Nested Learning for Multi-Level Classification

Raphaël Achddou, J. Matias Di Martino, Guillermo Sapiro

NESTED LEARNING FOR MULTI-LEVEL CLASSIFICATION

Raphaël Achddou,[1] *J.Matias Di Martino,*[2] *and Guillermo Sapiro*[2]

[1]LTCI Télécom Paris, Institut Polytechnique de Paris, [2]Duke University

ABSTRACT

Deep neural networks models are generally designed and trained for a specific type and *quality* of data. In this work, we address this problem in the context of *nested learning*. For many applications, both the input data, at training and testing, and the prediction can be conceived at multiple nested quality/resolutions. We show that by leveraging this multi-scale information, the problem of poor generalization and prediction overconfidence, as well as the exploitation of multiple training data quality, can be efficiently addressed. We evaluate the proposed ideas in six public datasets: MNIST, Fashion-MNIST, CIFAR10, CIFAR100, Plantvillage, and DBPEDIA. We observe that coarsely annotated data can help to solve fine predictions and reduce overconfidence significantly. We also show that hierarchical learning produces models intrinsically more robust to adversarial attacks and data perturbations.

1. INTRODUCTION

Deep Neural Networks (DNNs) tend to be overconfident about their predictions and limited to the task and data they have been trained on [1, 2, 3]. In this paper, we argue that this happens, among other reasons, because models are designed to learn a specific task in an end to end fashion [4]. Humans, in contrast, learn in a nested and hierarchical way. For example, learning to identify people before recognizing individuals, vehicles before appreciating different car brands, and so forth. In the present paper, we discuss a set of crucial ideas to transform end to end DNNs into a model that can be trained with data of different quality, and that provides prediction at multiple resolutions (with their associated calibrated confidence). Some of the main aspects we address are: how to train a nested model effectively, how to design nested architectures (framing the discussion with information theory), and how to combine nested outputs.

Recently, Bilal et al. showed that convolutional neural networks (CNNs) naturally tend to learn high-level hierarchical features that discriminate groups of classes in the first layers, while the deeper layers develop more specialized feature detectors [5]. We design a neural network framework that explicitly enforces this behavior by creating a sequence of low dimensional feature embeddings for each level in the labels' taxonomy, thanks to a series of information bottlenecks (see Fig. 1) [6, 7]. We show that skipped connections allow finer embeddings to access information (if available) and empirically evaluate both the information flow and the impact in the model's performance. The code and experiments associated with this work are open source.[1]

2. NESTED LEARNING

Preliminaries and notations. An input sample x is represented as a realization of a random variable X. We represent as $\mathcal{X}$ the alphabet of X. Associated with each input x, we consider a *ground truth label* y, also modeled as the realization of the random variable Y. Of course, Y and X are not independent; the problem of classification can be stated as inferring y from an observed sample x, i.e., $Y \rightarrow X \rightarrow \hat{Y}$. $\hat{Y}$ denotes a new random variable (estimated from X) which *approximates* Y. Subscripts will be used to indicate the granularity of each label, i.e., Y_{i-1} is the closest coarse level of Y_i (e.g., $y_1 =$ vehicle, $y_2 =$ car, and $y_3 =$ sport car).

Definition 2.1. We define $Y_1, ..., Y_n$ as a discrete sequence of nested labels if $H(Y_i|Y_{i+1}) = 0 \; \forall i \in [1, n-1]$. H denotes the standard definition of entropy for discrete random variables.

Definition 2.2. A discrete sequence of nested labels $Y_1, ..., Y_n$ is strictly nested if $H(Y_i|Y_{i-1}) < H(Y_i) \; \forall i \in [2, n]$.

2.1. Nested network

Nested information bottlenecks. Assume the input X has information about a sequence of strictly nested labels Y_i . Exploiting this, we will sequentially *compress* the information on X using standard DNN layers (convolutional, pooling, normalization, and activation) as we schematically illustrate in Fig. 1. We begin by guiding the network to find a low dimensional feature representation f_1 such that $H(f_1(X)) \ll H(X)$ while, $I(f_1(X), Y_1)$ is *close* to $I(X, Y_1)$. $I(U, V)$ stands for the standard mutual information between discrete random variable U and V. DNNs are remarkably efficient at compressing and extracting the mutual information between high dimensional inputs and target labels [6].

The second step consists of learning complementary information that, combined with the representation f_1, allows

[1]https://github.com/raphaelachddou/ICASSP_2021_Nested_learning

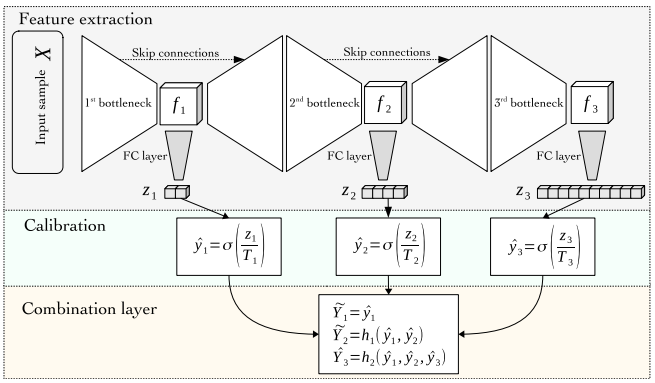

Fig. 1. Example of an architecture template for the proposed nested models. The first block extracts unrefined nested predictions. The calibration block presented in Section 2.2 reduces the overconfidence and allows a quantitatively meaningful combination of predictions, which further improves robustness and accuracy.

to achieve a second representation f_2 from which the second hierarchical label Y_2 can be inferred. To this end, skipped connections play a critical role, as we will discuss next. Using the definition of mutual information and the property that the sequence $\{Y_i\}$ is a set of strictly nested labels, we have

$$I(X, Y_i) = H(X) - \underbrace{H(X|Y_i)}_{>H(X|Y_{i+1})} < I(X, Y_{i+1}). \quad (1)$$

On the other hand, we want each feature embedding f_i to compress the information of X while $I(f_i(X), Y_i) \approx I(X, Y_i)$. If we do not consider skipped connections, $X \rightarrow f_i(X) \rightarrow f_{i+1}(X)$ forms a Markov chain where $I(X, f_{i+1}(X)) \leq I(X, f_i(X))$ (data-processing inequality) contradicting (1). While in most DNNs architectures skipped connections are included to encourage the model compactness and to mitigate vanishing gradients, in the present work they are included to circumvent the data-processing inequality (see Section 3 for empirical validation).

Output combination layer. Since nested predictions are related to each other, we implement a non-trainable layer that combines nested outputs to refine finer predictions, i.e., $\{\hat{Y}_1, ..., \hat{Y}_i\} \rightarrow \tilde{Y}_i$.

As described by Hein et al. [2], DNN models tend to produce over-confident predictions, and it is indeed frequent that the output confidence is significantly larger than the prediction accuracy. This phenomenon is known as the problem of uncalibrated prediction. This mismatch can be addressed by calibrating the score outputs, which consists of mapping output scores to estimate the actual class probability. Calibration is well defined and thoroughly explained in [8].

Let us denote $P_{\hat{Y}_i}(q)$ the calibrated output of the network that approximates $P(Y_i = q)$. Then, we can use the estimated probability associated to a fine label $P_{\hat{Y}_i}$ to compute the conditional probability $P(Y_i = y_i|Y_{i-1} = k)$. This is achieved by re-normalizing the finer labels associated to the same coarse label, i.e.,

$$P_{\hat{Y}_i|\hat{Y}_{i-1}}(q) = \frac{P_{\hat{Y}_i}(q)}{\sum_{w \in \mathcal{Y}_i^{k_q}} P_{\hat{Y}_i}(w)}, \quad (2)$$

where $\mathcal{Y}_i^{k_q}$ denotes the set of labels at granularity level i that share with q the same coarser label k_q. Finally, the estimated conditional probability is combined with the prior of the coarser prediction to recompute the fine prediction $P'_{\hat{Y}_i}(q) = P_{\hat{Y}_i|\hat{Y}_{i-1}}(q) P_{\hat{Y}_{i-1}}(k_q)$, which is then refined recursively until we reach the coarser level: $P'_{\hat{Y}_i}(q) = P_{\hat{Y}_i|\hat{Y}_{i-1}} P_{\hat{Y}_{i-1}|\hat{Y}_{i-2}} ... P_{\hat{Y}_0}$. This is a generalization of the combination method for two nested levels [9].

2.2. Nested training and calibration

Training. Let $G_{\theta,\eta}(x) = (f_i(x, (\theta_j)_{j=1,..,i}), g_i(f_i, \eta_i))_{i=1,..,m}$ be the function coded by our network, where m denotes the number of granularity levels and as before x represents an input sample. Each sub-function g_i corresponds to the output of granularity i (computed from the feature bottleneck f_i). G depends on parameters $(\theta_j)_{j=1,..,i}$ which are common to the sub-functions of coarser granularities, and some granularity-specific parameters η_i. The architecture is composed of a trunk of convolutional layers with parameters θ and fully connected layers for each intermediate outputs with parameters η (Fig. 1).

Training this type of model with a disparity of samples per granularity is challenging, and naively sampling random batches of training data leads to a noisy gradient computation [10]. To overcome this, we organize the training samples and train the network in a cascaded manner. First, the dataset $\mathcal{D}$ is organized in subsets of samples labeled up to granularity i for $i = 1, .., m$ (since we are focusing on strictly nested problems, knowing a fine label implies knowing all the coarser labels, the reciprocal is false). $\mathcal{D} = (\mathbf{x}, \mathbf{y})$ with $\mathbf{x}$ the set of inputs and $\mathbf{y}$ the set of labels. We consider that $\mathbf{x} = (\mathbf{x_i})_{\mathbf{i=1,..,m}}$ and $\mathbf{y} = (\mathbf{y_i})_{\mathbf{i=1,..,m}}$, where $\mathcal{D}_i = (\mathbf{x_i}, \mathbf{y_i})$ represents the subset of data for which the label is known up to the granularity level i.

We train the model to solve a sequence of optimization problems using $(\mathbf{x_i}, \mathbf{y_i})$ as the training examples at each step. The training sequence can be expressed as (P_i) : $\min_{(\theta_j,\eta_j)_{j=1,..,i}} \sum_{j=1}^{i} \alpha_j \mathcal{L}_{n_j}(\hat{Y}_j, Y_j)$, where $\mathcal{L}_n$ is the n-categorical cross-entropy and α are the weights for the loss associated to each prediction level. Training starts on the coarser level, proceeding with the consecutive finer level iteratively. We empirically compared the proposed training methodology with other schemes.

Calibration. To mitigate prediction overconfidence, we implement a two-step calibration method. First, we add a "rejection" class for each level of granularity. Synthetic samples associated with this class are generated from a uniform distribution. This is a simple and effective idea to mitigate

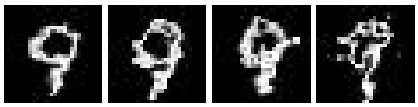

Fig. 2. Four levels of the 'turbulence-like" image distortion inspired by [14].

out-of-distribution overconfidence. Still, a naive implementation would be untractable, since a dense coverage on the input space requires a number of synthetic samples that grows exponentially with the number of dimensions, leading to very time consuming and memory intensive training. Our architecture design is suitable for a practical solution to this problem by injecting the samples associated with the rejection class at the low-dimensional bottleneck representation (so overconfidence is tackled at the multiple resolutions).

A second calibration step aims to reduce the prediction overconfidence in the input space regions, where the probability of multiple classes overlap. We adopt temperature scaling introduced by Guo et al. [1]. In short, this technique consists of scaling the output of the fully-connected layer before the softmax activation by an optimal temperature parameter.

3. EXPERIMENTS AND DISCUSSION

Method	DBPEDIA			Plantvillage			CIFAR100	
	C	M	F	C	M	F	C	F
# samples	50k	50k	50k	4.5k	4.5k	9k	25k	25k
End to end	91.8	81.8	76.4	95.2	94.5	92.3	70.0	59.6
Nested	**98.1**	**93.4**	**84.7**	**97.9**	**97.5**	**94.6**	**79.3**	**64.9**

Table 1. Accuracy of the end to end and nested models for: Dbpedia [11], Plantvillage [12] and Cifar100 [13]. C, M and F stands for Coarse, Middle, and Fine, respectively. In the first row we report the amount of coarse, middle, and fine data that was used for the nested training. The amount of data used for the end to end training corresponds to the amount of fine data.

Dataset	Coarse category 1		Coarse category 2		
	Middle 1	Middle 2	Middle 3		Middle 4
MNIST	3,8,5	0,6	9,4,7		1,2
F-MNIST	shirt,t-shirt, dress	coat, pull-over	boots,sneakers, sandals		pants, bags
	Middle 1	Middle 2	Middle 3	Middle 4	Middle 5
Cifar10	truck, car	plane, boat	cat dog	deer, horse	bird, frog

Table 2. Visually based three-level nested taxonomies for MNIST, Fashion-MNIST, and Cifar10.

We consider six publicly available datasets for experimental evaluation: the handwritten digits from MNIST [15], the small clothes images from Fashion-MNIST [16], CIFAR10 [17], CIFAR100 [13], the Plantvillage dataset [12], and DBPEDIA [11](Wikipedia's articles). We created visually based taxonomies for the first three datasets (Table 2), and used the nested categories provided for the remaining ones.

3.1. Nested learning vs. end to end learning

For a fair comparison, we compare two identical architectures, one trained on an end to end fashion (only optimizing for the finer prediction), and the second one following nested learning steps described in Section 2. We refer to these as "end to end" and "nested", respectively.

Can we improve our knowledge of the fine task by looking at the coarse ones? To understand how coarse annotations impact the performance on a finer task, we compared end to end models trained exclusively with fine data $\mathcal{D}_A \equiv \mathcal{D}_3$ and nested models trained with the same amount of fine data plus coarse data $\mathcal{D}_B \equiv \mathcal{D}_3 \cup \mathcal{D}_2 \cup \mathcal{D}_1$. Naturally, training with additional coarse and middle data improves the accuracy of the coarse and intermediate prediction, as we can see in Table 1. More interestingly, we also observe that additional coarse annotations lead to better fine models. More precisely, on MNIST, Fashion MNIST, and Cifar10 datasets, classification robustness improved when test samples drifted from the train examples (see Fig. 2) with an average accuracy gain of 2%, and a 7% reduction in the prediction overconfidence (gap between the predicted confidence and the actual accuracy).

With a fixed training budget, what is the better trade-off? We studied for a specific budget (this is, getting coarse annotations comes at the expense of less fine annotations), which are the level of annotations that contribute the most to improve the learning process? We tested models with more fine annotations, or more coarse and middle annotations. Again, we observed (on MNIST, Fashion-MNIST, and CIFAR-10) that the models trained with additional coarse and middle samples tend to be more robust to distortions (average 2.1 % gap in fine accuracy across the datasets) and less overconfident (7.5% decrease), even compared with models trained with 1.5 times more fine annotations.

Robustness to adversarial attacks. In previous experiments, the noise and distortions applied to the test data is agnostic to the classification task. Complementing previous experiments, we tested models' performance to active (adversarial) perturbations. To this end, most popular state-of-the-art gradient-based attacks such as the fast gradient sign method (FGSM) [18], Deepfool [19] and Saliency based attacks [20] were implemented.

We empirically observed that to reach a given error rate for FGSM, the attacker needs to add more than twice as much adversarial noise to the network trained in a nested fashion compared with its standard counterpart (additional details and numerical results are provided as supplementary material). In addition, when we fit an attack on the fine output, the coarse and intermediate predictions are significantly less affected for models learned with nested learning, as reported in Table 3.

3.2. Ablation studies

Skipped connections. As discussed in previous sections, skipped connections (SC) are included to allow information

Type of attack	coarse acc.	middle acc.	fine acc.
FGSM nested	**86.8**	**60.9**	0.0
FGSM end to end	25.2	15.6	0.0
Deepfool nested	**57.2**	**35.7**	0.0
Deepfool end to end	44.5	27.5	0.0
Saliency nested	**82.2**	**71.5**	0.0
Saliency end to end	27.4	17.2	0.0

Table 3. Accuracy for the middle and coarse prediction when the fine prediction is adversarially attacked. In this experiment we increased for each test sample the magnitude of the attack until its fine prediction becomes incorrect. Then, we compute the middle and coarse prediction for the end to end and nested models. The nested model explicitly provides middle and coarse outputs, while for the end to end model the nested and coarse labels are computed from the fine prediction.

to flow from the input to the finer feature representation. To test how the ideas outlined in Section 2.1 affect deep models in practice, we compared equivalent models with and without SC for the MNIST model, and we empirically measured both models performance and the flow of information. To estimate the mutual information between 2 high dimensional random variables, we implemented the MINE algorithm [21].

We observed that the model with SC performs slightly better on images from the original distribution, and much better on distorted data, with a 10% average gap for the fine task. We also see that the performance gap on the coarse prediction is relatively small (2%), while, as expected, the gap increases for the middle (7.5%) and fine predictions (10%). In addition, the difference in the mutual information between coarse and fine feature embeddings is doubled when SC are removed.

Cascaded Training. We experimentally compared the proposed cascaded methodology with the standard methodology (i.e., selecting batches of random samples out of the training set). We observed that cascaded training achieves substantially better performances than the traditional approach, with a 4% increase on the fine task, a 3% increase on the intermediate task, and a 2% on the coarse task. Additionally, we studied the behavior of the network during training, e.g., see the results presented in Fig. 3. The proposed protocol is more suitable for the multi-level problem leading to faster convergence and better models; also, prediction accuracy becomes more stable. These findings provide further evidence that cascaded training mitigates the noises of the stochastic gradient estimation.

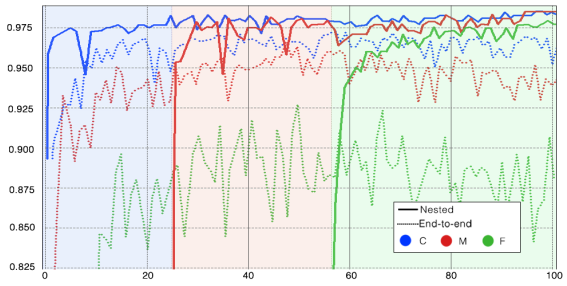

Fig. 3. End to end versus nested training. We compare the accuracy of the same model trained with a cascaded and a traditional training scheme. Blue, evolution of the accuracy of the coarse prediction; red, accuracy of the intermediate prediction; and green, the fine prediction

4. RELATED WORKS

The development of hierarchical and nested solutions has received significant attention recently. A central example is the work of Kim et al. [22], who proposed a nested sparse architecture with the emphasis on having a resource-aware and versatile implementation. In contrast with our work, they do not study how to combine these nested outputs into a refined single prediction, nor engineer a reliable confidence measure associated with them. From an architectural perspective, the main difference with our work is that their model predictions are obtained at the bottom of the network (similarly to an end to end approached), whereas we enforce sequential information bottlenecks with nested and intermediate predictions. Another relevant example is the work proposed by Yan et al. [9]. They introduced hierarchical deep CNNs (HD-CNNs), embedding CNNs into a two-level category hierarchy. Similarly to our work, they propose to distinguish a coarse class using an initial classifier and then to refine the classification into a second level for each coarse category. In contrast to ours, their components are designed specifically for a two-level hierarchy; in that sense, our work generalizes their ideas to multiple levels. In addition, while Yan et al. main goal is to propose a practical solution and solve many fundamental implementation challenges, we also focus on framing and contextualizing nested learning into a theoretical framework. Other relevant and similar works are [23] [24].

5. CONCLUSION

We proposed a practical solution and framed in the theory of information the problem of nested learning. In several diverse datasets, we showed that it is possible to leverage heterogeneously annotated data, meaning labeled with a different level of precision, to obtain more robust models. One of the main advantages is that the proposed models provide predictions at different resolutions (e.g., this is a picture of a vehicle, particularly a car, and specifically a car model). We demonstrated that when the input data shifts from the training distributions, even if the fine prediction becomes unreliable, coarser predictions can still be made. In contrast, end to end approaches tend to be "all or nothing." Nested learning leads to solutions more robust to adversarial attacks, even though there is no specific effort nor particular adaptations to tackle them. We theoretically argued and empirically demonstrated that measuring the flow of mutual information can provide meaningful input to define the architecture profile of optimal implementations.

6. REFERENCES

[1] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML*, 2017.

[2] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf, "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem," *CoRR*, 2018.

[3] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *CoRR*, 2014.

[4] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.

[5] Alsallakh Bilal, Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren, "Do convolutional neural networks learn class hierarchy?," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, 2018.

[6] Ravid Shwartz-Ziv and Naftali Tishby, "Opening the black box of deep neural networks via information," *CoRR*, 2017.

[7] Naftali Tishby, Fernando Pereira, and William Bialek, "The information bottleneck method," *Proceedings of the 37th Allerton Conference on Communication, Control and Computation*, vol. 49, 07 2001.

[8] Bianca Zadrozny and Charles Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002.

[9] Zhicheng Yan, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Robinson Piramuthu, "HD-CNN: hierarchical deep convolutional neural network for image classification," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[10] Iasonas Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[11] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*. "Springer Berlin Heidelberg", 2007.

[12] Sharada P Mohanty, David P Hughes, and Marcel Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in plant science*, 2016.

[13] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, "Cifar-100 (canadian institute for advanced research)," .

[14] Enric Meinhardt-Llopis and Mario Micheli, "Implementation of the centroid method for the correction of turbulence," *Image Processing On Line*, 2014.

[15] Yann LeCun and Corinna Cortes, "MNIST handwritten digit database," 2010.

[16] Han Xiao, Kashif Rasul, and Roland Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, 2017.

[17] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, "Cifar-10 (canadian institute for advanced research)," .

[18] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[19] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[20] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017.

[21] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm, "MINE: Mutual information neural estimation," *arXiv preprint arXiv:1801.04062*, 2018.

[22] Eunwoo Kim, Chanho Ahn, and Songhwai Oh, "NestedNet: learning nested sparse structures in deep neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[23] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros, "Hierarchical multi-label classification networks," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 5075–5084.

[24] Isaac Triguero and Celine Vens, "Labelling strategies for hierarchical multi-label classification techniques," *Pattern Recognition*, 2016.