enriched version) F for the m different number of clusters specified in K. Third, the Gene Ontology term enrichment technique is used to create m features for each function $a \in A$, corresponding to the number of clusters in K.

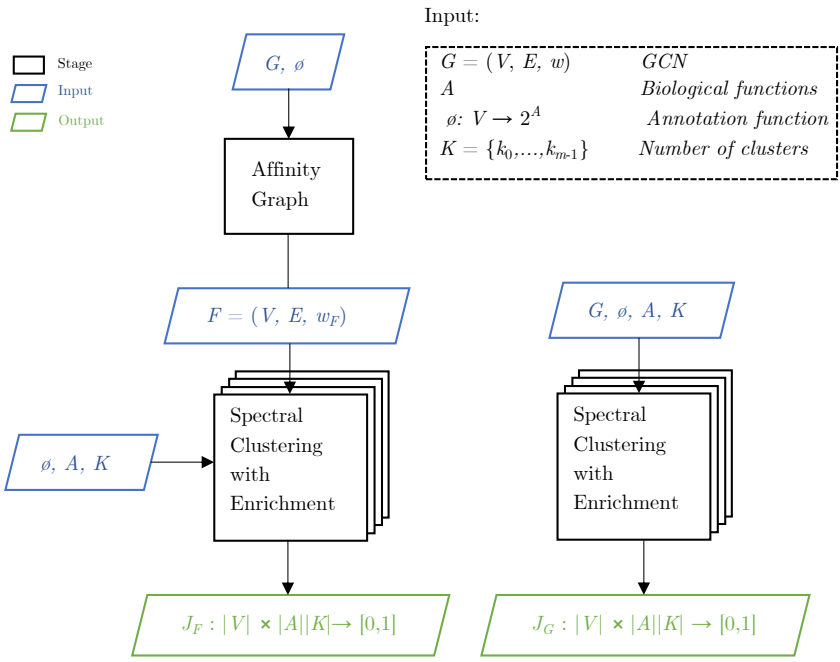


Figure 4.1: The clustering-based feature extraction method consists of three stages. Namely, creation of affinity graph, clustering computation, and Gene Ontology term enrichment. Its inputs are a GCN, denoted by $G = (V, E, w)$, a set of functions A, an annotation function $\phi : V \to 2^A$, and a set $K = \{k_0, \ldots, k_{m-1}\}$. Its output are two feature matrices (for both G and its enriched version F) of dimension $|V| \times |A||K| \to [0, 1]$ that specify how likely it is for the genes to be associated to the functions in A when the graph is decomposed into m clusters, each of size k_i, for $0 \leq i \leq m$.

4.2.1 Affinity Graph Creation

An affinity graph $F = (V, E, w_F)$ between G and ϕ is built. Its weight function is defined as the mean between the co-expression weight specified by w and the proportion of shared functions between genes specified by ϕ.

Definition 8. *The* weight function $w_F : V \times V \to [0, 1]$ *is defined for any* $u, v \in V$ *as*

$$w_F(u, v) = \frac{1}{2}\left(\frac{w(u, v) - 1}{\max(w) - 1} + \frac{|\phi(u) \cup \phi(v)|}{|\phi(u) \cap \phi(v)|}\right),$$

where $\max(w)$ *denotes the maximum value in the range of* w *(which exists because* w *is finite).*

Under the assumption that at least one element in the range of w is greater than 1, it is guaranteed that the range of w_F is $[0, 1]$ (because $w : V \times V \to [1, \infty)$). This is indeed the case, in practice, because the co-expression between two genes in the GCN is quantified in terms of the z-score, which is highly unlikely to be 1 for all pairs of genes.

4.2.2 Gene Clustering

The spectral clustering algorithm is applied independently to each graph $\mathcal{G} \in \{G, F\}$ to decompose $\mathcal{G}$ (i.e., group the genes V) using the number of clusters specified by $K = \{k_0, \ldots, k_{m-1}\}$. The decomposition of $\mathcal{G}$ is performed m times, once per each k in K. The adjacency matrices of the weighted and undirected graphs G and F are used as the precomputed affinity matrices required for the spectral clustering algorithm. The outcome of the clustering algorithm is an assignment from nodes to clusters of size k, for each $k \in K$. More precisely, the outputs of this stage are the matrices $M_{\mathcal{G}} : V \times K \to [0, 1]$, where each column $0 \leq i < m$ represents the decomposition of $\mathcal{G}$ in k_i clusters.

4.2.3 Gene Enrichment

The goal of this stage is to produce a matrix $J_{\mathcal{G}} : |V| \times |A||K| \to [0, 1]$ for each $\mathcal{G} \in \{G, F\}$, specifying how likely it is for the genes to be associated to every function $a \in A$ when $\mathcal{G}$ is decomposed in the given number of clusters.

For each decomposition from the previous stage (i.e., each column of the matrices $M_{\mathcal{G}}$) and function $a \in A$, the resulting clusters are used to compute whether

a significant number of members associated to function a is (locally) present. Intuitively, if genes that are grouped together have a strong co-expression relation and most of the group are associated to gene function a, then the remaining genes are also likely to be associated to a (i.e., guilt by association, see [59]). In this way, for each $v \in V$, $a \in A$, and $k \in K$, the entry $J_{\mathcal{G}}(v, a \cdot k)$ is a p-value indicating if the function a is over-represented in the decomposition of k clusters of $\mathcal{G}$. This process is commonly known as Gene Ontology term enrichment and may use different statistical tests, such as, Fisher's exact test [91].

4.3 Hierarchical Multi-label Classification for Gene Function Prediction

This section presents the method for gene function prediction using HMC to create a predictor, enriched with the information of the features created in Section 4.2.

The GO hierarchy contains three types of annotations: biological processes, molecular functions, and cellular component [23]. This work focuses on biological processes, i.e., a subgraph of the GO hierarchy that contains 28 roots (i.e., functions in the GO hierarchy with null indegree). This subgraph is denoted as $H = (A, R)$, where A is the set of biological processes and R the binary relation representing ancestral relations between pairs of biological processes (i.e., $(a, b) \in R$ means that function a is parent of function b in the GO hierarchy). The topological-sorting traversal algorithm presented in Chapter 3 is used to transform the GO hierarchy of biological processes into a tree. As a result, the hierarchy is split into several components, i.e., subtrees of H called sub-hierarchies. Each sub-hierarchy, $H' = (A', R')$ with $A' \subseteq A$, $R' \subseteq R$, and $r \in A'$ the root, is associated to a subgraph $G' = (V', E', w)$ containing all genes $v \in V$ associated to r, i.e., $V' = \phi^{-1}(r)$. The proposed method is independently applied to each sub-hierarchy.

The inputs of the method are a sub-hierarchy $H' = (A', R')$, a subgraph of the GCN, denoted by $G' = (V', E', w)$, where $V' \subseteq V$ and $E' \subseteq E$, an annotation function $\phi : V \rightarrow 2^{A'}$, the matrices J_G and J_F resulting from Section 4.2, and a constant value $c \in [0, 1]$ for feature selection. The output is a function $\psi : V' \times A' \rightarrow [0, 1]$, specifying the probability $\psi(v, a)$ of v being associated to function $a \in A'$ for each gene $v \in V'$.

First, sub-matrices J'_G and J'_F are created from J_G and J_F, by respectively considering only the genes $V' \subseteq V$ and functions $A' \subseteq A$. These sub-matrices represent structural properties of the GCN subgraph G', and associations

between genes and functions based on multiple partitions of each graph. Figure 4.2 illustrates the prediction method. The reminder of this section is devoted to detailing the prediction method.

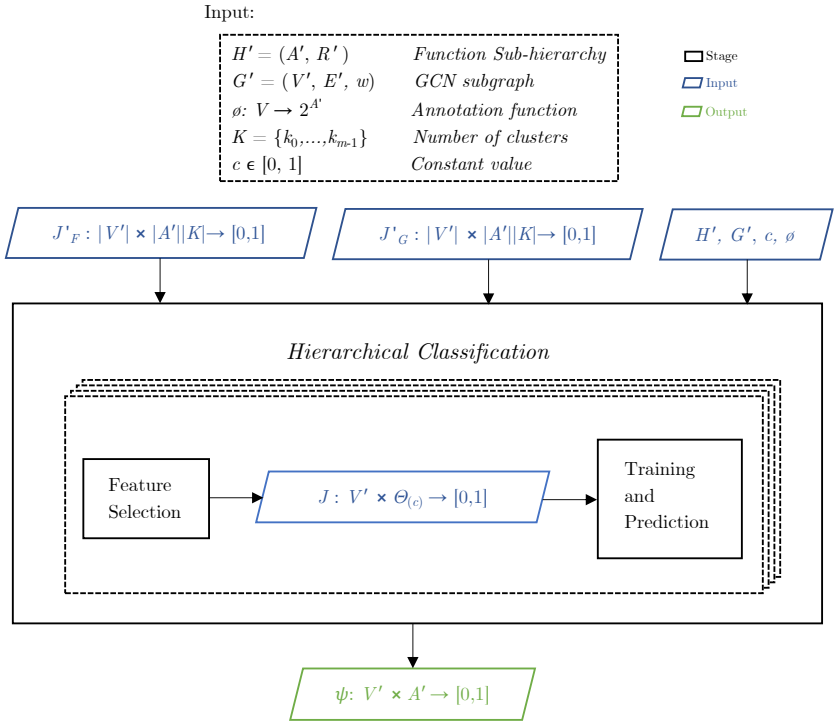


Figure 4.2: The prediction method mainly consists of two stages, feature selection with SHAP and hierarchical multi-label classification. Its inputs are a sub-hierarchy $H' = (A', R')$, a subgraph of the GCN $G' = (V', E', w)$, an annotation function $\phi : V \to 2^{A'}$ that satisfy the sub-hierarchy H', the sub-matrices of J_G and J_F containing only the functions A' and genes V', and a constant value $c \in [0, 1]$ for feature selection. Its output is a function $\psi : V' \times A' \to [0, 1]$, which indicates for each gene $v \in V'$, the probabilities $\psi(v, a)$ of v being associated to function $a \in A'$.

SHAP filters the extracted features with more impact in the prediction task, and HMC is used to predict associations between genes and functions without inconsistencies (i.e., complying the true-path rule). Since local HMC methods use more than one predictor per hierarchy, the feature selection is executed for each predictor independently, considering only the features related to the

functions being predicted, denoted by $A'' \subseteq A'$. For example, consider the function hierarchy and a local classifier per level method depicted in Figure 4.3. The predictor for level 2 predicts functions c, d, e, and f, so only the features associated to functions c, d, e, and f are considered for the feature selection.

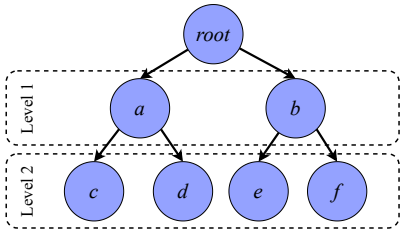


Figure 4.3: Gene function prediction considering the function hierarchy and using a local classifier per level method. The predictor for the level 1 predicts functions a and b, so only the features from J'_G and J'_F associated to functions a and b are considered for feature selection.

4.3.1 Feature Selection

The aim of feature selection is to produce a matrix $J : V' \times \Theta(c) \rightarrow [0, 1]$ by selecting a reduced number of significant features from J'_G and J'_F. The number of selected features is denoted by $0 \leq \Theta(c) \leq 2m \cdot |A''|$, where $m \cdot |A''|$ is the number of features in each matrix J'_G and J'_F, denoted as q (i.e., $q = m \cdot |A''|$).

Feature selection is conveyed from J'_G and J'_F to J using SHAP. Let J'_{G+F} denote the matrix resulting from extending J'_G with the q features of J'_F. That is, for each $v \in V'$, the expression $J'_{G+F}(v, \_)$ denotes a function with domain $[0, 2q)$ and range $[0, 1]$, where the values in $[0, q)$ denote the p-values associated to v in G and the values in $[q, 2q)$ the ones associated to v in the enriched version of G. For each entry $J'_{G+F}(v, j)$, with $v \in V'$ and $0 \leq j < 2q$, the mean absolute SHAP value $s_{(v,j)}$ is computed after a large enough number of Shapely values are computed (executions of SHAP). Features are selected based on the cutoff

$$c \cdot \sum_{j=0}^{2q-1} s_{(v,j)},$$

i.e., on the sum of mean absolute values by a factor of the input constant c. The first $\Theta(c)$ features, sorted from greater to lower mean absolute SHAP value, are selected as to reach the given cutoff.

Note that the input constant c is key for selecting the number of significant features. The idea is to set c so as to find a balance between prediction efficiency and the computational cost of building the predictor.

4.3.2 Training and Prediction

This stage comprises a process that combines two supervised machine learning techniques to build the predictor ψ. In particular, stratified k-fold cross-validation and hierarchical multi-label classification are used sequentially in a pipeline.

The pipeline takes as input the matrix J, which specifies the significant features of J'_G and J'_F, the sub-hierarchy H', and the annotation function ϕ. First, k-fold is applied to split the dataset into k different folds for cross validation (note that k is not related to the input K). That is, each fold is used as a test set, while the remaining $k-1$ folds are used for training. Recall that k-fold cross validation aims to overcome overfitting in training. Furthermore, one or multiple random forest classifiers are built and used for prediction, the number of classifiers depends on the HMC method. Random forest is selected for this method because it is a tree-based and multi-label classification algorithm, and SHAP can be applied. The parameter values used for random forest classifiers, differently from the default scikit-learn values, are: 200 estimators (*n_estimators*) and minimum number of samples of 5 (*min_samples_split*).

Some HMC methods require an extra step to keep prediction consistent w.r.t. the sub-hierarchy H' (i.e., to comply the true-path rule). The probability of association between a function $v \in V'$ and a function $b \in A'$ must be lower than the probability of association between the same gene and the ancestor of b in H'. To satisfy this constraint, cumulative probabilities are computed for all the paths in H'. That is, for each gene $v \in V$ and functions $(a, b) \in R$, the predicted probability of the association between v and b is multiplied by the predicted probability of association between v and a (its ancestor). This process is repeated for every path in the hierarchy from the root to the leaves.

The output of this stage is the predictor ψ, i.e., the probabilities of associations between the genes in V' and functions A'. Note that the predictor ψ satisfies the true-path rule.

4.3.3 Performance Evaluation

It is often the case in HMC datasets that individual classes have few positive instances. In genome annotation, only a few genes are associated to specific

functions typically. This implies that for most classes (deeper in the hierarchy), the number of negative instances by far exceeds the number of positive instances. Hence, the real focus is recognizing the positive instances (predict associations between genes and functions), rather than correctly predicting the negative ones (predict that a function is not associated to a given gene). Although ROC curves are better known, their area under the curve is higher if a method correctly predicts negative instances, which is not suitable for HMC problems.

For this reasons, the measures (based on the precision-recall (PR) curve) introduced in [85] are used for evaluation.

Area under the average PR curve. This metric transforms the multi-label problem into a binary one by computing the precision and recall for all functions A'. This corresponds to micro-averaging the precision and recall.

The output of the prediction stage are the probabilities of associations between genes V' and functions A'. Thereby, instead of selecting a single threshold to compute precision and recall, multiple thresholds are used to create a PR curve. In the PR curve, each point represents the precision and recall for a given threshold; it can be computed as:

$$\overline{\text{Prec}} = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FP_i}, \quad \text{and} \quad \overline{\text{Rec}} = \frac{\sum_i TP_i}{\sum_i TP_i + \sum_i FN_i}.$$

Note that i ranges over all functions A', i.e., precision and recall are computed for all functions together. The area under this curve is denoted as $\text{AU}(\overline{\text{PRC}})$.

Average area under the PR curves. The second metric corresponds to the (weighted) average of the areas under the PR curves for all functions A'. This metric, referred as macro-average of precision and recall, can be computed as:

$$\overline{\text{AUPRC}}_{w_1,w_2,\ldots,w_{|A'|}} = \sum_i w_i \cdot \text{AUPRC}_i.$$

If the weights of all functions are the same (i.e., $1/|A'|$) the metric is denoted as $\overline{\text{AUPRC}}$. In addition, weights can also be defined based on the number of genes associated to functions in ϕ, i.e., $w_a = |\phi^{-1}(a)|/\sum_i |\phi^{-1}(i)|$ for each $a \in A$. In the latter case, denoted as $\overline{\text{AUPRC}}_w$, more frequent functions get higher weight. Note that one point in the weighted PR curve corresponds to the (weighted) average of the AUPRC of all functions in A' for a given threshold.

4.4 Case Study: *Zea mays*

We describe a case study on applying the feature extraction and prediction method presented in Sections 4.2 and 4.3 to maize (*Zea mays*). First, the maize data used for the case study is described. Second, the proposed method is applied to the data. Lastly, the performance of the proposed method is compared to two models trained using each set of features J_G and J_F, independently.

4.4.1 Data Description and Feature Extraction

The co-expression information used in the study is imported from the ATTED-II database [54]. The gene co-expression network $G = (V, E, w)$ comprises 26,131 vertices (genes) and 44,621,533 edges. In this case, a z-score threshold of 1 is used as the cut-off measure for G, i.e., E contains edges e that satisfy $w(e) \geq 1$ (most of them satisfying $w(e) > 1$). Note that the highest value is assigned to the strongest connections. The functional information for this network is taken from DAVID Bioinformatics Resources [28] (2021 update); it contains annotations of biological processes, i.e., pathways to which a gene contributes. It is important to note that genes may be associated to several biological processes, and biological processes may be associated to multiple genes. The database comprises 3,924 biological processes A and 7,021 ancestral relations R between these functions, that represent the hierarchy $H = (A, R)$ of the GO [23]. A total of 255,865 association between genes and functions are considered; these associations are identified in the annotation function $\phi : V \to 2^A$.

The feature extraction method is applied on the inputs G, A, ϕ, and $K = \{10, 20, \ldots, 100\}$ (values are incremented in steps of 10 up to 100). The outputs are the feature matrices J_G and J_F that specify how likely it is for the maize genes V to be associated to the biological processes A when the graph is decomposed in each of the number of clusters in K.

Only functions associated to more than 200 genes are considered, so the number of functions in the resulting sub-hierarchies is tractable regarding the dimension of the output of SHAP (see Section 2). Recall that the Gene Ontology hierarchy splits into 28 sub-hierarchies when considering only biological processes. Additionally, all sub-hierarchies with less than 10 functions are discarded and the topological-sorting algorithm introduced in Chapter 3 is used to transform the sub-hierarchies, represented as DAGs, into trees. For each ancestral relation $(a, b) \in R$ (a is parent of b), the algorithm assigns a weight as the ratio of the number of genes associated to b to the number of genes associated to a. Then, for each function $b \in A'$ with more than one parent, only the parent with the higher weight remains (ties are broken arbitrarily).

Root	Description	Functions	Genes	Functions per level
GO:0050896	response to stimulus	13	1733	5/5/2
GO:0051179	localization	25	1497	3/5/9/6/1
GO:0065007	biological regulation	37	2647	2/5/11/10/4/2/2
GO:0008152	metabolic process	92	6596	8/18/38/12/7/6/2
GO:0009987	cellular process	92	8005	13/19/19/17/13/8/2

Table 4.1: Resulting sub-hierarchies H' of biological processes for maize. The identifier and description of each root function r is presented in the first and second columns, respectively. The third column shows the number of functions A' within each sub-hierarchy and the fourth column shows the number maize genes in the GCN subgraph G' associated to H'. The last column shows the number of functions per level, e.g., the first sub-hierarchy has 3 levels and there are 5, 5, and 2 functions on each level.

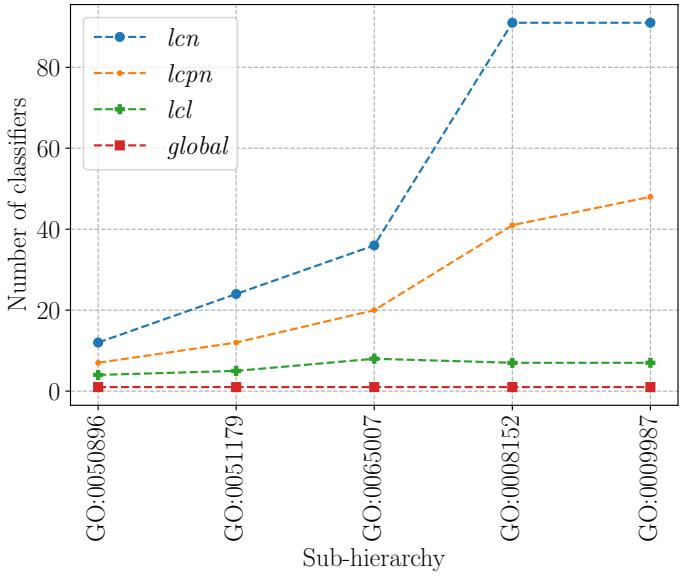


Figure 4.4: Number of classifiers trained per HMC method and sub-hierarchy. The *lcn* requires $|A'| - 1$ classifiers. The *lcpn* requires as many classifiers as functions with children in H'. The *lcl* requires as many classifiers as the number of levels in H'. At last, the global method requires one classifier per hierarchy.

As result, there are 5 sub-hierarchies of biological processes. Table 4.1 describes each sub-hierarchy H', starting by the root term r and its description, following the number of functions A' and the number of genes V' in the associated GCN subgraph G'. The prediction method is applied to each sub-hierarchy H' independently. The remaining input parameter for the prediction method is $c = 0.9$ (recall that this parameter is used to filter the most relevant features according to their mean SHAP value). Figure 4.4 depicts the number of classifiers trained per HMC method and sub-hierarchy. Note that the global method requires one classifier per hierarchy, while the *lcn* requires $|A'| - 1$ classifiers per hierarchy.

4.4.2 Summary of Results

Figure 4.5 presents the prediction performance of the proposed method measured with the AU($\overline{\text{PRC}}$) (denoted as *micro*) for four HMC methods, namely, local classifier per node (*lcn*), local classifier per parent node (*lcpn*), local classifier per level (*lcl*), and global classifier. In general, it can be seen that all methods get a high area under the average PR curve, but the global classifier outperforms the local methods for all sub-hierarchies. The proposed method identifies the associations between genes and functions by using the features extracted from the GCN G and the affinity graph F, and complying with the ancestral relations of the biological processes. The global method obtains the best performance, followed by the *lcpn* and the *lcl*. Using multi-label classifiers is better than using a binary classifier for each function, i.e., *lcn* method.

The micro score measures the overall performance of all functions within a sub-hierarchy without distinguishing between them. Figure 4.6 presents the prediction performance measured with the $\overline{\text{AUPRC}}$, denoted as *macro*. The macro score measure the prediction performance for each function individually and then takes the average. The conclusion is similar: the global method outperforms the local ones.

Finally, Figure 4.7 illustrates the prediction performance measured with the $\overline{\text{AUPRC}}_w$, denoted as *macro weighted*. This score weights the individual performance of each function according to the number of genes associated to it. Thereby, the leaves and deeper functions in a sub-hierarchy always get lower weight than the others. Note that the deeper a functions is in a sub-hierarchy, the lower the predicted probabilities becomes. The global method outperforms the locals again. The conclusion is consistent with the three metrics, using clustering techniques to extract features from the GCN and considering the hierarchical structure of the biological processes seems to be key for the gene function production task.

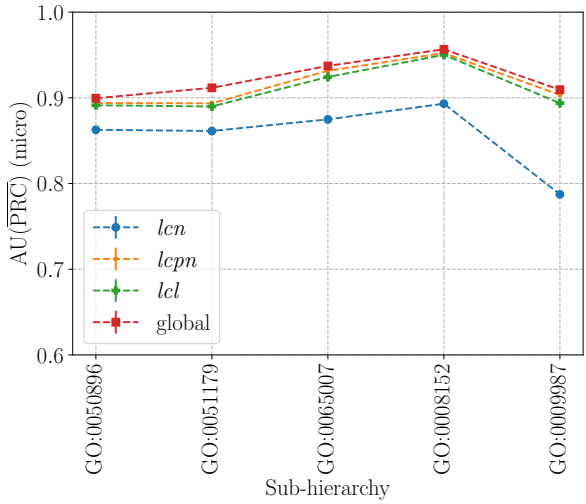


Figure 4.5: Prediction performance of the proposed method measured with the area under the average PR curve, i.e., AU($\overline{\text{PRC}}$). The performance is measured independently per sub-hierarchy.

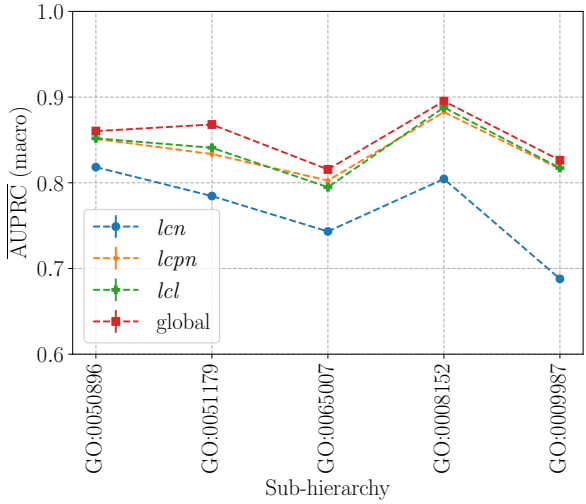


Figure 4.6: Prediction performance of the proposed method measured with the average area under the PR curve, i.e., $\overline{\text{AUPRC}}$. The performance is measured independently per sub-hierarchy.

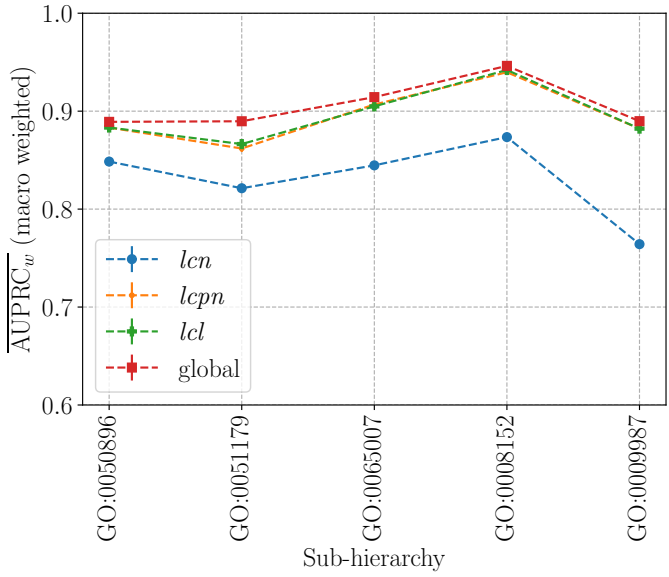


Figure 4.7: Prediction performance of the proposed method measured with the average area under the PR curve, i.e., $\overline{\text{AUPRC}_w}$. The performance is measured independently per sub-hierarchy.

Root	Total	Filtered
GO:0050896	239	124
GO:0051179	479	263
GO:0065007	713	402
GO:0008152	1812	796
GO:0009987	1813	853

Table 4.2: Number of extracted and filtered features used for the global method per sub-hierarchy. Recall that the extracted features are filtered using the mean SHAP values to select the more important with a cutoff defined by the input constant c.

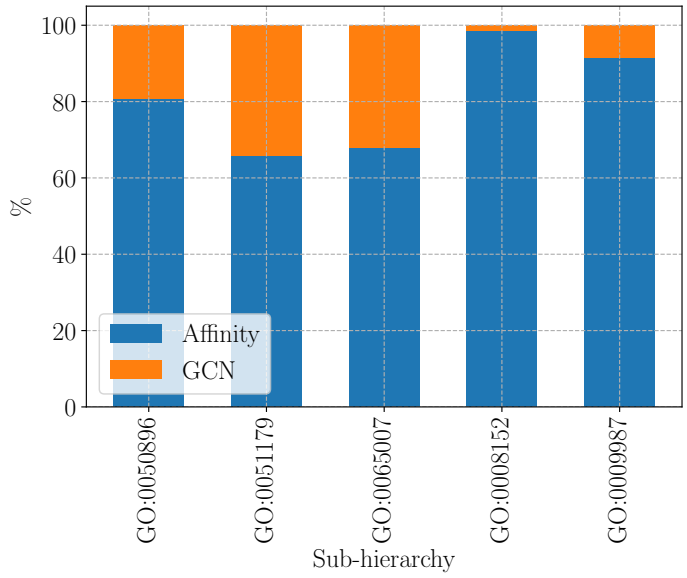


Figure 4.8: Distribution of the filtered features from J_G and J_F for the global method per sub-hierarchy.

It has been shown in [68] that the new features built from the GCN, and the associations between genes and functions with the spectral clustering algorithm are key to improve the prediction performance in the gene annotation problem (w.r.t. other features of the GCN and gene functional information). However, the feature extraction method presented in Section 4.2 produces two different sets of features, namely, J_G and J_F, that are combined and used for prediction. The individual relevance of each set of features for the gene annotation problem is analyzed by: (i) looking at the distribution of the filtered features for the global method and (ii) comparing the performance of the prediction task using each set of features independently. Table 4.2 presents the number of extracted and filtered features used for the global method per sub-hierarchy. Recall that the features are filtered using the mean SHAP values to select the more important ones with a cutoff defined by the input constant c.

Figure 4.8 illustrates the distribution of the filtered features for the global method per sub-hierarchy. Note that, even though the features from the affinity graph F (i.e., J_F) are more important, features from the GCN G (i.e., J_G) are also selected for all sub-hierarchies. Figure 4.9 shows the prediction performance of the global HMC method trained using the features J_G and J_F independently,

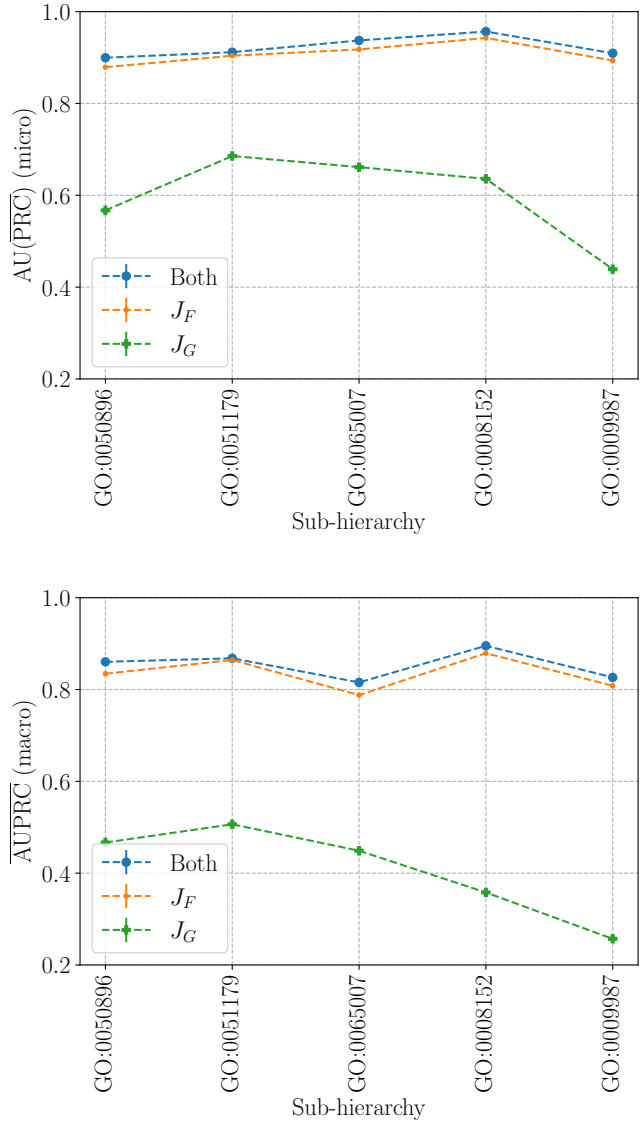


Figure 4.9: Prediction performance of the global method trained using the features J_G and J_F independently, and the proposed method (i.e., their combination) measured with AU(PRC) and AUPRC. The performance is measured independently per sub-hierarchy.

and the proposed method (i.e., their combination) measured with AU($\overline{\text{PRC}}$) and $\overline{\text{AUPRC}}$. The combination of both sets of features, extracted from the GCN and the affinity graph, is key to improve the performance of the proposed method for all sub-hierarchies.

4.5 Concluding Remarks

By combining network-based modeling, cluster analysis, interpretable machine learning, and hierarchical multi-label classification, the method presented in this chapter introduces a novel method to address the gene function prediction problem. It aims to predict the association probability between each gene and function by taking advantage of the GCN spectral decomposition, the information available of associations between genes and functions, and the ancestral relations between the functions (i.e., the GO hierarchy).

A case study on *Zea mayz* (maize) is presented. Using the structural information of the gene co-expression network (extracted by a spectral clustering algorithm) and considering the hierarchical structure of the biological processes (using HMC) seems to be the key for the improved performance of the proposed method. More precisely, the global HMC method, which considers all features available for a sub-hierarchy to build a single classifier, outperforms the other methods in relation to the three metrics that were used (namely, AU($\overline{\text{PRC}}$), $\overline{\text{AUPRC}}$, and $\overline{\text{AUPRC}_w}$).

The results presented in [68] show that the features extracted from the GCN using spectral clustering lead to better prediction performance in the gene function prediction task (addressed as an independent binary classification problem per function). In this chapter, it has been shown that considering the ancestral relations between functions to produce an outcome that satisfies its hierarchical structure (i.e., complies the true-path rule or hierarchical constraint), based on the features extracted from the GCN, improves the performance in the gene function prediction task (addressed as a hierarchical multi-label classification problem).

Availability of Data and Materials The datasets analyzed for the current study are publicly available from different sources. Gene co-expression data of rice is available at ATTED-II (version r21c) [54], functional data of rice is available at DAVID Bioinformatics Resources [28], and hierarchical data of Gene Ontology terms is available at GOATOOLS Python library [36].

The data collected, cleaned, and processed from the above sources as used in the case study can be requested to the authors. The proposed method was implemented in Python 3 and is publicly available in [69].

Chapter 5

Hierarchy Exploitation to Detect Missing Annotations on Hierarchical Multi-Label Classification

This chapter has been submitted for publication as:
Romero, M., Nakano, F.K., Finke, J., Rocha, C., and , Vens C. Hierarchy exploitation to detect missing annotations on hierarchical multi-label classification. Computers in Biology and Medicine, submitted for publication (2022).

Chapter Summary

The availability of genomic data has grown exponentially in the last decade, mainly due to the development of new sequencing technologies. Based on the interactions between genes (and gene products) extracted from the increasing genomic data, numerous studies have focused on the identification of associations between genes and functions. While these studies have shown great promise, the problem of annotating genes with functions remains an open challenge. In this chapter, we present a method to detect missing annotations in hierarchical multi-label classification datasets. We propose a method that exploits the class hierarchy by computing aggregated probabilities to the paths of classes from the leaves to the root for each instance. The proposed method is presented in the context of predicting missing gene function annotations, where these aggregated probabilities are further used to select a set of annotations to be verified through *in vivo* experiments. The experiments on *Oriza sativa Japonica*, a variety of rice, showcase that incorporating the hierarchy of classes into the method often improves the predictive performance. The method yields superior results when compared to competitor methods from the literature.

Connections to Previous Chapter

The HMC methods for predicting gene functions introduced in the previous chapters aim to predict all possible gene-function pairs. That is, a probability or value is predicted for each association between genes and functions that determines if the association exists according to the classifier. Unlike Chapters 3 and 4, this chapter introduces a method to detect missing annotations in a HMC task, i.e., the method focuses on identifying associations between genes and functions that have not been previously identified. The predictive performance of the method is evaluated using two versions of the functional information, which allow to verify if the associations detected by the model are correct or not. In addition, a global HMC classifier is used based on the results from Chapter 4.

Gene functional annotation is generally addressed as a (hierarchical) multi-label classification problem. A key assumption is that the functional information available (present annotations as well as the absent ones) can be trusted and can be used as a training set to construct an inductive model. However, it is well known that functional information of genes (and genes products) is incomplete [82]. Thus, it is important to consider methods that are able to handle incomplete labeled data and that focus on detecting missing annotations [93, 72].

This chapter introduces *hieRarchical multi-labEl clAsSification to diScover mIssinG aNnotations* (*REASSIGN*), a method to detect missing annotations based on HMC that exploits the class hierarchy to select a set of annotations (e.g., gene-function associations). Its specific purpose is twofold: first, it can be used to complete a given annotation dataset; second, the completed dataset can be used to create better supervised models.

To this aim, HMC classifiers based on tree ensembles are used to compute the probability of association of every instance-class pair (e.g., genes and biological functions). For each gene, aggregated probabilities are computed for the paths in the class hierarchy, where a path is a sequence of classes with ancestral relations from a leaf to the root of the hierarchy. Aggregated probabilities are used to group paths of classes instead of using single instance-pair associations. Based on the aggregated probabilities of the paths and genes, a set of annotations absent in the given annotation dataset and complying with the hierarchy constraint is selected as output.

The method is evaluated on *Oryza sativa Japonica*, a variety of rice, and it is compared to different methods from the literature [72, 52]. In addition, eight HMC datasets of biological processes from the GO hierarchy for rice are introduced. These datasets correspond to subsets of rice genes and biological processes (GO sub-hierarchies), whose features are structural properties and embeddings of the gene co-expression network. The results show that the proposed method outperforms the comparison methods in most cases. We find that exploiting the hierarchy of functions helps to better identify gene-function associations. The evidence suggests that this is a promising approach for reducing the cost, time, and effort required for experimental verification in a lab.

The remainder of the chapter is organized as follows. Section 5.1 reviews related work. Section 5.2 introduces the method to detect missing annotations exploiting the class hierarchy. Section 5.3 describes the datasets and experimental setup for the gene function prediction in *Oryza sativa Japonica*, followed by the results and discussion in Section 5.4. Finally, Section 5.5 draws concluding remarks and future research directions.

5.1 Related Work

Some studies have focused on hierarchical multi-label classification across different domains. For example, Dimitrovski et al. [20] presented a global method that addresses HMC using random forests of predictive clustering trees (PCTs) to annotate images. Ramírez et al. [60] introduced a local method based on chained path evaluation. It used a classifier to train non-leaf classes (i.e., classes with at least one descendant) in the hierarchy, including information on ancestral relations through extra features with the prediction of parent classes.

Other studies have focused on the gene (or gene products) function prediction problem. For example, Jiang et al. [30] proposed Hierarchical Binomial-Neighborhood, a probabilistic and local HMC method to predict protein functions in yeast. Their results showed that their method outperforms other methods based on independent class prediction. However, it requires a high computational cost to compute probabilities of every protein-function pair. Yu et al. [93] presented a method to replenish the missing function labels and to predict functions for unlabeled proteins in a hierarchical manner assuming that the labeled data was incomplete. Their method combines the hierarchical structure of functions and the similarity between labels to identify interaction between proteins and functions using guilt by association [59].

Zhao et al. [95] presented Gene Ontology Hierarchy Preserving Hashing, a gene function prediction method that retains the hierarchical order between GO functions. It used a hashing technique based on the taxonomic similarity between functions to capture the GO hierarchy and predict gene functions. Their results showed that their method preserved the GO hierarchy and helped to improve prediction performance. Nakano et al. [52] performed a comparison among publicly available HMC methods. According to their results, clus-ensemble, a random forest of predictive clustering trees adapted to HMC [76], provided superior results. However, the authors did not develop a method to identify missing annotations on HMC problems.

Zhou et al. [96] presented a method to predict functions of maize proteins, called Deep Graph Convolutional network model. It uses amino acid sequences of proteins and the GO hierarchy to predict functions of proteins. Their results showed that their method is a powerful tool to integrate amino acid data and the GO structure to accurately annotate proteins. Similarly, Cruz et al. [17] aimed to predict the phenotypes and functions associated to maize genes using: (i) hierarchical clustering based on datasets of transcriptome (set of molecules produced in transcription) and metabolome (set of metabolites found within an organism); and (ii) GO enrichment analyses. Their results showed that profiling individual plants is a promising experimental design for narrowing down the

lab-field gap. At last, Romero et al. [68] presented a method that combines the functional information with the gene co-expression network of an organism to extract features that capture the details of the GO hierarchy using spectral clustering. Their results showed that the extracted features are key to improve the performance of the gene function prediction task on rice, using a global HMC approach of random forests of decision trees.

5.2 Detecting Missing Annotations

This section introduces the definition of the problem of predicting missing gene functions annotations and present a general method to detect missing annotations in HMC problems.

5.2.1 Problem Definition

Given a set of genes V, a set of biological functions A, and an annotation function $\phi : V \rightarrow 2^A$, where each gene is associated with the collection of biological functions to which it is known to be related (e.g., verified through *in vivo* experiments). The goal of detecting missing annotations in the context of gene function prediction is to use the information represented by ϕ, together with additional information about V (e.g., genomic sequences or gene co-expression data), to obtain a function $\psi : V \rightarrow 2^A$ that augments ϕ with previously undetected annotations. In other words, the difference between ϕ and ψ is that the latter includes new associations between genes and functions not present in the former. The new associations identified by ψ are suggestions that need to be verified through *in vivo* experiments. The function ψ can be built from a predictor of gene functions, e.g., based on a supervised machine learning model.

Formally, the gene function prediction problem is defined as a task of detecting missing annotations as follows:

Definition 9. *Let V be a set of genes, A a set of biological functions, and $\phi : V \rightarrow 2^A$ a function describing known annotations. The objective is to obtain a function $\psi : V \rightarrow 2^A$ that augments ϕ by including new associations between gene and functions, and complies with the hierarchy constraint.*

5.2.2 REASSIGN

Given a HMC problem with instances I and a class hierarchy $(C, \leq_h)$, we introduce *hieRarchical multi-labEl clAsSification to diScover mIssinG*

aNnotations (*REASSIGN*), a method to detect missing annotations for I.

The input of the method are a dataset X comprising instances I and features, the class hierarchy represented as a tree with $|C|$ vertices (e.g., hierarchy of biological functions) and an annotation function (e.g., ϕ) represented as a label matrix Y with an assignment of each instance in I to a subset of classes from C (i.e., $Y : I \times C \rightarrow \{0, 1\}$). The output of the method is a suggestion of missing annotations in Y, i.e., a set of annotations whose value in Y is originally 0, but which are believed to be false negatives. Naturally, the suggested annotations must still satisfy the hierarchy constraint.

HMC datasets often have a large and imbalanced label set, especially on deeper levels. In particular, despite being more informative, deeper classes in the hierarchy have less annotations (are sparse), leading to low predicted probabilities and predictive performance overall. As a possible solution, we propose a method that exploits the hierarchy of classes to compute an aggregated probability per instance and path of classes in the hierarchy, relying on the prediction probabilities provided by a HMC classifier. As a result, at most $I \cdot p$ aggregated probabilities are computed corresponding to all combinations between instances and paths, where p is the total number of paths from the leaves to the root in the hierarchy. The aggregated probabilities of the paths are then used as the criterion to select a set of annotations.

The proposed method consists of 3 steps. First, a HMC classifier is used to compute the probability of every instance-class association, i.e., compute $Y' : I \times C \rightarrow [0, 1]$. Any local or global HMC classifier can be used (e.g., tree ensembles or neural networks), providing that the hierarchy constraint is satisfied.

Second, aggregated probabilities are computed for each instance and each path from the leaves to the root in the class hierarchy by using the predicted probabilities of the annotations in Y'. That is, only the paths that go to leaves in the class hierarchy are considered, because the addressed problem is leaf mandatory [78]. Importantly, only new potential annotations are used to compute aggregated probabilities, i.e., instance-class associations satisfying

$$(\forall i, c | i \in I \wedge c \in C : Y[i, c] = 0 \wedge Y'[i, c] > 0).$$

Different ways of aggregating the probabilities can be used; in this work, we used the average, sum, and minimum. Each aggregation function is considered an independent variation of the proposed method:

- **REASSIGN (min)**: aggregates probabilities by using the minimum probability along the path considered. Paths are identified by their most informative (deepest) class;

- **REASSIGN (sum)**: aggregates probabilities by using the sum of the probabilities along the path considered. The longer and deeper the path, the larger the sum, and therefore, more informative classes are considered; and
- **REASSIGN (average)**: aggregates probabilities by using the average of the probabilities along the path considered. It balances the probabilities of the more informative (deeper) and less informative (shallower) classes.

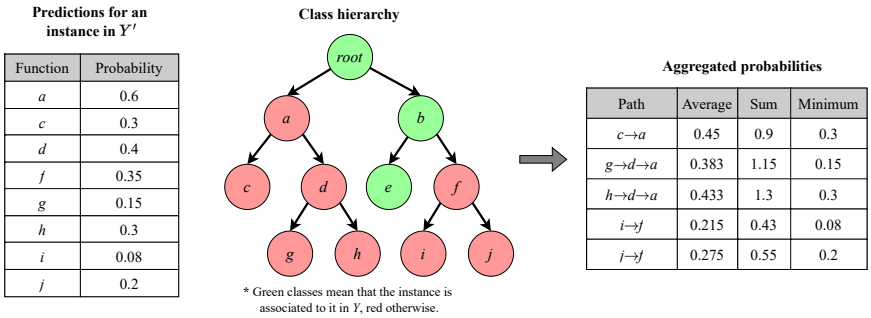


Figure 5.1: Given an instance, aggregated probabilities are computed for every path in the class hierarchy from the leaves to the root using the average, sum and minimum of the probabilities of the annotations in the path. Only those annotations whose value is 0 in Y are considered in the paths.

These variations are illustrated in Figure 5.1, where we exemplify how aggregated probabilities of the paths in the hierarchy are computed for a given instance. Classes coloured with green denote that the instance $i \in I$ is already associated to the class $c \in C$ in Y (i.e., $Y[i, c] = 1$), while classes coloured in red denote that the instance is not associated to the class (i.e., $Y[i, c] = 0$). Since the method is focused in detecting missing annotations (i.e., instance-class pairs associations that have not been identified), only classes coloured with red are used to compute aggregated probabilities. Reinforcing the objective of detecting missing annotations through new paths in the hierarchy. Note that the path $d \rightarrow a$ is not considered since it does not lead to a leaf node.

Some paths are selected using their aggregated probability as selection criterion. The number of paths to be selected is a parameter denoted as N_p. All annotations within the top N_p paths with higher aggregated probability are selected. Note that there might be common annotations between the paths, so duplicates have to be removed. For instance, paths $i \rightarrow f$ and $h \rightarrow f$ in Figure 5.1 share the class f, hence in case both paths are selected, the association between the instance and class f will be duplicated. The resulting number of annotations

is denoted as N and used for comparison with other methods. A detailed description of the proposed method is presented in Algorithm 5.1.

Algorithm 5.2.1: Hierarchical multi-label classification to discover missing annotations (REASSIGN)

```
input:
  X: dataset
  (C, ≤_h): class hierarchy
  Y : I × C → {0,1}: instance-class associations
  f(_): aggregation function
  N_p: Number of paths to select
output:
  top_annot ⊆ {(i,c)|i ∈ I ∧ c ∈ C}: subset of annotations

compute Y' : I × C → [0,1] using a HMC method s.t. Y' complies to
    ↪ the hierarchy constraint.
set all_paths = {}
foreach instance i ∈ I
  foreach path ∈ (C, ≤_h)
    set probs = {} and annot = {}
    foreach c ∈ path
      if Y[i,c] = 0 ∧ Y'[i,c] > 0
        add Y'[i,c] to probs
        add (i,c) to annot
      end
    end
    add (f(probs), annot) to all_paths
  end
end
sort all_paths in decreasing order
set top_paths = all_paths[0 ... N_p)
set top_annot = {}
foreach (x, annot) ∈ top_paths
  add annot to top_annot
end
remove duplicates from top_annot
return top_annot
```

5.3 Experimental Setup

In this section, a detailed description of the employed databases, comparison methods, and evaluation measures is presented.

5.3.1 Datasets

Two datasets are built using the functional information and the gene co-expression network (GCN) for *Oryza sativa Japonica*, a variety of rice [41, 15, 73].

The functional information depicts associations between genes and functions previously identified through *in vivo* or *in silico* experiments. For this work, the functional information is imported from DAVID Bioinformatics Resources [28], that contains annotations of biological processes, i.e., pathways to which a gene contributes. Note that genes may be associated to several biological processes, and biological processes may be associated to multiple genes. The datasets are built using two different versions of the functional information of rice. That is, each dataset has a different label matrix, but they share the instances and features. One dataset (the older version) is used to train and build the models, whereas the other one is used to evaluate the detected missing annotations.

The first version of the functional information is from 2018 and it comprises 3,531 biological processes and 6,367 hierarchical relations that are part of the GO hierarchy [23]. A total of 197,194 associations between genes and functions are considered in this version. The second version is from 2021 and since it is only used for performance evaluation, the same set of biological processes and hierarchical relations are considered. This version comprises a total of 289,407 associations.

The GCN is built using the co-expression information imported from the ATTED-II database (version r17c) [54]. Recall that a GCN is represented as an undirected and weighted graph where each vertex represents a gene and each edge the level of co-expression between two genes [2, 84]. The GCN of rice $G = (V, E, w)$ comprises 19,663 vertices (genes) and 550,813 edges. In this case, a mutual rank threshold of 100 is used as the cut-off measure for G, i.e., E contains edges e that satisfy $w(e) \leq 100$. Note that the lowest value is assigned to the strongest connections.

Biological processes are a subset of the functions in the GO hierarchy, where each function in the topmost level represents a sub-hierarchy. However, as functions can have more than one parent, sub-hierarchies might not be independent (i.e., functions might belong to multiple sub-hierarchies) and there might be several paths between two functions. The topological-sorting traversal algorithm presented in Chapter 3 is used to transform the hierarchy into a tree so that there is unique path between all pair of function in the sub-hierarchies and all sub-hierarchies are independent. Each sub-hierarchy is denoted as $H = (A, \leq_h)$, where A is the subset of biological processes and $\leq_h$ the binary relation representing ancestral relations between pairs of biological processes, i.e., $a \leq_h b$ means that function b is parent of function a in the sub-hierarchy H.

As a result, 8 sub-hierarchies of biological processes are used. Table 5.1 describes each sub-hierarchy H, starting by the root term and its description, followed by the number of biological processes A, the number of genes, the number of new annotations (i.e., 0s that became 1s from 2018 to 2021 version), and the number of functions per level. Note that the functional information from 2021 includes more annotations (i.e., 0s that became 1s), but also drops some of them (i.e., 1s that became 0s). Annotations are dropped from one version to other because it was experimentally verified that such associations between genes and functions do not exist. The prediction method is independently applied to each sub-hierarchy H.

Root	Description	Functions	Genes	$0 \rightarrow 1$	Functions per level
GO:0032501	multicellular organismal process	26	538	184	8/10/6/1
GO:0019752	carboxylic acid metabolic process	63	505	180	7/15/23/15/2
GO:0032502	developmental process	68	871	537	10/19/25/11/2
GO:0006796	phosphate-containing compound metabolic process	71	1142	669	9/16/22/16/7
GO:0051179	localization	112	1285	1630	4/9/21/31/23/16/5/1/1
GO:0065007	biological regulation	291	2137	2879	3/19/57/94/69/23/13/8/3/1
GO:0008152	metabolic process	514	5348	14601	14/47/149/98/72/60/45/18/8/1/1
GO:0009987	cellular process	594	5867	16520	32/67/117/144/93/66/46/16/10/1/1

Table 5.1: Resulting sub-hierarchies of biological processes for rice. The identifier and description of each root function r is presented in the first and second columns, respectively. The following columns show the number of functions A within each sub-hierarchy, the number of genes associated to it, and the number of new annotations (i.e., 0s that became 1s). The last column shows the number of functions per level, e.g., the first sub-hierarchy has 4 levels and there are 8, 10, 6, and 1 function on each level, respectively.

For each sub-hierarchy, we compute and combine two sets of features: structural properties and node embeddings of the GCN. Given a sub-hierarchy H and its associated genes, structural properties of the GCN are computed as features. In this case, the properties included for each gene u are the following:

- degree: number of edges incident to u (including u);
- average neighbor degree: average degree of the neighbors of u;
- eccentricity: maximum shortest distance from u to any node in its connected component;
- clustering coefficient: ratio between the number of triangles (3-loops) and the maximum number of 3-loops that could that pass through u;
- closeness centrality: reciprocal of the average shortest path length from u;

- betweenness centrality: the amount of influence that u has over the interactions of other nodes in the network, measured as the number of shortest paths that pass through u;
- Kleinberg's hub scores: defined as the principal eigenvector of $\mathbf{A}\mathbf{A}^T$, where $\mathbf{A}$ is the adjacency matrix of the graph [35]. Hubs are vertices linked to many other vertices;
- Kleinberg's authority score: defined as the principal eigenvector of $\mathbf{A}^T\mathbf{A}$. Authorities are the most central vertices on a network, which are connected to many different hubs; and
- coreness: the highest order k-core containing the vertex u, where a k-core is a maximal subgraph in which each vertex has at least degree k.

These measures are computed using igraph [31], an open source and free collection of network analysis tools. Additionally, a low-dimension embedding of the GCN is computed to capture gene expression patterns. An embedding is a continuous representation of nodes into a low-dimensional space that captures node similarity and the network structure. The goal is for properties in the embedded representation to approximate properties in the original network [26]. In other words, embeddings are vector representations that capture characteristics of the nodes by using less data, thus being more tractable and accessible to machine learning. The dimension of the embedding for each sub-hierarchy corresponds to the number of biological processes in it (i.e., $|A|$).

5.3.2 Comparison Methods

In this work, we employ 6 methods for comparison. More specifically, we present a comparison method from the literature, followed by 2 baseline methods and 3 variants of our proposed method.

Despite providing insights on how prediction probabilities may be used, Nakano et al. [52] did not explicitly propose a method to identify missing annotations on HMC problems, despite being the most recent work in this context. These authors have, however, showed that random forests have superior predictive performance than other HMC methods. For this reason, we used a global HMC classifier based on random forests of decision trees as the baseline classifier for all methods (including the proposed one), where all functions of the sub-hierarchy are considered at once. The parameter values used for random forest classifiers are: 200 estimators (*n_estimators*) and minimum number of samples of 5 (*min_samples_split*), whereas the number of folds used is $k = 5$.

The work of [93] could be employed as a comparison. Nonetheless its authors addressed the problem of identify missing functional annotations of proteins using a probabilistic model based on similarities between functions and guilt by association. Their method associates a protein and a function based on the correlation of functions in the hierarchy and the information of related proteins in the protein-protein interaction network (i.e., guilt by association). Thus, it can not be seen or extended as a HMC method.

Apart from that, the literature presents several works that are capable of identifying missing or wrong annotations in binary classification [9, 79, 72, 74, 94]. Unfortunately, these works require adaptation since they were evaluated in the context of binary classification. Among these, the recently proposed method presented by [72] seems to be the the most related to REASSIGN, since it relies specifically on random forests, and it can be straightforwardly adapted to HMC. In this work, this method is referred to as *Noise detect*.

A detailed description of each method included in the experiments is presented next:

- **Noise detect.** Sabzevari et al. [72] recently proposed a method that employs an ensemble of decision trees to identify mislabeled instances in binary classification. More specifically, an instance is marked as noise if its misclassification rate is higher than a threshold, where the misclassification rate is defined as the proportion of predictors in which the instance is misclassified based on the number of predictors where the instance is out-of-bag. In this work, we adapt this method by selecting the top N annotations with higher misclassification rate.
- **No aggr.** A variant of our proposed method that does not consider the hierarchy of classes. That is, no aggregation method is employed, and the predictions are employed directly from the classifier. This variant is included to highlight the importance of the hierarchical relationships.
- **Random.** A baseline random method that selects annotations without any criterion and complies with the hierarchy constraint. This method is included as reference point to validate the use of machine learning methods.
- **REASSIGN (min).** A variant of our proposed method that aggregates probabilities by using the minimum probability along the path considered.
- **REASSIGN (sum).** A variant of our proposed method that aggregates probabilities by using the sum of the probabilities along the path considered.

- **REASSIGN (average).** A variant of our proposed method that aggregates probabilities by using the average of the probabilities along the path considered.

Since *Noise detect* was built for binary classification problems, a default threshold of 0.5 is used to define the labels. However, in HMC, the predicted probabilities vary according to their level in the hierarchy, the deeper a class is, the lowest the probabilities get. For this reason, we adapt this method by using a different threshold for each function according to its level in the sub-hierarchy. The threshold is set as $t = 0.5 \cdot 0.75^{l-1}$ (similar to weights proposed by [85]), where l is the level of the function. For instance, a threshold $t = 0.5$ is used for functions in the first level and $t = 0.88989$ is used for functions in the seventh level.

5.3.3 Evaluation Measures

The performance evaluation of the methods is based on the true positive and false positive measures, because the aim of this work is to detect missing annotations (i.e., identifying 0s that became 1s) and there are two versions of the datasets available that allows us to verify the predictions. That is, the evaluation is focused on annotations that are not detected, but might show up in the future and can be verified using the newer version of the datasets.

We use precision@N as the first evaluation metric. Given N, the number of annotations selected in a dataset (derived from the number of paths to be selected N_p), the precision for the selected annotations is computed as

$$\text{precision@}N = \frac{tp_N}{tp_N + fp_N}.$$

Moreover, different values of N are used to avoid bias in the prediction and evaluation, hence the number of selected paths can be set according to the resources available for *in vivo* biological experimentation.

In addition, we use the area under the curve generated between the different values of N (in the x-axis) and the precision@N (in the y-axis) as second evaluation metric, denoted as, AUP@NC. This metric aims to analyze the performance of the methods regardless of the value of N and to remove subjectiveness caused by using only one value for each method and dataset. The area under the precision@N curve is defined as

$$\text{AUP@NC} = \sum_{N_i} \text{area}(N_i, \text{precision@}N_i)$$

where N_i represents the different values of N.

The number of paths to be selected N_p by the proposed method is defined as a proportion of the number of 1s in the older version of the dataset, i.e., the total number of associations between genes and functions occurring in Y. In particular, we set $n \in [0, 1]$ as the proportion of occurring annotations and $N_p = \sum Y \cdot n$. The number of annotations within the top N_p paths after removing duplicates is denoted as N. That is, N and n have a positive proportional relationship, i.e., lower values of n lead to lower values of N and higher values of n lead to higher values of N. We use 20 different values of n, from $n = 0.01$ incremented in steps of 0.01 up to $n = 0.2$. Note that the values of N_p and N are different for each dataset, whereas the values of n are the same for all datasets.

Furthermore, to provide statistical evidence, the Friedman-Nemenyi test is used. At first, the Friedman test verifies if any of the compared methods performs statistically significantly different from others. Next, the Nemenyi test ranks the methods where methods with superior results are ranked in higher positions. Graphically, methods connected by a horizontal bar, of length less or equal to a critical distance, are not statistically significantly different. As input to this test, we employ the area under the precision@N curve.

5.4 Results and Discussion

In this section, the experiments and results are presented. At first, we analyze the predictive performance of the proposed and comparison methods using the precision@N measure, followed by a discussion on how our method differs from its comparison counterparts through the area under the precision@N curve. Lastly, we analyze the deepness of the annotations predicted by the proposed method.

5.4.1 Comparison Between All Methods of the Precision@N

Figure 5.2 illustrates the predictive performance of all sub-hierarchies measured with precision@N. Sub-hierarchies are shown in the same order as in Table 5.1, from smallest (top left) to largest (bottom right). The predictive performance of the methods is measured based on a selection of the same number of annotations for each dataset.

The variants of our method are mostly associated with superior performance. More specifically, we highlight the results of *REASSIGN (avg)* on the datasets

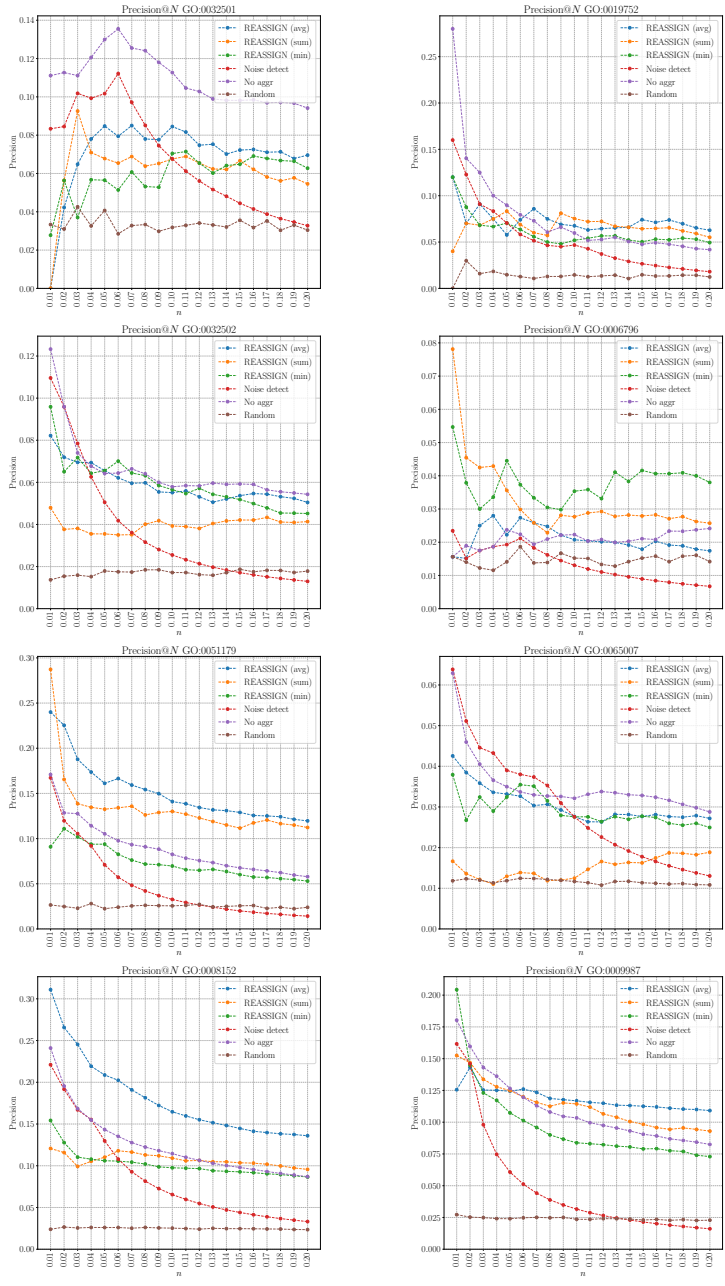


Figure 5.2: Precision@N of all sub-hierarchies for 20 different values of n (N is derived from n) considering all evaluated methods.

GO:0051179, GO:0008152, and GO:0009987, where its curve is majoritarily above the others considering most values of n. A noteworthy advantage in performance is seen in the GO:0008152 dataset where *REASSIGN (avg)* achieves 5% higher precision than the closest competitor, *No aggr*, for all values of n.

The other variants of our method, *REASSIGN (min)* and *REASSIGN (sum)*, also provided superior results in three cases: GO:0006796, GO:0051179 and GO:0009987. Precisely, in the GO:0006796 dataset, both methods are remarkably preferable over the competitors due to superior performance in all values of n. Likewise, these variants also yielded the best results in the GO:0051179 (*REASSIGN (sum)*) and GO:0009987 (*REASSIGN (min)*) datasets when $n = 0.01$ is considered. The performance of these methods, associated with the performance of *REASSIGN (avg)*, endorses the necessity of incorporating the hierarchical relationships among the classes.

Complementary, *No aggr*, the variant that overlooks the hierarchy, managed to have the upperhand only in a few cases, such as in the datasets GO:0032501, GO:0032502, GO:0019752, and GO:0065007, especially when the value of n is small. We suspect that this is related to size of the sub-hierarchies. As presented in Table 5.1, the datasets GO:0032501, GO:0032502, and GO:0019752 have relatively smaller sub-hierarchies, thus incorporating the hierarchy does not necessarily lead to better results. The behaviour observed in GO:0065007 seems peculiar since it was the only dataset where the method *Noise detect* yields the highest performance, followed by *No aggr*, especially when smaller values of n are employed. However, it is worth mentioning that this difference is marginal, as their performance ranges from approximately 6.5% to 4%, when compared to our proposed method using the average as the aggregation method.

In contrast, *Random* provides very underwhelming results in most of the experiments where its performance is barely superior to 0, making it negligible. Curiously, in some very specific scenarios, *Random* was capable of overcoming the *Noise detect* method, as seen in the GO:0032502, GO:0006796, GO:0051179, and GO:0009987, when larger values of n are analyzed. We attribute this counter-intuitive finding to the performance of *Noise detect* as a whole. As shown in Figure 5.2, there is a perceivable deterioration in performance as the value of n increases, whereas a less prominent worsening was detected in the other methods.

Such deterioration is expected, since smaller values of n lead directly to smaller subsets of annotations to be evaluated, artificially increasing the value of the precision. Hence, selecting a smaller subset of annotations does provide relatively better results.

Despite yielding superior results when compared to the literature, our method

shows that detecting missing annotations is a rather challenging task, as seen in datasets such as GO:0065007 and GO:0006796 where the best methods merely achieved 6% and 8% precision, respectively. We suspect that the low availability of annotations, especially in deeper levels of the hierarchy, plays a significant role in this matter.

5.4.2 Analysis of the Area Under the precision@N Curve

Table 5.2 shows the area under the precision@N curve for all methods and sub-hierarchies. For each sub-hierarchy the best method is highlighted with boldface. In all cases, variations of our method always provide the highest AUP@NC. More specifically, *REASSIGN(avg)* provides the best performance in 4 datasets, followed by *No aggr* and *REASSIGN(min)* on 3 and 1, respectively. The competitor method *Noise detect* did not manage to have the best performance in any dataset.

Among the 3 cases where *No aggr* had the upperhand, its superiority was more pronounced only in one dataset, GO:0032501, where it yielded 0.0208 AUP@NC and the second best method, *REASSIGN(avg)*, provided only 0.0137. In the other two cases, GO:0032502 and GO:0065007, *No aggr* was only marginally better.

A different behaviour is seen in *REASSIGN(avg)* where its performance was considerably better in 3 out of the 4 cases. Precisely, in GO:0051179, GO:0008152, and GO:0009987, *REASSIGN(avg)* provided visibly superior results in comparison to the runner-up method. When compared solely against *Noise detect*, our most prominent variant, *REASSIGN(avg)* is consistently superior.

Root	*REASSIGN (avg)*	*REASSIGN (sum)*	*REASSIGN (min)*	*No aggr*	*Random*	*Noise detect*
GO:0032501	0.0137	0.0121	0.0114	**0.0208**	0.0063	0.0129
GO:0019752	**0.0141**	0.0128	0.0113	0.0139	0.0027	0.0096
GO:0032502	0.0111	0.0075	0.0111	**0.0122**	0.0033	0.0067
GO:0006796	0.0040	0.0060	**0.0071**	0.0040	0.0028	0.0025
GO:0051179	**0.0286**	0.0249	0.0139	0.0166	0.0047	0.0088
GO:0065007	0.0057	0.0028	0.0055	**0.0066**	0.0022	0.0055
GO:0008152	**0.0339**	0.0203	0.0191	0.0233	0.0048	0.0160
GO:0009987	**0.0225**	0.0213	0.0180	0.0207	0.0046	0.0087

Table 5.2: Area under the curve generated between the different values of n (in the x-axis) and the precision@N (in the y-axis), i.e., AUP@NC. The proposed method (*REASSIGN (avg)*) outperforms *No aggr* and *Noise detect* methods in 5 and all sub-hierarchies, respectively.

These results are further attested in the Friedman-Nemenyi presented in Figure 5.3. It can be seen that there is a significant difference between *REASSIGN (avg)* and the competitor method *Noise detect*, nonetheless no significant difference is observed among the variant of our proposal.

Precisely, *REASSIGN (avg)* is ranked in the first position followed by *No aggr*, *REASSIGN (sum)* and *REASSIGN (min)*. The competitor method *Noise detect*, the variant *REASSIGN (min)* and Random are not statistically significantly different.

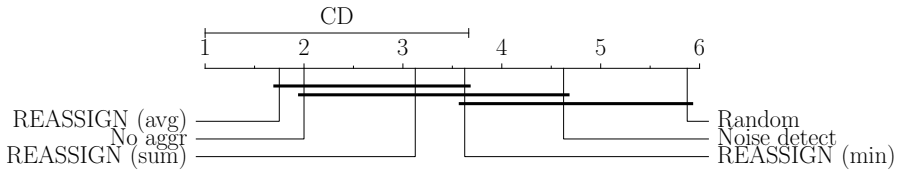


Figure 5.3: Friedman–Nemenyi test evaluating the area under the precision@N curve, i.e., the curve generated between the different values of n (in the x-axis) and the precision@N (in the y-axis). Methods connected by a horizontal bar, of length less or equal to a critical distance, are not statistically significantly different. The proposed method is significantly different from the noise detection and the random methods.

5.4.3 Comparison of True Positives Through the GO Hierarchy Levels

To further evaluate the two best methods, *REASSIGN (avg)* and *No aggr*, we have investigated their performance per level. More specifically, we analyze the precision per level on two datasets: GO:0032501 (Table 5.3) and GO:0009987 (Table 5.4). These were selected due to their difference in hierarchy size (4 and 11 levels, respectively) and in performance.

Table 5.3 suggests that *No aggr* focuses substantially on annotations present in the first level of the hierarchy where it correctly predicts 26 annotations, whereas *REASSIGN (avg)* managed to obtain only 10. In the second and third level, however, *REASSIGN (avg)* was capable of accurately identifying more missing annotations. We believe that the aggregation function is responsible for this difference, as classes located in deeper levels are often associated to very low prediction probabilities, making their selection very unlikely by *No aggr*.

Level	# 0 → 1	*REASSIGN (avg)*	*No aggr*
1	97	10/77 (**12.99%**)	26/185 (12.05%)
2	51	6/87 (6.90%)	3/24 (**12.50%**)
3	33	1/50 (**0.02%**)	0/5 (0%)
4	3	0/0 (0%)	0/0 (0%)
Total	184	17/214 (7.94%)	**29/214 (13.55%)**

Table 5.3: Number of predicted annotations per level for the proposed *REASSIGN (avg)* and *No aggr* methods for the sub-hierarchy GO:0032501. The second column shows the number of missing annotations (0s that became 1s) per level, followed by the number of true positives, the number of predicted annotations and the precision per level for both methods. The last row shows the total number of missing annotations in the sub-hierarchy and the total number of true positives predicted by each method.

Level	# 0 → 1	*REASSIGN (avg)*	*No aggr*
1	2334	581/2871 (**20.24%**)	261/1913 (13.64%)
2	3976	502/3121 (**16.08%**)	938/7458 (12.58%)
3	3833	302/2748 (**10.99%**)	221/2025 (10.91%)
4	2402	202/2197 (**9.19%**)	116/1268 (9.15%)
5	1729	87/1625 (5.35%)	76/793 (**9.58%)**
6	1109	20/699 (2.86%)	6/52 (**11.54%**)
7	721	12/271 (4.43%)	5/46 (**10.87%**)
8	304	6/34 (**17.65%**)	1/16 (6.25%)
9	85	0/19 (0%)	0/14 (0%)
10	24	0/0 (0%)	0/0 (0%)
11	3	0/0 (0%)	0/0 (0%)
Total	16,520	**1712/13,585 (12.60%)**	1624/13,585 (11.95%)

Table 5.4: Number of predicted annotations per level for the proposed *REASSIGN (avg)* and *No aggr* methods for the sub-hierarchy GO:0009987. The second column shows the number of missing annotations (0s that became 1s) per level, followed by the number of true positives, the number of predicted annotations and the precision per level for both methods. The last row shows the total number of missing annotations in the sub-hierarchy and the total number of true positives predicted by each method.

A slightly different behaviour in performance is noticed at Table 5.4 where *REASSIGN (avg)* had the upper hand with 1712 over 1624 provided by *No aggr*. Despite of that, a similar tendency in the distribution of the annotations was noticed: the missing annotations identified by *No aggr* are mostly located in the shallow levels of the sub-hierarchy, specially on the second level in this case, whereas *REASSIGN (avg)* seeks deeper annotations. Nevertheless, *REASSIGN (avg)* detects in average the double of missing annotations than *No aggr* in all levels, except for the second one.

Hence, we may assume that *No aggr* is more likely to provide desirable results when sub-hierarchies with fewer levels (in this case, 4) are considered. As opposed to that, employing the average as the aggregation function is preferred when deeper, and possibly more complex, sub-hierarchies are investigated. However, it is worth mentioning that detecting missing annotations in deeper levels still remains a challenge since no method was able to detect them in the deepest level of both hierarchies.

5.5 Concluding Remarks

In this chapter, we have presented a novel method to detect missing annotations in HMC datasets. More specifically, we proposed a method, with 3 possible variants, that exploits the class hierarchy by computing aggregated probabilities (e.g., average, sum and minimum) of the paths of classes from the leaves to the root for each instance. Furthermore, the proposed method is presented in the context of predicting missing gene function annotations, where these aggregated probabilities are further used to select a set of annotations to be verified through *in vivo* experiments.

Experiments on *Oriza sativa Japonica* showcased that our proposed method yields superior results when compared to competitor methods from the literature. Furthermore, we could also identify that incorporating the hierarchy of classes into the method often improves the results. Precisely, averaging the probabilities leads to the identification of missing annotations in deeper levels of the hierarchy, which is often regarded as more informative.

Even though the proposed method is focused in detecting missing annotations in the datasets, detecting annotations that were removed instead of added, may be of interest to identify wrong associations and to improve the quality of the datasets.

Availability of Data and Materials The datasets analyzed for the current study are publicly available from different sources. Gene co-expression data of rice is available at ATTED-II (version r17c) [54], functional data of rice is available at DAVID Bioinformatics Resources [28], and hierarchical data of Gene Ontology terms are available at GOATOOLS Python library [36].

The data collected, cleaned, and processed from the above sources as used in the case study can be requested to the authors. The proposed method was implemented in Python 3 and is publicly available in [67].

Chapter 6

Domain Adaptation and Hierarchical Multi-Label Classification for Gene Function Prediction

This chapter has been published as a preprint:
Romero, M., Nakano, F.K., Finke, J., Rocha, C., and Vens C. Domain adaptation and hierarchical multi-label classification for gene function prediction. To be submitted for publication (2022).

Chapter Summary

Predicting gene functions remains an open challenge partly because there are organisms without enough functional information available to train predictive models. This chapter introduces a method that combines hierarchical multi-label classification (HMC) and domain adaptation to address the problem of predicting gene functions by taking advantage of the functional information available for additional organisms. The method builds independent HMC classifiers for the additional and the target organism selected, and combines their predicted probabilities to get a final prediction for the target organism complying with the hierarchy constraint. The proposed method is applied on rice, as target organism, using arabidopsis, maize, and soybean as source organisms. The results show that combining the functional information of multiple organisms leads to improvements in the predictive performance of gene function prediction. The results also show that the method can handle additional organisms that are not related to the target organism.

Connections to Previous Chapter

The methods presented in this dissertation up to this chapter use the (co-expression and functional) information of a single organism, either rice or maize, to train a HMC classifier and predict gene functions. Although the method proposed in Chapter 4 combines the information of the GCN and affinity graph to train a HMC classifier, both networks are created with information available of the same organism. In this chapter, domain adaptation and HMC are combined into a method that uses the information of multiple organisms to predict gene functions. The method takes advantage of the information available for other organisms to improve the predictive performance in an organism with a possible shortage of functional information. In addition, we use (i) a global HMC classifier based on the results from Chapter 4; and (ii) the same process to build datasets for GO sub-hierarchies based on structural properties of GCNs presented in Chapter 5.

In contrast to traditional machine learning where a single domain is used for training and prediction, transfer learning use the knowledge from different source domains to improve the predictive performance of a target domain [57, 87, 39, 98]. Transfer learning assumes that the target and source domain are somehow related and the knowledge transferred improves predictive performance. Domain adaptation is a transfer learning approach where the target and source domains share the same feature and label spaces.

Relations between genes of different organisms can represent a source of additional information for the gene function prediction task. Genes can have homologous in other organisms that evolved from a single ancestral gene, called orthologs, which typically perform similar biological functions [38]. Transfer learning can be used to identify gene functions by taking advantage of the relations between genes of different organisms when their functional information is available.

This chapter introduces hierarchical multi-label Classification and dOmain adaptatioN for gene fuNction prEdiCTion (CONNECT), a method that combines hierarchical multi-label classification (HMC) and domain adaptation to address the problem of predicting gene functions. This method takes advantage of the functional information available for other organisms to improve the predictive performance in organisms whose functional information is scarce or unavailable. The method takes as input the information of several organisms, builds separate HMC classifiers for each one of them, and combines them by selecting per function the organisms whose predictions better approximate the true values of the target organism.

The proposed method is applied on rice, as target organism, using arabidopsis, maize, and soybean as source organisms. The results show that combining the functional information of multiple (related) organisms helps to improve the performance of predicting gene functions. In particular, the proposed method outperforms the comparison method in most datasets. Besides, the results highlight that the proposed method can handle organisms that are not related to the target organism.

The remainder of the chapter is organized as follows. Section 6.1 reviews related work. Section 6.2 introduces CONNECT, our proposed method for predicting gene functions based on HMC and domain adaptation. Section 6.3 describes the datasets and experimental setup for the prediction of gene function in rice, followed by the results and discussion in Section 6.4. Finally, Section 6.5 draws concluding remarks and future research directions.

6.1 Related Work

The gene function prediction problem has been addressed using multiple approaches, including probabilistic models, HMC (mostly based on tree ensembles), and deep learning [30, 76, 93, 4, 52, 95, 96, 65]. However, these approaches are generally based on a single organism (domain) for training and prediction. To the best of our knowledge, the literature does not present a single work for transfer learning (domain adaptation) and HMC for gene function prediction.

Transfer learning has been used to address other biological problems, such as, prediction of phenome-genome associations based on protein-protein interaction networks and GO [58]; protein function prediction based on sequences of several protein structural domains [90]; prediction of the CYP2D6 haplotype enzyme based on genome sequences [48]; prediction of Lysine propionylation based on protein sequences [43]; identification of heat shock proteins based on protein sequences [50]; prediction of transcription factor binding based on information about transcription factor binding events [53]; gene prioritization for cancer diagnosis based on expression and pathogenic data of genes [86]; and prediction of functional non-coding variants based on genomic sequences [11].

Other studies focused in multi-label problems using transfer learning approaches. However these problems do not present hierarchical dependencies between classes. These studies include image classification for early diagnosis of breast cancer using a pre-trained convolutional neural network [16]; image classification to identify movie genre using pre-trained convolutional neural network [40]; classification of commit messages on code repositories using neural networks [75]; sentiment analysis using pre-trained neural network [81]; image classification to detect ophthalmological diseases using pre-trained convolutional neural networks [25]; and image classification in autonomous driving using neural networks [44]. Most of these studies are based on neural networks, being convolutional neural networks the commonest approach. Pre-trained neural networks are commonly used to improve the prediction performance in multiple problems. However, these networks are mostly available for image classification tasks. In general, these works do not take into account the hierarchical dependencies between classes, as they focus on binary, multi-class, or multi-label problems instead. Therefore, such studies cannot be compared directly to the proposed method.

6.2 Hierarchical Multi-label Classification and Domain Adaptation

In this section, we present a method for gene function prediction based on HMC and domain adaptation.

6.2.1 Problem Definition

In the gene function prediction problem, the feature space $\mathcal{X}$ represents information or properties of the genes V, and the label space $\mathcal{Y}$ represents the biological functions A, their hierarchy $(A, \leq_h)$, and their associations with the genes V (through an annotation function ϕ). Note that the feature space of the genes can be common for different organisms, i.e., the same information or properties can be extracted for genes regardless of the organism under study. Note that, biological functions (as well as their hierarchy) are common for all organisms, as defined by [23]. In contrast, the probability p of the functional annotations is expected to be different for each organism although they can be similar for related organisms according, e.g., to the phylogenetic tree [62]. That is, the feature and label space can be equal for different organisms, but their probability density function may differ.

For these reasons, we extend the definition of the gene function prediction problem by including additional organisms for domain adaptation, as follows:

Definition 10. *Let $\mathcal{T}$ be the organism under study (target) with genes $V_\mathcal{T}$, a set of biological functions A organized into a hierarchy $(A, \leq_h)$, and an annotation function $\phi_\mathcal{T} : V_\mathcal{T} \to 2^A$ (a function describing known annotations); and $\mathcal{S}_1, \ldots, \mathcal{S}_m$ be m additional (source) organisms with genes $V_{\mathcal{S}_1}, \ldots, V_{\mathcal{S}_m}$ and annotation functions $\phi_{\mathcal{S}_1}, \ldots, \phi_{\mathcal{S}_m}$, respectively, which share the same set of biological functions A (and their hierarchy) with $\mathcal{T}$.*

The objective is to obtain a function $\psi_\mathcal{T} : V_\mathcal{T} \to 2^A$ that augments $\phi_\mathcal{T}$ and complies with the hierarchy constraint, using the information available for the additional organisms $\mathcal{S}_1, \ldots, \mathcal{S}_m$.

Organism-specific components are marked with the subscripts $\mathcal{T}$ and $\mathcal{S}_1, \ldots, \mathcal{S}_m$. Note that although the genes of the organism could be different, their feature spaces can be equal, given that the feature space refers to the information or properties extracted from the genes.

6.2.2 CONNECT

We introduce hierarchical multi-label Classification and dOmain adaptatioN for gene fuNction prEdiCTion (CONNECT), a method that combines HMC and domain adaptation to address the problem of predicting gene functions. Given a target organism, denoted as $\mathcal{T}$, with a (possible) shortage of functional information available, the proposed method aims to use the functional information available from m additional (source) organisms $\mathcal{S}_1, \ldots, \mathcal{S}_m$ to improve the predictive performance on $\mathcal{T}$.

For each organism $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$ the input of the method are a dataset $X_\mathcal{O}$ comprising $|V_\mathcal{O}|$ genes (instances) and features, and an annotation function $\phi_\mathcal{O}$ represented as a label matrix $Y_\mathcal{O}$ with an assignment of genes in $V_\mathcal{O}$ to a subset of biological functions from A (i.e., $Y_\mathcal{O} : V_\mathcal{O} \times A \rightarrow \{0, 1\}$). Note that the set of biological functions A, their hierarchy $(A, \leq_h)$, and the number of features are the same for each $\mathcal{O}$. The output of the method is a function $\psi_\mathcal{T}$ that augments $\phi_\mathcal{T}$ and complies with the hierarchy constraint, i.e., an extended annotation function with new associations between genes in $V_\mathcal{T}$ and biological functions in A. The annotation function $\psi_\mathcal{T}$ is represented as a label matrix denoted as $Y^E_\mathcal{T}$.

The proposed method is based on HMC classifiers that are trained using the information of each organism independently. Therefore, the first step is to build m independent HMC classifiers $f_\mathcal{O}$ using all the information of each source organism $\mathcal{O} \in \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$ for training. Any local or global HMC classifier can be used (e.g., tree ensembles or neural networks), providing that the hierarchy constraint is satisfied.

The classifiers, trained independently for each organism, are combined to compute the annotation function $\psi_\mathcal{T}$ using the probabilities predicted for the target organism $\mathcal{T}$. The combination of the predicted probabilities can be achieved in different ways, such as finding a linear combination. The proposed method focuses on finding the organism that better approximates $\mathcal{T}$ per label: a weight is computed per label and organism to measure the percentage of genes (instances) in which the predicted probabilities are closer to the true values of $\mathcal{T}$.

An inner cross validation (CV) strategy is used to compute the weights per organism and label while preventing overfitting. The organisms that better approximate $\mathcal{T}$ are selected based on these weights. In addition, a HMC classifier $f_\mathcal{T}$ is built using the information of $\mathcal{T}$ over the inner CV. For $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$, predicted probabilities are denoted as $Y'_{\mathcal{T},\mathcal{O}}$ where the first subscript means that the prediction is done on $\mathcal{T}$ and the second subscript means that the classifier was trained using information from $\mathcal{O}$. Based on $Y'_{\mathcal{T},\mathcal{O}}$,

the weight for each function $a \in A$ and each organisms $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$, denoted as $w_{a,\mathcal{O}}$, represents the percentage of genes where $Y'_{\mathcal{T},\mathcal{O}}$ is the closest prediction to the true values $Y_{\mathcal{T}}$ for a.

The average weights within the inner CV, denoted as $\overline{w_{a,\mathcal{O}}}$, are used to identify the organism that better approximates to the true values for each function $a \in A$. The output prediction for each function $a \in A$ is computed using the selected organism and is denoted as $Y^E_{a,\mathcal{T}}$. That is,

$$Y^E_{a,\mathcal{T}} = Y'_{a,\mathcal{T},\mathcal{M}} \text{ s.t. } \mathcal{M} = \arg\max_{\mathcal{O}} \overline{w_{a,\mathcal{O}}}$$

where $Y'_{a,\mathcal{T},\mathcal{M}}$ represents the prediction for the function a for $\mathcal{T}$ using the trained model $f_{\mathcal{M}}$ and $\mathcal{M}$ is the organism with higher average weight for a. Note that $Y^E_{\mathcal{T}}$ results from the union of all $Y^E_{a,\mathcal{T}}$ for $a \in A$.

Algorithm 6.2.1: Hierarchical multi-label classification and domain adaptation to predict gene functions (CONNECT)

input:
$\mathcal{T}$: target organism
$\mathcal{S}_1, \ldots, \mathcal{S}_m$: additional (source) organisms
$X_{\mathcal{O}}$ for each $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$: datasets
$(A, \leq_h)$: function hierarchy
$Y_{\mathcal{O}}$ for each $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$: annotation functions
output:
$Y^E_{\mathcal{T}}$: label matrix augmenting $Y_{\mathcal{T}}$

$\forall \mathcal{O} \in \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$ train a HMC classifier using $X_{\mathcal{O}}$ and $Y_{\mathcal{O}}$ ↪ denoted as $f_{\mathcal{O}}$
foreach fold k_i in inner CV:
train a HMC classifier using $X_{\mathcal{T}}$ and $Y_{\mathcal{T}}$ denoted as $f_{\mathcal{T}}$ on ↪ fold k_i
$\forall \mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$ compute $Y'_{\mathcal{T},\mathcal{O}}$ using $f_{\mathcal{O}}$ s.t. $Y'_{\mathcal{T},\mathcal{O}}$ complies ↪ with the hierarchy constraint
compute $w_{a,\mathcal{O}}$ based on $Y'_{\mathcal{T},\mathcal{O}}$ for each organism ↪ $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$ and function $a \in A$ on fold k_i
end
compute the average weight $\overline{w_{a,\mathcal{O}}}$ for each $\mathcal{O} \in \{\mathcal{T}, \mathcal{S}_1, \ldots, \mathcal{S}_m\}$ ↪ and $a \in A$ over all folds
foreach $a \in A$
calculate $\mathcal{M} = \arg\max_{\mathcal{O}} \overline{w_{a,\mathcal{O}}}$
compute $Y'_{a,\mathcal{T},\mathcal{M}}$ for function a using organism $\mathcal{M}$
$Y^E_{a,\mathcal{T}} = Y'_{a,\mathcal{T},\mathcal{M}}$
end
compute $Y^E_{\mathcal{T}} = \bigcup_{a \in A} Y^E_{a,\mathcal{T}}$
verify $Y^E_{\mathcal{T}}$ to satisfy the hierarchy constraint

24 **return** $Y_{\mathcal{T}}^{E}$

Since the the predictions for each function can be computed using different organisms, it is necessary to verify that the predicted probabilities for $\mathcal{T}$ (i.e., $Y_{\mathcal{T}}^{E}$) comply with the hierarchy constraint. To that aim, a top-down traversal of the hierarchy is performed such that for every $a \leq_h b$ where $a, b \in A$ (i.e., b is ancestor of a) must hold $Y_{a,\mathcal{T}}^{E} \leq Y_{b,\mathcal{T}}^{E}$. If $Y_{a,\mathcal{T}}^{E} > Y_{b,\mathcal{T}}^{E}$ for some $a \leq_h b$, then the probabilities predicted for b are assigned to a, i.e., $Y_{a,\mathcal{T}}^{E} = Y_{b,\mathcal{T}}^{E}$. That is, the predicted probabilities of a function are lower or equal than predicted probabilities of their ancestors. A detailed description of the proposed method is presented in Algorithm 6.1.

6.3 Experimental Setup

In this section, a detailed description of the databases, comparison methods, and evaluation measures is presented.

6.3.1 Datasets

We use rice (*Oryza sativa Japonica*) as our target organism. Despite being one of three major staple foods in the world and the fourth most consumed staple food in Latin America (supplying more calories to the diet than wheat, maize, cassava, or potatoes), its functional information available online is scarce mainly due to its commercial and industrial importance [80, 8, 41].

As source organisms, we have selected arabidopsis (*Arabidopsis thaliana*), maize (*Zea mays*), and soybean (*Glycine max*). On the one hand, maize and soybean are selected because they are closely related to rice in the phylogenetic tree. More specifically, the former belongs to the same family of plants as rice and the latter is the most consumed bean in the world, with a large amount of data available online. On the other hand, arabidopsis is selected because it is one of the most studied model plants with a large amount of data available online, even though it is not closely related to rice [84].

Datasets for each organisms are built using functional information and gene co-expression networks (GCNs) [65]. The functional information depicts associations between genes and functions previously identified through *in vivo* or *in silico* experiments. The functional information of the four organisms is imported from DAVID Bioinformatics Resources [28]. GCNs are built using the co-expression information imported from the ATTED-II database [54] (version

r21c). GCNs are represented as undirected and weighted graphs, denoted as $G = (V, E, w)$, where vertices V represent genes and edges E the level of co-expression between genes [2]. The co-expression level is measured through the weight function w where lowest values are assigned to the strongest connections.

Organism	Genes $\|V\|$	Functions $\|A\|$	Annotations
Rice (Target)	22,698	1,901	288,470
Arabidopsis	18,957	5,694	648,553
Maize	26,131	2,773	180,716
Soybean	33,331	4,163	562,895

Table 6.1: Functional and co-expression (GCN) data imported for target and source organisms. The second column shows the number of genes (vertices) in the GCN, followed by the total number of biological functions and the number of gene-function associations available for each organism.

Table 6.1 summarizes the functional and co-expression data for each organism. The second column shows the number of genes (vertices) in the GCN, followed by the total number of biological functions and the number of gene-function associations available for each organism. Note that rice is the organism associated to less biological functions, whereas arabidopsis and soybean have almost twice as much functional information as rice and maize. Besides, soybean is the organisms with the largest GCN followed by maize, rice, and arabidopsis.

We use 10 groups (sub-hierarchies) of biological processes from the GO hierarchy. Each sub-hierarchy is represented by its root function, i.e., biological processes of depth 1 in the GO hierarchy. Table 6.2 describes the sub-hierarchies starting by their root function and its description, followed by the number of functions included in the sub-hierarchy and the number of genes annotated (associated) with the root function for each organism. Note that more common biological processes are associated to a larger number of genes and their sub-hierarchy will be bigger (in the number of genes). Additionally, Table 6.3 describes each sub-hierarchy per level, starting with the root function, followed by the number of functions per level and the number of annotations (gene-function associations) per level for each organism (rice, arabidopsis, maize, and soybean). The GO sub-hierarchies are generated using the pre-processing stage of the method presented in Chapter 3.

For each sub-hierarchy and organism, a dataset is built using the same approach presented in Chapter 5, i.e., a dataset consisting of two sets of features: structural properties and node embeddings of the GCN. These features are computed for the genes associated to each sub-hierarchy, as described in Table 6.2.

Root	Description	Functions	# genes rice (target)	# genes arabidopsis	# genes maize	# genes soybean
GO:0040007	growth	9	167	488	41	85
GO:0002376	immune system process	13	714	750	49	181
GO:0044419	biological process involved in interspecies interaction between organisms	19	795	997	55	208
GO:0032501	multicellular organismal process	29	999	2463	242	507
GO:0022414	reproductive process	45	803	1461	152	359
GO:0032502	developmental process	88	1321	2764	253	698
GO:0050896	response to stimulus	160	3813	5249	1038	2851
GO:0051179	localization	163	1828	2308	913	2343
GO:0065007	biological regulation	379	3194	4824	1401	4647
GO:0008152	metabolic process	706	7780	9688	4765	11146

Table 6.2: Sub-hierarchies of biological processes presented in decreasing order according to their size. The root function and its description are presented in the first and second columns, respectively. The third column shows the number of biological processes included in each sub-hierarchy. The following columns show the number of genes associated to each sub-hierarchy per organism. For example, the first (and smaller) sub-hierarchy, whose root function is GO:0040007, comprises 9 biological functions and is associated to 167, 488, 41, and 85 genes of rice, arabidopsis, maize, and soybean, respectively.

6.3.2 Comparison Methods

Ee employ 2 methods for comparison. More specifically, we present a baseline and our proposed method.

The authors of [52] showed that random forests have superior predictive performance than other HMC methods. Hence, we used a global HMC classifier based on random forests as the classifier for the baseline and proposed methods, where all functions of the hierarchy are considered at once. The parameter values used for random forest classifiers are: 200 estimators (*n_estimators*) and minimum number of samples of 5 (*min_samples_split*), whereas the number of folds used are 5 and 3 for the usual and inner CV, respectively.

To the best of our knowledge, the literature does not present a single work for gene function prediction based on transfer learning (domain adaptation) and HMC. The literature presents, however, several works on multi-label classification using transfer learning [58, 90, 16, 40, 75, 81, 25, 44]. Unfortunately, these works cannot be directly compared to our method, since most of them use pre-trained neural networks focused on image classification problems. Among these, the method presented by [90] seems to be the the most related to our proposed method, since it addressed the protein function prediction problem. However this work is based on protein specific data for transfer learning, more specifically, the structural domain of proteins. Their work uses information of

Root	Functions per level	Annotations per level in rice (target)	Annotations per level in arabidopsis	Annotations per level in maize	Annotations per level in soybean
GO:0040007	2/2/2/1/1	242/134/93/23/22	709/397/281/111/88	49/19/14/3/3	90/41/40/13
GO:0002376	2/2/6/2	732/738/869/74	807/887/1056/119	52/52/66/5	188/203/218/26
GO:0044419	3/7/6/2	843/1638/869/74	1228/1787/1056/119	74/112/66/5	239/349/218/26
GO:0032501	11/10/6/1	1450/859/370/24	4516/2344/1376/89	358/151/92/8	721/217/99/5
GO:0022414	12/19/12/1	1218/984/711/119	2383/2726/2068/379	278/209/126/17	715/468/238/40
GO:0032502	11/27/32/14/3	2464/2853/2236/565/58	6281/9766/7759/2160/264	459/668/478/133/16	1150/1497/734/170/9
GO:0050896	9/28/48/45/22/5/2	11044/11851/8527/4081 /1245/93/27	14443/15870/12157/5321 /1733/220/87	1811/1444/969/475/182 /34/8	6038/5710/4210/1966/661 /123/17
GO:0051179	6/15/27/47/35/23 /6/2/1	2968/4080/4799/3294 /1744/797/215/43/20	4190/5990/6845/4767 /2477/1061/359/91/26	1528/2206/2268/1319 /594/232/178/55/17	4981/6834/8618/7118 /3327/1250/402/72/23
GO:0065007	3/21/65/118/93/30 /27/14/6/1	3672/7403/13212/12552 /7939/2458/2130/839 /197/33	5632/11924/21092/20419 /12315/3914/3312/1412 /235/29	1597/2696/5149/4762 /3125/879/843/214 /66/11	5096/9363/17528/18605 /11559/3804/3641/2082 /403/63
GO:0008152	17/50/182/131/92 /88/62/48/29/5/1	33098/36145/51566/14962 /8145/5721/3404/1736 /762/180/25	43322/48565/70104/20284 /11193/7070/4526/2140 /876/212/14	19428/21795/28721/8574 /4682/2848/1971/1062 /406/81/3	50463/55325/79688/22907 /12246/8998/5941/2864 /1258/402/26

Table 6.3: GO sub-hierarchies of biological processes. The root function is presented in the first column, followed by the number of functions per level, e.g., the first sub-hierarchy has 5 levels and there are 2, 2, 2, 1, and 1 functions on each level. The following columns show the number of annotations (gene-function associations) per level for rice, arabidopsis, maize, and soybean, respectively. For example, the first sub-hierarchy has 49, 19, 14, 3, and 3 annotations for maize.

other types of proteins in the same organism to predict their function rather than using the information of other organisms.

A detailed description of each method included in the experiments is presented next:

- **base-HMC**: A baseline HMC method that do not consider the information of additional organisms. That is, domain adaptation is not employed and only the information of the target organism is used for training and prediction. This baseline is included to highlight the importance of combining the information available for related organisms through domain adaptation;
- **CONNECT**: The proposed method, which combines HMC and domain adaptation;

6.3.3 Evaluation Measures

Given that datasets of functional annotations are, in general, highly imbalanced (i.e., the label matrices are sparse), deeper functions in the hierarchy have less annotations and get lower predicted probabilities. That is, deeper functions in the hierarchy are scarce despite being more informative. Hence, we prioritize obtaining true positives over false positive and focus the evaluation on the precision measure.

We construct an overall precision curve using multiple thresholds. That is, the area under the curve generated between the different values of the threshold (in the x-axis) and the precision (in the y-axis) is used as evaluation metric, denoted as, pooled AUPREC. This metric aims to summarize the performance of the methods considering all thresholds employed. The pooled AUPREC is defined as

$$\text{pooled AUPREC} = \sum_{t_i} \text{area}(t_i, \text{precision@}t_i)$$

where t_i represents the different values of the threshold and precision@t_i corresponds to the precision for a given threshold t_i. We use 50 different values of threshold, from $t_i = 0$ incremented in steps of 0.02, up to $t_i = 1$ [85].

6.4 Results and Discussion

In this section, the experiments and results are presented. First, the predictive performance of the proposed and comparison methods, using rice as target

organism, are analyzed based on the pooled AUPREC measure. Then, we discuss how the predictive performance of the proposed and comparison methods vary when other organisms are used as target, e.g., arabidopsis, maize, and soybean.

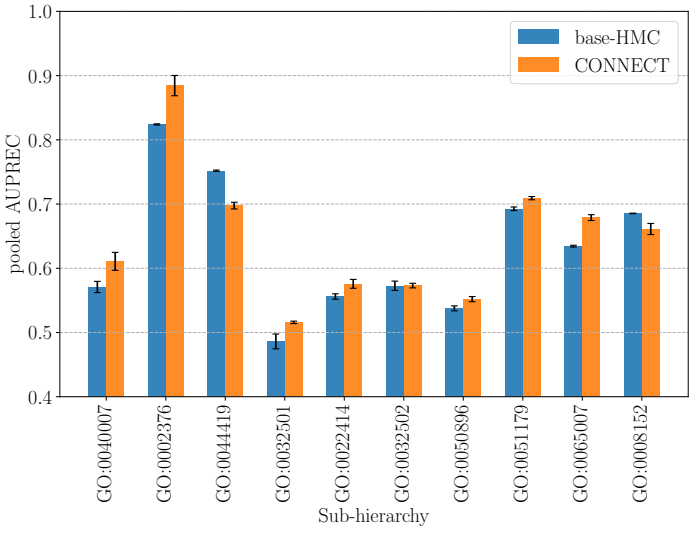


Figure 6.1: Predictive performance of the proposed (CONNECT) and comparison (base-HMC) methods, using rice as target organism, for the 10 GO sub-hierarchies of biological processes based on the pooled AUPREC measure. This Figure is best viewed in colors.

Figure 6.1 illustrates the predictive performance for the GO sub-hierarchies of biological processes presented in Table 6.2 (in the same order). The predictive performance of the proposed (CONNECT) and comparison (base-HMC) methods, for the gene function prediction task using rice as target organism, is measured with the pooled AUPREC.

The average pooled AUPREC and its standard deviation for 10 executions of the experiments is presented in Figure 6.1. The proposed method outperforms base-HMC in seven sub-hierarchies with improvements between 1.4% and 6.1%, considering the average pooled AUPREC. base-HMC outperforms CONNECT in two sub-hierarchies, GO:0044419 and GO:0008152, with 5.4% and 3.4% higher performance, respectively. Besides, in the sub-hierarchy GO:0032502 there is no significant difference (considering the standard deviation) in the performance of both methods. Note that using the additional functional information from arabidopsis, maize, and soybean does not help to improve the predictive

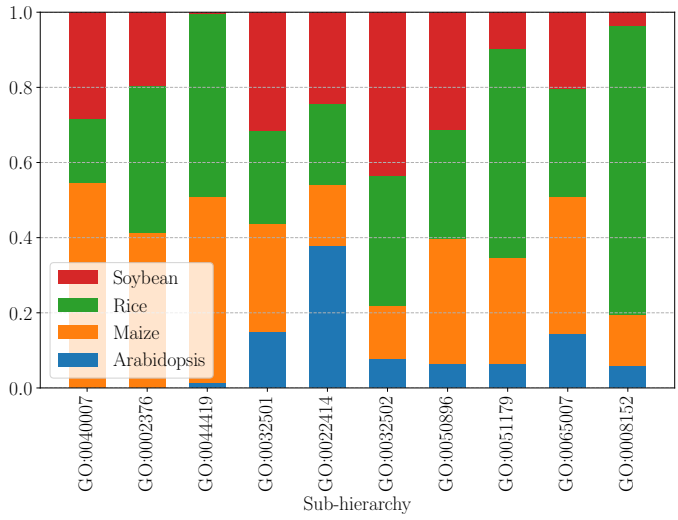


Figure 6.2: Influence of the target and source organisms in the prediction for the proposed method. The influence is computed as the percentage of functions in which each organisms get the higher weight for each sub-hierarchy. This Figure is best viewed in colors.

performance for rice in the sub-hierarchies GO:0044419, GO:0008152, and GO:0032502.

However, no relation seems to exists between the size of the sub-hierarchies and the predictive performance of the proposed method. CONNECT outperforms base-HMC in seven sub-hierarchies of different sizes in terms of the number of functions, the number of levels, and the number of annotations, including the smallest and second largest datasets. Furthermore, CONNECT underperforms base-HMC in three sub-hierarchies: third, sixth and tenth according to their size.

6.4.1 Contribution of the Organisms in the Prediction

We analyze the influence or contribution of the target and source organisms for each sub-hierarchy, where the influence of the organisms in the prediction is measured through the average weights $\overline{w_{a,\mathcal{O}}}$. In particular, the influence is computed as the percentage of functions in which each organisms gets the

higher weight for each sub-hierarchy. Note that the influence measures how often an organism is being selected in a sub-hierarchy.

The influence of each source organism is correlated to its relation with the target organism. If the source organism is not related to the target, then it will not have influence in the prediction (it will not be selected). Besides, if the predictive performance of the proposed method does not improve, then it is expected that the source organisms do not have much influence in the prediction, even if they are related.

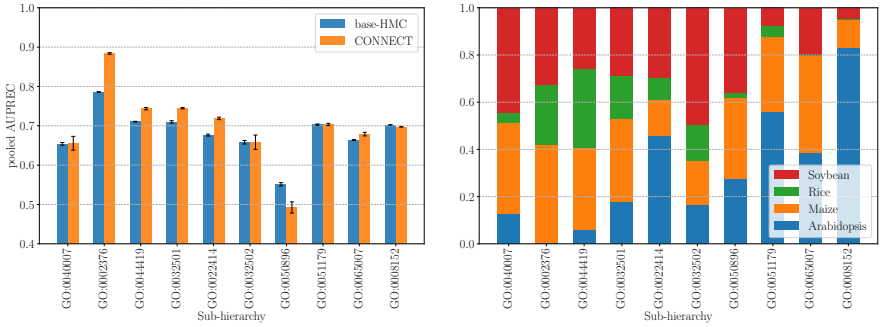
Given that maize and soybean are closely related to rice based on the phylogenetic tree, it is expected that both organisms have a high influence in the prediction (i.e. both organisms are highly selected in all sub-hierarchies). This can be observed in Figure 6.2, which depicts the influence of the target and source organisms for each sub-hierarchy. Improvement in the predictive performance of CONNECT is in general related to the high influence of maize and soybean. In particular, the influence of both organisms ranges from 40% to 83% in the sub-hierarchies GO:0040007, GO:0002376, GO:0032501, GO:0022414, GO:0050896, and GO:0065007.

On the contrary, the influence of arabidopsis is, in general, lower than the other organisms and negligible in most sub-hierarchies (it is lower than 15% in 9 cases). This may be the case because arabidopsis is not closely related to rice. It also shows that the proposed method can handle such "unrelated" organisms that do not contribute to the prediction without affecting the predictive performance.

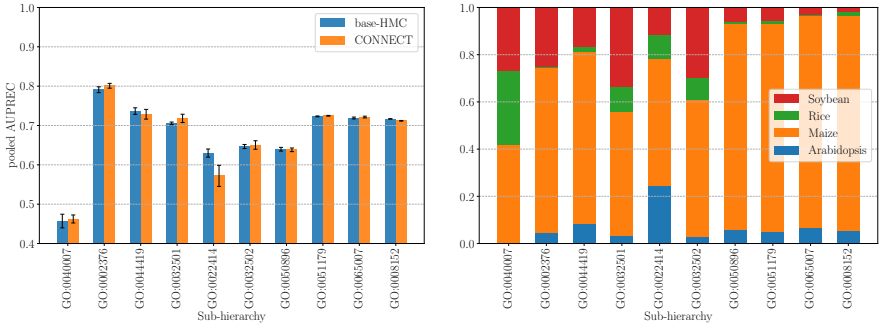
Note that there seems to be a relation between the influence of rice and the predictive performance of CONNECT. On the one hand, the sub-hierarchies where CONNECT outperforms base-HMC show a lower influence of rice w.r.t. to the other organisms. The influence of rice in the sub-hierarchies GO:0050896, GO:0022414, GO:0032501, GO:0040007, GO:0065007, and GO:0002365 is 29%, 21.7%, 24.6%, 16.8%, 28.6%, and 39.2%, respectively. On the other hand, rice has higher influence in the sub-hierarchies where HMC outperforms CONNECT. For instance, the influence of rice in the sub-hierarchies GO:0044419 and GO:0008152 is 48.6% and 76.8%, respectively.

Sub-hierarchies GO:0032502 and GO:0051179 have a different behavior. The former sub-hierarchy has a low influence of rice (34.6%), but there is no a significant difference in the performance of both methods. The influence of rice in the latter is 55.6%, but the performance of CONNECT is higher than the base-HMC.

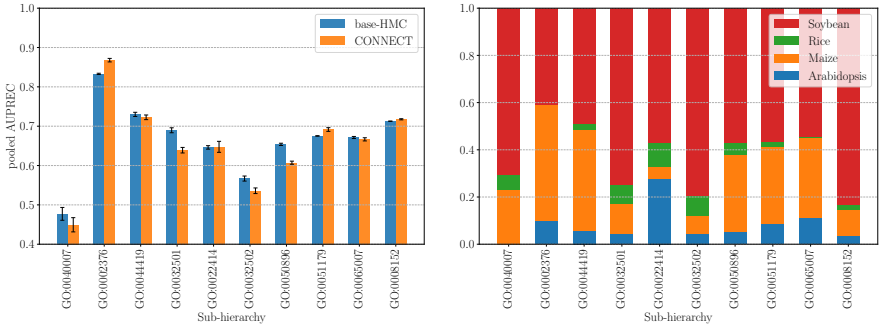
The results show that using the functional information of maize and soybean helped to improve the performance of predicting gene functions in rice for most sub-hierarchies. Nevertheless, it is not clear if the combination of the functional



(a) Target organism: **arabidopsis**



(b) Target organism: **maize**



(c) Target organism: **soybean**

Figure 6.3: Predictive performance of the proposed (CONNECT) and comparison (base-HMC) methods and influence of the target and source organisms in the prediction for CONNECT using **(a)** arabidopsis, **(b)** maize, and **(c)** soybean as target organisms. The performance is measured for the 10 GO sub-hierarchies of biological processes using the pooled AUPREC and the influence is computed as the percentage of functions in which each organisms get the higher weight for each sub-hierarchy. This Figure is best viewed in colors.

information of these four organisms will also improve the predictive performance in arabidopsis, maize, and soybean.

6.4.2 Arabidopsis, Maize and Soybean as Target Organisms

Arabidopsis, maize, and soybean are used independently as target organisms with the same experimental setup as rice. That is, each organism (arabidopsis, maize, and soybean) is used separately as target, while the remaining three organisms including rice are used as source organisms. For example, if arabidopsis is used as target, maize, soybean, and rice are used as source organisms.

Figure 6.3 illustrates the predictive performance and the influence of each organism for the 10 sub-hierarchies when arabidopsis, maize, and soybean are used independently as target organism. The sub-hierarchies are presented in the same order of Table 6.2, i.e., in increasing order by their number of functions. Figure 6.3a summarizes the results when arabidopsis is the target organism. Note that CONNECT outperforms base-HMC in five cases, underperforms in one case, and has similar performance in the remaining four cases. Additionally, the influence of rice seems to be negligible in six cases (less than 9%), but it is higher than 20% in the sub-hierarchies where CONNECT outperforms base-HMC, GO:0002376, GO:0044419, GO:0032501, and GO:0022414. The influence of maize and soybean is in general higher than 15% (except for GO:0008152) and their combined influence varies from 40% to 83% in nine sub-hierarchies.

The results are substantially different when maize and soybean are the target organisms. Figure 6.3b shows the results for maize, in which HMC outperforms CONNECT in two sub-hierarchies and they have similar performance in the remaining cases. This can be explained by the influence of the source organisms in the prediction, where maize has more than 70% influence in six cases and more than 42% in the others. That is, the functional information of the source organisms is barely used. In particular, arabidopsis and rice are not used in most sub-hierarchies, where influence is less than 11% in nine cases. Although soybean has higher influence, it ranges between 15% and 33.6% in six cases, and it is lower than 11% in the remaining ones.

The analysis is similar when soybean is the target organism, as illustrated by Figure 6.3c. In terms of predictive performance, CONNECT outperforms HMC in two sub-hierarchies, underperforms in four, and has similar performance in the remaining four. In other words, the functional information of the source organisms does not help to improve the performance of the proposed method, which coincides with the influence of the source organisms in the prediction. Note that the influence of rice and arabidopsis is lower than 10% in most cases (except for GO:0022414 where arabidopsis has 27.6% of influence), and the

influence of maize ranges from 22% to 48.9% in six cases. Even though the information of maize is used in some sub-hierarchies, the influence of soybean is substantially higher, ranging between 40.8% and 83.2%.

Finally, there are two sub-hierarchies that show a similar pattern in the predictive performance and influence for all targets. First, sub-hierarchy GO:0002376 (immune system process), in which CONNECT outperforms base-HMC for all organisms, specially for arabidopsis and rice, and the influence of maize and soybean seems to dominate arabidopsis and rice. Since GO:0002376 has only 13 functions, the results may be related to the lack of data and highlight the importance of considering additional sources of information in the prediction. Second, sub-hierarchy GO:0008152 (metabolic process), where both methods show a similar performance for all organisms, that is, the additional information is either not being used or not useful for the prediction. In fact, metabolic processes (the largest sub-hierarchy) are specific for each organism; note that the influence of the target organism is higher than 80% in all cases.

6.5 Concluding Remarks

In this chapter, we have presented a novel method that combines HMC and domain adaptation to predict gene functions. The proposed method aims to improve the performance of predicting gene functions in a (target) organism by taking advantage of the functional information available for other (source) organisms, while complying with the hierarchy constrain. Domain adaptation is used for tackling the gene function prediction problem under the hypothesis that the feature and label spaces can be the same for any organism.

Experiments on rice as target organism showcased that the proposed method can incorporate data from other organisms, leading to better performance than the comparison method. In particular, the results show that incorporating the functional information of maize and soybean improve the predictive performance of rice in seven GO sub-hierarchies of biological processes. Furthermore, the results show that the proposed method can handle organisms that are not related to the target, since arabidopsis was barely selected.

Availability of Data and Materials The datasets analyzed for the current study are publicly available from different sources. Gene co-expression data of rice is available at ATTED-II (version r21c) [54], functional data of rice is available at DAVID Bioinformatics Resources [28], and hierarchical data of Gene Ontology terms are available at GOATOOLS Python library [36].

The data collected, cleaned, and processed from the above sources as used in the case study can be requested to the authors. The proposed method was implemented in Python 3 and is publicly available in [66].

Chapter 7

Conclusion and Future Work

This chapter presents the conclusion of this dissertation, followed by a discussion on future research directions.

7.1 Conclusion

The focus of this dissertation has been on developing four machine learning methods for addressing the problem of gene function prediction, where genes can be associated to multiple functions and the functions have a hierarchical structure. These novel machine learning methods focus on: (i) taking into account the hierarchical relations between functions using hierarchical multi-label classification; (ii) extracting new features from GCNs and functional information, using clustering techniques, to enrich the training of predictive models; (iii) detecting missing annotations for genes based on paths of functions in the hierarchy instead of single gene-function pairs; and (iv) integrating multiple organisms in the prediction task by using domain adaptation to take advantage of additional information.

First, the HMC method introduced in Chapter 3 allows us to address classification problems where classes obey a hierarchical structure. Taking into account hierarchical dependencies between functions to produce predictions complying with the hierarchy constrain (or true-path rule) is key for improving the predictive performance of the method. The results show that regardless of the classifier being used (either tree ensembles or graph convolutional networks) computing cumulative probabilities through the hierarchy (from the top to the

bottom) leads to improvements when compared to the state of the art in the literature. The results highlight the importance of using gene co-expression data by means of structural properties of GCNs to train the classifiers. Moreover, selecting appropriate evaluation measures is critical to analyze HMC datasets, where most of the functions are associated to a small set of genes (i.e., datasets are highly imbalanced). For such scenarios, measures based on the number of true positives are more convenient (e.g., precision, recall, or f1-score).

Second, developing new strategies to extract features from GCNs is required to make the most of the information available, which is scarce for some organisms. Chapter 4 presents a method to extract features using a combination of clustering techniques and biological concepts. Extracted features represent groups of genes in the GCN that are likely to be involved in the same biological functions. The results show that the new features improve the performance of local and global HMC methods for the prediction of gene functions. In particular, the global method gets the best performance in all datasets and is more efficient than the local methods, since it considers all functions of the hierarchy at once (i.e., only one model is trained by hierarchy, in contrast with the local methods that require more than one model per hierarchy).

Third, functional information of genes is in general highly imbalanced, i.e., functions in the deeper levels of the hierarchy have less associations with genes, and therefore, get lower predicted probabilities. In Chapter 5, the problem of predicting gene functions is redefined as a problem of detecting missing annotations, where the goal is to select a set of annotations instead of computing a probability for every gene-function association. Moreover, a method to detect missing annotations is introduced, which focuses on identifying associations at deeper levels of the hierarchy. The method selects paths of functions based on aggregated probabilities computed for every gene. The results show that exploiting the paths of the hierarchy help in better identifying deeper annotations, which are often regarded as more informative.

Finally, taking advantage of the information available for multiple organisms is key to analyze organisms whose functional information is either not available or is not enough to train predictive models. The method introduced in Chapter 6 combines domain adaptation and HMC to integrate information of multiple sources under the hypothesis that their feature and label spaces are the same. The method aims to improve the performance of predicting gene functions while complying with the hierarchy constrain. Experiments on rice show that including information of other organisms (namely, arabidopsis, maize, and soybean) lead to better performance in the prediction of gene functions for rice. The results highlight that this method can handle scenarios in which some of the sources organisms included may not be related to the target organism.

The contributions presented in this dissertation, including, the developed methods, their implementation, and the case studies on rice and maize, are a significant step forward to computationally address the gene function prediction problem, a current challenge in the area of bioinformatics.

7.2 Future Work

There are many possible extensions of the work presented in this dissertation. This section presents the most promising directions for future work.

First, transfer learning is a promising research topic for gene function prediction that has not been widely explored. The possibility to integrate different types of data into the prediction task, besides gene co-expression, opens multiple directions for further research. For example:

- Exploiting gene orthology to map genes of different organisms and filter the data of additional sources that is used for training may lead to better performance of predictors;
- Combining the problems of predicting functions of genes and proteins will benefit from the information of GCNs and the protein-protein interaction networks (this approach will require a mapping between genes and proteins for each organisms); and
- Using pre-trained neural networks for HMC problems will facilitate the transformation or adaptation of additional data (however it might be necessary to built and train such neural network since it might not exist).

Second, the methods presented in this dissertation focus on different aspects of the gene function prediction problem: considering hierarchical relations between functions, creating data representation to built predictive models, detecting deeper annotations in highly imbalanced datasets, and taking advantage of the information available for multiple organisms. Besides HMC, these aspects have been addressed separately although they can be complementary. In particular, there are two promising combinations:

- Extraction of features is a preliminary step for building predictive models, where the feature extraction method can be used together with the detection of missing annotations or the domain adaptation methods to create new representations of the data that lead to better performance of the predictive models; and

- The domain adaptation method can be combined with the problem of detecting missing annotations, where organisms with more associations in deeper levels of the hierarchy can be used as source of information.

Finally, although this dissertation focuses on computational approaches for the problem of predicting gene functions, the results of the methods should be used as input for *in vivo* experimental verification. More specifically, the annotations identified or predicted by the methods should be used to reduce the search space and facilitate the experimentation in a lab. To that aim, it is necessary to collaborate with research institutes (e.g., the International Center for Tropical Agriculture (CIAT) or the Colombian Research Center for Sugar Cane (Cenicaña)) that have data available for different organisms (and varieties), and can carry out the *in vivo* experimentation. The case studies presented here are based on publicly available data from different sources. However, to obtain accurate predictions, the methods should be trained using data of the same varieties that will be used for *in vivo* experimentation (i.e., methods must be applied to more specific data).

Bibliography

[1] Abu-El-Haija, S., Perozzi, B., Kapoor, A., and Lee, J. N-gcn: Multi-scale graph convolutionfor semi-supervised node classification. In *Conference on Uncertainty in Artificial Intelligence (UAI)* (2019).

[2] Aoki, K., Ogata, Y., and Shibata, D. Approaches for extracting practical information from gene co-expression networks in plant biology. *Plant and Cell Physiology 48*, 3 (2007), 381–390.

[3] Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. Gene Ontology: Tool for the unification of biology. *Nature Genetics 25*, 1 (May 2000), 25–29.

[4] Banerjee, S., Akkaya, C., Perez-Sorrosal, F., and Tsioutsiouliklis, K. Hierarchical transfer learning for multi-label text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, 2019), Association for Computational Linguistics, pp. 6295–6300.

[5] Bergstra, J., and Bengio, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research 13*, 10 (2012), 281–305.

[6] Bhagat, S., Cormode, G., and Muthukrishnan, S. Node classification in social networks. In *Social Network Data Analytics*, C. C. Aggarwal, Ed. Springer US, Boston, MA, 2011, pp. 115–148.

[7] Bi, W., and Kwok, J. T. Multi-label classification on tree- and DAG-structured hierarchies. In *Proceedings of the 28th International Conference on International Conference on Machine Learning* (Madison, WI, USA, 2011), ICML'11, Omnipress, p. 17–24.

[8] Calvert, L. A., Sanint, L. R., Châtel, M., and Izquierdo, J. Rice production in Latin America at critical crossroads. *International Rice Commission Newsletter* (2006).

[9] Cao, J., Kwong, S., and Wang, R. A noise-detection based AdaBoost algorithm for mislabeled data. *Pattern Recognition 45*, 12 (2012), 4451–4465.

[10] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research 16* (June 2002), 321–357.

[11] Chen, L., Wang, Y., and Zhao, F. Exploiting deep transfer learning for the prediction of functional non-coding variants using genomic sequence. *Bioinformatics 38*, 12 (June 2022), 3164–3172.

[12] Chen, Q., Li, Y., Tan, K., Qiao, Y., Pan, S., Jiang, T., and Chen, Y.-P. P. Network-based methods for gene function prediction. *Briefings in Functional Genomics 20*, 4 (July 2021), 249–257.

[13] Chen, T., and Guestrin, C. XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), 785–794.

[14] Chen, Z., Zhao, P., Li, F., Leier, A., Marquez-Lago, T. T., Wang, Y., Webb, G. I., Smith, A. I., Daly, R. J., Chou, K.-C., and Song, J. iFeature: A Python package and web server for features extraction and selection from protein and peptide sequences. *Bioinformatics 34*, 14 (July 2018), 2499–2502.

[15] Childs, K. L., Davidson, R. M., and Buell, C. R. Gene coexpression network analysis as a source of functional annotation for rice genes. *PLoS ONE 6*, 7 (2011), e22196.

[16] Chougrad, H., Zouaki, H., and Alheyane, O. Multi-label transfer learning for the early diagnosis of breast cancer. *Neurocomputing 392* (June 2020), 168–180.

[17] Cruz, D. F., De Meyer, S., Ampe, J., Sprenger, H., Herman, D., Van Hautegem, T., De Block, J., Inzé, D., Nelissen, H., and Maere, S. Using single-plant-omics in the field to link maize genes to functions and phenotypes. *Molecular Systems Biology 16*, 12 (Dec. 2020).

[18] Data61, C. Stellargraph machine learning library. https://github.com/stellargraph/stellargraph, 2018.

[19] Diestel, R. *Graph Theory*, vol. 173 of *Graduate Texts in Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017.

[20] Dimitrovski, I., Kocev, D., Loskovska, S., and Džeroski, S. Detection of visual concepts and annotation of images using ensembles of trees for hierarchical multi-label classification. In *Recognizing Patterns in Signals, Speech, Images and Videos*, D. Ünay, Z. Çataltepe, and S. Aksoy, Eds., vol. 6388. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 152–161.

[21] Elshawi, R., Al-Mallah, M. H., and Sakr, S. On the interpretability of machine learning-based model for predicting hypertension. *BMC Medical Informatics and Decision Making 19*, 1 (July 2019), 146.

[22] Emamjomeh, A., Saboori Robat, E., Zahiri, J., Solouki, M., and Khosravi, P. Gene co-expression network reconstruction: A review on computational methods for inferring functional information from plant-based expression data. *Plant Biotechnology Reports 11*, 2 (Apr. 2017), 71–86.

[23] Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research 47*, D1 (Jan. 2019), D330–D338.

[24] Gligorijević, V., Barot, M., and Bonneau, R. deepNF: Deep network fusion for protein function prediction. *Bioinformatics 34*, 22 (2018), 3873–3881.

[25] Gour, N., and Khanna, P. Multi-class multi-label ophthalmological disease detection using transfer learning based convolutional neural network. *Biomedical Signal Processing and Control 66* (Apr. 2021), 102329.

[26] Grover, A., and Leskovec, J. node2vec: Scalable feature learning for networks, 2016.

[27] Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, Dec. 2017), NIPS'17, Curran Associates Inc., pp. 1025–1035.

[28] Huang, D. W., Sherman, B. T., and Lempicki, R. A. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols 4*, 1 (Jan. 2009), 44–57.

[29] Jia, H., Ding, S., Xu, X., and Nie, R. The latest research progress on spectral clustering. *Neural Computing and Applications 24*, 7-8 (June 2014), 1477–1486.

[30] Jiang, X., Nariai, N., Steffen, M., Kasif, S., and Kolaczyk, E. D. Integration of relational and hierarchical network information for protein function prediction. *BMC Bioinformatics 9*, 1 (2008), 350.

[31] Ju, W., Li, J., Yu, W., and Zhang, R. iGraph: An incremental data processing system for dynamic graph. *Frontiers of Computer Science 10*, 3 (June 2016), 462–476.

[32] Kanehisa, M., and Goto, S. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research 28*, 1 (01 2000), 27–30.

[33] Khan, S. S., and Madden, M. G. A survey of recent trends in one class classification. In *Artificial Intelligence and Cognitive Science*, L. Coyle and J. Freyne, Eds., vol. 6206. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 188–197.

[34] Kipf, T. N., and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)* (2017).

[35] Kleinberg, J. M. Authoritative sources in a hyperlinked environment. *Journal of the ACM 46*, 5 (Sept. 1999), 604–632.

[36] Klopfenstein, D. V., Zhang, L., Pedersen, B. S., Ramírez, F., Warwick Vesztrocy, A., Naldi, A., Mungall, C. J., Yunes, J. M., Botvinnik, O., Weigel, M., Dampier, W., Dessimoz, C., Flick, P., and Tang, H. GOATOOLS: A Python library for Gene Ontology analyses. *Scientific Reports 8*, 1 (Dec. 2018), 10872.

[37] Knuth, D. E. *The Art of Computer Programming*, 3rd ed ed. Addison-Wesley, Reading, Mass, 1997.

[38] Koonin, E. V. Orthologs, paralogs, and evolutionary genomics. *Annual Review of Genetics 39*, 1 (Dec. 2005), 309–338.

[39] Kouw, W. M., and Loog, M. An introduction to domain adaptation and transfer learning. *CoRR abs/1812.11806* (2018).

[40] Kundalia, K., Patel, Y., and Shah, M. Multi-label movie genre detection from a movie poster using knowledge transfer learning. *Augmented Human Research 5*, 1 (Dec. 2020), 11.

[41] Kurata, N., and Yamazaki, Y. Oryzabase: An integrated biological and genome information database for rice. *Plant Physiology 140*, 1 (Jan. 2006), 12–17.

[42] Levatić, J., Kocev, D., and Džeroski, S. The importance of the label hierarchy in hierarchical multi-label classification. *Journal of Intelligent Information Systems 45*, 2 (Oct. 2015), 247–271.

[43] Li, A., Deng, Y., Tan, Y., and Chen, M. A transfer learning-based approach for lysine propionylation prediction. *Frontiers in Physiology 12* (Apr. 2021), 658633.

[44] Li, G., Ji, Z., Chang, Y., Li, S., Qu, X., and Cao, D. ML-ANet: A transfer learning approach using adaptation network for multi-label image classification in autonomous driving. *Chinese Journal of Mechanical Engineering 34*, 1 (Dec. 2021), 78.

[45] Lundberg, S., and Lee, S.-I. A unified approach to interpreting model predictions. *arXiv:1705.07874 [cs, stat]* (Nov. 2017).

[46] Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence 2*, 1 (2020), 2522–5839.

[47] Makrodimitris, S., van Ham, R. C. H. J., and Reinders, M. J. T. Automatic gene function prediction in the 2020's. *Genes 11*, 11 (Oct. 2020), 1264.

[48] McInnes, G., Dalton, R., Sangkuhl, K., Whirl-Carrillo, M., Lee, S.-b., Tsao, P. S., Gaedigk, A., Altman, R. B., and Woodahl, E. L. Transfer learning enables prediction of CYP2D6 haplotype function. *PLoS Computational Biology 16*, 11 (Nov. 2020), e1008399.

[49] Mills, P. Solving for multi-class: A survey and synthesis. *arXiv:1809.05929 [cs, stat]* (Jan. 2021).

[50] Min, S., Kim, H., Lee, B., and Yoon, S. Protein transfer learning improves identification of heat shock protein families. *PLoS ONE 16*, 5 (May 2021), e0251865.

[51] Mu, Z., Yu, T., Liu, X., Zheng, H., Wei, L., and Liu, J. FEGS: A novel feature extraction model for protein sequences and its applications. *BMC Bioinformatics 22*, 1 (Dec. 2021), 297.

[52] Nakano, F. K., Lietaert, M., and Vens, C. Machine learning for discovering missing or wrong protein function annotations: A comparison using updated benchmark datasets. *BMC Bioinformatics 20*, 1 (Dec. 2019), 485.

[53] Novakovsky, G., Saraswat, M., Fornes, O., Mostafavi, S., and Wasserman, W. W. Biologically relevant transfer learning improves transcription factor binding prediction. *Genome Biology 22*, 1 (Dec. 2021), 280.

[54] Obayashi, T., Aoki, Y., Tadaka, S., Kagaya, Y., and Kinoshita, K. ATTED-II in 2018: A plant coexpression database based on investigation of the statistical property of the mutual rank index. *Plant and Cell Physiology 59*, 1 (Jan. 2018), e3–e3.

[55] Obayashi, T., and Kinoshita, K. COXPRESdb: A database to compare gene coexpression in seven model animals. *Nucleic Acids Research 39*, Database (Jan. 2011), D1016–D1022.

[56] Oti, M., van Reeuwijk, J., Huynen, M. A., and Brunner, H. G. Conserved co-expression for candidate disease gene prioritization. *BMC Bioinformatics 9*, 1 (2008), 208.

[57] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (Oct. 2010), 1345–1359.

[58] Petegrosso, R., Park, S., Hwang, T. H., and Kuang, R. Transfer learning across ontologies for phenome–genome association prediction. *Bioinformatics* (Oct. 2016), btw649.

[59] Petsko, G. A. Guilt by association. *Genome Biology 10*, 4 (2009), 104.

[60] Ramírez-Corona, M., Sucar, L. E., and Morales, E. F. Hierarchical multilabel classification based on path evaluation. *International Journal of Approximate Reasoning 68* (Jan. 2016), 179–193.

[61] Ranganathan, S., Gribskov, M. R., Nakai, K., and Schönbach, C. *Encyclopedia of Bioinformatics and Computational Biology*. Elsevier, 2019. OCLC: 1052465484.

[62] Reaume, C. J., and Sokolowski, M. B. Conservation of gene function in behaviour. *Philosophical Transactions of the Royal Society B: Biological Sciences 366*, 1574 (July 2011), 2100–2110.

[63] Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. d. F., and Rodrigues, F. A. Clustering algorithms: A comparative approach. *PLoS ONE 14*, 1 (Jan. 2019), e0210236.

[64] Romero, M., Finke, J., and Rocha, C. Node Classification. https://github.com/migueleci/node_classification, 2021.

[65] Romero, M., Finke, J., and Rocha, C. A top-down supervised learning approach to hierarchical multi-label classification in networks. *Applied Network Science 7*, 1 (Dec. 2022), 8.

[66] Romero, M., Nakano, F. K., Finke, J., Rocha, C., and Vens, C. CONNECT: Hierarchical multi-label classification and domain adaptation for gene function prediction. https://github.com/migueleci/connect, 2022.

[67] Romero, M., Nakano, F. K., Finke, J., Rocha, C., and Vens, C. REASSIGN: Hierarchical multi-label classification to discover missing annotations. https://github.com/migueleci/reassign, 2022.

[68] Romero, M., Ramírez, Ó., Finke, J., and Rocha, C. Supervised gene function prediction using spectral clustering on gene co-expression networks. In *Complex Networks & Their Applications X*, vol. 1016. Springer International Publishing, Cham, 2022, pp. 652–663.

[69] Romero, M., Ramírez, O., Finke, J., and Rocha, C. Clustering HMC. https://github.com/migueleci/clustering_hmc, 2022.

[70] Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence 1*, 5 (May 2019), 206–215.

[71] Rust, A. G., Mongin, E., and Birney, E. Genome annotation techniques: New approaches and challenges. *Drug Discovery Today 7*, 11 (May 2002), S70–S76.

[72] Sabzevari, M., Martínez-Muñoz, G., and Suárez, A. A two-stage ensemble method for the detection of class-label noise. *Neurocomputing 275* (Jan. 2018), 2374–2383.

[73] Sakai, H., Lee, S. S., Tanaka, T., Numa, H., Kim, J., Kawahara, Y., Wakimoto, H., Yang, C.-c., Iwamoto, M., Abe, T., Yamada, Y., Muto, A., Inokuchi, H., Ikemura, T., Matsumoto, T., Sasaki, T., and Itoh, T. Rice annotation project database (RAP-DB): An integrative and interactive database for rice genomics. *Plant and Cell Physiology 54*, 2 (Feb. 2013), e6–e6.

[74] Samami, M., Akbari, E., Abdar, M., Plawiak, P., Nematzadeh, H., Basiri, M. E., and Makarenkov, V. A mixed solution-based high agreement filtering method for class noise detection in binary classification. *Physica A: Statistical Mechanics and its Applications 553* (2020), 124219.

[75] Sarwar, M. U., Zafar, S., Mkaouer, M. W., Walia, G. S., and Malik, M. Z. Multi-label classification of commit messages using transfer learning. In *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)* (Coimbra, Portugal, Oct. 2020), IEEE, pp. 37–42.

[76] Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., and Džeroski, S. Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics 11*, 1 (Dec. 2010), 2.

[77] Serin, E. A. R., Nijveen, H., Hilhorst, H. W. M., and Ligterink, W. Learning from co-expression networks: Possibilities and challenges. *Frontiers in Plant Science 7* (Apr. 2016).

[78] Silla, C. N., and Freitas, A. A. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery 22*, 1-2 (Jan. 2011), 31–72.

[79] Sluban, B., Gamberger, D., and Lavrač, N. Ensemble-based noise detection: Noise ranking and visual performance evaluation. *Data mining and knowledge discovery 28*, 2 (2014), 265–303.

[80] Takamiya, K., and Tsutsui, H. Rice and irrigation in Latin America. *Rural and Environment Engineering 2000*, 38 (2000), 5–19.

[81] Tao, J., and Fang, X. Toward multi-label sentiment analysis: A transfer learning based approach. *Journal of Big Data 7*, 1 (Dec. 2020), 1.

[82] Valentini, G. True path rule hierarchical ensembles. In *Multiple Classifier Systems*, J. A. Benediktsson, J. Kittler, and F. Roli, Eds., vol. 5519. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 232–241.

[83] van Dam, S., Võsa, U., van der Graaf, A., Franke, L., and de Magalhães, J. P. Gene co-expression analysis for functional classification and gene–disease predictions. *Briefings in Bioinformatics* (Jan. 2017).

[84] Vandepoele, K., Quimbaya, M., Casneuf, T., De Veylder, L., and Van de Peer, Y. Unraveling transcriptional control in Arabidopsis using cis-regulatory elements and coexpression networks. *Plant Physiology 150*, 2 (June 2009), 535–546.

[85] Vens, C., Struyf, J., Schietgat, L., Džeroski, S., and Blockeel, H. Decision trees for hierarchical multi-label classification. *Machine Learning 73*, 2 (Nov. 2008), 185–214.

[86] Wang, Y., Xia, Z., Deng, J., Xie, X., Gong, M., and Ma, X. TLGP: A flexible transfer learning algorithm for gene prioritization based on heterogeneous source domain. *BMC Bioinformatics 22*, S9 (Aug. 2021), 274.

[87] Weiss, K., Khoshgoftaar, T. M., and Wang, D. A survey of transfer learning. *Journal of Big Data 3*, 1 (Dec. 2016), 9.

[88] Xiao, S., Wang, S., Dai, Y., and Guo, W. Graph neural networks in node classification: Survey and evaluation. *Machine Vision and Applications 33*, 1 (Nov. 2021), 4.

[89] Xu, D., Shi, Y., Tsang, I. W., Ong, Y.-S., Gong, C., and Shen, X. Survey on multi-output learning. *IEEE Transactions on Neural Networks and Learning Systems 31*, 7 (2020), 2409–2429.

[90] Xu, Y., Min, H., Wu, Q., Song, H., and Ye, B. Multi-instance metric transfer learning for genome-wide protein function prediction. *Scientific Reports 7*, 1 (Mar. 2017), 41831.

[91] Yon Rhee, S., Wood, V., Dolinski, K., and Draghici, S. Use and misuse of the gene ontology annotations. *Nature Reviews Genetics 9*, 7 (July 2008), 509–515.

[92] Yu, and Shi. Multiclass spectral clustering. In *Proceedings Ninth IEEE International Conference on Computer Vision* (Nice, France, 2003), IEEE, pp. 313–319 vol.1.

[93] Yu, G., Zhu, H., and Domeniconi, C. Predicting protein functions using incomplete hierarchical labels. *BMC Bioinformatics 16*, 1 (Jan. 2015), 1.

[94] Zhang, H., Chen, F., Shen, Z., Hao, Q., Zhu, C., and Savvides, M. Solving missing-annotation object detection with background recalibration loss. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2020), pp. 1888–1892.

[95] Zhao, Y., Fu, G., Wang, J., Guo, M., and Yu, G. Gene function prediction based on Gene Ontology Hierarchy Preserving Hashing. *Genomics 111*, 3 (2019), 334–342.

[96] Zhou, G., Wang, J., Zhang, X., Guo, M., and Yu, G. Predicting functions of maize proteins using graph convolutional network. *BMC Bioinformatics 21*, S16 (Dec. 2020), 420.

[97] Zhou, X., Kao, M.-C. J., and Wong, W. H. Transitive functional annotation by shortest-path analysis of gene expression data. *Proceedings of the National Academy of Sciences 99*, 20 (Oct. 2002), 12783–12788.

[98] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A comprehensive survey on transfer learning. *Proceedings of the IEEE 109*, 1 (Jan. 2021), 43–76.

Curriculum

Miguel Romero was born in Bogotá on June 10, 1991. He earned a B.S. degree in Economics (in 2013) and Systems Engineering (in 2017) from the Escuela Colombiana de Ingeniería, Bogotá (Colombia). He participated in multiple programming contest from 2014 to 2016, including the Regional Latin America Programming Contest, North Region ACM-ICPC, in which his team reached the seventh place in the region. He has worked as teaching assistant at the Escuela Colombiana de Ingeniería and Pontificia Universidad Javeriana, Cali (Colombia) teaching courses of differential calculus, and algorithm analysis and design. In addition, he has worked as a research assistant at the Pontificia Universidad Javeriana with fundings from the Colombian Administrative Department of Science, Technology and Innovation (COLCIENCIAS) between 2017 and 2019, and at the Research Institute in Omics Sciences iÓMICAS between 2019 and 2022. He published papers in international conferences, such as the NASA Formal Methods (NFM), the International Workshop on Rewriting Logic and its Applications (WRLA), and the International Conference on Complex Networks and their Applications. He also published journal articles in Social Network Analysis and Mining (SNAM), Applied Network Science (ANS), and Plant Direct.

In 2019, he joint the Doctorate in Engineering and Applied Sciences at the Pontificia Universidad Javeriana, Cali funded by the In-silico Multiscale Optimization of Sustainable Agricultural Crops (ÓMICAS). He works on the development of mathematical models and algorithms that allow identifying, from in-silico omic characterization, functions of genes in different varieties of crops based on machine learning and graph theory under the supervision of profs. Camilo Rocha and Jorge Finke. He has experience in rewriting logic, network analysis, algorithms, competitive programming, and machine learning.

List of Publications

Articles in Internationally Reviewed Academic Journals

Romero, M., Finke, J., and Rocha, C. A top-down supervised learning approach to hierarchical multi-label classification in networks. Applied Network Science 7, 8 (2022).

Romero, M., Ramírez, O., Finke, J., and Rocha, C. Feature extraction with spectral clustering for gene function prediction using hierarchical multi-label classification. Applied Network Science 7, 28 (2022).

Romero, M., Nakano, F.K., Finke, J., Rocha, C., and , Vens C. Hierarchy exploitation to detect missing annotations on hierarchical multi-label classification. Computers in Biology and Medicine, submitted for publication (2022).

Article in Conference Post-Proceedings Published as a Book Chapter

Romero, M., Finke, J., Quimbaya, M., and Rocha, C. (2020). In-silico gene annotation prediction using the co-expression network structure. Complex Networks and Their Applications VIII. COMPLEX NETWORKS 2019. Studies in Computational Intelligence, vol 882. Springer, Cham.

Romero, M., Ramírez, Ó., Finke, J., and Rocha, C. (2022). Supervised gene function prediction using spectral clustering on gene co-expression networks. Complex Networks and Their Applications X. COMPLEX NETWORKS 2021. Studies in Computational Intelligence, vol 1016. Springer, Cham.

López-Rozo, N., **Romero, M.**, Finke, J., and Rocha, C. (2022). A network-based approach for inferring thresholds in co-expression networks. COMPLEX NETWORKS 2022, accepted.

Abstracts, Presented at Scientific Conferences

Romero M. Integration of function hierarchy data for gene function prediction. Virtual Symposium in Plant Omics Science. Cali, Colombia, 23-27 November (2020).

Preprints

Romero, M., Nakano, F.K., Finke, J., Rocha, C., and , Vens C. Domain adaptation and hierarchical multi-label classification for gene function prediction (2022).

DOCTORATE IN ENGINEERING AND APPLIED SCIENCES
FACULTY OF ENGINEERING AND SCIENCES
Calle 18 No. 118-250
Cali, Colombia
doctoradoingenieria@javerianacali.edu.co