Semi-Supervised Hierarchical Classification

Jonathan Serrano Pérez, Luis Enrique Sucar Succar

Technical Report No. CCC-22-002
May, 2022

Instituto Nacional de Astrofísica, Óptica y Electrónica

Luis Enrique Error 1,
Santa María Tonantzintla,
72840, Puebla, México

Abstract

Hierarchical classification is commonly seen as a special type of multi-label classification, that is, the instances can be associated to multiple labels, but labels are arranged in a predefined structure, a hierarchy. Moreover, the predictions in hierarchical classification have to fulfill the hierarchical constraint that states if an instance is associated to a node, then it also has to be associated to the ancestors of that node.

On the other hand, scarce data is a common problem in supervised classification, and therefore, it is also present in hierarchical classification. Scarce data occurs when hand-labeling data is expensive, time-consuming or difficult to do. Even though, labels arranged in hierarchies are found in multiples domains, such as text categorization, image classification, biology and music, just a few works address the problem of scarce data in a hierarchical classification scenario. Hence, we hypothesized that training a semi-supervised hierarchical classifier (SSHC) with labeled and unlabeled data will produce a classifier with better performance than a classifier trained only on labeled data.

Therefore, the main goal of this research it to develop a *semi-supervised hierarchical classifier* that can be trained with labeled and unlabeled data. Furthermore, it is expected to be able to handle hierarchies of tree and DAG type, as well as, predict a single and multiple paths of labels.

A preliminary SSHC based on local information (SSHC-BLI) was developed, which pseudo-labels the unlabeled data and later used them to train a hierarchical classifier. The nearest labeled instances to each unlabeled instance are used to build a pseudo-label, so, if the unlabeled data is *similar* to its nearest labeled instances, it is pseudo-labeled, else stays unlabeled. Preliminary results on a subset of the Functional Catalogue (FunCat) datasets shows promising results, because the SSHC-BLI tends to outperform a supervised hierarchical classifier (Top-Down) trained only on labeled instances.

Contents

Acronyms

AD Artificial Datasets.

DAG Directed Acyclic Graph.

FD Full Depth.

FunCat Functional Catalogue.

GO Gene Ontology.

HC Hierarchical classification.

hF Hierarchical F-measure.

hP Hierarchical Precision.

hR Hierarchical Recall.

HS Hierarchical Structure.

LCN Local Classifier per Node.

LCPN Local Classifier per Parent Node.

MPL Multiple Paths of Labels.

MPP Multiple Path Prediction.

NMLNP Non-Mandatory Leaf Node Prediction.

PD Partial Depth.

SC Supervised Classification.

SISI Similarity of an Instance with a Set of Instances.

SPL Single Path of Labels.

SPP Single Path Prediction.

SSHC Semi-Supervised Hierarchical Classifier.

SSHC-BLI Semi-Supervised Hierarchical Classifier Based on Local Information.

SSL Semi-Supervised Learning.

T Tree.

TD Top-Down.

TL Transfer Learning.

1 Introduction

Supervised classification (SC) consists on building a model from labeled instances, which is later used to predict the class of *new* instances. Nevertheless, scarce data is a common problem of SC, this occurs when hand-labeling data is time-consuming, expensive or difficult to label. Consequently, the problem when a classifier is trained with few labeled data is that an unreliable classifier could be obtained.

Furthermore, the problem of scarce data may be found in a scenario of multi-label classification, that is, when instances can be associated to multiple labels (classes). As shown by Bielza et al. [2011], Sucar et al. [2014] the labels can be related, hence the performance of a classifier can be improve by taking into account that relations, however, if the labels are independent, the performance of the classifier will not improve.

Even more, hierarchical classification (HC) can be seen as a *special* type of multi-label classification, that is, an instance can be associated to multiple labels, but the labels are already arranged into a predefined structure, that contains the relations among the labels, which is commonly a Tree but in its general form is a directed acyclic graph (DAG). Additionally, the predictions in HC have to fulfill the *hierarchical constraint*, which states that if an instance is associated to a node (label), the instance has to be associated to the ancestors of that node (label). So HC with scarce data is the problem to be addressed in this research.

There are two main approaches that try to addressed the problem of scarce data. The first is generation of artificial data, for example the SMOTE method [Chawla et al., 2002]. The second consists on making use of unlabeled data along with labeled data, this approach is known as semi-supervised classification [van Engelen and Hoos, 2019].

For large amounts of (unlabeled) information can be obtained from different sources, for instance the internet, such as video, text, images, ..., this research will be focused in the semi-supervised classification approach. So that, applications can make use of that (unlabeled) information along with the available labeled information.

The aim of this research is to develop a *Semi-Supervised Hierarchical Classifier* (SSHC) which can be trained with labeled and unlabeled data. Furthermore, the SSHC has to make use of the relations among the labels (hierarchy), and it is expected that the SSHC can predict both single and multiple paths of labels.

An initial SSHC based on local information (SSHC-BLI) was developed (it can handle tree hierarchies and predicts a single path of labels), whose main idea is to pseudo-label the unlabeled data, which are later used to train a hierarchical classifier. The nearest labeled instances to each unlabeled instances are used to build a pseudo-

label, then, the similitude of the unlabeled instance with its nearest labeled instances is estimated, if they are similar, the unlabeled instance is pseudo-labeled, else it stays unlabeled. Experiments on a subset of the Functional Catalogue (FunCat) datasets show promising results, because the SSHC-BLI tends to outperform the baseline, a supervised hierarchical classifier (Top-Down) trained only on labeled instances.

2 Background

2.1 Introduction

This section presents the background required for this research. An introduction to supervised learning is given in section 2.2, introductions to semi-supervised learning and transfer learning are given in sections 2.3 and 2.4, respectively. Finally in section 2.5 hierarchical classification is presented.

2.2 Supervised Classification

Supervised classification is essentially a mapping from the measurement space to a set of classes that represent the types of interest to the user. The steps in applying a classifier (based on Richards [2013b]) usually include:

1. Deciding the set of classes.
2. Choosing known representative instances for each of the classes, and describe them in the space of attributes $\mathbb{R}^d$, where d is the number of attributes. These form the training set.
3. Using the training set to estimate the parameters of the particular classifier algorithm to be employed.
4. Assessing the performance of the classifier using a labeled testing set, which is different from the training set.
5. Using the trained classifier to label unknown instances.

The first two steps can be skipped if we already have real world or artificial databases. Furthermore, based on the results obtained in step 4, the training process can be refined in order to improve its performance.

Some supervised learning algorithms Richards [2013b], Géron [2017] are: K-nearest neighbors, Minimum Distance Classification, Naive Bayes Classifier, Maximum likelihood classification, Parallelepiped Classification, Support Vector Machines, Decision Trees and Random Forest, Artificial Neural Networks.

2.2.1 Multi-label classification

The multi-label classification problem correspond to searching a function h that assigns to each instance a subset of labels. First, let $\mathbf{x}$ be an instance represented by a vector of m features:

$$\mathbf{x} = (X_1 = x_1, ..., X_m = x_m) \tag{1}$$

where each feature X_i could be a numeric or categorical variable, let Y be a vector of d labels:

$$Y = (L_1 = l_1, ..., L_d = l_d) \tag{2}$$

where each label L_j can take the values $\{1, 0\}$, that is, positively or negatively classified. Hence, h is a function that given an instance x returns the subset of labels to which it is associated:

$$h : (x_1, ..., x_m) \mapsto (l_1, ..., l_d) \tag{3}$$

Thus, the h function should assign to each instance $\mathbf{x}$ the most likely combination of labels, that is:

$$arg\, max_{l_1,...,l_d}\, P(L_1 = l_1, ..., L_d = l_d|\mathbf{x}) \tag{4}$$

There are two basic strategies for multilabel classification. The first is *binary relevance* [Luaces et al., 2012] where a binary classifier is trained for each label and the decision rule assigns to an instance all the labels whose classifiers predicted it positively, nevertheless, this strategy do not consider dependencies between labels. The second is *label power set*, which defines a new compound of class variables, whose values are all the possible combinations of values of the original labels, then, it can be treated as a multiclass problem, this strategy implicitly considers interaction between labels, but, it could be computationally expensive, because for N labels it will generate a compound of 2^N classes. For example, if there are $N = 10$ labels, the compound will have $2^{10} = 1024$ classes.

2.3 Semi Supervised Learning

Unsupervised classification can be used as a stand-alone technique, when reliable training data for supervised classification cannot be obtained or is too expensive to

acquire [Richards, 2013a]. Furthermore, while in supervised classification an instance has associated an output value, in unsupervised classification there is no specific output value for the instances. Some unsupervised algorithms are k-means, k-medians and hierarchical clustering.

On the other hand, Semi-Supervised Learning (SSL) can be seen as the branch of machine learning that aims to combine supervised and unsupervised learning [Chapelle et al., 2010, Zhu, 2008]. That is, SSL uses labeled and unlabeled data to perform learning tasks.

Semi-supervised classification methods are appropriated to scenarios where labeled data is scarce, and a reliable classifier could be hard to obtain. Scarce labeled data occurs when it is expensive or difficult to obtain, like computer-aided diagnosis, drug discovery and part-of-speech tagging [van Engelen and Hoos, 2019].

2.3.1 Assumptions of Semi-Supervised Learning

In SSL there are some recognized assumptions, which are the foundation of most semi-supervised learning algorithms, which depend on one or more of them being satisfied (explicitly or implicitly) [Chapelle et al., 2010]. They are briefly described below:

- **Smoothness assumption**: It states that, for two input points $x_i, x_j \in X$ that are close by the input space, the corresponding labels y_i, y_j should be the same. This assumption has a benefit, it can be applied transitively to unlabeled data. For example, let $x_0 \in X_L$ be a labeled point and let $x_1, x_2 \in X_U$ be unlabeled points, if x_1 is close to both x_0 and x_2, but x_0 is not close to x_2, x_2 can be labeled with the same label than x_0, that is, the label was transitively propagated through x_1.
- **Low-density assumption**: It states that the decision boundary of a classifier should preferably pass through low-density regions in the input space. That is the same as saying that the decision boundary should not pass through high-density areas.
- **Manifold assumption**: It states that:
  - The input space is composed of multiple lower dimension manifolds on which all data points lie.
  - Data points on the same manifold have the same label.

An example of the smoothness and low-density assumption is shown in Fig. 1 a), example for manifold assumption is shown in Fig. 1 b).

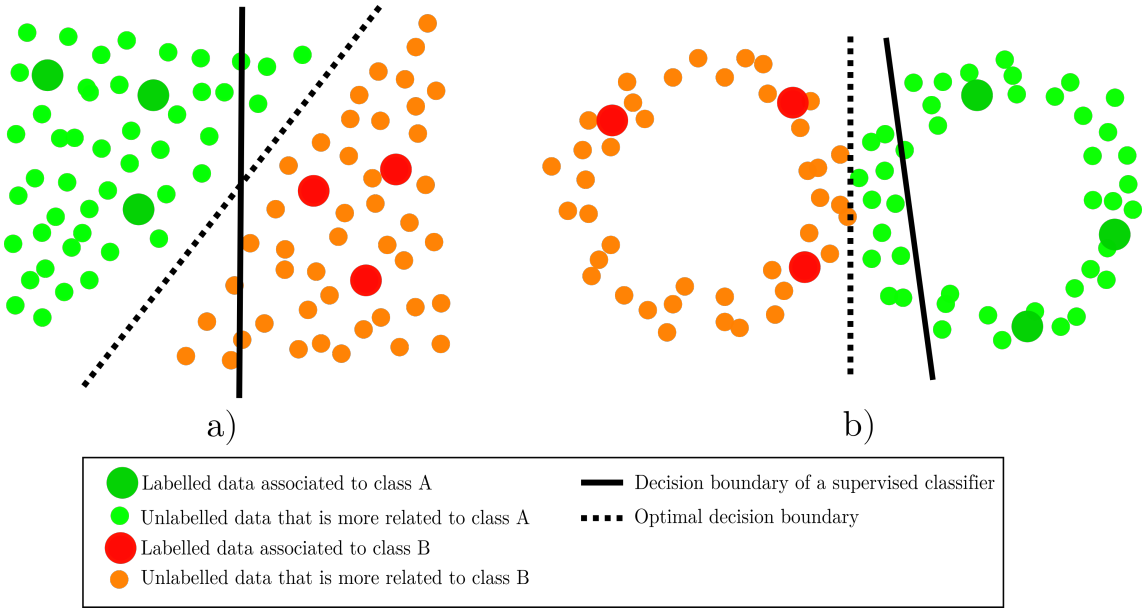


Figure 1: Semi-supervised learning assumptions [van Engelen and Hoos, 2019]. a) Smoothness and low-density assumptions, b) Manifold assumption. (Best seen in color)

2.3.2 Taxonomy of Semi-Supervised Learning Methods

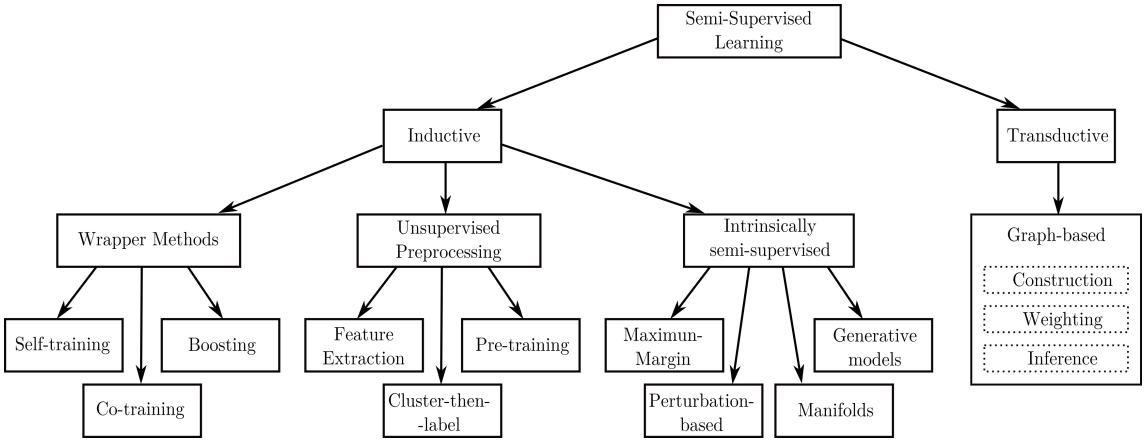


Figure 2: A taxonomy that groups the different approaches of semi-supervised learning.

van Engelen and Hoos [2019] proposed a taxonomy to group the SSL methods, the taxonomy is shown in Fig. 2. First, the SSL methods are divided into two main groups, inductive and transductive, the former produces a classification model (which can be used to label new instances), while the second is only focused with labeling the unlabeled data points.

The inductive methods are the most interesting for this research proposal,

because a classification model for predicting/labeling new data, different from the training set (labeled and unlabeled data), is required. However, inductive methods can be grouped based on the way they incorporate the unlabeled data:

- **Wrapper methods**: They are the most known algorithms for SSL. The procedure commonly consists of two alternating steps, *training*: one or more classifiers are trained with labeled and pseudo-labeled data (if available), and *pseudo-labeling*: the resulting classifiers are used to infer labels for previously unlabeled data and those with the most confident predictions are pseudo-labeled and used in the next iteration. Wrapper methods can be divided into the following categories:
  - *Self-training*: It was first proposed by Yarowsky [1995] for word sense disambiguation in text documents. They consist of a single classifier iteratively trained with labeled and pseudo-labeled data.
  - *Co-training*: It is an extension of self-training to multiple supervised classifiers, that is, two or more classifiers are trained on labeled data, and each one adds its most confident predictions to the labeled data of the other classifiers in each iteration. However, in co-training is important that the base classifiers are not strongly correlated in their predictions, in order to provide each other with useful information, this condition is called the *diversity criterion* [Wang and Zhou, 2010]. In co-training methods, we can find *multi-view co-training* methods [Xu et al., 2013, Blum and Mitchell, 1998] where multiple *views* (subsets of features) exist, *single-view co-training* methods [Du et al., 2011, Jiao Wang et al., 2008] only there is one view, so methods split the data into multiple views, and *co-regularization* methods [Sindhwani and Rosenberg, 2008, Sindhwani et al., 2005] where the ensemble quality and the disagreement between base learners are simultaneously optimized.
  - *Boosting*: semi-supervised boosting methods have been proposed such as *SSMBoost* [Grandvalet et al., 2001, d'Alché-Buc et al., 2002], *Adaptative Supervised Ensemble* [Bennett et al., 2002] and *SemiBoost* [Mallapragada et al., 2009]
- **Unsupervised preprocessing**: They extract useful features from unlabeled data, pre-cluster the data or determine the initial parameters of a supervised method in a unsupervised manner. That is, the supervised classifier is only trained with the labeled data.
- **Intrinsically semi-supervised methods**: They directly optimize an objective function with elements for labeled and unlabeled instances. Intrinsically semi-supervised methods can be divided into the following categories:

- *Maximum-margin methods*: This approach is focused in the low-density assumption. That is, these classifiers attempt to maximize the distance between the given data and the decision boundary. Some based on support vector machines are S3VM [Bennett and Demiriz, 1998, Ding et al., 2017] and S4VM [Li and Zhou, 2015].
- *Perturbation based methods*: A predictive model should be robust to local perturbation in its input, because the smoothness assumption. That is, if a data point is perturbed with a *small* amount of noise, the prediction for the noise and *clean* data points should be the same.
- *Manifolds*: This approach is based on the manifold assumption. Two different techniques are presented, *manifold regularization techniques* that define a graph over data points and penalize differences in predictions for instances with small geodesic distance, and *manifold approximation techniques* that explicity estimate the manifolds where the data lie while optimize an objective function.
- *Generative Models*: The main goal of these methods is to model the distribution that generated the data.

2.4 Transfer Learning

Transfer learning (TL) is commonly used in cases when labeled and unlabeled data are difficult to collect. Hence, TL is focused on transferring the knowledge across domains. Zhuang et al. [2021] formally defines TL as follows:

Given some/an observation(s) corresponding to m^s source domain(s) and task(s) (i.e, $\{(\mathcal{D}_{S_i}, \mathcal{T}_{S_i})|i = 1, ..., m^s\}$), and some/an observation(s) about m^T target domain(s) and task(s) (i.e., $\{(\mathcal{D}_{T_j}, \mathcal{T}_{T_j})|j = 1, ..., m^T\}$), transfer learning utilizes the knowledge implied in the source domain(s) to improve the performance of the learned decision functions $f^{T_j} = (j = 1, ..., m^T)$ on the target domain(s).

Note this definition covers both situations, single-source transfer learning and multi-source transfer learning.

When TL techniques are designed three issues should be considered [Aggarwal, 2014, Pan and Yang, 2010]: first, *what to transfer?*, that asks which part of the knowledge can be transferred from source domain to target domain; second, *how to transfer?*, once it is known which knowledge can be transferred, the algorithms for transferring the knowledge need to be developed; and third, *when to transfer?*, this last has to do with in which situations it is appropriate to use transfer learning.

It is worth to mention that in some situations where the source and target domains are no related to each other, the TL may result unsuccessful. In other

words, the performance in the target domain could be worsened, this situation is known as *negative transfer*.

2.5 Hierarchical Classification

Hierarchical classification is a special type of multilabel classification in which the labels are arranged in a predefined structure, the structure can be a Tree or in its general form a Directed Acyclic Graph (DAG). Thus the *Hierarchical Structure* (HS) can be denoted with the notation of graphs:

$$HS = (L, E) \tag{5}$$

where L is the set of labels (nodes), E is the set of edges that linked the labels (nodes) and HS is a DAG. Note, a Tree is a DAG where all nodes have only one parent, except the root node which does not have parents.

Furthermore, in hierarchical classification the *Hierarchical constraint* exists, which states that if an instance x is associated to the label $l \in L$ then x has to be associated to the ancestors of l, $Anc(l)$, given by the HS:

$$\forall x \in l \rightarrow x \in z, \forall z \in Anc(l) \tag{6}$$

Thus, a *valid path* or *consistent path* is a subset of the labels that complies the hierarchical constraint.

2.5.1 Hierarchical Classification Problems

In hierarchical classification there are different problems, thus, it is important to know the hierarchical classification problem, in order to choose a suitable method to train and predict. Silla and Freitas [2011] describe the different hierarchical problems as a 3-tuple $< \Upsilon, \Psi, \Phi >$ where:

- Υ: specifies the type of hierarchical structure in which the labels are arranged, so, it can take one of two values, T if it is a tree or DAG if it is a Direct Acyclic Graph.
- Ψ: specifies whether an instance can be associated either one or multiple paths. Thus the values that it can take are: Single Path of Labels (SPL) and Multiple Paths of Labels (MPL).
- Φ: describes the depth of the paths of the instances, two values are permitted: Full Depth (FD) if the path (or paths) of all instances reach a leaf node, and Partial Depth (PD) if at least one path of an instance does not reach a leaf node.

This implies that exist at least eight different hierarchical classification problems, Table 1 lists the different problems. Furthermore, in Fig. 3 are shown examples of paths of instances that represent the first four problems of Table 1, that is, those with HS tree type, in Fig. 4 are shown the next four examples of paths of instances that represent the problems with DAG type HS. Note that the hierarchical structures for Trees and DAG's were designed almost the same, this for easy visualization and comparison between the different hierarchical classification problems.

Table 1: This table lists the different hierarchical classification problems. Υ: hierarchical structure, Ψ: number of paths, Φ: depth of paths, T: Tree, DAG: Directed Acyclic Graph, SPL: Single Path of Labels, MPL: Multiple Paths of Labels, FD: Full Depth, PD: Partial Depth.

#	Υ	Ψ	Φ
1	T	SPL	PD
2	T	SPL	FD
3	T	MPL	PD
4	T	MPL	FD
5	DAG	SPL	PD
6	DAG	SPL	FD
7	DAG	MPL	PD
8	DAG	MPL	FD

2.6 Evaluation

In order to evaluate the performance of the proposed semi-supervised hierarchical classifier (SSHC), the evaluation measures commonly used for hierarchical classification will be used, this is because the SSHC produces a model that will be used to classify instances that were not used in the training phase.

In hierarchical classification different evaluation measures have been proposed [Nakano et al., 2017, Silla and Freitas, 2011] which assess whether the predictions of the method were correct or partially correct. Let N be the number of instances in the test set, let Y be the real subset of labels to which an instance is associated and let $\widehat{Y}$ be the subset of predicted labels. Some evaluation measures are described below: .

- **Exact Match**: It is the most strict measure, because, the prediction of an instance has to be equal to real subset of labels. So, it returns the percentage

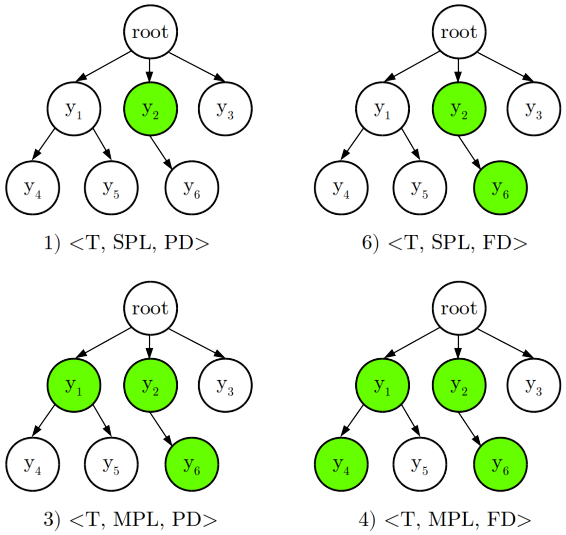


Figure 3: Examples of the four problems with HS tree type. The nodes shaded in green for each HS indicate the labels to which some instance is associated, for example, the instance x showed in the problem 3) is associated to the subset $\{y_1, y_2, y_6\}$.

of instances classified correctly.

$$Exact\_Match = \frac{1}{N}\sum_{i=1}^{N} 1_{Y=\widehat{Y}} \tag{7}$$

- **Accuracy**: For each instance the ratio of labels predicted correctly to the union of real and predicted labels is calculated. So, the average of all the instances is returned.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} \frac{\left|Y_i \cap \widehat{Y}_i\right|}{\left|Y_i \cup \widehat{Y}_i\right|} \tag{8}$$

- **Hierarchical F-measure (hF)**: F-measure for hierarchical classification.

$$hF = \frac{2 * hP * hR}{hP + hR} \tag{9}$$

$$hP = \frac{\sum_{i=1}^{N}\left|Y_i \cap \widehat{Y}_i\right|}{\sum_{i=1}^{N}\left|\widehat{Y}_i\right|} \tag{10}$$

$$hR = \frac{\sum_{i=1}^{N}\left|Y_i \cap \widehat{Y}_i\right|}{\sum_{i=1}^{N}|Y_i|} \tag{11}$$

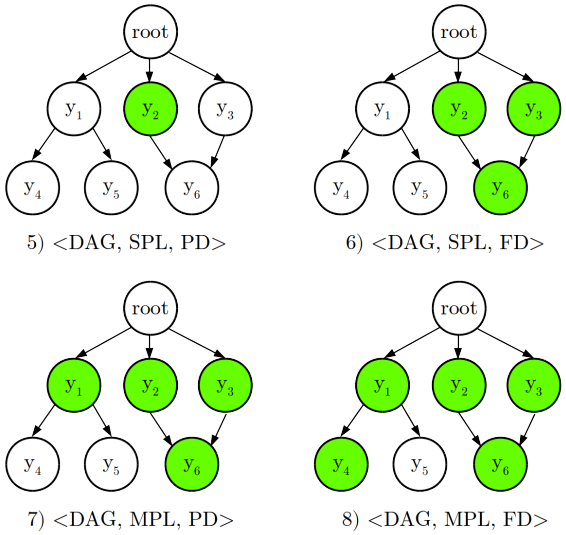


Figure 4: Examples of the four problems with DAG type HS. The nodes shaded in green for each HS indicate the labels to which some instance is associated, for example, the instance x showed in the problem 6) is associated to the subset $\{y_2, y_3, y_6\}$.

Where hR is the hierarchical Precision, which calculates the ratio of correct predictions over the number of predictions in the complete dataset, and hP is hierarchical Recall, which calculates the ratio of correct predictions over the number of real labels in the complete dataset.

Even though, the previous evaluation measures can be used to evaluate any hierarchical classification method, they are mainly used to evaluate hierarchical classification methods of single path prediction type. Additionally, there are others measures such as Area Under the ROC Curve (AUC) Robinson et al. [2015], Area Under the PR Curve (AUPRC) Bi and Kwok [2011], Sun et al. [2017], Area under the average PR-curve ($AU(\overline{PRC})$) Cerri et al. [2014, 2016] and Weighted average of the areas under the individual PR-curves $\overline{AUPRC}_w$ Cerri et al. [2014, 2016], however, those measures are mainly used in hierarchical classification problems where the instances are associated to multiple paths of labels.

3 Related Work

3.1 Hierarchical Classification

Some classical approaches for hierarchical classification are described in this section.

3.1.1 Flat classification

Flat classification for hierarchical classification is perhaps the simplest method, because this method ignores almost completely the hierarchical structure (HS) and focus its training and predictions over the leaf nodes of the HS. Figure 5 shows examples of the nodes used to generate a *multiclass* problem in either tree or DAG structures, hence any multiclass classifier or strategy can be applied. Nevertheless, the main drawback of this method is that it ignores the hierarchical structure in training and predictions phases. Therefore useful information provided by the hierarchy is wasted.

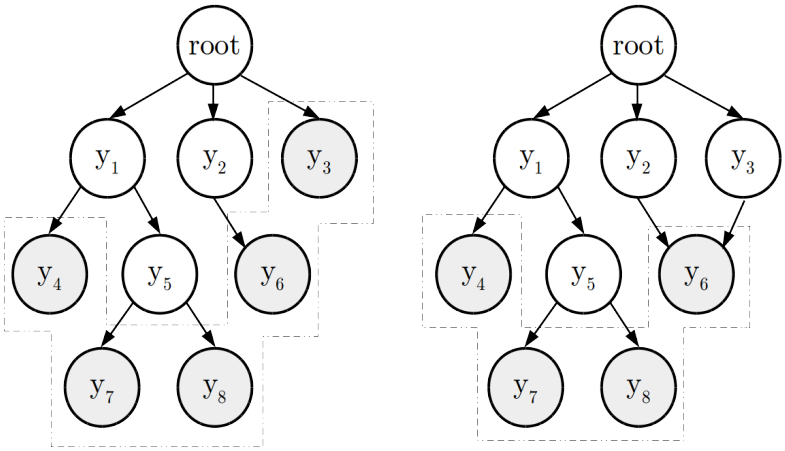


Figure 5: Flat classification. Leaf a tree; Right a DAG. A multiclass classifier is trained considering only the leaf nodes (shaded in gray).

3.1.2 Local Classifier per Parent Node Approach

In this approach Local Classifiers per Parent Node (LCPN) are trained, that is, for each non-leaf node a multiclass classifier is trained to predict its children nodes. This approach is shown in Fig. 6, where the different multiclass problems are inside boxes. It is not natural to use LCPN in hierarchical problems with DAG type hierarchies, since nodes with multiple parents obtain a prediction from each parent node, as can be seen on node y_6. However, strategies to combine multiple predictions can be used, such as Ramírez-Corona et al. [2016] that average the scores obtained by the different multiclass classifiers.

Once multiclass classifiers are trained for each non-leaf node, the standard prediction for an instance follows the *Top-Down* (TD) procedure, that is, the instance is evaluated in the multiclass classifier of the root node, then the prediction advances toward the child node with the highest score, and so on, until a leaf node is reached.

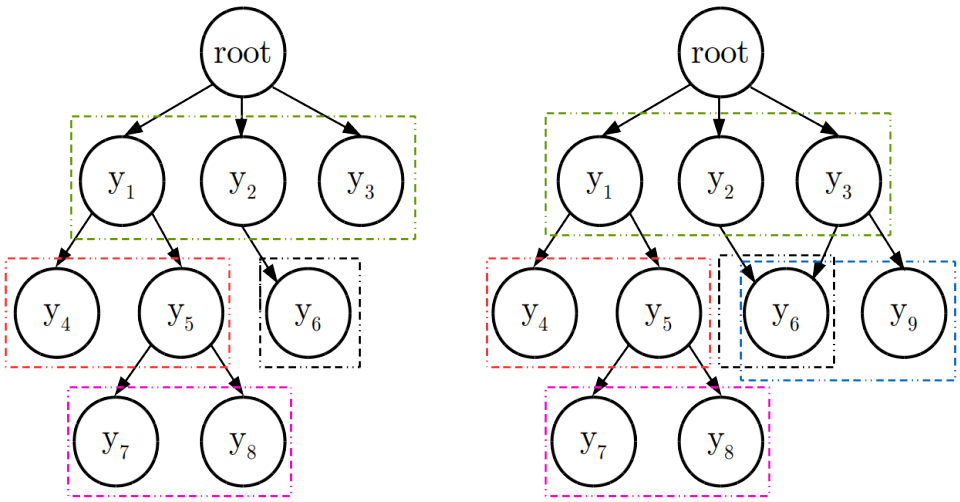


Figure 6: Local Classifier per Parent Node approach. Left a tree; right a DAG. Each box contains the nodes/labels for the multiclass problem of the node that all they are children. For example, nodes $\{y_4, y_5\}$ represent the multiclass problem of node y_1. (Best seen in color)

Therefore, the prediction for an instance is the path generated by the Top-Down procedure.

The Top-Down procedure has a very well known problem, called *error-propagation*, this error occurs when it makes a mistake in some prediction, this implies that all the next predictions will also be wrong. Furthermore, the Top-Down procedure is unable to correct wrong predictions.

3.1.3 Local Classifier per Node (LCN) Approach

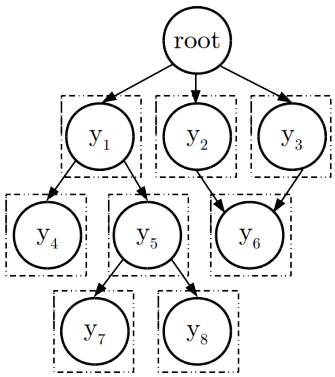


Figure 7: Local classifier per node approach. For each node, except the root, a binary classifier is trained.

In this approach a binary classifier is trained for each node in the hierarchy, except for the root, see Fig. 7. These classifiers will predict whether an instance is associated to the label or not. For each LCN the training set of positive and negative instances has to be selected. Some policies for selecting them have been proposed:

- The *exclusive policy*: for node y_j the positive instances are only those instances for which its most specific label is y_j, and the negatives are the rest of instances.
- The *less exclusive policy*: for a node y_j the positive instances are only those instances which its most specific label is y_j, and the negatives are the rest of instances, except those that are descendents of y_j.
- The *less inclusive policy*: in this case, for a node y_j the positive instances are those associated to y_j (not only instances which its most specific label is y_j), and the negative instances are the rest.
- The *inclusive policy*: for a node y_j the positive instances are those associated to y_j (not only instances which its most specific label is y_j), and the negative instances are the rest except instances that are associated to ancestors of y_j.
- The *siblings policy*: for a node y_j the positive instances are those associated to y_j (not only instances which its most specific label is y_j), and the negative instances are those associated to the siblings of y_j.
- The *exclusive policy*: for node y_j the positive instances are only those instances which its most specific label is y_j, and the negatives are those instances which its most specific label is some sibling of y_j.
- The *balanced Bottom-Up policy*: for a node y_j the positive instances are those associated to y_j, and the negatives are those associated to siblings, then to uncles, and so on, until the amount of negatives instances is equal to the positives.

These policies were proposed by Eisner et al. [2005], Fagni and Sebastiani [2007]. Using those policies for LCN is not mandatory, variants of these policies and new ones are permitted. For example, Feng et al. [2018] use a sibling policy for the nodes, then adds artificial instances to the minority class (positive or negative) to balance the number of instances.

For predicting the path of an instance, the procedure Top-Down is also compatible with this approach. The prediction starts in the root node advancing toward its child node with the highest score, and so on until a leaf node is reached.

3.1.4 Local classifier per level approach

Local classifier per level approach consists in training a multiclass classifier for each level of the hierarchy. Nevertheless, in the prediction phase is important to correct predictions in order of avoid inconsistent paths. Nevertheless, this approach is much less used than the previous approaches.

3.1.5 HC methods

Several methods have been proposed for the different hierarchical classification problems, some approaches that have been researched are: improve standard methods, modify the hierarchy, modeling probabilistically the hierarchy.

CLUS-HMC[1] was proposed by Vens et al. [2008a] for multiple path prediction and predictions can finish on internal nodes. The method creates a decision tree where each leaf contains the probability of each node in the hierarchy. A threshold is required in order to determinate if an instance is associated to a class.

Naik and Rangwala [2016] studied feature selection for each non-leaf node, then they proposed two approaches for choosing relevant number of features at each non-leaf node. First, *Global Feature Selection* where the number of features is the same for all the non-leaf nodes. Second, *Adaptive Feature Selection* that selects different number of features at each node, so, validation sets are required to tested different number of features. However, a Top-Down procedure is used in the prediction phase, hence the problem of error propagation still exists.

Naik and Rangwala [2016] proposed *rewHier*, which applies different operations to modify the hierarchy. The operations are: *Node Creation*, that groups together similar class pairs which have different parents; *Parent-child rewiring* assigns a leaf node from one parent to another parent node in the hierarchy, this when the node is identified to be similar to all the sibling leaf nodes; the last is *Node deletion*, this operation deletes nodes that do not have leaf nodes, this can happen after applying the two first operations. The threshold τ is selected empirically, which is used to determine whether a pair of classes is similar or not, thus, different values for τ will produce different modified hierarchies.

Nakano et al. [2017] proposed a modification of the hierarchy (only for Trees) which consists in replicating internal nodes and then adding them as subclasses of themselves. This method is NMLNP. Two variants were proposed, the first is *non-Leaf Local Classifier per Parent Node* (nLLCPN) which trains LCPN for the modified hierarchy. Then, in the prediction phase to obtain the prediction of an instance a Top-Down procedure is used. The second is *Local Classifier per Parent*

[1] CLUS-HMC is type MPP

Node and Branch, in this variant, LCPN are trained for the modified hierarchy, but in the prediction phase, they use a measure to score paths, which is the average of the scores of the labels in the path, that is, it is equal to Sum of Probabilities of Hernandez et al. [2013]. Panta et al. [2019] analyzed the performances of LCPNB and nLLCPN using eight different base classifiers, where SVM outperformed the rest of classifiers.

Ramírez-Corona et al. [2016] proposed the method *Chained Path Evaluation* (CPE). This method consists in training LCPN combined with the strategy Bayesian Chained Classifiers (BCC), thus, the prediction of each classifier is influenced by the prediction of its parents. This strategy theoretically works for Trees and DAG hierarchies, but due to the use of LCPN in DAG hierarchies, a node would be influenced by predictions of other parents of its siblings, information that could be useless or add noise.

Mukti et al. [2018] generate a hierarchy from a set of classes related to the problem of diabetic retinopathy, after, the Top-Down method is applied. Later, Yamashita et al. [2019] use an approach similar to Mukti et al., that is, they generate a hierarchy from a set of classes, where each level in the hierarchy is in charge of classifying a new instance in a class, so if the prediction is negative then it goes to the next level (similar to a Top-Down procedure). Furthermore, at each level of the hierarchy feature selection and classifier selection are carried out.

Bayesian Networks with Chained Classifiers were proposed by Serrano-Pérez and Sucar [2019], the method trains chained classifiers over independent binary classifiers that fed a Bayesian Network, in this way, the predictions of the nodes are influenced by the previous predictions of their neighbors. The prediction of an instance is the path that maximizes a score. Three variants were proposed, that are different by the neighbors that influence the predictions of the chained classifier HCP-parents, HCA-ancestors and HCC-children.

3.2 Semi-Supervised Hierarchical Classification

This section presents some works related with semi-supervised learning and hierarchical classification that have been proposed. Also a discussion and analysis about them is given, which include an empirical comparison between related works and the research proposal.

The first method for semi-supervised hierarchical classification was proposed by Metz and Freitas [2009]. It is a Top-Down (Tree, SPP, NMLNP) with LCN, where the positive instances of a label are those associated to the label or its descendants, and the negative instances are those associated to its siblings or its siblings descendants. Each label classifier is self-trained following one of three strategies:

self-train A: all the instances in the labeled set are used, *self-train B*: the instances classified as positive in the parent class are used, and *self-train C*: only the instances classified as positive instances during the self-train step on the parent node are used as unlabeled set in the children nodes. It is not really clear the difference between self-train C and self-train B, even worst, they reported double results for self-train B and none for self-train C.

Santos and Canuto [2014a] proposed Hierarchical Multi-label classification using Semi-Supervised Label Powerset (HMC-SSLP). First, a HMC-Label Powerset (HMC-LP) [Cerri et al., 2009] is trained with labeled data, then it is used to label a (predefined) proportion of the unlabeled data which are added to the training set, this process is iterated until all the unlabeled data are labeled. HMC-LP [Cerri et al., 2009] combines all the classes of an example to generate a new hierarchy, nevertheless, examples of how to combine paths of different lengths are not given. On the other hand, the positive instances for each new class could be very few, which can result on unreliable classifiers. Furthermore, if HMC-LP is supposed to be applied with a semi-supervised approach (because there are only few labeled data), the amount of positive instances will be ridiculously few for each new class. So, using the HMC-SSLP for semi-supervised hierarchical classification does not seem a good option in any way.

Furthermore, Santos and Canuto [2014a] proposed Hierarchical Multi-label using Semi-Supervised Random k-Labelsets (HMC-SSRAkEL) which is the semi-supervised version of HMC-RAkEL [Santos and Canuto, 2014a]. This method trains LCPN, that is, for each parent node a RAkEL (multi-label) classifier [Tsoumakas et al., 2011] is trained. Then, a Top-Down procedure is use to label a (predefined) proportion of unlabeled data, which are added to the training set, this process is iterated until all the unlabeled data are labeled.

Hierarchical Multi-label Classification using Semi-Supervised Binary Relevance (HMC-SSBR) was proposed by Santos and Canuto [2014b], which is the semi-supervised version of HMC-BR [Cerri et al., 2009] (a Top-Down method with LCN, siblings policy). However, Santos and Canuto indicate that BR is replaced by SSBR [Santos and Canuto, 2012], a semi-supervised method for *multi-label classification*, that is, each node of the HMC-BR is self-trained and the prediction for each unlabeled data follows the Top-Down procedure.

The Top-Down procedure for labeling instances in HMC-SSRAkEL and HMC-SSBR can predict multiple labels in the *same* level, that is, it advances for different paths if the instance is classified as positive in the following label/node. On the other hand, for HMC-SSRAkEL, HMC-SSBR and HCM-SSLP methods a proportion of unlabeled data to be labeled in each iteration is predefined, the proportions used are 17%, 33% and 50%, which results in 6, 3 and 2 iterations, respectively, however a justification for choosing proportions is not provided. Furthermore, when labels

are predicted for unlabeled data, instead of choosing those with the most confident predictions, all of them are pseudo-labeled and added to the labeled data; that is, these methods lack a way to select the instances with the most confident predictions.

Xiao et al. [2019] proposed Path Cost-Sentive Algorithm with Expectation-Maximization (PCEM) which consists in the following steps, first the base classifier Path Cost-Sentive Naive Bayes Classifier (PCNB) [Xiao et al., 2019] is trained with the labeled data, then pseudo-label all the unlabeled instances and train the PCNB with labeled and pseudo-labeled instances, this is iterated until the parameters of the PCNB converge. PCNB was proposed for hierarchical text classification, where the document representation is a vector called bag-of-word, that is, the number of attributes is equal to the number of words in the corpus, and each cell contains the frequency of the word in the document. Due that PCNB is designed using the bag-of-word representation, it is not straightforward how to apply PCEM in non-text domains.

3.3 Analysis

Table 2 shows a comparison among the state of the art methods and the proposed methods for this research. Note, the proposed methods are different by the type of paths that they predict, the first predicts a single path of labels while the second is able to predict multiple path of labels. However, we highlight the following points:

- **Hierarchical Structure**: The related works were proposed for problems with hierarchy Tree type, which is a limitation of the methods. The proposed method will overcome this limitation and be able to work in any hierarchy (DAG type).
- **Number and depth of paths**: The related works are proposed for a predefined number of paths and depth. In this way, the proposed method (*Proposed SPP*) will be extended to be able to predict multiple paths with partial depth (*Proposed MPP*).
- **Labeling strategy**: Since self-training approach will be used, a labeling strategy to select the unlabeled data with the most confident predictions is required. But this labeling strategy has to consider the information provided by the hierarchy. Note that some related works lack of a labeling strategy.

As can be seen, we would like to address two hierarchical problems with hierarchical structure Directed Acyclic Graph type, that have not been addressed before. Furthermore, we will use a self training approach for the semi-supervised learning but proposing labeling strategies suitable for hierarchical classification.

Table 2: Comparative table of the related works and the proposed methods. HS: hierarchical structure, MD: maximum depth of datasets on which it was tested, HC: hierarchical classifier, LCN: local classifier per node, (N)MLNP: (Non) mandatory leaf node predictions, DAG: directed acyclic graph, N/A: not apply.

Method	HS	Paths	Depth	MD	SSL approach	Base HC	Policy (LCN)	Labeling strategy
Metz and Freitas [2009]	Tree	Single	NMLNP	6	Self training	Top-Down (decision trees)	extended siblings	Threshold plus self-train {A,B,C}
HMC-SSBR [2014]	Tree	Multiple	NMLNP	2	Self training	Top-Down (HMC-BR)	siblings	No
HMC-SSLP [2014]	Tree	Multiple	NMLNP	2	Self training	Top-Down* (HMC-LP)	N/A	No
HMC-SSRAkEL [2014]	Tree	Multiple	NMLNP	2	Self training	Top-Down (HMC-RakEL)	N/A	No
PCEM [2019]	Tree	Single	MLNP	3	Generative Model	PCNB	N/A	N/A
Proposed SPP	Tree/ DAG	Single	MLNP	?	Self training	any/ based on ANN	-	not defined yet
Proposed MPP	Tree/ DAG	Multiple	NMLNP	?	Self training	any/ based on ANN	-	not defined yet

4 Research Proposal

4.1 Motivation

A common problem in supervised classification is lack of data. This may be because hand-labeling data is time consuming and costly or just hard to label [van Engelen and Hoos, 2019]. Hence, training a classifier with few labeled data could produce a unreliable classifier.

This is even more notorious in hierarchical classification, because the data of a node is split among its children, hence, nodes in deeper levels of the hierarchy only have a little fraction of data. So, an alternative way is to use semi supervised learning, that is, use unlabeled data along with the labeled to train a classifier. Moreover, considering that upper nodes contains general information while lower nodes contains specific information, transfer learning (TL) may be applied, that is, upper nodes could share their learned information to the lower ones.

Furthermore, large amounts of information can be obtained from different sources of information, such as the internet. Information such as text, images, videos, etc., is commonly desired, nevertheless, most of that information is unlabeled. Moreover, unlabeled information could be required in scenarios where instances can have associated multiple labels, like hierarchical classification, which makes more challenging make use of unlabeled data. So strategies that take advantage of that information are required.

Hierarchical classification methods have been applied in multiple domains, showing better performance than flat classification (algorithms that do not consider

the hierarchy in any way), some of them are functional genomics [Giunchiglia and Lukasiewicz, 2020, Serrano-Pérez and Sucar, 2019], text classification [Wu and Saito, 2018, Kowsari et al., 2017] and image classification [Sali et al., 2020, Kowsari et al., 2020, Murtaza et al., 2019]. So, we consider that a suitable Semi Supervised Hierarchical Classification algorithm, that consider the hierarchy, trained with labeled and unlabeled data, can produce a hierarchical classifier with better performance than using only the few labeled data.

4.2 Justification

This research will address the following problems related to hierarchical classification:

- **Few labeled data (Hierarchical classification)**: Training classifiers with few labeled data could produce an unreliable classifier, but use unlabeled data together with a suitable semi-supervised hierarchical classifier could help to improve the performance of the classifier.
  - Furthermore, It is well known, that in hierarchical classification, the amount of instances is split from a node to its children, which result in deeper nodes with very few instances.
- **Single and Multiple path prediction**: The unlabeled data is used in a scenario where the instances can be associated to multiple labels. So, keep the most confident pseudo-label instances is necessary, to train a hierarchical classifier.

4.3 Problem Statement

When training a hierarchical classifier with few labeled data, it can result in an unreliable HC. Using unlabeled data could help to improve the performance of the HC. In the literature (see section 2.3) have been proposed several methods that make use of labeled and unlabeled data to perform learning tasks. Nevertheless, most of them were proposed for *flat* classification and applying them directly on hierarchical classification means that the information provided by the hierarchy is ignored.

Hence, a suitable semi-supervised hierarchical classifier (SSHC) is required, that is, a SSHC that pseudo-labels unlabeled instances and selects the best of them to retrain itself, in order to get a better performance than the HC trained only in the labeled data.

Formally: let $X = \{x_1, x_2, ..., x_n\}$ and $U = \{x_{n+1}, x_{n+2}, ..., x_{n+m}\}$ be instances sets where $x_i \in \mathbb{R}^d$, that is, each instances x_i is described by a vector of d attributes, let $|L|$ be the number of labels, let $Y = \{y_1, y_2, ..., y_n\}$ be the set of labels for X,

where $y_i \in \{0,1\}^{|L|}$, that is, each $y_{i,j}$ indicates if the i-th instance is associated to the j-th label, in this way (X, Y) is a labeled set and U is an unlabeled set. Additionally, let $HS = (L, E)$ be the Directed Acyclic Graph (DAG) that represents the hierarchy, where L is the set of nodes and E is the set of edges that link the labels.

Therefore, from (X, Y, U, HS) is required a classifier to predict labels for new instances: $f_{SSHC} : \mathbb{R}^d \to \{0,1\}^{|L|}$, and whose performance is better than the classifier trained only with the labeled data (X, Y, HS): $f_{HC} : \mathbb{R}^d \to \{0,1\}^{|L|}$, that is:

$$performance(f_{SSHC}) > performance(f_{HC}) \tag{12}$$

4.4 Research Questions

- Will training a semi supervised hierarchical classifier (SSHC) with unlabeled and few labeled data produce a classifier with better performance than the hierarchical classifier (HC) trained only in the few labeled data?
- Will transfer learning help to improve the performance of nodes by taking into account the information learned from their ancestors?
- Will pseudo-labeling instances with multiple paths of labels produce a SSHC (that can predict multiple paths of label) with better performance than pseudo-labeling instances with a single path of labels?

4.5 Hypothesis

Training a semi supervised hierarchical classifier (SSHC), that uses as base hierarchical classifier to HC, with unlabeled and labeled data will produce a classifier with better performance than training the hierarchical classifier HC only on labeled data.

4.6 Objectives

Propose a semi-supervised hierarchical classifier for hierarchical multi label classification, that can be trained with labeled and unlabeled data; whose performance, in tree hierarchies, is competitive with state of the art methods, and in DAG[2] hierarchies its performance is better than the supervised classifier trained on labeled data.

[2]We could not find any previous work on semi-supervised hierarchical classification for DAG hierarchies.

4.7 Specific objectives

- Propose a *methodology* for semi supervised hierarchical classifiers based on self training approach.
- Propose a strategy for labeling unlabeled data considering the hierarchy, while keeping the best ones for training.
- Propose a SSHC based on the proposed methodology.
- Incorporate transfer learning between neighboring nodes.
- Extend the SSHC method to Multiple Paths Prediction.

4.8 Scope and Limitations

In hierarchical classification there are different hierarchical classification problems (description and examples can be found in section 2.5.1), however, the problems to cover in this research are the following:

- Problems with Tree/DAG type hierarchical structure, where instances are associated to a single path of labels, and the paths always reach a leaf node (full depth).
- Problems with Tree/DAG type hierarchical structure, where instances are associated to multiple paths of labels, and the paths can finish in internal nodes (partial depth).

On the other hand, there are multiple approaches in semi supervised learning for *flat* classification. Nevertheless, most of them have not been explored/researched in hierarchical classification. Hence, this research is focused in the self training approach.

4.9 Expected Contributions

The expected contributions are:

- A methodology for semi supervised hierarchical classification, independent of any base hierarchical classifier.
- A SSHC method which considers the hierarchy when the model is built.
- A SSHC method that can handle hierarchies of tree and DAG type, and is able to predict single and multiple paths of labels.

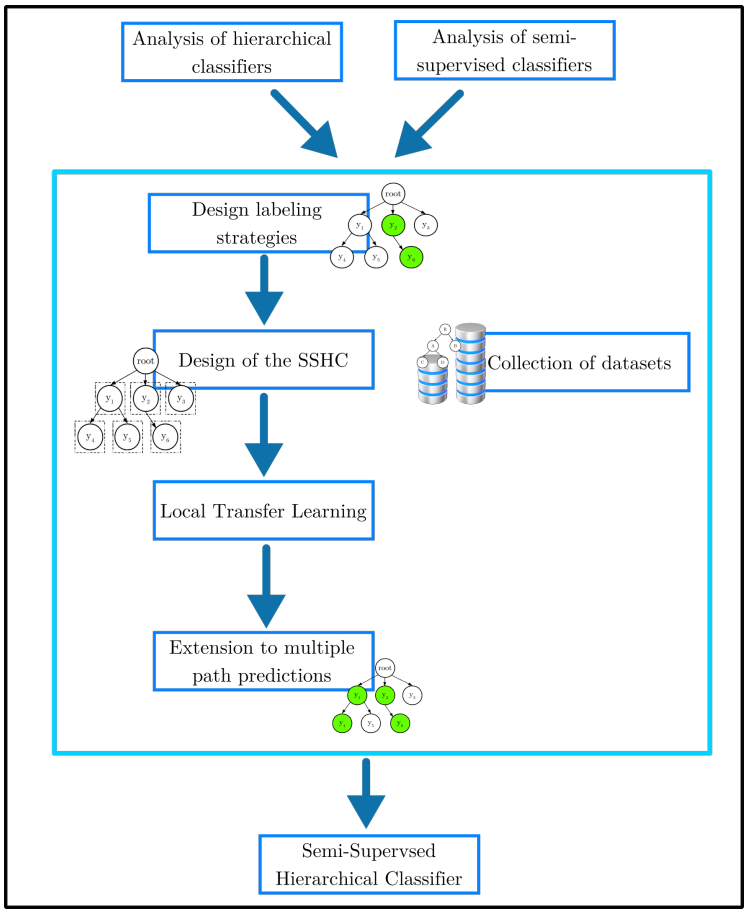


Figure 8: Diagram of the research methodology. An analysis of hierarchical classifiers will be carried out, also an analysis of semi-supervised classifiers. Later, labeling strategies to pseudo-label unlabeled data will be designed. Then, a semi-supervised hierarchical classifier (SSHC) will be designed taking into account the previous analysis and incorporating the labeling strategy. The proposed SSHC will be evaluated with a collection of datasets. Transfer learning strategies will be incorporated to the SSHC taking into account the hierarchy in which the labels are arranged. Finally, the proposed SSHC will be extended to predict multiple paths of labels.

4.10 Methodology

The methodology is composed by the points below, additionally, in Fig. 8 is shown the diagram.

1. **Analysis of hierarchical classifiers**: Analysis of the hierarchical classifier (HC) is important, due that HC are proposed for a special type of hierarchical problems, so the selection of those methods that try to solve hierarchical problems equal to the addressed by this proposal is required.

Furthermore, there are methods that used independent local classifiers, such as those based on the approaches LCN and LCPN, while others builds a global classifier.

2. **Analysis of semi-supervised classifiers**: In hierarchical classification only a few semi-supervised methods have been proposed, but for *flat* classification several methods and approaches have been proposed [van Engelen and Hoos, 2019]. The analysis of those techniques can help to overcome the situation where few labeled data is available in HC. Approaches such as *self-training* and *co-training* are of greatest interest, because they pseudo-label the unlabeled data and use them to retrain a model, so it keeps learning until there is no unlabeled data.

3. **Creation and collection of datasets**: Datasets will be collected to evaluate the performance of the SSHC in real world datasets. Furthermore, artificial datasets (AD) will be generated because they are a *good* option to extend the evaluation and analysis of methods.

    - *Design and creation of artificial datasets*: They are useful to show the behaviour of classifier under certain conditions. So, in order to extend the evaluation and analysis of methods, some AD will be generated, Serrano-Pérez and Sucar [2021] proposed a method to generate AD for hierarchical classification, hence we could take advantage of that method to generate labeled and unlabeled data, so both sets would be generated from the same distribution, and the amount of data would not be a limitation.
    - *Collection of datasets*: Real world datasets that will be used to evaluate the performance of the proposed SSHC. Currently, we have access to two sets of challenging hierarchical datasets from the field of functional genomics [Vens et al., 2008b]:
        - Functional Catalogue (FunCat)
        - Gene Ontology (GO)

4. **Design of labeling strategies**: Labeling strategies are required in order to pseudo-label and select the most confident unlabeled instances for re-training the hierarchical classifier. The labeling strategies have to be focused on building *paths of labels* instead of a fully local pseudo label process. Pseudo-labeling an unlabeled instance with a path of labels assures consistence when the classifier is trained, because the hierarchical constrain is satisfied.

5. **Design the SSHC**: The design of the SSHC has to be based on the semi-supervised learning approach selected, as well as it has to consider the hierarchy when is trained. Furthermore, it incorporates the labeling strategy designed in step 4.

In this stage, the SSHC can predict only a single path of labels. Initially, it can handle tree hierarchies, which will be later extended to handle any hierarchy of directed acyclic graph type.

6. **Incorporate transfer learning**: Transfer learning is a set of techniques that could help the model to converge faster. Two different strategies are considered:

    - In hierarchical classification, the upper nodes represent general information, while the lower nodes represent specific information. In this way, the information learned by a upper node may be shared to the lower ones, for instance, to its children.
    - Taking into account that the classifier will iterate, the trained model could be re-trained in the next iteration, instead of training a new hierarchical classifier from scratch.

7. **Extension of the SSHC to multiple path predictions**: Initially, the method predicts a single path of labels, hence the SSHC will be extended to be able to predict multiple path of labels.

Furthermore, evaluation measures such as *accuracy*, *hierarchical precision* (hP), *hierarchical recall* (hR) and *hierarchical F-measure* (hF) will be used to evaluate the performance of the proposed method, details of the measures are found in section 2.6.

4.11 Schedule

In figure 9 is depicted the schedule of activities for the development of this research.

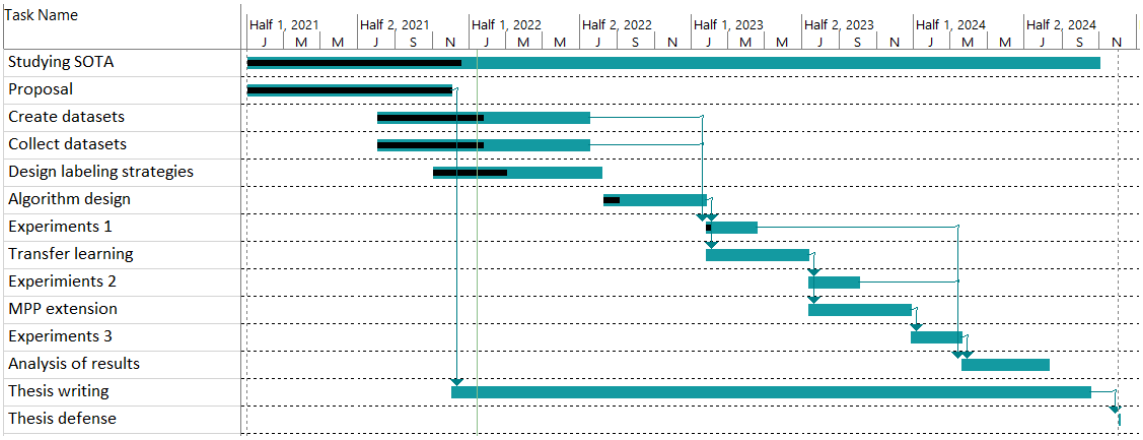


Figure 9: Gantt chart of the work plan.

4.12 Publications Plan

- *Conference paper*: the manuscript "Semi-Supervised Learning for Hierarchical classification" (tentative name) will be written and submitted to "International Conference on Machine Learning" (2022).
- *Conference paper*: the manuscript "Semi-Supervised Hierarchical Classification with Transfer Learning" (tentative name) will be submitted to "European Conference on Artificial Intelligence" (2023).
- *Journal paper*: The article "Semi-Supervised Hierarchical Classification" (tentative name) will be written and submitted to the journal "Data Mining and Knowledge Discovery" (2024).

5 Preliminary Results

Finally, a summary of results is presented in this section.

5.1 A *methodology* for Semi-Supervised Hierarchical Classification

The steps that we are considering for proposing a SSHC (so far) are the following:

1. Select a hierarchical classifier (HC).
2. The HC is trained with labeled data.
3. Predict *positive* probabilities (scores) for each node for the unlabeled data.
4. Select the unlabeled instances with the most confident predictions and add them to training set. (labeling strategies.)
5. The HC is trained with labeled and pseudo-labeled data.
6. If all unlabeled data was pseudo-labeled or maximum number of iteration was reached, finish, else go to step 3.

In order to select the instances with the most confident probabilities *labeling strategies* are required, which should consider the hierarchy. An example of a labeling strategy is proposed in section 5.2.2.

5.2 Semi-Supervised Hierarchical Classifier Based on Local Information (SSHC-BLI)

A preliminary version of the semi-supervised hierarchical classifier has been developed. This method is mainly based on the smoothness assumption, that is, neighboring instances must have the same (or similar) paths of labels. SSHC-BLI begins building pseudo-labels for each unlabeled instances using its neighboring labeled instances, but if the unlabeled instance is not similar to its labeled neighbors, it stays unlabeled; this process iterates until the all the pseudo-labels do not change.

Algorithm 1 shows the general steps. It is an iterative method where the unlabeled data is pseudo-labeled using its k-nearest neighbors (lines 6 - 8), details of how pseudo-labels are built (line 8) are shown in subsection 5.2.2. Also, the similitude of the unlabeled point with its neighbors (line 10) is considered, if they are not *similar* the unlabeled point stays unlabeled, details of how the similitude is estimated are shown in subsection 5.2.3. The method finishes when the pseudo-labels for the unlabeled data did not change from an iteration to other, or whether the maximum number of iterations was reached.

5.2.1 Variants

Variant 1: It correspond to the version described in Algorithm 1.

Variant 2: Due that in each iteration the pseudo-labels for the whole unlabeled set are re-estimated, after the first iteration an instance u_j that was added to training set, it will have as one of the k-nearest neighbors to itself (with distance zero). This results in a new pseudo-label *biased*, because the instance is contributing to itself. In order to avoid this, the function *getKNN* (line 6) is modified so that it guarantees that none of k-nearest neighbors is the instance itself.

Variant 3: Considering that the number of instances in the training set could increase in each iteration, it may be interesting increase the number of nearest neighbors. In this way, from variant 2, variant 3 allows to increase the value of k after a predefined number of iterations.

5.2.2 Pseudo-label an instance

A way to pseudo-label instances is required. So for building a pseudo label for an unlabeled instance, labeled instances close to it are required.

Let $Y = [y_1, ..., y_k]$ be the labels of k instances close to the unlabeled instance x, where $y_i \in \{0, 1\}^{|L|}$, that is, $y_{i,j}$ is 1 if the i-th instance is associated to the j-th

Algorithm 1 Algorithm of SSHC-BLI

Require: (X, Y): labeled data, U: unlabeled data, k: number of nearest neighbors, THR: similitude threshold, $t2label$: threshold to pseudo-label an instance, HS hierarchy, $maxIterations$: maximum number of iteratios.

Ensure: f_{sshc}: semi supervised hirarchical classifier

$T \leftarrow 1$ ▷ Iteration
$LD \leftarrow X$ ▷ LD: Labeled data
$CL \leftarrow L$ ▷ Labels of labeled data
while $True$ **do**
  for each $u_j \in U$ **do**
    $IND_j \leftarrow getKNN(k, u_j, LD)$ ▷ Get k-nearest neighbors
    $DI_j \leftarrow distancesKNN(IND_j, u_j, LD)$ ▷ Get distances to the knn
    $PSL_j \leftarrow getPseudoLabel(IND_j, LD, t2label)$ ▷ Pseudo label for u_j
  for each $u_j \in U$ with valid PSL_j **do**
    if $similitude(u_j, IND_j) < THR$ **then**
      $PSL_j = \emptyset$ ▷ Invalid pseudo-label
  if $(T > maxIterations)$ or $(PSL^T == PSL^{T-1})$ **then**
    break cycle (while)
  else ▷ join labeled data with valid pseudo-labeled data
    $CL \leftarrow Y \cup valid(PSL)$
    $LD \leftarrow X \cup U[valid(PSL)]$
  $T \leftarrow T + 1$
$f_{SSHC} \leftarrow trainHC(LD, CL, HS)$ ▷ Train a hierarchical classifier

label, 0 otherwise. So the probabilities for each individual label can be estimated in the following way:

$$ppsl = \left[\frac{\sum_{i=1}^{k} y_{i,1}}{k}, \frac{\sum_{i=1}^{k} y_{i,2}}{k}, ..., \frac{\sum_{i=1}^{k} y_{i,|L|}}{k} \right] \tag{13}$$

Then a threshold ($t2label$) is used to determine if an instance is associated to the label:

$$psl_j = \begin{cases} 1 & ppsl_j >= t2label \\ 0 & ppsl_j < t2label \end{cases}, \forall j \in L \tag{14}$$

$$1 >= t2label >= 0 \tag{15}$$

Finally, psl is the pseudo label for x, but if psl is full of *zeros*, then it is a invalid path, therefore, x is still unlabeled.

Fig. 10 shows an example of how a pseudo label for an instance is built. Labels of the labeled instances in vector representation are required, which are used to calculate $ppsl$, that contains the probability of each label, then a threshold is

applied to generate the pseudo-label. 3 thresholds are applied that generate different pseudo labels, in this example, the threshold 1 generate an invalid pseudo-label, so the instance is still unlabeled.

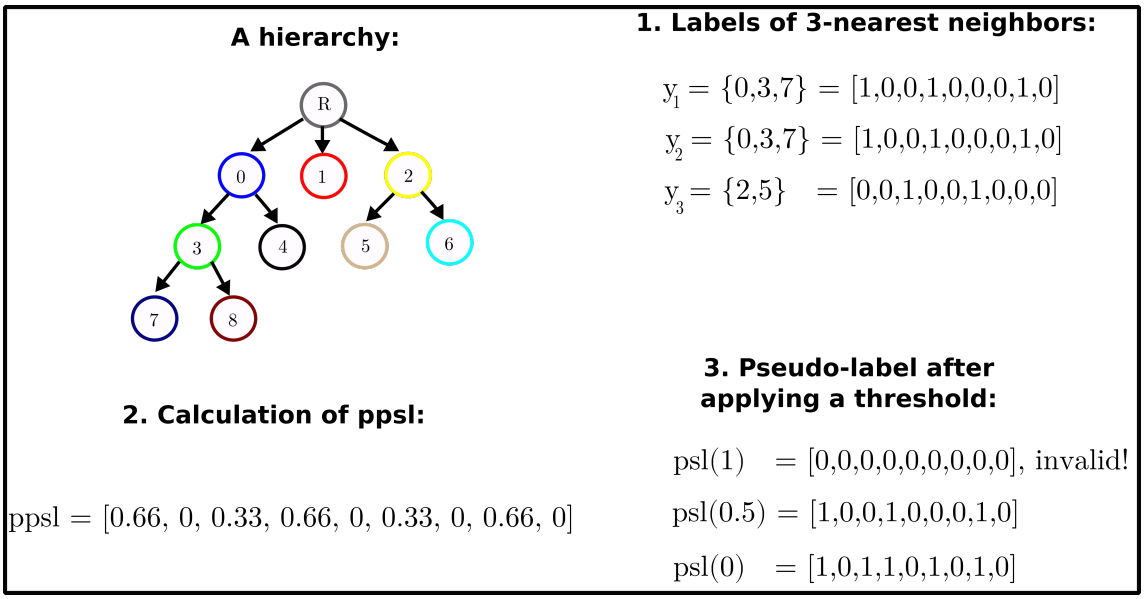


Figure 10: Example of how pseudo-label an instance. 1. Labels of the nearest neighbors are required in vector representation, 2. *ppsl* is calculated from the labeled instances, 3. A threshold is applied to *ppsl* to obtain the pseudo-label (*psl*), results of three thresholds are shown.

5.2.3 *SISI*: Similarity of an instance with a set of instances

In this work, a *similarity* function to know how similar or how close is an instance to a small set of instances is required. Also, it is required that the result is in the interval $[0, 1]$, that is, 1 if the instance is similar to the set of instances, or 0 in the opposite case. Nevertheless, it is not known[3] a *similarity* function that complies the previous requisites by the author.

The heuristic function *Similarity of an Instance with a Set of Instances* (**SISI**) is proposed in this work. This is a *local* measure, because it does not consider the complete data distribution, but just the instance of interest and the set of instances.

SISI takes into account the distances among the set of instances A, $lavg$, and the distances of an instance p with the instances of set A, $uavg$. Both equation are

[3]Even though, there is the Mahalanobis distance, it is focus on the distance between a point and a distribution. Furthermore, this measure does not comply with the interval result, $[0, 1]$.

shown below:

$$lavg = \frac{\sum_{i=1}^{k} \sum_{j=i+1}^{k} d(x_i, x_j)}{\frac{k(k-1)}{2}} \tag{16}$$

$$uavg = \frac{\sum_{i=1}^{k} d(p, x_i)}{k} \tag{17}$$

where $x_i \in A$, k is the length of A and $d(A, B)$ is any metric (see Appendix A). Additionally, two assumptions are made:

Assumption 1: It is assumed that if $uavg$ is equal or lower than $lavg$, the instance p is similar to the set of instances A with probability 1.

Assumption 2: It is assumed that if $uavg$ is greater than n times $lavg$, with $n > 1$, the instance p is not similar to the set of instances A, that is, probability 0.

Hence, from Assumptions 1 and 2, the equation of the line that passes through points $(lavg, 1)$ and $(n * lavg, 0)$ is defined as:

$$score = \frac{lavg - uavg}{(n-1)lavg} + 1 \tag{18}$$

That is, equation 18 scores the similitude of the point p with the set of instances A in interval $(lavg, n * lavg)$.

Finally, from assumption 1,2 and equation 18, function *SISI* is defined as follow:

$$SISI = \begin{cases} 1 & uavg <= lavg \\ 0 & uavg >= n * lavg \\ \frac{lavg - uavg}{(n-1)lavg} + 1 & otherwise \end{cases} \tag{19}$$

$$n > 1 \tag{20}$$

The general behavior of SISI is shown in Fig. 11. However, SISI still requires a parameter, n. For simplicity it was set to 3, therefore, the experiments shown in this work were made with this configuration.

5.3 Datasets

Artificial and real world datasets were used to evaluate the performance of the proposed SSHC.

5.3.1 Artificial Datasets

An artificial dataset was generated in order to show the behavior of the different methods in a simple case. The hierarchy is tree type and the instances are associated

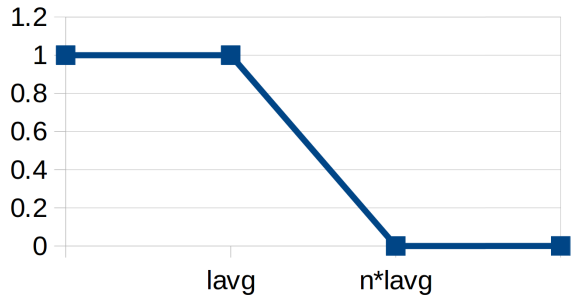


Figure 11: General behavior of function similitude of an instance with a set of instances, SISI. If $uavg$ is lower or equal than $lavg$ then SISI returns 1, but if $uavg$ is greater or equal than $n * lavg$ then SISI returns 0, else, a score is calculated with equation 19.

Table 3: Description of artificial datasets. MD: maximum depth.

Dataset	#Labeled	#Unlabeled	#Test	#Attr.	#Nodes	MD
HAD_01	24	600	300	2	9	3

to a single path of labels which always reach a leaf node, that is, it is a hierarchical problem type (Tree, SPL, FD). The instances were sampled from half moons[4], that is, each leaf node has associated a half moon. Description of dataset is shown in Table 3, the hierarchy is the same as the shown in Fig. 10. In this case, the unlabeled data is initially separated from the labeled.

5.3.2 Real world datasets

A subset of real world datasets from the field of functional genomics are used. The datasets are a subset of the Functional Catalogue (FunCat) Vens et al. [2008b], their hierarchies are trees, the instances are associated to a single path of labels which always reach a leaf node, that is, they are hierarchical problems type (Tree, SPL, FD). Description of datasets is shown in Table 4.

5.4 Results SSHC-BLI

The three variants of the SSHC-BLI were applied to the HAD_02 dataset, the parameters for each variant are shown in Table 5. The hierarchical base classifier is Top-Down, that trains support vector machines (regularization parameter: 1000, kernel:rbf, gamma=$1/n\_features$) for each node, and the *less inclusive* policy is used to select positive and negative instances in each node.

[4] scikit-learn make_moons

Table 4: Description of FunCat datasets. MD: maximum depth.

Dataset	Instances	Attr.	Classes	MD
cellcycle_FUN	2339	77	36	4
derisi_FUN	2381	63	37	4
eisen_FUN	1681	79	25	3
gasch1_FUN	2346	173	36	4
gasch2_FUN	2356	52	36	4

Table 5: Parameters of the SSHC-BLI variants. Note, the parameters *k, ST* and *t2label* are the same for the three variants. THR: similitude threshold, *t2label*: threshold to positively label an instances, MI: maximum number of iterations.

Variant	k	ST	t2label	MI	Increase k
V1				∞	False
V2	3	0.5	0.5	10	False
V3				∞	each 5 iter.

Fig. 12 shows how the variants pseudo-label the unlabeled data. In the first iteration all of them pseudo-label the unlabeled data in the same way, as shown in Fig. 12 a). Afterwards, variant 1 pseudo-labeled the complete unlabeled data, nevertheless, most of wrongly pseudo-labeled instances are found at the ends of the half moons (circled in blue), as can be seen in Fig. 12 b).

Variant 2 got better results at the ends of the half moons (circle in blue), see Fig. 12 c), but still has some issues. Also, points that were pseudo-labeled in the first iteration finished unlabeled, because only neighbors are considered (avoiding that an instance contribute itself to maintain its pseudo-label), also there are unlabeled points that are surrounded by points with the same labels, so its natural to think that they should have the same set of labels, nevertheless, they were not pseudo-label because the estimation of similitude with its k-nearest neighbors.

Finally, variant 3 seems to smooth the results obtained by the 2nd variant, see Fig. 12 d).

The pseudo-labeled instances are used to train a hierarchical classifier. So results of the SSHC with its different variants are shown in Table 6. The column TD correspond to the hierarchical classifier, Top-Down, trained only on labeled data, while the variants of the SSHC are trained with labeled and pseudo-labeled data. As can be seen, using *KNN* strategy to pseudo-label instances helped to improve the performance of the hierarchical classifier. Additionally, the variants 2 and 3 tend to outperform the other one.

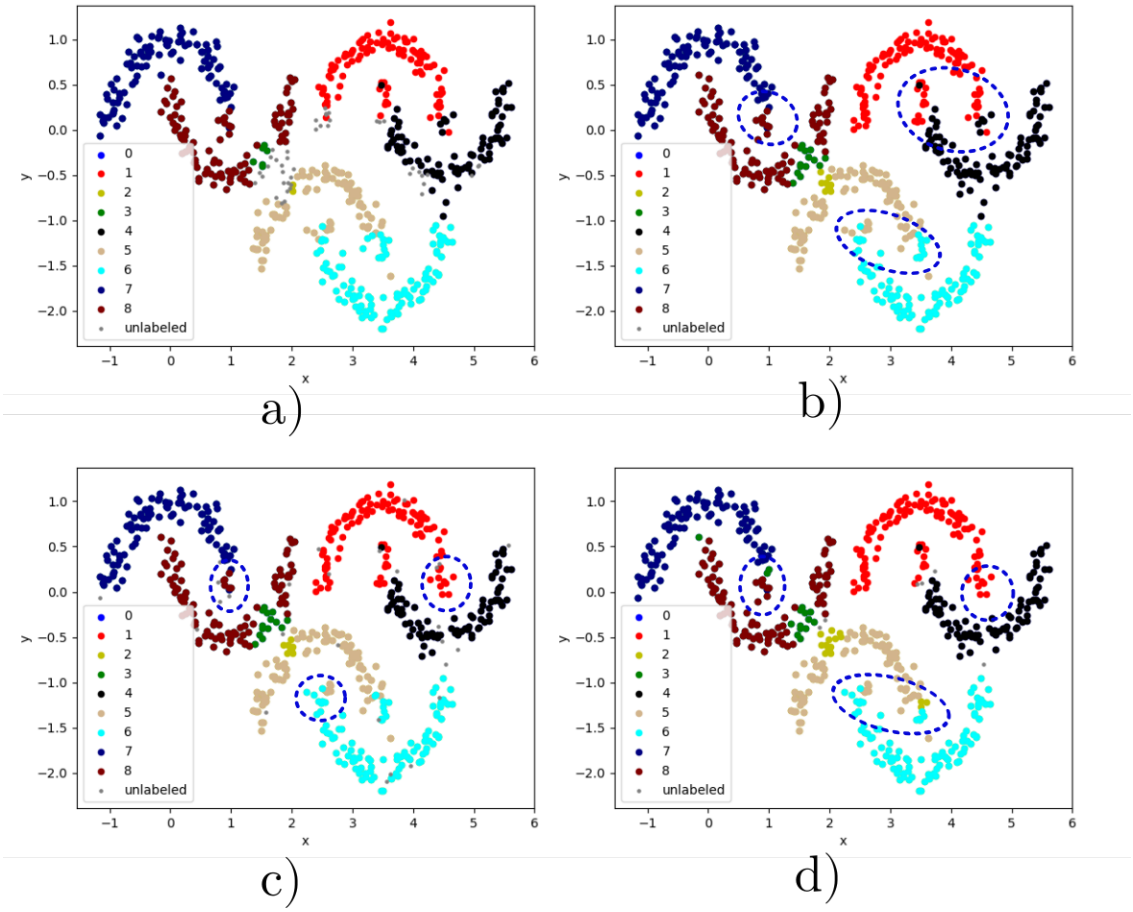


Figure 12: Pseudo-labels for HA_02 datasets. a) Pseudo labels at first iteration for all variants. b) pseudo labels of variant 1. c) pseudo-labels of variant 2. d) pseudo-labels of variant 3. (Best seen in color)

5.4.1 Results in real world datasets

The SSHC-BLI was applied to the real world datasets, the parameters its variants are shown in Table 7. The hierarchical base classifier is Top-Down, that trains a random forest classifier (number of trees: 100, criterion: *gini*, bootstrap: *True*) for each node, and the *balanced bottom-up* policy is used to select positive and negative instances in each node.

Furthermore, the datasets were stratified split in the following way:

- Test: 20%
- Training (80%), which also was stratified split in labeled and unlabeled:

Table 6: Results of SSHC-BLI (variants 1,2 and 3) and the supervised classifier Top-Down (TD) in artificial dataset HA_02. In bold the best score.

	TD	V1	V2	V3
Exact Match	0.5000	**0.9233**	**0.9233**	0.9200
Accuracy	0.4662	0.8928	**0.8951**	0.8930
h Recall	0.7692	**0.9492**	0.9477	**0.9492**
h Precision	0.7082	0.9551	0.9595	**0.9596**
hF	0.7375	0.9522	0.9536	**0.9544**

Table 7: Parameters of the SSHC-BLI variants. Note, the parameters *k, THR, t2label* and *MI* are the same for the three variants. THR: similitude threshold, *t2label*: threshold to positively label an instances, MI: maximum number of iterations.

Variant	k	THR	t2label	MI	Increase k
V1					False
V2	3	0.3, 0.5, 0.7	0.5	45	False
V3					each 10 iter.

- Labeled: {10, 30, 50, 70, 90}%
- Unlabeled: {90, 70, 50, 30, 10}%, complement with respect to labeled.

The division of training was randomly carried out 3 times, so results are the averages of 3 executions.

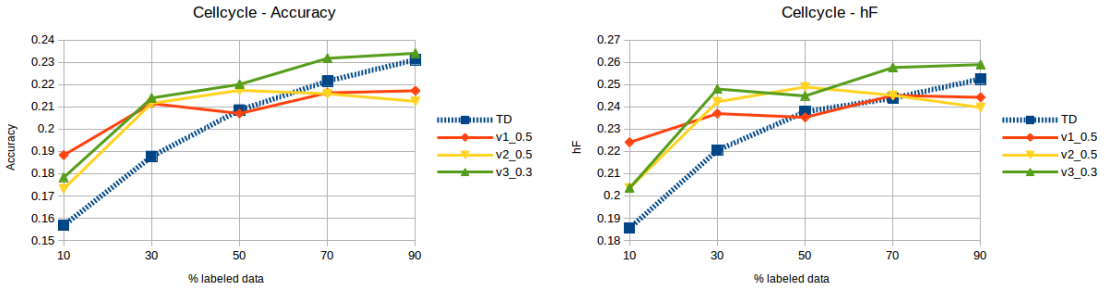
The similitude threshold varies in each variant of the SSHC-BLI, so, for each variant the threshold results in the highest average rank among the datasets is selected. That is, for variant 1 $THR = 0.5$, variant 2 $THR = 0.5$ and variant 3 $THR = 0.3$.

The results in terms of accuracy and hF for the datasets cellcycle, derisi, eisen, gasch1 and gasch2 are shown in Figs. 13, 14, 15, 16 and 17, respectively. Each graph compares the three variants of SSHC-BLI with the baseline, TD, that considers only labeled data, for different percentages of labeled /unlabeled data.

Table 8 summarizes the results of the SSHC-BLI variants and the supervised classifier, that is, it presents the average rank of each classifier over the datasets.

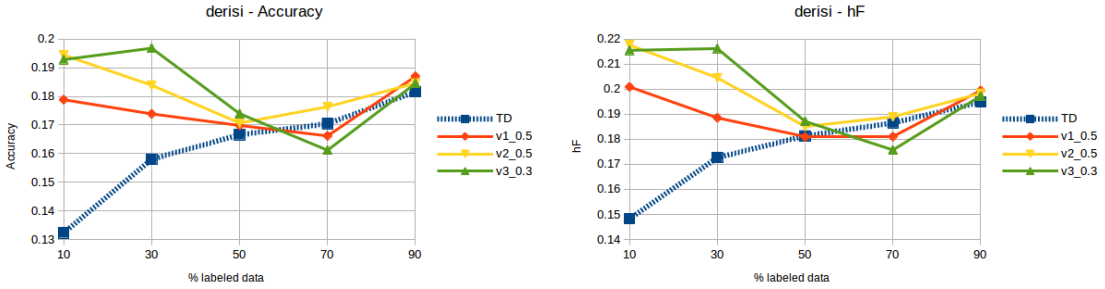
5.4.2 Statistical Comparison

Demšar [2006] recommends the Wilcoxon signed-rank test when the performances of



(a) Accuracy (b) hF

Figure 13: Results for cellcycle dataset. (Best seen in color)



(a) Accuracy (b) hF

Figure 14: Results for derisi dataset. (Best seen in color)

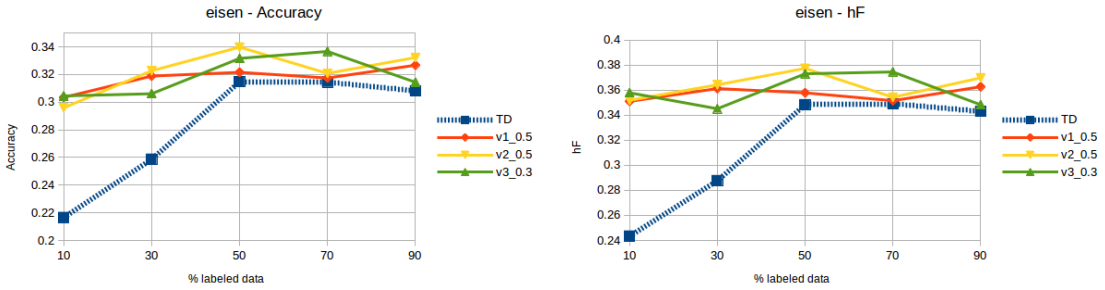
two classifier are compared. In this case, the performance of each SSHC-BLI variant is compared against the supervised classifier TD.

The null hypothesis (one-side) states that the median of the performance differences is negative, against the alternative that states it is positive. 25 real world datasets (5 datasets x 5 divisions) were considered, and the confidence level was set to $\alpha = 0.05$. Therefore, the results are the following:

- For all variants (variant 1 ($THR = 0.5$), variant 2 ($THR = 0.5$), variant 3 ($THR = 0.3$)): the null hypothesis can be rejected in favor of the alternative, that is, the median is greater than zero (positive) for both evaluation measures, accuracy and hF.

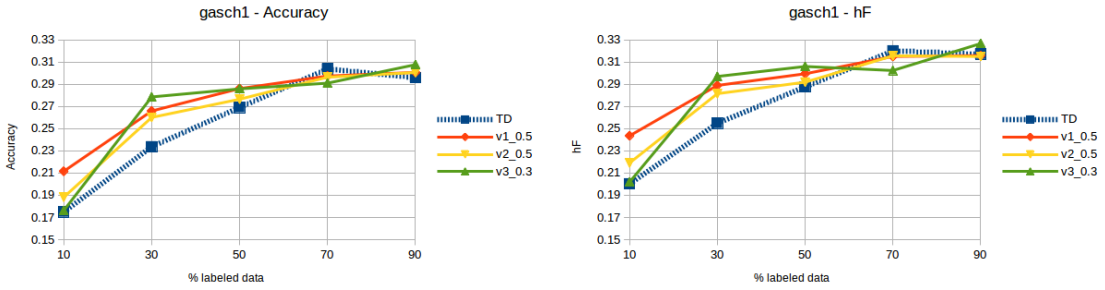
5.4.3 Discussion

The previous experiments show that using unlabeled information in a hierarchical classification scenario can help to improve the performance of a hierarchical classifier trained only on labeled data.



(a) Accuracy (b) hF

Figure 15: Results for eisen dataset. (Best seen in color)



(a) Accuracy (b) hF

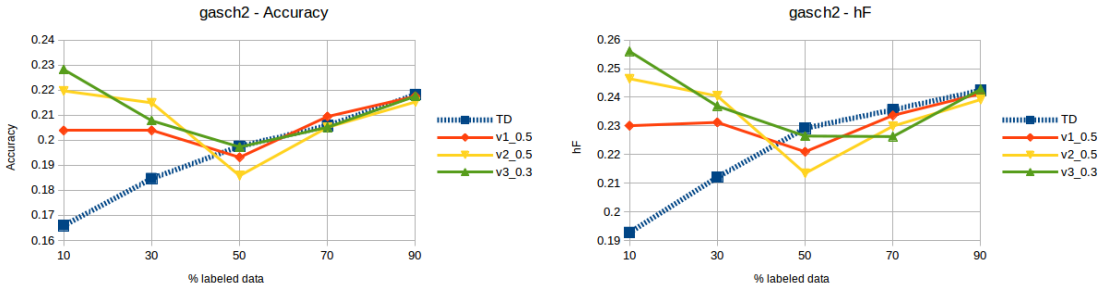
Figure 16: Results for gasch1 dataset. (Best seen in color)

The amount of labeled and unlabeled information vary from 10% to 90% which is useful to see the behavior of the semi-supervised classifier with different amount of information. As can be seen in the result graphs, the SSHC-BLI variants tend to outperform the baseline when there is few labeled data. However, as the labeled information increased and the unlabeled decreased, the performance difference among the SSHC-BLI and the supervised tends to reduce, which is a expected behavior.

Finally, a statistical comparison was carried out using the Wilcoxon signed-rank test, where is shown that the SSHC-BLI variants get better performance than the supervised classifier TD with a confidence level of 0.05.

6 Conclusions

In this work the research proposal was presented. The aim of this research is to develop a semi-supervised hierarchical classifier (SSHC), which can be trained with labeled and unlabeled data.



(a) Accuracy (b) hF

Figure 17: Results for gasch2 dataset. (Best seen in color)

Table 8: Average rank of each classifier in the FunCat datasets. In bold the best (lower is better).

	TD	**V1(0.5)**	**V2(0.5)**	**V3(0.3)**
Acuracy	3.28	2.4	2.4	**2.14**
hF	3.24	2.64	2.16	**1.96**

Currently, large amounts of information can be obtained from the internet, such as images, text and videos, nevertheless, that information is unlabeled. Moreover, methods that take advantage of unlabeled information in a hierarchical contexts are scarce. Furthermore, hierarchical classification naturally suffers of scarce data, mainly in deeper nodes of the hierarchy, because the amount of data is split from a node into its children, which result in deeper nodes with few labeled.

Consequently, the SSHC will be useful in scenarios where the labels (classes) are arranged in a hierarchy, and there is few labeled data but unlabeled information is available. The final version of the SSHC is expected to be capable of handling tree and DAG type hierarchies, as well as be capable of predicting single and multiple paths of labels.

Until now, we have been working on the first three specific objectives. First, a *initial* methodology for semi-supervised hierarchical classification was proposed, which is based on self-training approach. Second, a strategy to pseudo label and keep the most confident instances was proposed (section 5.2.2). And third, the semi-supervsied classifier based on the local information (SSHC-BLI; section 5.2) was developed, which is based in the methodology and makes use of the proposed labeling strategy.

Experiments were conducted on artificial and real world datasets and, where the results obtained by the SSHC-BLI are promising, because in the artificial dataset the baseline (hierarchical classifier trained on labeled data) was outperformed. Fur-

thermore, the SSHC-BLI tends to outperform the baseline in the real world datasets, even with statistical significance.

The immediate future work will be focused in three points: first, extend the SSHC-BLI to be able to handle any hierarchy of directed acyclic graph type; second, generation of artificial datasets to extended the analysis of the semi-supervised methods; and third, write a manuscript to report the obtained results so far, and submit it to a conference.

References

C. C. Aggarwal. *Data Classification: Algorithms and Applications*. Chapman & Hall/CRC, 1st edition, 2014. ISBN 1466586745.

K. P. Bennett and A. Demiriz. Semi-supervised support vector machines. In *Proceedings of the 11th International Conference on Neural Information Processing Systems*, NIPS'98, page 368–374, Cambridge, MA, USA, 1998. MIT Press.

K. P. Bennett, A. Demiriz, and R. Maclin. Exploiting unlabeled data in ensemble methods. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, page 289–296, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 158113567X. doi: 10.1145/775047.775090. URL https://doi.org/10.1145/775047.775090.

W. Bi and J. T. Kwok. Multi-label classification on tree- and dag-structured hierarchies. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 17–24, USA, 2011. Omnipress. ISBN 978-1-4503-0619-5. URL http://dl.acm.org/citation.cfm?id=3104482.3104485.

C. Bielza, G. Li, and P. Larrañaga. Multi-dimensional classification with bayesian networks. *International Journal of Approximate Reasoning*, 52(6):705 – 727, 2011. ISSN 0888-613X. doi: https://doi.org/10.1016/j.ijar.2011.01.007. URL http://www.sciencedirect.com/science/article/pii/S0888613X11000314.

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, page 92–100, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 1581130570. doi: 10.1145/279943.279962. URL https://doi.org/10.1145/279943.279962.

R. Cerri, A. de Carvalho, and A. F. Comparing local and global hierarchical multi-label classification methods using decision trees. 01 2009.

R. Cerri, R. C. Barros, and A. C. de Carvalho. Hierarchical multi-label classification using local neural networks. *Journal of Computer and System Sciences*, 80(1):39 – 56, 2014. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2013.03.007. URL http://www.sciencedirect.com/science/article/pii/S0022000013000718.

R. Cerri, R. C. Barros, A. C. P. L. F. de Carvalho, and Y. Jin. Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC Bioinformatics*, 17(1):373, Sep 2016. ISSN 1471-2105. doi: 10.1186/s12859-016-1232-1. URL https://doi.org/10.1186/s12859-016-1232-1.

O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010. ISBN 0262514125.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16 (1):321–357, jun 2002. ISSN 1076-9757.

J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248547.1248548.

S. Ding, Z. Zhu, and X. Zhang. An overview on semi-supervised support vector machine. *Neural Comput. Appl.*, 28(5):969–978, May 2017. ISSN 0941-0643. doi: 10.1007/s00521-015-2113-7. URL https://doi.org/10.1007/s00521-015-2113-7.

F. d'Alché-Buc, Y. Grandvalet, and C. Ambroise. Semi-supervised marginboost. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, volume 14, page 553–560, Cambridge, MA, USA, 2002. MIT Press. URL https://proceedings.neurips.cc/paper/2001/file/931af583573227f0220bc568c65ce104-Paper.pdf.

J. Du, C. X. Ling, and Z. Zhou. When does cotraining work in real data? *IEEE Transactions on Knowledge and Data Engineering*, 23(5):788–799, 2011. doi: 10.1109/TKDE.2010.158.

R. Eisner, B. Poulin, D. Szafron, P. Lu, and R. Greiner. Improving protein function prediction using the hierarchical structure of the gene ontology. In *2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–10, Nov 2005. doi: 10.1109/CIBCB.2005.1594940.

T. Fagni and F. Sebastiani. On the selection of negative examples for hierarchical text categorization. *Proceedings 3rd Lang Technology Conference*, 01 2007.

S. Feng, P. Fu, and W. Zheng. A hierarchical multi-label classification method based on neural networks for gene function prediction. *Biotechnology & Biotechnological Equipment*, 32(6):1613–1621, 2018. doi: 10.1080/13102818.2018.1521302. URL https://doi.org/10.1080/13102818.2018.1521302.

E. Giunchiglia and T. Lukasiewicz. Coherent hierarchical multi-label classification networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9662–9673. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/6dd4e10e3296fa63738371ec0d5df818-Paper.pdf.

Y. Grandvalet, F. d'Alché Buc, and C. Ambroise. Boosting mixture models for semi-supervised learning. In G. Dorffner, H. Bischof, and K. Hornik, editors, *Artificial Neural Networks — ICANN 2001*, pages 41–48, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44668-2.

A. Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 1st edition, 2017. ISBN 1491962291, 9781491962299.

J. Hernandez, L. Sucar, and E. Morales. A hybrid global-local approach for hierarchical classification. *FLAIRS 2013 - Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference*, pages 432–437, 01 2013.

Jiao Wang, Si-wei Luo, and Xian-hua Zeng. A random subspace method for co-training. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 195–200, 2008. doi: 10.1109/IJCNN.2008.4633789.

K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari Meimandi, M. S. Gerber, and L. E. Barnes. Hdltex: Hierarchical deep learning for text classification. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2017. doi: 10.1109/icmla.2017.0-134. URL http://dx.doi.org/10.1109/ICMLA.2017.0-134.

K. Kowsari, R. Sali, L. Ehsan, W. Adorno, A. Ali, S. Moore, B. Amadi, P. Kelly, S. Syed, and D. Brown. Hmic: Hierarchical medical image classification, a deep learning approach. *Information*, 11(6):318, Jun 2020. ISSN 2078-2489. doi: 10.3390/info11060318. URL http://dx.doi.org/10.3390/info11060318.

Y. Li and Z. Zhou. Towards making unlabeled data never hurt. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):175–188, 2015. doi: 10.1109/TPAMI.2014.2299812.

O. Luaces, J. Díez, J. Barranquero, J. J. del Coz, and A. Bahamonde. Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence*, 1(4):303–313, Dec 2012. ISSN 2192-6360. doi: 10.1007/s13748-012-0030-x. URL https://doi.org/10.1007/s13748-012-0030-x.

P. K. Mallapragada, R. Jin, A. K. Jain, and Y. Liu. Semiboost: Boosting for semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):2000–2014, 2009. doi: 10.1109/TPAMI.2008.235.

J. Metz and A. A. Freitas. Extending hierarchical classification with semi-supervised learning. In *Proceedings of the UK Workshop on Computational Intelligence*, pages 1–6, 2009.

F. Mukti, C. Eswaran, N. Hashim, C. C. Ho, and M. U. A. Ayoobkhan. An automated grading system for diabetic retinopathy using curvelet transform and hierarchical classification. *International Journal of Engineering and Technology(UAE)*, 7:154–157, 01 2018. doi: 10.14419/ijet.v7i2.15.11375.

G. Murtaza, L. Shuib, G. Mujtaba, and G. Raza. Breast cancer multi-classification through deep neural network and hierarchical classification approach. *Multimedia Tools and Applications*, 79:15481–15511, 2019.

A. Naik and H. Rangwala. Embedding feature selection for large-scale hierarchical classification. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1212–1221, Dec 2016. doi: 10.1109/BigData.2016.7840725.

A. Naik and H. Rangwala. Filter based taxonomy modification for improving hierarchical classification. *CoRR*, abs/1603.00772, 2016. URL http://arxiv.org/abs/1603.00772.

F. K. Nakano, W. J. Pinto, G. L. Pappa, and R. Cerri. Top-down strategies for hierarchical classification of transposable elements with neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2539–2546, May 2017. doi: 10.1109/IJCNN.2017.7966165.

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. doi: 10.1109/TKDE.2009.191.

M. Panta, A. Mishra, M. T. Hoque, and J. Atallah. Machine learning based prediction of hierarchical classification of transposable elements, 2019.

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

J. A. Richards. *Clustering and Unsupervised Classification*, pages 247–318. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013a. ISBN 978-3-642-30062-2. doi: 10.1007/978-3-642-30062-2_9. URL https://doi.org/10.1007/978-3-642-30062-2_9.

J. A. Richards. *Supervised Classification Techniques*, pages 247–318. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013b. ISBN 978-3-642-30062-2. doi: 10.1007/978-3-642-30062-2_8. URL https://doi.org/10.1007/978-3-642-30062-2_8.

P. N. Robinson, M. Frasca, S. Köhler, M. Notaro, M. Re, and G. Valentini. A hierarchical ensemble method for dag-structured taxonomies. In F. Schwenker, F. Roli, and J. Kittler, editors, *Multiple Classifier Systems*, pages 15–26, Cham, 2015. Springer International Publishing. ISBN 978-3-319-20248-8.

R. Sali, S. Adewole, L. Ehsan, L. A. Denson, P. Kelly, B. C. Amadi, L. Holtz, S. A. Ali, S. R. Moore, S. Syed, and D. E. Brown. Hierarchical deep convolutional neural networks for multi-category diagnosis of gastrointestinal disorders on histopathological images, 2020.

A. Santos and A. Canuto. Applying semi-supervised learning in hierarchical multi-label classification. *Expert Systems with Applications*, 41(14):6075–6085, 2014a. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2014.03.052. URL https://www.sciencedirect.com/science/article/pii/S0957417414001900.

A. Santos and A. Canuto. Applying the self-training semi-supervised learning in hierarchical multi-label methods. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 872–879, 2014b. doi: 10.1109/IJCNN.2014.6889565.

A. M. Santos and A. M. P. Canuto. Using semi-supervised learning in multi-label classification problems. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2012. doi: 10.1109/IJCNN.2012.6252800.

J. Serrano-Pérez and L. E. Sucar. Hierarchical classification with bayesian networks and chained classifiers. In *Proceedings of the Thirty-Second International Florida Artificial Intelligence Research Society Conference, Sarasota, Florida, USA, May 19-22 2019.*, pages 488–493, 2019.

J. Serrano-Pérez and L. E. Sucar. Artificial datasets for hierarchical classification. *Expert Systems with Applications*, 182:115218, 2021. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.115218. URL https://www.sciencedirect.com/science/article/pii/S0957417421006515.

C. N. Silla and A. A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1):31–72, Jan 2011. ISSN 1573-756X. doi: 10.1007/s10618-010-0175-9. URL https://doi.org/10.1007/s10618-010-0175-9.

V. Sindhwani and D. S. Rosenberg. An rkhs for multi-view learning and manifold co-regularization. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 976–983, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390279. URL https://doi.org/10.1145/1390156.1390279.

V. Sindhwani, P. Niyogi, and M. Belkin. A co-regularized approach to semi-supervised learning with multiple views. In *Proceedings of the 22nd ICML Workshop on Learning with Multiple Views*, 2005.

L. E. Sucar, C. Bielza, E. F. Morales, P. Hernandez-Leal, J. H. Zaragoza, and P. Larrañaga. Multi-label classification with bayesian network-based chain classifiers. *Pattern Recognition Letters*, 41:14 – 22, 2014. ISSN 0167-8655. doi:

https://doi.org/10.1016/j.patrec.2013.11.007. URL http://www.sciencedirect.com/science/article/pii/S0167865513004303. Supervised and Unsupervised Classification Techniques and their Applications.

Z. Sun, Y. Zhao, D. Cao, and H. Hao. Hierarchical multilabel classification with optimal path prediction. *Neural Processing Letters*, 45(1):263–277, Feb 2017. ISSN 1573-773X. doi: 10.1007/s11063-016-9526-x. URL https://doi.org/10.1007/s11063-016-9526-x.

G. Tsoumakas, I. Katakis, and I. Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011. doi: 10.1109/TKDE.2010.164.

J. E. van Engelen and H. Hoos. A survey on semi-supervised learning. *Machine Learning*, 109:373–440, 2019.

C. Vens, J. Struyf, L. Schietgat, S. Dzeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine learning*, 73(2):185–214, 2008a. ISSN 0885-6125.

C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine learning*, 73(2):185, 2008b.

W. Wang and Z.-H. Zhou. A new analysis of co-training. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 1135–1142, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Z. Wu and S. Saito. Hinet: Hierarchical classification with neural network, 2018.

H. Xiao, X. Liu, and Y. Song. Efficient path prediction for semi-supervised and weakly supervised hierarchical text classification. In *The World Wide Web Conference*, WWW '19, page 3370–3376, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313658. URL https://doi.org/10.1145/3308558.3313658.

C. Xu, D. Tao, and C. Xu. A survey on multi-view learning, 2013.

G. H. Yamashita, M. J. Anzanello, F. Soares, M. K. Rocha, F. S. Fogliatto, N. P. Rodrigues, E. Rodrigues, P. G. Celso, V. Manfroi, and P. F. Hertz. Hierarchical classification of sparkling wine samples according to the country of origin based on the most informative chemical elements. *Food Control*, 106:106737, 2019. ISSN 0956-7135. doi: https://doi.org/10.1016/j.foodcont.2019.106737. URL http://www.sciencedirect.com/science/article/pii/S0956713519303184.

D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ACL '95, page 189–196, USA, 1995. Association for Computational Linguistics. doi: 10.3115/981658.981684. URL https://doi.org/10.3115/981658.981684.

X. Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison, 07 2008.

F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2021. doi: 10.1109/JPROC.2020.3004555.

Appendices

A Metric

Let X be a set, and let d be a function defined on pair of elements of X. d is a metric of space X if the following axioms are satisfied for all $x_1, x_2, x_3 \in X$:

- $d(x_1, x_2) \geq 0$
- $d(x_1, x_2) = 0$, if and only if $x_1 = x_2$, *Identity.*
- $d(x_1, x_2) = d(x_2, x_1)$, *Symmetry.*
- $d(x_1, x_2) \leq d(x_1, x_3) + d(x_3, x_2)$, *triangle inequality.*