The Thirty-Second International Florida
Artificial Intelligence Research Society Conference (FLAIRS-32)

Hierarchical Classification with Bayesian Networks and Chained Classifiers

Jonathan Serrano-Pérez, L. Enrique Sucar
Instituto Nacional de Astrofísica Óptica y Electrónica, México
{js.perez,esucar}@inaoep.mx

Abstract

In this work is proposed a method for Hierarchical Classification, which takes advantage of the hierarchical structure to influence the prediction of local classifiers with their neighbors. To achieve this two strategies are combined. The first is to represent the hierarchical structure as a Bayesian network, and the second is to build chained classifiers that feed the Bayesian network as local classifiers. The proposed method was tested in several datasets of functional genomics, which consist of tree-structured hierarchies. The results of several variants of the proposed method are compared to the standard methods, Flat and Top-Down, as well as with a state of the art technique, showing superior performance under several metrics.

Introduction

Hierarchical classification is a special type of Multi-Label Classification, in which the classes are arranged in a Tree structure or in its general form as a Direct Acyclic Graph (DAG). Therefore, the subset of classes in which an instance is associated must satisfy the Hierarchical Constraint.

Several methods (Silla and Freitas 2011) have been proposed for the problem of hierarchical classification, nevertheless, those methods have some problems such as error propagation, do not use the hierarchical structure, among others. In this work is proposed a method for hierarchical classification where the hierarchical structure is used and the predictions of the different classes are influenced by their neighbors.

The proposed method consists of two levels. In the first level, the hierarchical structure is represented as a Bayesian Network, which represents the data distribution in the nodes while maintaining the hierarchical constraint. In the second level, Chained Classifiers are learned for each class (node), so their predictions are influenced by their neighbors in the hierarchy, and feed the Bayesian Network.

To test the method, several datasets from the field of functional genomic were used. Further, the method was compared versus standard methods, Top-Down and Flat, and a state of the art technique. The results show that the proposed method outperforms the previous methods, and obtains statistical significance over some measures.

Fundamentals

The *Hierarchical Structure* (HS) of a dataset is denoted with the notation of a Graph:

$$HS = (C, E) \tag{1}$$

where C is the set of classes/nodes and E is the set of edges that related to the nodes. The *Hierarchical Constraint* say, if an instance z is associated to the class C_i, the instance z must be associated to the ancestors of C_i ($Anc(C_i)$):

$$\forall z \epsilon C_i : z \epsilon Anc(C_i) \tag{2}$$

In this way, a *valid path* or *consistent path* is a subset of the classes that complies the hierarchical constraint. There are different problems of hierarchical classification, (Silla and Freitas 2011) describe them as a 3-tuple (Υ, Ψ, Φ), where Υ specifies the type of hierarchical structure, T if it is a tree or DAG if it is a Direct Acyclic Graph, Ψ specifies whether an instance can be associated to a single path (SPL) or multiple paths (MPL), and Φ indicates the depth of the paths of the instances, that is, FD if all the paths reach a leaf node or PD if at least one path of an instance do not reach a leaf node.

On the other hand, methods can be described according to the problem that they try to solve (Silla and Freitas 2011). Single Path Prediction (SPP) or Multiple Path Prediction (MPP), Mandatory Leaf Node Prediction (MLNP) or Non-Mandatory Leaf Node Prediction (NMLNP), and whether it work in Tree or DAG structures. Furthermore, the methods can make use of Local Classifier per Node (LCN), also known as binary classifier, which consist in predict whether an instance is associated to the node or not, Local Classifier per Level (LCL) that trains a single classifier for each level of the hierarchy, Local Classifier per Parent Node (LCPN) that consist of predicting an instance on the children of the node, and Global Classifiers (GC) that generally builds a classifier that takes into account the entire hierarchy.
In this work is proposed a method described as (SPP, MLNP, T, GC) for problems of type (T, SPL, FD), in addition, the proposed method has been tested (without additional modifications) in problems of type (T, SPL, PD), where results obtained are promising.

Related Work

There are two standard methods for hierarchical classification. The first is known as *Flat*, which consist in building binary classifiers for each leaf node ignoring completely the HS, so, a new instance is classified with the path that reaches the leaf node with the highest probability. The second method is known as *Top-Down* (TD), for each node a LCN is built, so an instance is classified with the path that start in the root node and finish in a leaf node, selecting at each level the node with the highest probability. This last approach has the problem of *Error Propagation*.

Some variants of the standard methods have been proposed. For example, (Secker et al. 2007) propose to select the best classifier for each node, later, (Silla and Freitas 2009) propose two approaches, in the first they make attributes selection for each node, and in the second, they make attributes selection and selection of the best classifier for each node. Other methods such as (Hernandez, Sucar, and Morales 2013) and (Kosmopoulos, Paliouras, and Androutsopoulos 2015) propose to build LCPN for each non-leaf node, then the instances are evaluated in all classifiers, and finally, scoring measures are used to select the best path for each instance.

There is another group of methods, where the main idea is to modify the HS. Some examples are the methods proposed by (Wang and Lu 2010), (Babbar et al. 2013), HierFlat of (Naik and Rangwala 2017), which try to flatten the HS, that is, this methods remove some internal nodes based in some criterion. (Naik and Rangwala 2016) also propose their method *rewHier* in which different operations are used to modify the HS such as node creation, Parent-Child rewiring and node deletion. Once the HS has been modified, the process of classification is Top-Down. Nevertheless, those methods imply loss of information from the HS.

(Barutcuoglu et al. 2007) propose the method *Hierarchical Bayesian Aggregation* (HBA), in which a Bayesian Network (BN) is built from the HS and independent binary classifiers are built which feed the BN, however, given that the nodes are arranged in the HS, to use independent classifiers does not seem right. On the other hand, (Ramírez-Corona, Sucar, and Morales 2016) proposed the method *Chained Path Evaluation* (CPE), in which *Chained Classifiers* are built, thus, the prediction of a node is influenced by the predictions of its parents; however, this method does not take into account the information that is represented as a BN.

Bayesian Network with Chained Classifiers

Bayesian Network

For modelling the HS, we propose a variant of Hierarchical Bayesian Aggregation (Barutcuoglu et al. 2007) (HBA, originally proposed for problems of type (DAG, MPL, PD)). This method consists of a Bayesian Network (BN) from the HS and adding to the BN an extra node for each node of the HS, as shown in figure 1. Some modifications have been made to the original method, which will be explained later.

From the BN (see Figure 1), the y_i nodes will have the probability that a new instance is associated to the class C_i, while the q_i nodes receive the prediction of the classifiers, thus, there is a binary classifier for each node q_i. Unlike the original method, the root node (R) does not have assigned a node q, since it is assumed that all the instances belong to this node, thus, it is not necessary to create the node q_R and its corresponding classifier.

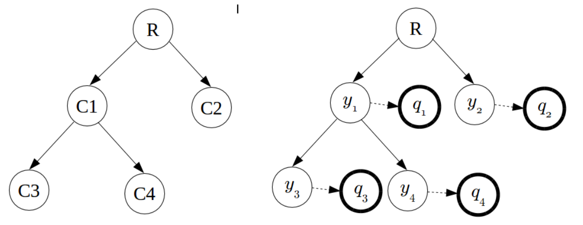

Figure 1: A hierarchical structure (on the left) is transformed into a Bayesian network (on the right).

Parameter estimation Once the BN is built, its parameters must be estimated, which correspond to the Conditional Probability Tables (CPT) for each node.

First, $pa(y_i)$ represents the set of the parents of y_i, thus, for each node y_i must be calculated $P(y_i|pa(y_i))$, which can be estimate by frequency from the training set.

Now, the parameters $P(q_i|y_i)$ have to be estimated. These parameters represent the base classifier output distribution to be expected on instances that were not used as training, that is, $P(q_i|y_i)$ can be estimated using the confusion matrices over validation data.

Chained Classifiers

As part of the method of HBA, (Barutcuoglu et al. 2007) proposed to use a binary classifier for each node q_i, where it is assumed conditional independence from a prediction q_i given any other prediction q_j. However, the nodes are arranged in a structure, thus, that assumption does not seem right. So in this work is proposed to use Chained Classifiers (Sucar et al. 2014; Zaragoza et al. 2011), in this way, the prediction of a node is influenced by the predictions of its neighbors, that are given by the HS.

Thus, $P(Cl_i|Nei(Cl_i), \mathbf{x})$ is obtained for each node i, where Cl_i is the classification for the i-th node, $Nei(Cl_i)$ are the predictions of its neighbors that are added as attributes and $\mathbf{x}$ is the set of attributes. In this way, the nodes are influenced by their neighbors.

Three variants of chained classifiers are consider:

- **Chained Classifiers of Parents (HCP)**: correspond to the original chained classifiers shown by Sucar et al.; Zaragoza et al., where each node receive as additional attributes the predictions of its parents ($Nei(Cl_i) \leftarrow Pa(Cl_i)$), for example, Cl_7 receives as additional attribute the prediction of Cl_3,see figure 2 a).
- **Chained Classifiers of Ancestors (HCA)**: each classifier receives as additional attributes the predictions of its ancestors ($Nei(Cl_i) \leftarrow Anc(Cl_i)$), for example, Cl_8 receives as additional attributes the predictions of Cl_1 and Cl_3, see figure 2 b).
- **Chained Classifiers of Children (HCC)**: in this version, each classifier receives as additional attributes the predictions of its children ($Nei(Cl_i) \leftarrow Ch(Cl_i)$), for example,

Cl_2 receives as additional attributes the predictions of Cl_5 and Cl_6, see figure 2 c).

Combination of the Bayesian Network with Chained Classifiers

The BN requires a classifier for each node q_i, and the Chained Classifiers satisfy it directly, as for each node in the HS a classifier Cl_i is built. Thus, each Cl_i classifier will feed its correspond q_i node, as it is shown in figure 3. The fact that chained classifiers are used implies that the calculation of the CPT for each $P(q_i|y_i)$ must be calculated using these same and not to use independent classifiers. This results in a two-level method, as can be seen in figure 3, in the first level is the Bayesian network, and in the second level are the chained classifiers.

Inference

Once the estimations are received from each local classifier, these are combined by probability propagation in the BN, after which the posterior probabilities for each class (node) are obtained. Then a path in the hierarchy is selected as explained below.

Mandatory Leaf Node Prediction This work is focused in the Mandatory Leaf Node Prediction (MLNP), that is, the prediction for an instance is the path of nodes/classes that starts in the root node and finishes in one leaf node (of course, this must comply the hierarchical constraint), and that has the best score.

To score the different paths, *Sum of Probabilities* (SP) has been used. Let be H the set of leaf nodes from the BN considering only the y nodes, let be $Path_h$ the subset of y nodes that form a path that finish in the node $h \epsilon H$, the score for $Path_h$ is defined in equation 3.

$$SP_{Path_h} = \frac{\sum_{y_x \epsilon Path_h} P(y_x = 1)}{|Path_h|} \tag{3}$$

Where $P(y_x = 1)$ is obtained from the BN by marginalization. Thus, the subset, $Path_h$ that maximizes equation 3 is returned as the prediction of the instance.

Summary of the proposed method

The proposed consists of two main phases:

Training

1. Given the HS, the method of Bayesian aggregation is applied.
   - The CPT of the y nodes are calculated using only the training set, that is, $P(y_i|Pa(y_i))$.
   - The CPT of the q nodes are calculated, based on the confusion matrix of the local classifiers, that is, $P(q_i|y_i)$.
2. The chained classifiers are built using the HS, and selecting one of the three variants of neighbors for building the chain. $Nei(Cl_i) \leftarrow Pa(Cl_i)|Anc(Cl_i)|Ch(Cl_i)$.

Classifying a new instance

1. The new instance is evaluated in all the chained classifiers, and the probabilities of the predictions are obtained.
2. The Bayesian Network receives all the probabilities obtained from the classifiers, then, the Junction-Tree algorithm is applied for inference.
3. The subset $Path_h$ that maximizes the Sum of Probabilities is returned as the prediction of the new instance.

Datasets and Preprocessing

The datasets used are a subset of ***FunCat***, from the field of Functional genomics (Vens et al. 2008)[1].

The preprocessing applied to the datasets are those used by (Ramírez-Corona, Sucar, and Morales 2016), with the intention of making a direct comparison with their results. The type of problem that the FunCat datasets initially have is (T, MPL, PD), so when applying the first preprocessing it results in a problem of type (T, SPL, PD) and applying the second preprocessing results in a problem of type (T, SPL, FD), in both cases the HS could be modified.

Let be $HS = (C, E)$ the original hierarchical structure, the two first steps are the same for both:

1. Given that the instances are associated to multiple paths, it is only considered the first path, and the rest are deleted.
2. Let be $nmin$ the minimum number of instances that the nodes of the HS must have. Nodes that have less than $nmin$ instances are pruned from the HS, which produces a reduced hierarchical structure HS_{p*}, and the following is true:

$$\begin{aligned} HS_{p*} &= (C_{p*}, E_{p*}) \\ C_{p*} &\subseteq C \\ E_{p*} &\subseteq E \end{aligned} \tag{4}$$

Preprocessing 1 (T, SPL, PD)

3. Instances selection:
   - If the instance is not associated to any node of HS_{p*}, that instance is deleted.
   - All the instances that are associated to at least one node of HS_{p*} are saved. This implies that the path of an instance does not need to reach a leaf node of HS_{p*}.
   - If an instance is associated to some node that is not in HS_{p*}, that means that the instance is associated to nodes that were pruned from the original HS, then the nodes of the path that are not in HS_{p*} are deleted, thus, the instance is saved with its path trimmed.

The datasets after applying this preprocessing are described in the table 1.

Preprocessing 2 (T, SPL, FD)

3. Instances selection:
   - The instances that are associated with a leaf node of HS_{p*} are saved. Thus, the instances that are no associated with a leaf node of HS_{p*} are deleted.

[1]download link: https://dtai.cs.kuleuven.be/clus/hmcdatasets/

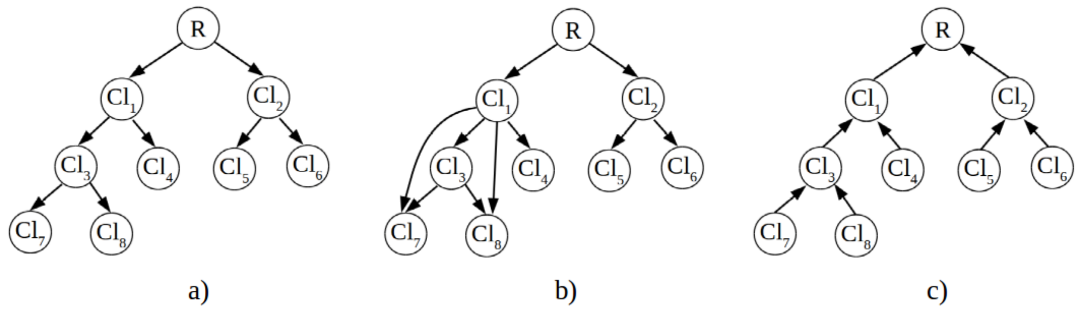

Figure 2: Chained Classifiers of a) Parents, b) Ancestors, c) Children

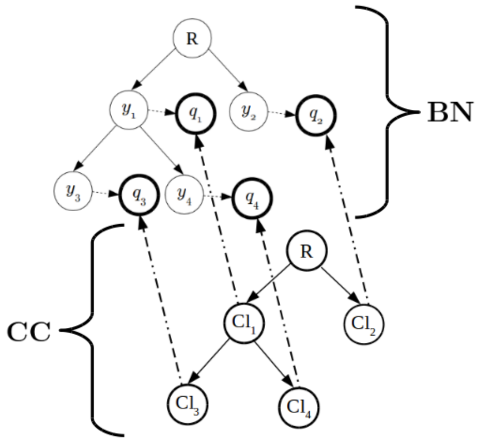

Figure 3: This image represents the method proposed in this work, which has two levels. A Bayesian Network (BN) fed by Chained Classifiers (CC).

- If an instance is associated to a leaf node of HS_{p*} and the instance is associated to some node that is not in HS_{p*}, that means that the instance is associated to nodes that were pruned from the original HS, then the nodes of the path that are not in HS_{p*} are deleted, thus, the instance is saved with its path trimmed.

The datasets after applying this preprocessing are described in the table 2.

Evaluation Measures

Let be N the number of instances in the test set, let Y be the real subset of classes to which an instance is associated and let be $\widehat{Y}$ the subset of predicted classes. The four evaluation measures that have been used are described below:

- **Exact Match**: Percentage of instances classified correctly.

$$Exact\_Match = \frac{1}{N}\sum_{i=1}^{N} 1_{Y=\widehat{Y}} \tag{5}$$

- **Accuracy**: Ratio of classes predicted correctly to the union of the real and predicted classes for each instance.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|Y_i \cap \widehat{Y}_i\right|}{\left|Y_i \cup \widehat{Y}_i\right|} \tag{6}$$

Table 1: Description of the dataset after applying the preprocessing 1 (T, SPL, PD), where the minimum number of instances per node is 50 ($nmin = 50$). MD: Maximum Depth

Dataset	Instances	Attr.	Classes	MD
cellcycle_FUN	3602	77	49	4
derisi_FUN	3575	63	49	4
eisen_FUN	2335	79	35	4
gasch1_FUN	3611	173	49	4
gasch2_FUN	3624	52	49	4

Table 2: Description of the dataset after applying the preprocessing 2 (T, SPL, FD), where the minimum number of instances per node is 70 ($nmin = 70$). 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

- **Hamming-Accuracy (H_Accuracy)**: Frequency of incorrect predictions.

$$H\_Accuracy = 1 - Hammming\_Loss \tag{7}$$

$$Hammming\_Loss = \frac{1}{N\,|C|}\sum_{i=1}^{N}\left|Y_i \oplus \widehat{Y}_i\right| \tag{8}$$

Where $\oplus$ is the $exclusiveOR$ operator.

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

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

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

Table 3: Results obtained in datasets with preprocessing 1. The symbols $\triangle$, $\star$ and $\bullet$ indicate superior results with statistical significance against TD, FLAT and CPE, respectively. Best results for each dataset/metric are shown in bold.

Dataset	**Evaluation M.**	**TD**	**FLAT**	**CPE**	**HBA_tp**	**HCP**	**HCA**	**HCC**
cellcycle	Exact Match	8.86	8	10.68	11.66$^{\triangle\star}$	12.1$^{\triangle\star\bullet}$	**12.13**$^{\triangle\star\bullet}$	**12.13**$^{\triangle\star\bullet}$
	Accuracy	13.21	14.26	16.55	20.36$^{\triangle\star\bullet}$	20.73$^{\triangle\star\bullet}$	**20.78**$^{\triangle\star\bullet}$	20.43$^{\triangle\star\bullet}$
	Hamming Acc.	91.48	91.01	92.69	93.07$^{\triangle\star\bullet}$	**93.09**$^{\triangle\star\bullet}$	93.08$^{\triangle\star\bullet}$	92.97$^{\triangle\star\bullet}$
	hF	16.49	17.89		24.38$^{\triangle\star}$	24.77$^{\triangle\star}$	**24.83**$^{\triangle\star}$	24.25$^{\triangle\star}$
derisi	Exact Match	6.41	6.07	8.66	11.27$^{\triangle\star\bullet}$	11.19$^{\triangle\star\bullet}$	**11.33**$^{\triangle\star\bullet}$	11.13$^{\triangle\star\bullet}$
	Accuracy	9.56	12.17	13.92	**20**$^{\triangle\star\bullet}$	19.64$^{\triangle\star\bullet}$	19.87$^{\triangle\star\bullet}$	19.28$^{\triangle\star\bullet}$
	Hamming Acc.	91	90.8	92.49	**93.04**$^{\triangle\star\bullet}$	92.94$^{\triangle\star\bullet}$	92.98$^{\triangle\star\bullet}$	92.82$^{\triangle\star\bullet}$
	hF	11.93	15.76		**23.93**$^{\triangle\star}$	23.38$^{\triangle\star}$	23.75$^{\triangle\star}$	22.81$^{\triangle\star}$
eisen	Exact Match	11.13	9.51	12.42	14.69*	**14.73***	14.52*	13.45
	Accuracy	14.59	15.12	18.86	**23.39**$^{\triangle\star\bullet}$	23.27$^{\triangle\star\bullet}$	23.15$^{\triangle\star\bullet}$	21.31$^{\triangle\star}$
	Hamming Acc.	88.69	88.05	90.59	90.8$^{\triangle\star}$	90.8$^{\triangle\star}$	**90.82**$^{\triangle\star}$	90.26$^{\triangle\star}$
	hF	17.35	18.61		**27.43**$^{\triangle\star}$	27.23$^{\triangle\star}$	27.21$^{\triangle\star}$	24.78$^{\triangle\star}$
gasch1	Exact Match	11.55	10.25	**13.85**	12.68$^{\triangle\star}$	12.74$^{\triangle\star}$	12.85$^{\triangle\star}$	12.88$^{\triangle\star}$
	Accuracy	16.86	16.97	21.62	21.55$^{\triangle\star}$	21.5$^{\triangle\star}$	**21.63**$^{\triangle\star}$	21.55$^{\triangle\star}$
	Hamming Acc.	91.85	91.36	92.79	**93.21**$^{\triangle\star\bullet}$	93.2$^{\triangle\star\bullet}$	**93.21**$^{\triangle\star\bullet}$	93.19$^{\triangle\star\bullet}$
	hF	20.66	20.61		25.95$^{\triangle\star}$	25.89$^{\triangle\star}$	**26.04**$^{\triangle\star}$	25.89$^{\triangle\star}$
gasch2	Exact Match	9.11	8.03	7.86	11.95$^{\triangle\star\bullet}$	**12.17**$^{\triangle\star\bullet}$	12.14$^{\triangle\star\bullet}$	12.14$^{\triangle\star\bullet}$
	Accuracy	12.75	14.17	14.26	20.8$^{\triangle\star\bullet}$	20.87$^{\triangle\star\bullet}$	**20.89**$^{\triangle\star\bullet}$	20.68$^{\triangle\star\bullet}$
	Hamming Acc.	91.93	90.81	92.6	**93.13**$^{\triangle\star\bullet}$	93.11$^{\triangle\star\bullet}$	93.11$^{\triangle\star\bullet}$	93.03$^{\triangle\star\bullet}$
	hF	15.73	17.39		25$^{\triangle\star}$	25.02$^{\triangle\star}$	**25.06**$^{\triangle\star}$	24.64$^{\triangle\star}$

Table 4: Results obtained in datasets with preprocessing 2. The symbols $\triangle$, $\star$ and $\bullet$ indicate superior results with statistical significance against TD, FLAT and CPE, respectively. Best results for each dataset/metric are shown in bold.

Dataset	**Evaluation M.**	**TD**	**FLAT**	**CPE**	**HBA_tp**	**HCP**	**HCA**	**HCC**
cellcycle	Exact Match	14.19	13.98	17.74	18.98$^{\triangle\star}$	19.02$^{\triangle\star}$	**19.2**$^{\triangle\star}$	19.15$^{\triangle\star\bullet}$
	Accuracy	17.41	18.08	22.93	**27.33**$^{\triangle\star\bullet}$	26.89$^{\triangle\star\bullet}$	26.81$^{\triangle\star\bullet}$	26.93$^{\triangle\star\bullet}$
	Hamming Acc.	89.12	88.88	90.98	**91.15**$^{\triangle\star}$	90.98$^{\triangle\star}$	90.89$^{\triangle\star}$	90.96$^{\triangle\star}$
	hF	20.08	20.23		**30.46**$^{\triangle\star}$	29.75$^{\triangle\star}$	29.52$^{\triangle\star}$	29.78$^{\triangle\star}$
derisi	Exact Match	11.59	12.73	13	**16.8**$^{\triangle\star\bullet}$	16.42$^{\triangle\star\bullet}$	16.38$^{\triangle\star\bullet}$	16.42$^{\triangle\star\bullet}$
	Accuracy	14.53	17.37	18.19	**24.67**$^{\triangle\star\bullet}$	23.29$^{\triangle\star\bullet}$	23.57$^{\triangle\star\bullet}$	23.56$^{\triangle\star\bullet}$
	Hamming Acc.	88.66	89.11	90.71	**90.92**$^{\triangle\star}$	90.51$^{\triangle\star}$	90.56$^{\triangle\star}$	90.57$^{\triangle\star}$
	hF	16.18	19.47		**27.28**$^{\triangle\star}$	25.37$^{\triangle\star}$	25.81$^{\triangle\star}$	25.81$^{\triangle\star}$
eisen	Exact Match	15.83	14.1	18.69	19.45*	19.21*	**19.75***	19.51*
	Accuracy	17.91	19.49	27.08	**28.55**$^{\triangle\star}$	27.78$^{\triangle\star}$	28.24$^{\triangle\star}$	28.02$^{\triangle\star}$
	Hamming Acc.	85.9	86.25	**88.4**	87.99$^{\triangle\star}$	87.7$^{\triangle\star}$	87.77$^{\triangle\star}$	87.76$^{\triangle\star}$
	hF	20.03	21.75		**33.4**$^{\triangle\star}$	32.3$^{\triangle\star}$	32.67$^{\triangle\star}$	32.45$^{\triangle\star}$
gasch1	Exact Match	20.38	20.08	21.74	23.19$^{\triangle\star\bullet}$	**23.79**$^{\triangle\star\bullet}$	23.23$^{\triangle\star}$	22.98$^{\triangle\star\bullet}$
	Accuracy	23.63	24.5	28.12	30.62$^{\triangle\star\bullet}$	**30.79**$^{\triangle\star\bullet}$	30.37$^{\triangle\star\bullet}$	30.36$^{\triangle\star\bullet}$
	Hamming Acc.	89.91	89.56	91.45	**91.51**$^{\triangle\star}$	91.46$^{\triangle\star}$	91.37$^{\triangle\star}$	91.44$^{\triangle\star}$
	hF	25.64	25.97		**33.47**$^{\triangle\star}$	33.34$^{\triangle\star}$	32.9$^{\triangle\star}$	33.19$^{\triangle\star}$
gasch2	Exact Match	14.52	15.03	13.35	19.61$^{\triangle\bullet}$	20.21$^{\bullet}$	**20.38**$^{\triangle\bullet}$	19.83$^{\bullet}$
	Accuracy	17.19	18.91	18.32	27.82$^{\triangle\star\bullet}$	28.02$^{\triangle\star\bullet}$	**28.07**$^{\triangle\star\bullet}$	27.51$^{\triangle\star\bullet}$
	Hamming Acc.	89.49	88.62	90.81	**91.18**$^{\triangle\star}$	91.11$^{\triangle\star}$	91.08$^{\triangle\star}$	91$^{\triangle\star}$
	hF	18.36	20.34		30.73$^{\triangle\star}$	**30.76**$^{\triangle\star}$	30.67$^{\triangle\star}$	30.16$^{\triangle\star}$

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

Where hP is the hierarchical Precision and hR is the hierarchical Recall.

Experiments and Results

The first experiment was to select the base classifier, we evaluated *Naïve Bayes*, *SVM* and *Random Forest*, and the best results were obtained with *Random Forest*, thus, the results shown in this work are obtained with this classifier.

The results obtained in this work are compared against the standard methods, *Top-Down* (TD) and *Flat*, and against a state of the art method, *Chained Path Evaluation* (CPE); the four variants of the proposed method are: (i) HBA_tp which includes only the BN, (ii) HCP, (iii) HCA, and (iv) HCC.

The results were obtained applying five-fold crossvalidation. For calculation of the statistical significance, a *t-Student* test with a significance of 0.05 was used.

Table 3 shows the results obtained in the datasets with preprocessing (1), in which the path of an instance could not reach a leaf node. Although the proposed method was not designed for problems of this type, the HCA variant obtains the best scores most of the times; furthermore, all the proposed variants obtain results with statistical significance compared to TD, FLAT and CPE, the later is able to predict trajectories of type NMLNP.

Table 4 shows the results obtained in the datasets with preprocessing (2), in which the path of the instances reaches a leaf node. The four proposed variants obtain good results compared to the standard methods and CPE, even with statistical significance; in this case HBA_tp obtains in most cases the best scores followed by the HCA variant.

Conclusions and Future Work

A novel scheme for hierarchical classification is proposed that combines a BN for providing global consistency, and chain classifiers to include the predictions of the neighbors in the hierarchy.

The experimental results show that the use of a BN to guarantee the hierarchical constraint improves previous results in all metrics, and in some cases including the chained local classifiers implies further gains.

As future work we plan to extend the method for DAG structures which is the natural next step, due that a BN can be built from a DAG, also chained classifiers work for DAG structures (Sucar et al. 2014). Furthermore, Multiple Path Prediction is considered as future work.

References

Babbar, R.; Partalas, I.; Gaussier, E.; and Amini, M.-R. 2013. Maximum-margin framework for training data synchronization in large-scale hierarchical classification. In Lee, M.; Hirose, A.; Hou, Z.-G.; and Kil, R. M., eds., *Neural Information Processing*, 336–343. Berlin, Heidelberg: Springer Berlin Heidelberg.

Barutcuoglu, Z.; Decoro, C.; E Schapire, R.; and G Troyanskaya, O. 2007. Bayesian aggregation for hierarchical classification.

Hernandez, J. N.; Sucar, L. E.; and Morales, E. F. 2013. A hybrid global-local approach for hierarchical classification. In *FLAIRS Conference*.

Kosmopoulos, A.; Paliouras, G.; and Androutsopoulos, I. 2015. Probabilistic cascading for large scale hierarchical classification. *CoRR* abs/1505.02251.

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

Naik, A., and Rangwala, H. 2017. Hierflat: flattened hierarchies for improving top-down hierarchical classification. *International Journal of Data Science and Analytics* 4(3):191–208.

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

Secker, A. D.; Davies, M. N.; Freitas, A. A.; Timmis, J.; Mendao, M.; and Flower, D. R. 2007. An experimental comparison of classification algorithms for hierarchical prediction of protein function. *Expert Update (Magazine of the British Computer Society's Specialist Group on AI)* 9(3):17–22.

Silla, C. N., and Freitas, A. A. 2009. Novel top-down approaches for hierarchical classification and their application to automatic music genre classification. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, 3499–3504.

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

Sucar, L. E.; Bielza, C.; Morales, E. F.; Hernandez-Leal, P.; Zaragoza, J. H.; and Larrañaga, P. 2014. Multi-label classification with bayesian network-based chain classifiers. *Pattern Recognition Letters* 41:14 – 22. Supervised and Unsupervised Classification Techniques and their Applications.

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

Wang, X., and Lu, B. 2010. Flatten hierarchies for large-scale hierarchical text categorization. In *2010 Fifth International Conference on Digital Information Management (ICDIM)*, 139–144.

Zaragoza, J. H.; Sucar, L. E.; Morales, E. F.; Bielza, C.; and Larrañaga, P. 2011. Bayesian chain classifiers for multidimensional classification. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, 2192–2197. AAAI Press.