Hierarchical Multi-Label Classification Networks

Jônatas Wehrmann [1] **Ricardo Cerri** [2] **Rodrigo C. Barros** [1]

Abstract

One of the most challenging machine learning problems is a particular case of data classification in which classes are hierarchically structured and objects can be assigned to multiple paths of the class hierarchy at the same time. This task is known as hierarchical multi-label classification (HMC), with applications in text classification, image annotation, and in bioinformatics problems such as protein function prediction. In this paper, we propose novel neural network architectures for HMC called HMCN, capable of simultaneously optimizing local and global loss functions for discovering local hierarchical class-relationships and global information from the entire class hierarchy while penalizing hierarchical violations. We evaluate its performance in 21 datasets from four distinct domains, and we compare it against the current HMC state-of-the-art approaches. Results show that HMCN substantially outperforms all baselines with statistical significance, arising as the novel state-of-the-art for HMC.

1. Introduction

The main focus of research in machine learning has been towards the induction of models for typical classification problems, where an object is associated with a single class from a set of disjoint classes. There is, however, a niche of tasks in which classes are not disjoint but organized into a hierarchical structure, namely *hierarchical classification* (HC). In HC, objects are associated with a given superclass and its corresponding subclasses. Depending on the task, the correspondence may be with all subclasses or with only a subset of them. The hierarchical structure that formalizes the relationship among classes can assume the form of a tree or of a Directed Acyclic Graph (DAG).

[1]School of Technology, Pontifícia Universidade Católica do Rio Grande do Sul [2]Universidade Federal de São Carlos. Correspondence to: Rodrigo C. Barros <rodrigo.barros@pucrs.br>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

In a more challenging scenario, there are HC problems in which each object can be associated to several different paths of the class hierarchy, namely *hierarchical multi-label classification* (HMC). Typical HMC problems are text classification (Rousu et al., 2006; Cesa-Bianchi et al., 2006; Mayne & Perry, 2009; Lewis et al., 2004; Klimt & Yang, 2004), image annotation (Dimitrovski et al., 2011), and bioinformatics tasks such as protein function prediction (Otero et al., 2010; Valentini, 2011; Bi & Kwok, 2011; Barros et al., 2013; Cerri et al., 2015; Triguero & Vens, 2016), which are the application domains we focus in this paper.

Algorithms that perform HMC must be capable of labeling objects as belonging to one or multiple paths in the class hierarchy. For such, they must optimize a loss function either locally or globally (Silla & Freitas, 2010). Algorithms that perform local learning attempt to discover the specificities that dictate the class relationships in particular regions of the class hierarchy, later combining the local predictions in order to generate the final classification. The idea is to generate a hierarchy of classifiers following a top-down strategy (Costa et al., 2007), in which each classifier is responsible for the prediction of either particular nodes (Kiritchenko et al., 2004) or particular hierarchical levels. Global approaches for HMC, on the other hand, usually consist of a single classifier capable of associating objects with their corresponding classes in the hierarchy as a whole (Cerri et al., 2012; 2013b). There are advantages and disadvantages of using either global or local approaches. Global approaches are usually cheaper than local approaches, and they do not suffer from the well-known error-propagation problem, though they are less likely to capture local information from the hierarchy, eventually underfitting. Local approaches are much more computationally expensive since they rely on a cascade of classifiers, but they are much more suitable for extracting information from regions of the class hierarchy, eventually overfitting.

In order to combine the advantages of both HMC approaches and to careful avoid their downfalls, we propose a paradigm shift: instead of choosing a particular strategy, we argue in favor of a hybrid method capable of simultaneously optimizing both local and global loss functions. Our novel approach, namely HMCN, implements recurrent and non-recurrent neural network architectures. In both versions it comprises multiple outputs, with one output layer per hi-

erarchical level of the class hierarchy plus a global output layer for the entire network. The rationale is that each local loss function reinforces the propagation of gradients leading to proper local-information encoding among classes of the corresponding hierarchical level. At the same time, the loss function optimized in the global output layer keeps track of the label dependency in the hierarchy as a whole. In addition, we introduce a hierarchical violation penalty for encouraging predictions that obey the hierarchical structure. We show that HMCN comfortably establishes itself as the novel state-of-the-art for HMC problems.

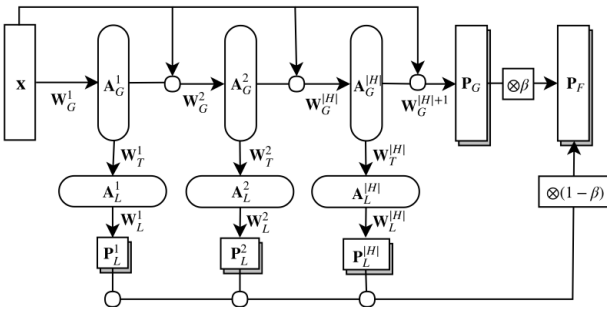

Figure 1. HMCN-F architecture.

2. HMCN

We propose HIERARCHICAL MULTI-LABEL CLASSIFICATION NETWORKS (HMCN), which is a multiple-output deep neural network specially designed to perform both local and global optimization in HMC problems. For that, HMCN propagates gradients from multiple network outputs. It comprises one local output per hierarchical level, and a local loss function is used for backpropagating the gradients from the classes in the corresponding level. The global output captures the cumulative relationships that were forwarded across the entire network, backpropagating the gradients from all classes of the hierarchy. We present both a feed-forward (HMCN-F) and a recurrent (HMCN-R) architecture of HMCN.

2.1. HMCN-F

HMCN-F is specifically designed for maximizing the learning capacity regarding the hierarchical structure of the labeled data. In HMCN-F, information flows in two ways: i) the main flow, which begins with the input layer and traverses all fully-connected (FC) layers until it reaches the global output; and ii) the local flows, which also begin in the input layer and pass by their respective global FC layers but also through specific local FC layers, finally ending at the corresponding local output. For generating the final prediction, all local outputs are then concatenated and pooled with the global output for a consensual prediction. The global flow is responsible to carry information from the i^{th} level to the $(i+1)^{th}$ hierarchical level. It is influenced by the local outputs that reinforce local level-wise relationships within the global information flow by backpropagating gradients specific to the set of classes from each level. Figure 1 depicts a graphical illustration of the structure of HMCN-F along with its respective notation.

Formally, let $\mathbf{x} \in \mathbb{R}^{|\mathbf{D}|\times\mathbf{1}}$ be the input feature vector, $\mathbf{C}^h$ be the set of classes of the h^{th} hierarchical level, $|D|$ be the number of features, $|H|$ the total number of hierarchical levels, and $|C|$ the total number of classes. Let $\mathbf{A}_G^1$ denote the activations in the first level of the global flow (first level of the class hierarchy) and is given by:

$$\mathbf{A}_G^1 = \phi(\mathbf{W}_G^1\mathbf{x} + \mathbf{b}_\mathbf{G}^\mathbf{1}) \tag{1}$$

where $\mathbf{W}_\mathbf{G}^\mathbf{1} \in \mathbb{R}^{|\mathbf{A}_\mathbf{G}^\mathbf{1}|\times|\mathbf{D}|}$ is a weight matrix and $\mathbf{b}_G^1 \in \mathbb{R}^{|A_G^1|\times 1}$ is the bias vector, which are the parameters for learning global information directly from the input, and ϕ is a non-linear activation function (e.g., ReLU). The subsequent global activations are given by

$$\mathbf{A}_G^h = \phi(\mathbf{W}_G^h(\mathbf{A}_G^{h-1} \odot \mathbf{x}) + \mathbf{b}_G^h) \tag{2}$$

where $\mathbf{W}_G^h \in \mathbb{R}^{|A_G^h|\times|A_G^{h-1}|}$ is the weight matrix for the h^{th} level of the hierarchy, $\mathbf{b}_G^h \in \mathbb{R}^{|A_G^h|\times 1}$ is the corresponding bias vector, and $\odot$ denotes vector concatenation. The HMC prediction based on global information $\mathbf{P}_G \in \mathbb{R}^{|C|\times 1}$ is then calculated by

$$\mathbf{P}_G = \sigma(\mathbf{W}_G^{|H|+1}\mathbf{A}_G^{|H|} + \mathbf{b}_G^{|H|+1}) \tag{3}$$

where $\mathbf{W}_G^{|H|+1} \in \mathbb{R}^{|C|\times|A_G^{|H|}|}$ is the weight matrix that connects the activations of the last hierarchical level with the global output that has $|C|$ neurons, $\mathbf{b}_G^{|H|+1} \in \mathbb{R}^{|C|\times 1}$ is the corresponding bias vector, and σ is the sigmoidal activation. Note that $\mathbf{P}_\mathbf{G}$ is a continous vector for which each position $P_G^{(i)}$ denotes the probability $P(C_i|\mathbf{x})$ for $C_i \in \mathbf{C}$. With respect to the local flows, let $\mathbf{A}_L^h$ denote the activations in the h^{th} layer of the local flow, which is calculated by

$$\mathbf{A}_L^h = \phi(\mathbf{W}_T^h\mathbf{A}_G^h + \mathbf{b}_T^h) \tag{4}$$

where $\mathbf{W}_T^h \in \mathbb{R}^{|A_L^h|\times|A_G^h|}$ is a transition weight matrix that maps a global hidden layer to a local hidden layer, and $\mathbf{b}_T^h \in \mathbb{R}^{|A_L^h|\times 1}$ is the transition bias vector. The local information of each hierarchical level is learned by parameters $\mathbf{W}_L^h \in \mathbb{R}^{|C^h|\times|A_L^h|}$ and $\mathbf{b}_L \in \mathbb{R}^{|C^h|\times 1}$. Hence, the local predictions for level h, $\mathbf{P}_L^h \in \mathbb{R}^{|C^h|}$, are given by

$$\mathbf{P}_L^h = \sigma(\mathbf{W}_L^h\mathbf{A}_L^h + \mathbf{b}_L^h) \tag{5}$$

where once again σ is necessarily sigmoidal and the i^{th} position of $\mathbf{P}_L^h$ denotes probability $P(C_i|\mathbf{x})$ for $C_i \in \mathbf{C}^h$. Note that $\mathbf{W}_L$ is a set of weight matrices that perform linear mappings from a hidden space vector into hierarchical multi-label classes. Given that each level may comprise a distinct number of classes, we use a distinct weight matrix per level. In order to use both local and global information, the final predictions $\mathbf{P}_G \in \mathbb{R}^{|C|\times 1}$ are calculated by

$$\mathbf{P}_F = \beta\left(\mathbf{P}_L^1 \odot \mathbf{P}_L^2 \odot, \ldots, \mathbf{P}_L^{|H|}\right) + (1-\beta)\mathbf{P}_G \quad (6)$$

where $\beta \in [0, 1]$ is the parameter that regulates the trade-off regarding local and global information. We set $\beta = 0.5$ as the default option in order to give equal importance to both local and global information from the class hierarchy.

To the best of our knowledge, this is the first work to propose a conjoint local-global optimization approach for HMC problems. Recent neural network approaches for HMC (Cerri et al., 2013a; 2016; Wehrmann et al., 2017) optimize exclusively local information. Recall that global approaches are better-suited to discover overall relationships by capturing information from all hierarchical layers at once. The global output in HMCN-F is a FC layer with $|C|$ neurons, where $|C|$ is the number of classes in the dataset. For a 6-level HMC problem, the sixth layer of the global flow receives as input the cumulative information from feature space and maps it at once to all classes of the hierarchy, hence defining HMCN-F as a global approach. Notwithstanding, we argue that also employing local outputs within the architecture is beneficial for three main reasons: (i) it feeds the network with specific local information that would otherwise be ignored; (ii) it prevents dead neurons by increasing the gradient signal; and (iii) it accelerates convergence, which is a direct consequence of increasing the gradient signal. Recall that multiple-output networks have also been applied with success to flat single-label and multi-label classification (see (Lee et al., 2015; Cissé et al., 2016) and references therein).

HMCN-F comprises l local outputs (one per hierarchical level) and virtually two global outputs: the main global output and the weight-average output. For instance, if one has prior knowledge that global information is much more relevant than local relationships in a given dataset, one can control that by decreasing the value of β. Such a decision automatically enforces the loss function to prioritize global features during learning, and the same could be done for each local output. There are no gradients flowing back from the trade-off output, only from the local and global outputs.

Another important architectural choice in HMCN-F is the reuse of the original input features in each layer of the main flow. By reusing the original features, HMCN-F improves the process of learning local information. More specifically, each layer becomes specialized in finding relationships between the original features and the information required for classifying classes in a given hierarchical level. Such an intuition is empirically demonstrated in Section 4.1, in which we show the gains in reusing the input features throughout the global flow. Input reuse is a choice often used in RNNs for keeping the input features easily accessible (which helps in tasks such as image captioning). In our work, this choice is grounded with an even stronger intuition: it allows each local network to learn features directly related to a given hierarchical level, which can be seen as specialist two-layered neural networks that are capable of leveraging information from the previous levels as well.

2.2. HMCN-R

One of the main problems of HMCN-F is that the amount of parameters grow with the number of hierarchical levels. Hence, we developed a modification over HMCN-F where the global flow shares weights throughout the hierarchy, acting as a recurrent path within the network. This modification resulted in a very unstable training process, making such an architecture unfeasible for practical use. Notwithstanding, we realized that HCMN could be properly modeled as a fully-recurrent network, even though HMC is not a case of sequential learning. Hence, we developed a recurrent version of HMCN inspired on Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997), hereafter called HMCN-R. In HMCN-R, each iteration resulting from unrolling the recurrent network concerns a hierarchical level. For instance, an HMC problem with 6 levels is modeled as a recurrent network that is unrolled into 6 iterations. We argue that the sequential flow of shared parameters in a recurrent network naturally acts as the global learning flow in HMCN-F, with the advantage of keeping the number of parameters virtually unchanged even for very deep class hierarchies.

In HMCN-R, gradients flow from not only the recurrent general flow, which is capturing global information, but also from the unrolled local outputs generated in each iteration. Recall that the local outputs are specialized in a particular level of the class-hierarchy, conceptually achieving the same desired behavior as in HMCN-F. In HMCN-R, the input reuse occurs naturally since the objects do not vary over time (i.e., they do not constitute a sequence). Therefore, the input of each iteration is the same original input, yielding the same effect of input reuse previously discussed. As in HMCN-F, HMCN-R weights the global output with the local outputs for balancing global and local information. Finally, modeling HMCN-R with an LSTM-based architecture provides the further advantage of gradient regulation, preventing the vanishing problem. Nevertheless, we still have to clip gradients so as to avoid the gradients

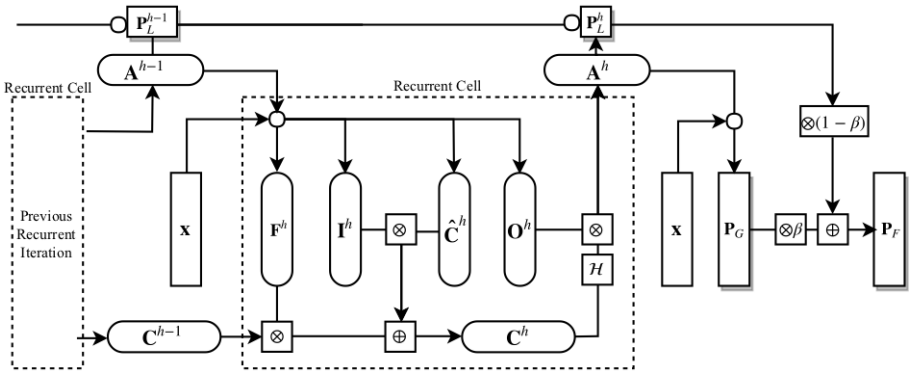

Figure 2. HMCN-R architecture.

from exploding. For such, we guarantee that the norm of the gradient vector does not exceed 1 (this constraint is applied in both HMCN versions). A graphical illustration of HMCN-R is depicted in Figure 2

Our implementation is based on the original LSTM incarnation. It encapsulates long-term dependencies of the h^{th} hierarchical level (i.e., recurrent iteration) in a cell state $\mathbf{C}^{h-1}$ that is accessed (and modified) via forget $\mathbf{F}^h$ and input $\mathbf{I}^h$ gates. Similarly to HMCN-F, we concatenate $\odot$ the input features in each iteration for easier access to the original data.

$$\mathbf{F}^h = \sigma\left(\mathbf{W}_F(\mathbf{A}^{h-1} \odot \mathbf{x}) + \mathbf{b}_F\right) \tag{7}$$

$$\mathbf{I}^h = \sigma\left(\mathbf{W}_I(\mathbf{A}^{h-1} \odot \mathbf{x}) + \mathbf{b}_I\right) \tag{8}$$

$$\hat{\mathbf{C}}^h = \mathcal{H}\left(\mathbf{W}_C(\mathbf{A}^{h-1} \odot \mathbf{x}) + \mathbf{b}_C\right) \tag{9}$$

$$\mathbf{C}^h = \mathbf{F}^h\mathbf{C}^{h-1} + \mathbf{I}^h\hat{\mathbf{C}}^h \tag{10}$$

The output gate $\mathbf{O}^h$ is calculated by using activations from the previous level $\mathbf{A}^{h-1}$ and the input features $\mathbf{x}$. The activation $\mathbf{A}^h$ for the current h^{th} hierarchical level is generated by modulating the output gate with the current cell state $\mathbf{C^h}$ that is capable of carrying information from previous layers:

$$\mathbf{O}^h = \sigma\left(\mathbf{W}_O(\mathbf{A}^{h-1} \odot \mathbf{x}) + \mathbf{b}_O\right) \tag{11}$$

$$\mathbf{A}^h = \mathbf{O}^h\mathcal{H}(\mathbf{C}^h) \tag{12}$$

Similarly to HMCN-F, we use a transition weight matrix to project global features $\mathbf{A}^h$ into local ones, and individual weight matrices for providing level-specific class predictions. To the best of our knowledge, this is the first approach to use independent weight matrices along with recurrent layers for improving HMC performance.

2.3. Loss Function

HMCN minimizes the sum of the local ($\mathcal{L}_L$) and global ($\mathcal{L}_G$) loss functions.

$$\mathcal{L}_L = \sum_{h=1}^{|H|}\left[\mathcal{E}(\mathbf{P}_L^h, \mathbf{Y}_L^h)\right] \tag{13}$$

$$\mathcal{L}_G = \mathcal{E}(\mathbf{P}_G, \mathbf{Y}_G) \tag{14}$$

Since classes are not mutually exclusive, for learning both global and local information, we minimize the binary cross-entropy $\mathcal{E}(\hat{\mathbf{Y}}, \mathbf{Y})$

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|C|}\Big[\mathbf{Y}_{ij} \times \log(\hat{\mathbf{Y}}_{ij}) + (1 - \mathbf{Y}_{ij}) \times \log(1 - \hat{\mathbf{Y}}_{ij})\Big] \tag{15}$$

where $\mathbf{Y}_G$ is the binary class vector (expected output) containing all classes in the hierarchy and $\mathbf{Y}_L^h$ is the binary class vector (expected output) containing only the classes of the h^{th} hierarchical level. By minimizing the proposed loss function, one cannot guarantee that a consistent hierarchical path would be predicted by the network. To circumvent this issue we propose penalizing predictions with hierarchical violations, and also the execution of a post-processing step to ensure that all predictions respect the hierarchy constraints.

A *hierarchical violation* happens when the generated prediction score of a node $\mathbf{Y}_{in}$ is larger than the score of its parent node $\mathbf{Y}_{ip}$. Those violations, which are conveniently threshold-independent, are applied according Equation 16. We employ $\lambda \in \mathcal{R}$ for regulating the importance of the penalty for hierarchical violations in the final loss function. When λ is too large, the network is biased towards predicting smaller values within deeper layers, which might

Table 1. SUMMARY OF THE 21 DATASETS: NUMBER OF ATTRIBUTES ($|A|$), NUMBER OF CLASSES ($|C|$), TOTAL NUMBER OF OBJECTS (TOTAL), AND NUMBER OF MULTI-LABEL OBJECTS (MULTI).

TAXONOMY	DATASET	$\|A\|$	$\|C\|$	TRAINING		VALIDATION		TEST	
				TOTAL	MULTI	TOTAL	MULTI	TOTAL	MULTI
TREE	ENRON	1000	56	692	692	296	296	660	660
TREE	DIATOMS	371	398	1085	1031	464	436	1054	998
TREE	IMCLEF07A	80	96	7000	7000	3000	3000	1006	1006
TREE	IMCLEF07D	80	46	7000	7000	3000	3000	1006	1006
TREE	REUTERS	1000	102	2100	2032	900	865	3000	2905
FUNCAT (TREE)	CELLCYCLE	77	499	1628	1323	848	673	1281	1059
FUNCAT (TREE)	DERISI	63	499	1608	1309	842	671	1275	1055
FUNCAT (TREE)	EISEN	79	461	1058	900	529	441	837	719
FUNCAT (TREE)	EXPR	551	499	1639	1328	849	674	1291	1064
FUNCAT (TREE)	GASCH1	173	499	1634	1325	846	672	1284	1059
FUNCAT (TREE)	GASCH2	52	499	1639	1328	849	674	1291	1064
FUNCAT (TREE)	SEQ	478	499	1701	1344	879	679	1339	1079
FUNCAT (TREE)	SPO	80	499	1600	1301	837	666	1266	1047
GENE ONTOLOGY	CELLCYCLE	77	4122	1625	1625	848	848	1278	1278
GENE ONTOLOGY	DERISI	63	4116	1605	1605	842	842	1272	1272
GENE ONTOLOGY	EISEN	79	3570	1055	1055	528	528	835	835
GENE ONTOLOGY	EXPR	551	4128	1636	1636	849	849	1288	1288
GENE ONTOLOGY	GASCH1	173	4122	1631	1631	846	846	1281	1281
GENE ONTOLOGY	GASCH2	52	4128	1636	1636	849	849	1288	1288
GENE ONTOLOGY	SEQ	478	4130	1692	1692	876	876	1332	1332
GENE ONTOLOGY	SPO	80	4116	1597	1597	837	837	1263	1263

constraint too much the optimization process. Conversely, if λ is too small the network is allowed to learn inconsistent paths more easily, i.e., it can be more influenced by statistical characteristics of the data rather than by the hierarchical structure at hand. Similarly to (Wehrmann & Barros, 2018; Wehrmann et al., 2018), the violation is given by

$$\mathcal{L}_{H_i} = \lambda \max\{0, \mathbf{Y}_{in} - \mathbf{Y}_{ip}\}^2 \tag{16}$$

The final loss function we optimize is thus given by

$$\min_{W} \left(\mathcal{L}_L + \mathcal{L}_G + L_H\right) \tag{17}$$

3. Experimental Methodology

We compare HMCN-(F/R) with four HMC algorithms which are considered the state-of-the-art for HMC: Clus-HMC (Vens et al., 2008), Clus-HMC-Ens (Schietgat et al., 2010), CSSA (Bi & Kwok, 2011), and HMC-LMLP (Cerri et al., 2016). All algorithms are executed over 21 freely-available datasets related to either protein function prediction (Vens et al., 2008), annotation of medical or microalgae images (Dimitrovski et al., 2011), or text classification (Lewis et al., 2004). Those datasets are structured as either trees (MIPS Functional Catalogue, for protein function prediction) or directed acyclic graphs (Gene Ontology). The following pre-processing step is performed before running HMCN over these datasets: all nominal attribute values were transformed into numeric values using the binary one-attribute-per-value approach. The attributes were then standardized (zero-centered with unit variance). All missing values were replaced by the corresponding mean or mode. Table 1 presents the characteristics of the employed datasets. They present challenging characteristics for deep neural networks: i) they comprise a rather limited amount of training samples; ii) they have a large variation in the number of features (varying from 52 to 1000); iii) they present a wide range on the number of hierarchical levels and classes (some present up to 4200 classes distributed in 13 hierarchical layers). There are several cases in which the classes are very sparse, making the optimization even more challenging.

The outputs of HMCN and baseline algorithms are probability values for each class. Hence, the final predictions (binary vector indicating the presence or absence of each class) are generated after thresholding those probabilities. The choice of optimal threshold is difficult and often subjective, thus we follow the trend of HMC research in which we avoid choosing thresholds by employing precision-recall curves (PR-curves) as the evaluation criterion for comparing the different approaches, and the single-value criterion we analyze is the area under the average precision-recall curve ($AU(\overline{PRC})$). We run all experiments $10\times$ and report averages of those runs as final results. For simplicity, we omit the standard deviation given that our networks have shown to be very stable with standard deviation varying between $[1 \times 10^{-3}, 3 \times 10^{-3}]$. Finally, when training for evaluation in test data, we re-train the models with both training and validation data. Hence, we evaluate models generated at the end of the final epoch.

3.1. Hyper-Parameters

We are dealing with relatively small datasets (though with a very large number of classes), which makes the choice of hyper-parameters harder and more important than it usually is. For training our networks, we use the Adam optimizer with learning rate of 1×10^{-3} and remaining parameters

Table 2. IMPACT OF INPUT REUSE AND LOCAL OUTPUTS ON THE PERFORMANCE OF HMCN-F. THE VALUES PRESENTED ARE OF $AU(\overline{PRC})$ REGARDING FUNCAT VALIDATION DATA. AVERAGE RANKING IS ALSO PROVIDED (LOWER IS BETTER).

DATASET	HMCN (GLOBAL ONLY)	HMCN-F (INPUT REUSE)	HMCN-F (LOCAL OUTPUTS)	HMCN-F
CELLCYCLE	0.200	0.212	0.216	**0.228**
DERISI	0.164	0.166	**0.173**	0.168
EISEN	0.253	0.263	0.266	**0.280**
EXPR	0.220	0.242	0.238	**0.271**
GASH1	0.217	0.243	0.241	**0.261**
GASH2	0.210	0.215	0.211	**0.227**
SEQ	0.109	0.258	0.236	**0.279**
SPO	0.158	0.167	**0.174**	0.170
AVERAGE RANKING	5.25	3.38	3.00	1.25

as suggested in (Kingma & Ba, 2014). For the HMCN-F version, the fully-connected layers comprise 384 ReLU neurons, followed by a batch normalization, residual connections, and dropout of 60%. Dropout is important given that these models could easily overfit the small training sets. For the HMCN-R models, we use independent dropout masks for the local outputs, though no dropout is used in the recurrent layers.

All networks minimize the loss function proposed in Section 2.3 via mini-batch gradient descent, where each iteration is computed over a mini-batch of 2 training objects. Even though the use of larger mini-batch sizes (e.g., 4, 8, 12, 16, 24) yields similar results, we noticed that one can achieve better results by training HMCN models with smaller batches. We do not perform per-dataset hyper-parameter optimization unlike the current state-of-the-art approaches. We kept the same hyper-parameters across all datasets to evaluate our method's capability of learning from distinct datasets with varying characteristics. The only hyper-parameter that we vary across datasets is regularization, which is regulated over validation data in order to mitigate eventual overfitting.

4. Experimental Analysis

We perform two distinct sets of experiments. First, we analyze the performance of different architectural choices for HMCN-F on validation data in all 8 FunCat protein function prediction datasets. Our goal is to show to the reader the impact of critical components such as input reuse and the local outputs within HMCN. In the second set of experiments, we compare HMCN-F and HMCN-R with the current state-of-the-art approaches on test data of all the 21 datasets employed in this paper.

4.1. Impact of the Architectural Components

We analyze the impact that the local outputs and input features reuse produce on HMCN-F. We show the results of the architecture when using only the global flow with no local outputs as a baseline (HMCN-F-Global Only). Such baseline version is a deep neural network with $|H|$ layers with ReLU (Nair & Hinton, 2010), batch normalization (Ioffe & Szegedy, 2015), and residual connections (He et al., 2016). Table 2 shows the results regarding $AU(\overline{PRC})$ on validation data of the Funcat datasets. Note that the best performance is achieved when using the proposed incarnation of HMCN-F that combines both input reuse and local outputs. The single most important design choice is the inclusion of local outputs, which indeed seems to be providing extra local information with respect to inter-class relationships. Input reuse also seems to largely increase the performance of the network. With those results in mind, we use only the full version of HMCN in both feedforward and recurrent versions (namely HMCN-Fand HMCN-R) for comparing with the state-of-the-art for the test sets of all 21 datasets in the following set of experiments.

4.2. Effect of Hierarchical Violation Penalty

Table 3 depicts the effect of λ for weighting hierarchical violations. By using $\lambda = 0.1$ in HMCN-F we can achieve slightly better results, while for HMCN-R the improvements are more significant due to more consistent predictions. In addition, a side effect of this penalty is the reduction in magnitude of the weights, which has an implicit regularizer effect. We have found that even with no penalty ($\lambda = 0.0$) the networks were capable of learning consistent hierarchical paths. However, λ indeed helps in mitigating consistency problems by reducing the amount of updates performed in the final post-processing step.

4.3. HMCN vs. State-of-the-Art

We compare the best HMCN-F version from the previous subsections (i.e., the complete version with $\lambda = 0.1$) with both HMCN-R and the current state-of-the-art approaches, this time on the test sets of all datasets from Table 1. Table 4 presents the results of the experiments with HMCN-F,

Table 3. IMPACT OF HIERARCHICAL VIOLATIONS IN HMCN. VALUES PRESENTED ARE OF $AU(\overline{PRC})$ ON VALIDATION DATA.

	HMCN-F			HMCN-R	
DATASET	$\lambda = 0.0$	$\lambda = 0.1$	$\lambda = 1.0$	$\lambda = 0.0$	$\lambda = 0.1$
CELLCYCLE	0.250	**0.252**	0.233	0.245	0.249
DERISI	0.191	**0.193**	0.185	0.188	0.189
EISEN	0.293	**0.298**	0.264	0.292	**0.298**
EXPR	0.296	**0.301**	0.263	0.295	0.300
GASCH1	0.281	**0.284**	0.259	0.278	0.283
GASCH2	0.252	**0.254**	0.232	0.244	0.249
SEP	0.283	**0.291**	0.262	0.285	0.290
SPO	0.210	**0.211**	0.197	0.205	0.207

HMCN-R, and the baseline approaches Clus-HMC-Ens, Clus-HMC, CSSA, and HMC-LMLP. Note that we do not compare with the recent PLS+OPP approach (Sun et al., 2016) since it does not generate probabilistic outputs, preventing the correct generation of precision-recall curves, which is the standard evaluation measure for experiments in this research area. We highlight the best absolute values (underlined) of $AU(\overline{PRC})$ that were obtained per dataset. HMC-LMLP (Cerri et al., 2016) does not provide results for the GO datsets.

The first analysis we perform is regarding HMCN-F and its performance when compared with the current state-of-the-art. Note that HMCN-F is the algorithm with the greatest number of wins (it reaches top performance in all datasets but one) and best average ranking (1.07), comfortably outperforming the state-of-the-art approaches. It is the first time that a method in the literature outperforms the ensemble of Clus-HMC and CSSA in all FunCat and GO datasets of protein function. Moreover, note that the only draws of HMCN-F are regarding our own recurrent network. HMCN-R, in turn, also provides better average ranking than the baseline approaches, outperforming them in 19 out of 21 datasets. HMCN-R demonstrates similar performance to HMCN-F, being slightly overperformed on most datasets. Even though HMCN-F seems to be better than HMCN-R, recall that HMCN-R is much lighter in terms of total amount of parameters. For instance, HMCN-R has $\approx$ 3M parameters for the Cellcyle GO dataset, whereas HMCN-F has $\approx$ 7M. For very large hierarchies, HMCN-R is probably a better choice than HMCN-F considering the trade-off *performance vs. computational cost*.

We also verify the statistical significance of the results following the recommendation of (Demšar, 2006). First, we execute the Friedman test, which indicates the existence of significant differences with a p-value of 1.58×10^{-32}. We then move to the post-hoc Nemenyi test, which is presented in Figure 3. For this particular analysis, we employ the graphical representation suggested in (Demšar, 2006), the so-called *critical diagrams*. In this diagram, a horizontal line represents the axis on which we plot the average rank values of the methods. We connect the groups of algorithms that do not differ significantly through a horizontal line. We also show the critical difference given by the test, which is $CD = 1.645$. Observe that HMCN-F outperforms all baseline approaches with statistical significance. The test does not show, however, a significant difference between HMCN-R and Clus-HMC-Ens, between Clus-HMC-Ens and CSSA, nor between CSSA and Clus-HMC.

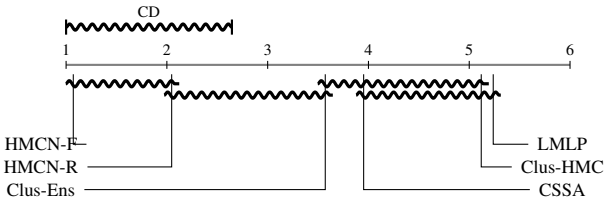

Figure 3. Critical diagram for the Nemenyi's statistical test.

5. Related Work

Vens et al. (Vens et al., 2008) proposed three classification algorithms based on the concept of Predictive Clustering Trees (PCT). The best of them, Clus-HMC, is a global approach that induces a single decision tree to deal with the entire class hierarchy. Schietgat et al. (Schietgat et al., 2010) proposed a bagging strategy for generating ensembles of Clus-HMC trees, namely Clus-HMC-Ens, considerably improving over Clus-HMC. Cesa-Bianchi et al. (Cesa-Bianchi et al., 2011) investigated the synergy between different local-based strategies related to gene function prediction in FunCat annotated genes. They integrated kernel-based data fusion tools and ensemble algorithms with cost-sensitive HMC methods (Cesa-Bianchi & Valentini, 2010; Valentini, 2011). Cerri et al. (Cerri et al., 2011; 2013a; 2016) proposed HMC-LMLP, a local HMC approach based on a chain of Multi-Layer Perceptrons (MLPs) with a single hidden layer each. Each MLP is responsible for predicting the classes in a given level of the hierarchy, and the input of a given MLP

Table 4. COMPARISON OF HMCN WITH THE BASELINE METHODS. THE VALUES PRESENTED ARE OF $AU(\overline{PRC})$. VALUES IN BOLD OUTPERFORM THE BEST RESULTS PUBLISHED SO FAR, AND UNDERLINED ARE THE NOVEL STATE-OF-THE-ART FOR EACH DATASET.

DATASET	HMCN-F	HMCN-R	CLUS-HMC	CSSA	CLUS-ENS	HMC-LMLP
CELLCYCLE (FUNCAT)	**0.252**	**0.247**	0.172	0.188	0.227	0.207
DERISI (FUNCAT)	**0.193**	**0.189**	0.175	0.186	0.188	0.183
EISEN (FUNCAT)	**0.298**	**0.298**	0.204	0.212	0.271	0.245
EXPR (FUNCAT)	**0.301**	**0.300**	0.210	0.220	0.271	0.243
GASCH1 (FUNCAT)	**0.284**	**0.283**	0.205	0.208	0.267	0.236
GASCH2 (FUNCAT)	**0.254**	**0.249**	0.195	0.210	0.227	0.211
SEQ (FUNCAT)	**0.291**	**0.290**	0.211	0.218	0.284	0.236
SPO (FUNCAT)	**0.211**	0.210	0.186	0.208	0.210	0.186
CELLCYCLE (GO)	**0.400**	**0.395**	0.357	0.366	0.387	-
DERISI (GO)	**0.369**	**0.368**	0.355	0.357	0.363	-
EISEN (GO)	**0.440**	**0.435**	0.380	0.401	0.433	-
EXPR (GO)	**0.452**	**0.450**	0.368	0.384	0.418	-
GASCH1 (GO)	**0.428**	**0.416**	0.371	0.383	0.415	-
GASCH2 (GO)	**0.465**	**0.463**	0.369	0.373	0.395	-
SEQ (GO)	**0.447**	**0.443**	0.386	0.387	0.435	-
SPO (GO)	**0.376**	**0.375**	0.345	0.352	0.372	-
DIATOMS	**0.530**	**0.514**	0.167	-	0.379	-
ENRON	**0.724**	**0.710**	0.638	-	0.681	-
IMCLEF07A	**0.950**	**0.904**	0.574	-	0.777	-
IMCLEF07D	**0.920**	**0.897**	0.749	-	0.863	-
REUTERS	0.649	0.610	0.562	-	**0.703**	-
AVERAGE RANKING	**1.07**	2.04	5.12	3.96	3.57	-

is given by the output of the previous MLP in the chain. All MLPs are trained separately with distinct strategies of data augmentation, and the results are presented for FunCat annotated protein data. Bi and Kwok (Bi & Kwok, 2011) proposed a problem transformation approach for tree and DAG hierarchies that can be used with any classifier. It employs kernel dependency estimation (KDE) to reduce the number of labels to a manageable number of single-label learning problems. To preserve the hierarchical information among labels, they developed a generalised Condensing Sort and Select Algorithm (CSSA), which finds optimal approximation subtrees. They project the labels to a 50-dimensional space and then use ridge regression in the learning step. Triguero and Vens (Triguero & Vens, 2016) studied alternatives to perform the final labelling in HMC problems. The authors evaluated the Clus-HMC-Ens method when using single and multiple thresholds to transform the continuous prediction scores into actual binary labels. To choose thresholds, two approaches were proposed: to optimize a given evaluation measure or to simulate training set properties within the test set. They concluded that selecting thresholds for each class is a good alternative, resulting in improved label-sets and faster execution time. Sun et al. (Sun et al., 2016) proposed PLS+OPP, in which the classification task is formulated as a path selection problem. They used partial least squares to transform the label prediction problem into an optimal-path prediction strategy. Each multi-label prediction is a connected subgraph that comprises a small number of paths, and the optimal paths are found and merged to provide the final predictions.

6. Conclusions and Future Work

We proposed novel deep neural network architectures for hierarchical multi-label classification in tree-structured and DAG-structured hierarchies, namely HMCN. We designed two distinct versions of HMCN: a more robust feedforward version (though with the potential of having much more parameters), namely HMCN-F, and a more efficient recurrent version that leverages shared weights and an LSTM-like structure for encoding hierarchical information, namely HMCN-R. To the best of our knowledge, HMCN is the first HMC method that benefits from both local and global information at the same time, while penalizing hierarchical violations. We performed several experiments using 21 datasets from four distinct domains. Results show that both HMCN-R and HMCN-R were capable of comfortably outperforming the state-of-the-art methods, establishing themselves as the novel state-of-the-art for HMC tasks. As future work, we intend to evaluate HMCN models for hierarchical classification based on raw images (e.g., using the ImageNet synsets), to further explore better text encoding strategies in HMC datasets, and provide visualization techniques for HMCN models.

Acknowledgements

We would like to thank Google and the Brazilian research agencies CAPES, CNPq, and FAPERGS for funding this research. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs that were used for running the experiments.

References

Barros, R. C., Cerri, R., Freitas, A. A., and Carvalho, A. C. P. L. F. Probabilistic Clustering for Hierarchical Multi-Label Classification of Protein Functions. In *European Conference on Machine Learning (ECML/PKDD 2013)*, pp. 385–400. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-40991-2.

Bi, Wei and Kwok, James. Multi-Label Classification on Tree- and DAG-Structured Hierarchies. In *International Conference on Machine Learning*, ICML'11, pp. 17–24. ACM, 2011.

Cerri, R., Barros, R. C., and de Carvalho, A. C. P. L. F. Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks. In *Intelligent Systems Design and Applications (ISDA)*, pp. 337 –343, nov. 2011.

Cerri, R., Barros, R. C., and Carvalho, A. C. P. L. F. A genetic algorithm for hierarchical multi-label classification. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pp. 250–255, New York, NY, USA, 2012. ACM.

Cerri, R., Barros, R. C., and Carvalho, A. C. P. L. F. Hierarchical multi-label classification using local neural networks. *Journal of Computer and System Sciences*, 80(1):39–56, 2013a. ISSN 0022-0000. doi: http://dx.doi.org/10.1016/j.jcss.2013.03.007.

Cerri, R., Barros, R. C., de Carvalho, A. C. P. L. F., and Freitas, A. A. A grammatical evolution algorithm for generation of hierarchical multi-label classification rules. In *2013 IEEE Congress on Evolutionary Computation*, pp. 454–461, June 2013b. doi: 10.1109/CEC.2013.6557604.

Cerri, R., Barros, R. C., and de Carvalho, A. C. P. L. F. Hierarchical classification of gene ontology-based protein functions with neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, July 2015. doi: 10.1109/IJCNN.2015.7280474.

Cerri, Ricardo, Barros, R. C., de Carvalho, A. C. P. L. F., and Jin, Yaochu. Reduction Strategies for Hierarchical Multi-Label Classification in Protein Function Prediction. *BMC Bioinformatics*, 17:373, 2016. doi: 10.1186/s12859-016-1232-1.

Cesa-Bianchi, N and Valentini, G. Hierarchical cost-sensitive algorithms for genome-wide gene function prediction. *Journal of Machine Learning Research*, 8:14–29, 2010.

Cesa-Bianchi, N., Re, M., and Valentini, G. Synergy of multi-label hierarchical ensembles, data fusion, and cost-sensitive methods for gene functional inference. *Machine Learning*, pp. 1–33, 2011. ISSN 0885-6125.

Cesa-Bianchi, Nicolò, Gentile, Claudio, and Zaniboni, Luca. Incremental algorithms for hierarchical classification. *Machine Learning*, 7:31–54, 2006. ISSN 1533-7928.

Cissé, Moustapha, Al-Shedivat, Maruan, and Bengio, Samy. Adios: Architectures deep in output space. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 2770–2779. JMLR.org, 2016.

Costa, Eduardo P., Lorena, Ana C., Carvalho, André C. P. L. F., and Freitas, Alex A. Comparing several approaches for hierarchical classification of proteins with decision trees. In *Brazilian Symposium on Bioinformatics*, volume 4643 of *LNBI*, pp. 126–137. Springer-Verlag, 2007.

Demšar, Janez. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Dimitrovski, Ivica, Kocev, Dragi, Loskovska, Suzana, and Džeroski, Sašo. Hierarchical annotation of medical images. *Pattern Recognition*, 44(10-11):2436–2449, 2011.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778, 2016.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456, 2015.

Kingma, Diederik and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kiritchenko, Svetlana, Matwin, Stan, and Famili, A. Fazel. Hierarchical text categorization as a tool of associating genes with gene ontology codes. In *European Workshop on Data Mining and Text Mining in Bioinformatics*, pp. 30–34, 2004.

Klimt, Bryan and Yang, Yiming. The enron corpus: A new dataset for email classification research. In *ECML '04: Proceedings of the 18th European Conference on Machine Learning – LNCS 3201*, pp. 217–226. Springer Berlin / Heidelberg, 2004.

Lee, Chen-Yu, Xie, Saining, Gallagher, Patrick W., Zhang, Zhengyou, and Tu, Zhuowen. Deeply-supervised nets. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015*, 2015.

Lewis, David D., Yang, Yiming, Rose, Tony G., and Li, Fan. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

Mayne, A. and Perry, R. Hierarchically classifying documents with multiple labels. In *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 133–139, 2009.

Nair, Vinod and Hinton, Geoffrey E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

Otero, Fernando, Freitas, Alex, and Johnson, Collin. A hierarchical multi-label classification ant colony algorithm for protein function prediction. *Memetic Computing*, 2: 165–181, 2010.

Rousu, Juho, Saunders, Craig, Szedmak, Sandor, and Shawe-Taylor, John. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626, 2006. ISSN 1533-7928.

Schietgat, Leander, Vens, Celine, Struyf, Jan, Blockeel, Hendrik, Kocev, Dragi, and Dzeroski, Saso. Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics*, 11:2, 2010.

Silla, Carlos and Freitas, Alex. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22:31–72, 2010. ISSN 1384-5810.

Sun, Zhengya, Zhao, Yangyang, Cao, Dong, and Hao, Hongwei. Hierarchical multilabel classification with optimal path prediction. *Neural Processing Letters*, pp. 1–15, 2016.

Triguero, Isaac and Vens, Celine. Labelling strategies for hierarchical multi-label classification techniques. *Pattern Recogn.*, 56(C):170–183, August 2016. ISSN 0031-3203.

Valentini, Giorgio. True path rule hierarchical ensembles for genome-wide gene function prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(3):832–847, May 2011. ISSN 1545-5963.

Vens, Celine, Struyf, Jan, Schietgat, Leander, Džeroski, Sašo, and Blockeel, Hendrik. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73: 185–214, 2008. ISSN 0885-6125.

Wehrmann, Jônatas and Barros, R. C. Bidirectional retrieval made simple. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2018)*, 2018.

Wehrmann, Jônatas, Barros, R. C., Dôres, Silvia N das, and Cerri, Ricardo. Hierarchical multi-label classification with chained neural networks. In *Proceedings of the Symposium on Applied Computing*, pp. 790–795. ACM, 2017.

Wehrmann, Jônatas, Mattjie, Anderson, and Barros, R.C. Order embeddings and character-level convolutions for multimodal alignment. *Pattern Recognition Letters*, 102: 15–22, 2018.